The Complete Reference

"Very well-written. The author expresses the same enthusiasm in this book as he shared with me when he trained me in SharePoint technologies...I highly recommend this book to all of my SharePoint friends and customers!"
—*Rich Snarski, SharePoint Architect, Microsoft Consulting Services*

Microsoft® Office SharePoint® Server 2007

- Set up, configure, and administer SharePoint Server 2007
- Learn how to use Microsoft Enterprise Content Management
- Leverage external system data and use advanced performance metrics and KPIs with the Business Data Catalog

David Sterling

Mc Graw Hill Osborne

Microsoft® Office SharePoint® Server 2007: The Complete Reference

About the Author

David Sterling has literally grown up with information technology, having started in the field in 1976 and establishing what is now Sterling International Consulting Group in 1984. As an executive consultant, he specializes in technical architecture, global program management, and design and development of software, and consults in the legal field as a legal expert and SME in patents, technical discovery, and legal document management. Since early 2000, he has become an established expert in collaboration and information architecture systems, producing "Portals—the next application platform" for WatchIT.com (his second program with them). He has managed hundreds of portal system deployments worldwide and has served as a speaker and trainer for many well-known companies. Since 2003, Sterling has been heavily involved in Microsoft Technologies (Commerce Server, MCMS, SharePoint), often consulting with Microsoft Consulting Services and other Gold Certified Partners. In addition to this work, he is also the author of *The MCMS Primer* and a contributing author to the SharePoint 2003 Technologies Resource Kit. Sterling consults organizations on all aspects of SharePoint, from acquisition and architecture to hands-on deployment.

About the Technical Editor

Vincent Simone has been involved in the IT industry since 1989. He has been a long-time applications developer and a large-team development manager for the past six years. He spent many years specializing in the C++ environment and now focuses on management and hands-on development of enterprise applications in C# and .NET. He has a great deal of knowledge in systems architecture and enjoys SharePoint.

Microsoft® Office SharePoint® Server 2007: The Complete Reference

David Matthew Sterling



New York Chicago San Francisco
Lisbon London Madrid Mexico City
Milan New Delhi San Juan
Seoul Singapore Sydney Toronto

The McGraw-Hill Companies

Copyright © 2008 by The McGraw-Hill Companies. All rights reserved. Manufactured in the United States of America. Except as permitted under the United States Copyright Act of 1976, no part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of the publisher.

0-07-159666-6

The material in this eBook also appears in the print version of this title: 0-07-149328-X.

All trademarks are trademarks of their respective owners. Rather than put a trademark symbol after every occurrence of a trademarked name, we use names in an editorial fashion only, and to the benefit of the trademark owner, with no intention of infringement of the trademark. Where such designations appear in this book, they have been printed with initial caps.

McGraw-Hill eBooks are available at special quantity discounts to use as premiums and sales promotions, or for use in corporate training programs. For more information, please contact George Hoare, Special Sales, at george_hoare@mcgraw-hill.com or (212) 904-4069.

TERMS OF USE

This is a copyrighted work and The McGraw-Hill Companies, Inc. ("McGraw-Hill") and its licensors reserve all rights in and to the work. Use of this work is subject to these terms. Except as permitted under the Copyright Act of 1976 and the right to store and retrieve one copy of the work, you may not decompile, disassemble, reverse engineer, reproduce, modify, create derivative works based upon, transmit, distribute, disseminate, sell, publish or sublicense the work or any part of it without McGraw-Hill's prior consent. You may use the work for your own noncommercial and personal use; any other use of the work is strictly prohibited. Your right to use the work may be terminated if you fail to comply with these terms.

THE WORK IS PROVIDED "AS IS." McGRAW-HILL AND ITS LICENSORS MAKE NO GUARANTEES OR WARRANTIES AS TO THE ACCURACY, ADEQUACY OR COMPLETENESS OF OR RESULTS TO BE OBTAINED FROM USING THE WORK, INCLUDING ANY INFORMATION THAT CAN BE ACCESSED THROUGH THE WORK VIA HYPERLINK OR OTHERWISE, AND EXPRESSLY DISCLAIM ANY WARRANTY, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. McGraw-Hill and its licensors do not warrant or guarantee that the functions contained in the work will meet your requirements or that its operation will be uninterrupted or error free. Neither McGraw-Hill nor its licensors shall be liable to you or anyone else for any inaccuracy, error or omission, regardless of cause, in the work or for any damages resulting therefrom. McGraw-Hill has no responsibility for the content of any information accessed through the work. Under no circumstances shall McGraw-Hill and/or its licensors be liable for any indirect, incidental, special, punitive, consequential or similar damages that result from the use of or inability to use the work, even if any of them has been advised of the possibility of such damages. This limitation of liability shall apply to any claim or cause whatsoever whether such claim or cause arises in contract, tort or otherwise.

DOI: 10.1036/007149328X

Professional



Want to learn more?

We hope you enjoy this McGraw-Hill eBook! If you'd like more information about this book, its author, or related books and websites, please click here.

To my wife Ludmila
for her incredible love and support
that made this book possible
and, of course,
to Amstel and Smokey,
who always stopped
by to say hello.

This page intentionally left blank

Contents at a Glance

1	MOSS: The Complete Reference	1
2	SharePoint Architecture	11
3	The Business Side of SharePoint	27
4	Installation Planning	39
5	Installing SharePoint	53
6	Post-Install Configuration	107
7	Exploring SharePoint Central Administration	197
8	SharePoint Site Administration	285
9	Exploring the SharePoint User Interface	335
10	Working with MOSS Sites, Office Workspaces, and Site Pages	355
11	SharePoint Lists and Libraries	375
12	Web Parts in SharePoint	405
13	The MOSS My Site	443
14	Site Columns, Content Types, and Content Management	453
15	SharePoint Site Definitions	517
16	SharePoint Workflow Tools	563
17	Features and Solutions	597
18	SharePoint Object Model and Web Services	611
19	Building Event Receivers	645
20	Building Web Parts	663
A	The STSAdm Command Tool	715
B	Setting Up Application Authentication	741
	Index	745

This page intentionally left blank

For more information about this title, click here

Contents

Acknowledgments xxiii
Introduction xxv

1 MOSS: The Complete Reference 1
Origins of SharePoint 1
The Birth of SharePoint Technologies 2
SharePoint Today 3
Anatomy of SharePoint 5
Summary 9

2 SharePoint Architecture 11
Breakdown of the Architecture 12
SharePoint Core 13
SharePoint Server Roles 15
SharePoint Shared Services 16
SharePoint Sites and Pages 16
Features and Solutions 18
Personalization 19
Services 21
Summary 26

3 The Business Side of SharePoint 27
SharePoint Governance 28
Site Governance 29
Site Policies 29
Site Templates 29
Content Governance and Education of Users 29
Methodology Overview 30
Breakdown of the Steps 30
Summary 37

4 Installation Planning 39
Planning Your Install 39
Planning the Network and Validating Connectivity 40
Choosing the Authentication Method 42
Domain Account and Active Directory Creation Modes 43

SharePoint Server Configurations 44
Single Server 45
Small Server Farm 45
Medium Server Farm 46
Large Server Farm 47
SharePoint Backup and Disaster Recovery Planning 48
Server Farm Summary 49
Hardware Requirements 49
Using Virtual Machines 50
Software Requirements 50
Web and Application Servers 50
SQL Server 51
Development Systems 51
Summary 52

5 Installing SharePoint 53
Installation Issues to Avoid 53
Set Up Windows Network and Windows Load Balancing 55
Testing the Network 55
Setting Up Load Balancing 55
Testing Load Balancing 57
Stopping Load Balancing 58
Step-by-Step Installation Checklists 58
Active Directory Accounts (Server Farm) Installation Checklist 59
SQL Server (Server Farm) Installation Checklist 59
Web/Application Servers (Server Farm) Installation Checklist 60
Complete Installation (Single Server) Installation Checklist 60
Step 1: Defining SharePoint Domain Accounts 61
Creating Accounts and SharePoint Administrators Group 62
Step 2: Installing the Operating System 68
Install the OS (All Server Types) 68
Post Install (Servers Joining an Existing Domain) 69
Step 3: Installing Internet Information Services 72
Install IIS Using the Wizard 72
Install IIS Using Manual Configuration 73
Upgrading from an Existing Windows 2000 Server 73
Step 4: Install .NET Frameworks 74
Set Up IIS to Use the Correct Framework 74
Step 5: Installing AD, DNS, and DHCP (Development Only) 75
Adding Domain Accounts 76
Step 6: Install E-mail Services (Development Only) 77
Step 7: Run Windows Update 78

Step 8: Install SQL Server (SQL Server in Farm or Development) 79
Installing SQL Server 2005 79
Account Setup in SQL 80
Apply SQL Server Service Pack 1 81
Apply SQL Server Service Pack 2 82
Step 9: Install Visual Studio (Development or QA) 83
Software Installation 83
Post Install 84
Adding Workflow Extensions 85
Apply the Visual Studio Service Packs 1 & 2 85
Prepare for SharePoint 85
Step 10: Install Microsoft Office (Development or QA) 86
Step 11: Installing SharePoint 3.0 86
Basic Install Overview (Single Server) 87
Advanced Install Overview (Server Farm Members) 96
Step 12: Install SharePoint Extensions for Visual Studio 105
Summary 106

6 Post-Install Configuration 107
Administrative Tasks 107
Where to Start 108
Adding the Application Pool Accounts to SQL 110
Checking for Install Errors 110
SQL Settings and Correcting Common Errors 110
Where to Now? 112
Overview of the SharePoint Administration Command-Line Tool 112
Overview of SharePoint 3.0 Central Administration 113
Post Install Administrative Tasks 114
Step-by-Step Configuration 116
Adding Servers to the Farm 117
Assigning Services to Servers 118
Setting Incoming E-mail Settings 126
Setting Outgoing E-mail Settings 127
Creating SharePoint Sites and Setting Up Shared Services 128
Configure Workflow Settings 149
Configure Session Throttles for InfoPath Forms Service (MOSS Only) 151
Add/Change Excel Services Trusted Locations (MOSS Only) 153
Service Level Settings for Shared Services (MOSS Only) 162
Create Alternate Access Mapping 179
Check Services Enabled in this Farm 181
Diagnostic Logging Settings (Optional) 182

Enable SSO in the Farm (Optional) 183
Add Antivirus Protection (Optional) 186
Post-Configuration Steps 188
Update Application Settings 188
Enable Enterprise Features (MOSS Only) 189
Check the System Event Logs 190
Joining an Existing Server Farm 190
Removing a Server from the Farm 192
Starting Windows Load Balancing Services 193
Repairing or Uninstalling SharePoint 193
Repairing 194
Uninstalling 194
Summary 195

7 Exploring SharePoint Central Administration 197
Using Central Administration 198
Central Administration Home 199
Central Administration Operations 200
Backup and Restore 231
Perform a Backup 231
Backup and Restore History 234
Backup and Restore Job Status 236
Data Configuration 237
Default Database Server 237
Data Retrieval Service 238
Content Deployment (MOSS Only) 238
Content Deployment Paths and Jobs 239
Content Deployment Settings 240
Check Deployment of Specific Content 241
Application Management 241
SharePoint Web Application Management 242
SharePoint Site Management 254
Create Site Collection 254
Delete Site Collection 255
Site Use Confirmation and Deletion 256
Quota Templates 257
Site Collection Quotas and Locks 258
Site Collection Administrators 258
Site Collection List 259
Application Security 260
Security for Web Part Pages 260
Self-Service Site Management 260

User Permissions for Web Application 261
Policy for Web Application 262
Authentication Providers 263
External Service Connections 264
Records Center 264
HTML Viewer 265
Document Conversions 266
Office Search (MOSS Only) 268
Manage Search Service 268
InfoPath Forms Services (MOSS Only) 270
Manage Form Templates 270
Configure InfoPath Forms Services 271
Upload Form Template 272
Manage Data Connection Files 274
Manage the Web Service Proxy 274
Office SharePoint Server Shared Services (MOSS Only) 275
Create or Configure This Farm's Shared Services 275
Grant or Configure Shared Services Between Farms 275
Check Services Enabled in this Farm 276
Configure Session State 276
Workflow Management (MOSS Only) 277
Workflow Settings 277
Shared Service Provider Administration 278
Summary 284

8 SharePoint Site Administration 285
Site Collection Site Administration Group 285
Users and Permissions 287
View Existing Users and Groups 291
Add/Modify User/Security Group to Site or Site Group 292
Delete User/Security Group to Site or Group 293
Add/Modify SharePoint Site Group 294
Enable/Disable Site Permissions 295
Manage Permissions Level Groups 296
Site Collection Administrators 297
Advanced Permissions 297
Look and Feel Group 297
Master Page (MOSS) 298
Title, Description, and Icon 298
Navigation (MOSS) 299
Page Layouts and Site Templates (MOSS) 300
Welcome Page (MOSS) 301

Tree View ... 302
Site Theme ... 302
Top Link Bar (WSS) ... 303
Quick Launch (WSS) ... 304
Save Site as Template (WSS) ... 304
Reset to Site Definition ... 305
Searchable Columns (MOSS) ... 306
Galleries Group ... 306
Master Pages (WSS) ... 306
Site Content Types ... 307
Site Columns ... 308
Site Templates ... 308
List Templates ... 309
Web Parts ... 310
Workflows ... 311
Master Pages and Page Layouts (MOSS) ... 312
Site Administration Group ... 312
Regional Settings ... 312
Site Libraries and Lists ... 313
Site Usage Reports ... 314
User Alerts ... 315
RSS ... 315
Search Visibility ... 316
Sites and Workspaces ... 316
Site Features ... 318
Delete This Site ... 318
Related Links Scope Settings (MOSS) ... 319
Site Output Cache (MOSS) ... 319
Content and Structure (MOSS) ... 320
Content and Structure Logs (MOSS) ... 321
Site Collection Administration Group ... 322
Search Settings (MOSS) ... 322
Search Scopes (MOSS) ... 322
Search Keywords (MOSS) ... 325
Recycle Bin ... 325
Site Directory Settings (MOSS) ... 326
Site Collection Usage Reports (MOSS) ... 326
Site Collection Features ... 326
Site Hierarchy ... 327
Portal Site Connection ... 328
Site Connection Audit Settings (MOSS) ... 328

Audit Log Reports (MOSS) 329
Site Collection Policies (MOSS) 329
Site Collection Output Cache (MOSS) 330
Site Collection Cache Profiles (MOSS) 330
Site Collection Object Cache (MOSS) 332
Variations (MOSS) 332
Variation Labels (MOSS) 333
Variation Logs (MOSS) 334
Translatable Columns (MOSS) 334

9 Exploring the SharePoint User Interface 335
SharePoint for Users 335
Where to Start 338
Site Components 338
Site Templates 340
Exploring Sites 341
Creating Sites 344
Using the Recycle Bin 352
Summary 354

10 Working with MOSS Sites, Office Workspaces, and Site Pages 355
Site Topology in MOSS 355
Creating MOSS Sites (Portal Level) 357
Setting Site Creation Permissions 358
Using the MOSS Site Content and Structure Page 358
Creating MOSS Sub Sites (Site Directory) 360
Creating Workspace Sites from Microsoft Office 361
Creating Pages 363
Accessing Pages 365
Creating a Basic Page 365
Creating a Web Part Page 367
Creating a Publishing Page (MOSS) 369
Publishing Pages, Editing, and Workflow (MOSS) 372
Summary 374

11 SharePoint Lists and Libraries 375
Anatomy of a SharePoint List 375
The Administrative Side of Lists 378
List Settings 379
List Columns 382
Create Columns 383
Edit Columns 385

Add from Existing Site Columns 385
Column Ordering 386
Indexed Columns 386
Content Types 386
List Views 387
Create/Edit View 387
Using a SharePoint List 391
Creating a List View Web Part and Setting the View 394
Using Libraries 398
Migrating Documents to SharePoint 401
SharePoint List Types 401
Summary 404

12 Web Parts in SharePoint 405
SharePoint Web Parts 405
Common Web Part Properties 406
Accessing Web Part Properties 408
Exercise: Connecting Web Parts 408
Exercise: Creating the Custom Lists 408
Exercise: Creating the List View Web Parts 410
Exercise: Connecting the Web Parts 411
Core Parts 412
Content Editor Web Part 414
Form Web Part 416
Image Web Part 417
Page Viewer Web Part 419
Relevant Documents 420
Site Users 421
User Tasks 422
XML Web Part 423
MOSS Web Parts 424
Site Aggregator Web Part 424
Content Query Web Part 427
I Need To... Web Part 429
RSS Viewer Web Part 431
Summary Link Web Part 433
Table of Contents Web Part 434
This Week in Pictures Web Part 435
Filter Web Parts 437
Search Web Parts 438
Categories Web Part 439

Sites in Category Web Part 440
Top Sites Web Part 440
Contact Details Web Part 441
Summary 442

13 The MOSS My Site 443
My Home Personal View 444
My Profile Public View 445
My Site Web Parts 446
Creating the User Blog (My Blog) 446
My Calendar Web Part (OWA) 446
My Contacts Web Part (OWA) 448
My Inbox Web Part (OWA) 448
My Mail Folder Web Part (OWA) 448
My Tasks Web Part (OWA) 448
Colleague Tracker 448
Colleagues 449
Get Started with My Site 449
In Common Between Us 449
Memberships 449
My Links 449
My Pictures 449
My SharePoint Sites 449
My Workspaces 449
Recent Blog Posts 450
SharePoint Documents 450
Creating Sub Sites in the My Site 450
User Personal Settings, Login and Out, and My Links 450
Using SharePoint Alerts 451
Summary 452

14 Site Columns, Content Types, and Content Management 453
Site Columns 453
Creating a Site Column 454
Changing a Site Column 455
Using a Site Column 455
Site Content Types 457
Exercise: Creating a Content Type 457
Dublin Core Columns 468
Site Content Types and InfoPath 468
Creating InfoPath Site Content Types 468
Updating InfoPath Site Content Types 472

Enterprise Content Management in SharePoint 474
ECM Components in SharePoint 475
Master Pages and Branding 476
Building a Master Page 477
Page Layout and Publishing Pages 483
Setting Up a Page Layout Page 485
Choosing a Content Type 486
Building a Page Layout Page 486
Web Content Authoring 489
Extending the Page Editing Toolbar 491
Smart Client Authoring 492
Creating Variations 497
SharePoint Content Deployment 501
Creating Custom Field Controls for Publishing 506
Page Caching and Controls 507
Developing Pages and Vary By Handlers 515
Summary 516

15 SharePoint Site Definitions 517
Windows SharePoint Services Sites 520
Content Storage 521
WSS Site Template Types 522
Microsoft Office SharePoint Server Sites 529
Portal Site Definitions 530
Enterprise Site Definitions 531
Records Center and Information Management Policies 536
Creating Sites Using Microsoft Office 545
Creating a New Site Definition with the User Interface 548
Using a Site Template Defined in the SharePoint UI 550
Creating a New Site Definition in SharePoint Designer 550
Creating a Site Template Using Designer 551
Creating a New Site Definition in Visual Studio 551
Deploying a Site Definition from Visual Studio 552
Customizing and Branding the SharePoint User Interface 553
Out-of-the-Box Modifications 553
Custom Modifications 553
Creating and Customizing SharePoint Master Pages 554
Creating and Customizing SharePoint Page Layouts 557
Customizing the My Site Template 561
Using a Different Default Master for the My Site 561
Removing the Organization Part on the My Site Person Page 562
Summary 562

16 SharePoint Workflow Tools 563
Who Does What? 564
Background on Windows Workflow Foundation 564
How SharePoint Implements Workflow Foundation 565
SharePoint Workflow Basics 566
Workflow Limitations 568
Configuring SharePoint Workflow 569
Using User Based Workflow 569
Starting a Workflow Manually 575
Workflow and Microsoft Office 576
Building Workflows Using SharePoint Designer 581
Preparing to Build the Workflow 584
Building the Workflow 584
Verifying the Workflow 591
Editing an Existing Workflow 594
Starting Workflow from Microsoft Office 594
Summary 596

17 Features and Solutions 597
Anatomy of a Feature 597
Solutions 598
Feature Manifest File 598
Elements Manifest File 598
Solution Manifest File 600
WSP File 601
Building a Feature 602
Assembling a Feature 602
Deploying the Feature 607
Problems Deploying Features 609
Building Complex Features 609
Summary 610

18 SharePoint Object Model and Web Services 611
Developing Against SharePoint 611
Referencing the SharePoint Object Model 612
SharePoint Security and Impersonation 613
Using Impersonation 613
Retrieving Impersonation Account from Registry 615
Retrieving Impersonation Account from Web.Config 618
Writing to the System Event Log 619
The SharePoint Site and Web Architecture 620
Connecting to a SharePoint Site 621

Connecting to a List 622
Using List Columns 623
Searching for a List Item 624
Creating a New List and Adding Items 625
Extracting a Document from a Library 627
Using SharePoint Web Services 628
Referencing Web Services 630
Example: Using the SharePoint Search Web Service 631
Microsoft Office Search 632
What Is Office Search? 632
How Search Works 635
Tweaking Search 636
Building Custom Search Tools 638
Summary 644

19 Building Event Receivers 645
What Receivers Are For 645
Anatomy of Event Receivers 646
Event Properties 649
Creating an Event Receiver 650
Building the Project 650
Deploying the Event Receiver 657
Pre-setup for Testing the Event Receiver 658
Testing and Debugging the Event Receiver 659
Deleting the Event Receiver 661
Summary 662

20 Building Web Parts 663
Anatomy of a Web Part 663
How Web Parts Work 665
Building Web Part Solutions 667
Methods for Building Web Parts 667
Building a One-off Web Part 668
Building a Web Part Solution 676
Fixing Common Loading Problems 706
Debugging 707
Removing a Web Part Solution 707
Building Web Parts in Visual Studio 2005 708
Using the Web Part Solution to Create a Farm Deployment Package 712
Removing a Web Part Solution Deployed by Visual Studio 712
Using MSBuild to Build WSP Files Automatically 712
Summary 714

A The STSAdm Command Tool 715
Schedule Parameter 736
Extending the STSAdm Tool 737

B Setting Up Application Authentication 741
Scope of Authentication 742
Selecting the Authentication Method 742
Choosing the Authentication Method on Create or Extend 744
Setting Non-Windows-Based Authentication Methods 744

Index 745

This page intentionally left blank

Acknowledgments

Many thanks to Carly Stapleton and Jane Brownlow for their support (and patience), Vincent Simone who took great strides to make sure I was accurate, and Bruce Edwards for helping my diction.

This page intentionally left blank

Introduction

Although this book is focused specifically on all things SharePoint, it is also intended to provide a guide to using SharePoint as an application framework and really utilize its full potential. Having been in the business for a few years, I've seen a lot of changes in the world of applications and development. I've seen methodologies come and go, whole forests of trees wiped out by a new buzzword, and even languages thought to be immortal no longer mentioned.

Over the years, much of the progress in development and application platforms has not been as major as it seemed at first. For example, we all thought Java would shake the world. You might say it did, but that's the point, it did; it's over, and we've moved on. I've seen this over and over, and it seems that most of the time, we still end up doing the same sort of things but using reworked tools and languages. In fact, many times these innovations simply add a new layer of complexity canceling any possible benefit. As developers, we've needed new tools for quite a while now and maybe Microsoft's been listening.

The past five years or so have seen somewhat of a quiet revolution in the development of software applications. "Portal technologies will be the new application platform," was a statement I made in my WatchIT.com presentation back in 2002. Even though I saw it coming, it is amazing where we are today. Not only have portals matured into full-fledged development platforms, but they've also brought the enterprise and the office together. Finally, IT isn't the only player in building applications.

As the proliferation of portals became more intense during the initial boom of the Web, designers building sites and applications saw that reuse and time to market were becoming critical aspects. In addition, designers and web-hosting companies saw that they were spawning a new industry of prebuilt websites in which "mom and pop shops" could be set up quickly—the most successful of these being eBay and the derivative blog spaces such as MySpace. As the business model of such enterprises matured, so did the tools used to create the sites and applications; new products emerged that enabled users to develop and deploy sites using prebuilt tools such as those eBay uses today. In fact, eBay is a perfect example. Although a portal itself, eBay subscribers can set up their own "portals" using eBay's tools.

At the same time portals were catching on in the web space, the operational side of the business was equally busy trying to find ways to pull together enterprise information from backend systems such as CRM, ERP, and Business Intelligence Systems. This began the start of the Application Server concept first implemented by Oracle, BEA's WebLogic, and IBM's WebSphere—the concept being that a single "application server" acts as a "go between" for transactions and communications between various applications and databases. Because the server handles inter-application communications, operational

and transactional data can be captured and stored for analysis, reporting, and so on. These application servers all provide some form of user interface to access the information and reports that provide either a web-based delivery or third-party tools used for reporting, such as Business Objects and Cognos.

In the background, the concept of web-based applications and enterprise information systems began to merge into a more generic platform for all types of portals. Early out were a few technologies like Plumtree (now Aqualogic) and Hummingbird, and in a very short time, there were dozens on the market. Many of the portal-related technologies were innovative at the time and formed the basis of what we have today. For example, Plumtree introduced the concept of Gadgets (later renamed Portlets), which are functionally synonymous with what SharePoint and .NET call web parts today. For all of those available, however, most were too focused on the specific vendor (like Oracle's tools) or too generic (like Aqualogic, which didn't include key elements like content management).

While Microsoft made a few feeble attempts in portal technology initially (the 2001 release of SharePoint Server was a bust), it did approach the concept in a slightly different way. Whereas all of the original technologies focused on pulling enterprise data to the desktop, Microsoft focused on pushing out content from the desktop with such early tools as Site Server and later Microsoft Content Management Server. The major difference was a focus on enabling the users to manage sites and information and move that effort away from IT. Although a terrific concept, Microsoft still didn't offer much to the enterprise outside of e-commerce.

In 2003, Microsoft made a major advance with SharePoint Portal Server—the first portal technology to provide enterprise-wide tools out of the box such as search, common user management, and content targeting. In a major overhaul, Microsoft brought together their FrontPage technology and their Project Central Team Sites product and created a user-based interface that provided end-users with the ability to design their own sites and create their own content. They also threw in some advanced features such as database storage for documents and a slew of out-of-the-box list types (web parts) that required no coding or development. Keeping with their original idea of linking the enterprise to the desktop and the desktop to the desktop, Microsoft integrated it tightly with the Microsoft Office Suite enabling users to publish content from an Office application directly to a website. For all its benefits and mainly the low price, SharePoint Portal Server 2003 started a boom in popularity as organizations of all sizes began to use the technology.

Now with the release of Windows SharePoint Services Version 3, Microsoft has created a true application and development platform targeting the needs of business and the business of development. On the application side, WSS is a complete application-development platform in which to build websites, e-commerce sites, Wiki sites, collaboration portals, desktop applications, remote applications, and pretty much any kind of application you can think of. In fact, all of Microsoft's applications are now either integrated with or built on top of WSS. For developers, this is revolutionary since the tedious part of developing an application is now gone and time can be spent on functionality.

This, ladies and gentlemen, is what we've been waiting for.

Who This book Is For

Ideally, anyone who is interested in SharePoint technologies and the Microsoft Office SharePoint Server system in general should get some benefit from this book, but as a technical reference, it is primarily the designer and developer who will get the most out of it. However, to create a balanced approach across the technical spectrum, the content is intended to provide the following benefits:

- CTO, technical managers, and directors
  - Grasp the true power of Windows SharePoint Services and where it can benefit information management, enterprise content management, and collaboration
  - Understand the use of key elements, including the Business Data Catalog, Content Types, and Key Performance Indicators
- Technical leaders
  - Understand the underlying technology and foundation for MOSS development
  - Gain a better understanding of working with development aspects and knowing where to configure versus develop
- Developers
  - Understand how to develop using the SharePoint technology
  - Understand key aspects of SharePoint, including ECM, building web parts, creating workflow, and event receivers
- Professional users
  - Utilize the power of the SharePoint Designer tool to customize SharePoint and create custom workflows
  - Understand the conceptual use of SharePoint technologies as a prelude to learning development

How This book Is Organized

The basis of this book is to provide a technical reference for SharePoint's features and functionality, and it also focuses on what you do in the order it should be done. Although certainly not an end all, this book is a complete primer for getting started and a quick reference for down the road. To set up a logical order, I start out by defining what you are trying to do with SharePoint, and then I move on to the "technical" steps—planning the install, installing, configuring, administering (the user side), using SharePoint (the front-end User Interface), and then into extending SharePoint (the developer side). In my own learning, I tend to read cover to cover. Hopefully, you will find this easy to do.

Planning and Preparation Chapters 1–3 cover SharePoint architecture and some of the business aspects to take into account when considering using SharePoint. Whether you are a CTO, project manager, developer, or designer, these chapters provide an extensive overview of the technology.

Installation and Configuration Chapters 4–7 will cover the raw basics—hardware and software requirements, installing the software, and configuring it for the first time. The chapters are formatted in a checklist style to help with covering all of the steps and will make a good reference for later.

Administration and the User Interface Chapters 7–16 cover the true operational use of SharePoint from administration to using the out-of-the-box features, including enterprise features such as Content Types and Content Management.

Design and Development and Enterprise Services in SharePoint Chapters 17–20 focus on the real-world development aspects of SharePoint technologies, including using the Object Model, creating user-defined workflows, and building event receivers and web parts. The final chapter (available online) covers the enterprise tools—the Business Data Catalog and Excel Web Services.

Business Data Catalog in SharePoint The online Web Chapter 1 covers the use of the SharePoint Business Data Catalog, including step-by-step instructions on how to create data source definitions, define localization, and how to include business data in SharePoint Search. You can access this chapter by going to http://www.mhprofessional.com/product.php?isbn=007149328X.

Appendixes Appendix A covers the use of the SharePoint command-line administration tool, and Appendix B provides an overview of the multiple connection methods available.

NOTE *Web Chapter 1 on the Business Data Catalog as well as the source code examples for the chapters are available online (http://www.mhprofessional.com/product.php?isbn=007149328X).*

CHAPTER 1

MOSS: The Complete Reference

Welcome to *Microsoft Office SharePoint Server (MOSS): The Complete Reference*, a comprehensive introduction to SharePoint Technologies—the new development platform from Microsoft. Right from the start, let me be one of the first to welcome you to a brand-new kind of information management and enterprise content management, as well as a completely new approach to enterprise systems and application development. While SharePoint is not a new product, the 2007 release represents a completely revised platform (that's right, platform). As such, knowledge of prior versions of SharePoint is by no means necessary.

Since this book is on SharePoint Technologies specifically, I'm not going to waste any of your time explaining the history of computing (I've always hated that). I will, however, touch briefly on the history of SharePoint. As I've mentioned, while it's not necessary to have experience with past versions, knowing SharePoint's roots will help you understand what it does and it can be helpful to know where all the terminology comes from.

Origins of SharePoint

SharePoint was originally a little-known product called Team Services that came bundled with Microsoft Project Central (circa 1997). An add-on tool for Project, Team Services used "project sites" (template sites based on FrontPage), which were intended for individual projects. These sites included things like task lists, issues lists, and calendars, which could be "packaged up" into a single file and redeployed elsewhere.

In 2000, Microsoft saw the rising popularity in "portal technology" and decided to try using Team Services and FrontPage as a pseudo portal/web management system. Microsoft also added collaboration-type features leveraging Exchange technologies, specifically messaging, calendaring, and using web folders for document storage. The result was the dashboard-based SharePoint Portal Server (SPS) 2001. Team sites were still at the core of the software, but were rebranded as Windows SharePoint Services 1.0 (WSS 1.0). Not a bad idea, but alas too late—web folder technologies and dashboard concepts were already proving unpopular by then.

Around the same time that SPS 2001 was released, Microsoft had several partners creating content management systems to support Microsoft Commerce Server. Microsoft could see the need for content management systems for both e-commerce and publishing web sites, so they later acquired one of these systems, which later became Microsoft Content Management Server (MCMS) 2001. MCMS provided the capability for structuring web sites and creating template-based web pages that could reuse format, style, and functionality, while also providing dynamic content with the first version of Active Server Pages (ASP). While the original version of MCMS could co-exist with SPS, there was no direct connection between the two.

After seeing some of the failings in SPS 2001, Microsoft went back to work and began with a different approach. They knew that some features were working—for example, fast deployment of sites, "team-based" approaches to lists, and document storage were all hits. In contrast, dashboards requiring extensive configuration and web folders that simply emulated network file sharing (and the problems with it) were not.

The result of this regroup was Windows SharePoint Services Version 2 and SharePoint Portal Server 2003. This was quite a leap forward in most aspects, including the use of database document storage. The SharePoint team gave up on the "dashboard" approach and expanded the site concept to create a whole portal (a multipage site where the topology could be arranged at will). They integrated database-managed storage for documents for eliminating the use of web folders, created many out-of-the-box lists and tools for enriching the user experience, and, to top it off, succeeded in bringing the desktop and the enterprise together by integrating it directly into Microsoft Office. This made it the first portal to marry enterprise use with a primary source of content.

To get it noticed, Microsoft made SPS 2003 quite inexpensive and Windows SharePoint Sevices was available for free. The intranet license fee for SPS was a paltry $12k retail with Client Access Licenses (CALs) roughly $65–75 per seat; its nearest competitor started around $500k for the software with *yearly* user licenses at $165. SharePoint Portal Server 2003 was the fastest growing product in Microsoft's history (and it made the top of Gartner's Ratings in less than three years).

Great products, but by no means perfect, both WSS and SPS offered the same basic user features with SPS offering the "enterprise" level features of multiple site content search and the ability to create site hierarchies using common security. There were some issues though; 2003 was short in content management because there weren't any publishing tools built in. Leaning more towards the collaboration (sharing) side of things, SPS content was either live or not—with no in between—because there wasn't any way to stage it or provide any more than a one-level approval process. There was an attempt (called the Spark project) to bridge this gap in 2004, which enabled MCMS 2002 to publish content directly to SharePoint, but this use was clunky at best because the products were clearly aimed at different goals.

The Birth of SharePoint Technologies

With the huge growth and popularity of SharePoint 2003, Microsoft began to discern what people were really using and more important, what the products were missing. They found that more flexibility in Intranet, Extranet, and Internet use was needed, that true (enterprise) content management had to be included seamlessly, and that because information workflow was crucial, things like projects shouldn't be silos of information. For those in the know, these are key elements in the Microsoft Information Worker vision.

Microsoft also saw heavy growth in organizations not just using SharePoint, but building on top of it. Partners and vendors began building cut-and-paste applications that could utilize

the features within SharePoint and extend its capabilities (workflow tools alone spawned at least three companies). Web parts (one of the ways to add custom application functionality to SharePoint) began proliferating around the globe (I know, I've built/sold them myself). Microsoft realized that this was the real potential: SharePoint's concepts represented a true technology and not just a product. By presenting SharePoint as a "technology platform" base to build on instead of just a suite of applications, they would be able to leverage it within their other products internally. Coupled with the opportunity to create a new development platform for outside of the product, Microsoft found that the idea was simply too good to pass up.

WSS and MOSS 2007 are intended to mature SharePoint as an extensible development platform based on SharePoint's concepts and provide a real technology base (similar to the .NET Framework) for building, developing and deploying of web-based applications. And as far as I can tell, they've succeeded. SharePoint as a technical platform literally transforms not only what we can do in it but what we can do *with* it. As a platform, it provides us an extremely reduced time in development and tools that give us capabilities we've never even dreamed we'd have. Just imagine if, even a few years ago, your business group came to you with a request to incorporate document storage and versioning inside of an application. Even with a third party tool, you'd be up to your ears in code just getting the basics to work. Using SharePoint Technologies, the code needed to load and store a document (in the database no less) is a whopping five lines.

In addition to just platform enhancements, SharePoint as a product has had a number of new capabilities added to it. For one, it can now provide enterprise content management and publishing (this version replaces Microsoft Content Management Server). Another (and probably most significant) is the new Business Data Catalog which enables use of external data sources directly in SharePoint.

SharePoint Today

So let's talk about what SharePoint is today. Aside from a development platform integrated into the operating system, the SharePoint Product Line is composed of the following:

Windows SharePoint Services (WSS) WSS is the platform that provides the core SharePoint features and services including application management, site provisioning, and farm level service management. In technical terms, WSS is the Application Server managing all services on top of Internet Information Server (IIS). Features include team sites, lists, libraries, workflow, and Office integration. Version 3 is free for internal use (a connector is required for external use).

Microsoft Office SharePoint Server 2007 Standard Edition (MOSS SE) MOSS SE is an application built on top of WSS and it provides the user interface for the basic collaboration/publishing features, along with farm-wide level services collaboration, enterprise content management, workflow, shared services, My Site services, user profiles, and personalization and enterprise wide search.

Microsoft Office SharePoint Server 2007 Enterprise Edition (MOSS EE) MOSS EE adds additional enterprise features including the Business Data Catalog, Excel Services, the Report Center, Infopath Forms Services, Key Performance Indicators (KPIs) and Filter Web Parts (the difference between SE and EE is licensing only). Note that both SE and EE install from the same media; enterprise features are enabled when the license is entered.

Microsoft Office SharePoint Server 2007 for Search SE/EE MOSS for Search provides only the SharePoint search capabilities that can be used as a stand-alone search engine and/or be incorporated into other applications. Functionally, there is no difference between the Standard and Enterprise Editions, though the SE edition is limited to a maximum of 500,000 documents.

In addition to the products, SharePoint as a technology represents a major shift in Microsoft's product strategies, becoming the all-encompassing application server for the enterprise—from content management to EDI. From a technical perspective, this puts SharePoint into the same class as IBM's WebSphere and BEA's WebLogic. While every Microsoft product uses (or is based on) SharePoint, a quick highlight of the World of SharePoint is shown in Figure 1-1. It's not possible to list every product here, but I've included the most common, which should give you a rough idea of how the products layer on the platform.

Microsoft has benefited a lot by centralizing some core application functionality into SharePoint, so that its own products can leverage the technology, but there's much more

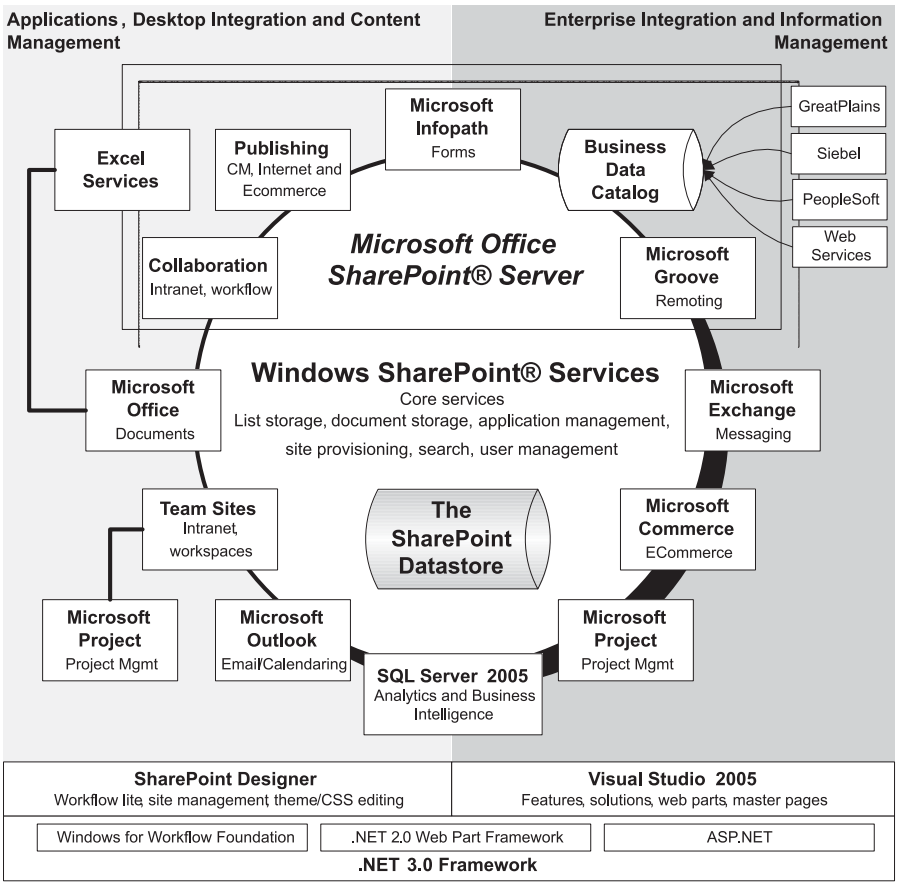


FIGURE 1-1 SharePoint and the Microsoft product mix

to it than that. More often than not, you'll hear the word "extensible" when Microsoft talks about SharePoint—this is a reference to the use of SharePoint as both a development platform and foundation that you can use to build any application.

Those of us that have worked with Microsoft over the years know that Microsoft's enterprise products (such as Commerce Server) have *always* been an 80/20 deal—meaning that they give you 80% and you go the rest (sometimes the most painful part). With SharePoint, Microsoft went the distance to give you 100% of a platform, with a great application in MOSS as icing on the cake. The flip-side for development is that it now takes you 80% less time to build an application and you can do a whole lot more! To be sure, Microsoft would be more than happy to hear that your custom program was built entirely on top of SharePoint!

Anatomy of SharePoint

While we'll get into much more detail on the architecture of SharePoint in the next chapter, the anatomy of SharePoint is worth reviewing at a high level. A few key concepts about SharePoint that are often misunderstood are how it dynamically stores sites and where documents go. In addition, the use (and true definition) of sites, webs, pages, and web parts has always been confusing. SharePoint Portal 2003 didn't help; it introduced areas that were sort of pages, sort of sites, but that didn't have the full complements of either. (Areas are no longer in this version.)

A lot has been cleared up in this release, but there are still a few confusing terms in the object model (developers should be forewarned; they've changed some functions but kept old names!). To begin with, let's look at the overall view of how Windows SharePoint® Services is implemented in Figure 1-2.

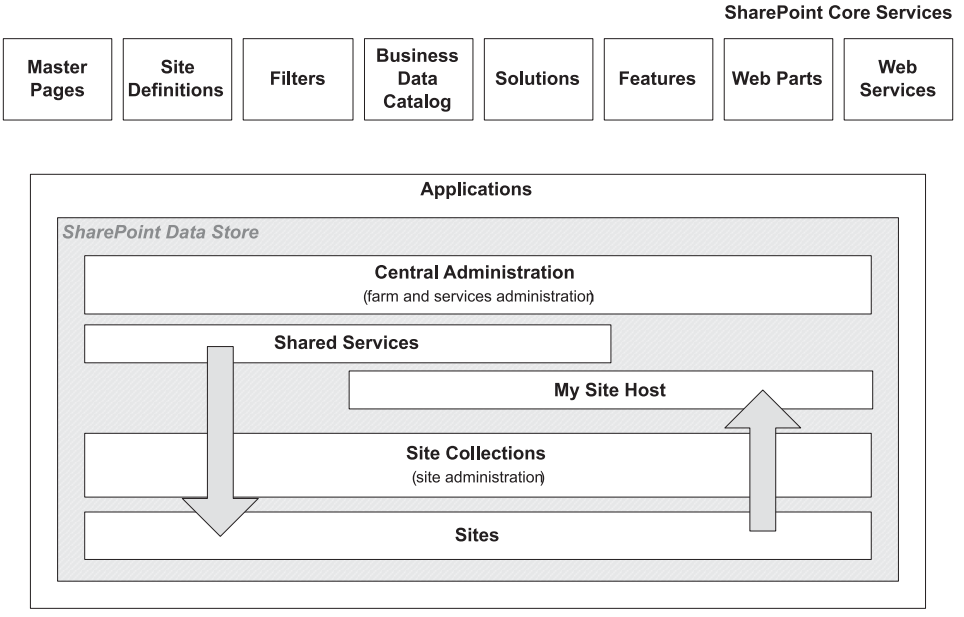


Figure 1-2 SharePoint internals

Windows SharePoint Services itself can be considered a product, though it embeds itself between (and is reliant) on several Windows services. For example, while I don't indicate it in Figure 1-2, Internet Information Server is the underlying technology in management and in provisioning of web sites and web services. What makes SharePoint unique is that nearly everything in and about it is stored in the SharePoint Data Store, and that the footprint of the product is really nothing more than some images, XML and .NET forms, and controls. If you were to examine what makes SharePoint tick, you would see that there are a lot of Windows stand-alone products it uses or provides access to, such as:

- **Internet Information Server** Site provisioning, virtual directory management
- **ASP.NET and the .NET Framework** SharePoint's object model based on it, forms, controls
- **InfoPath Forms Server** XML forms, form viewing
- **Windows Workflow Foundation** Workflow processing, notification and reporting
- **Excel Calculation Services** Remote calculation services, external data sources
- **SQL Server** Data storage and repository

Leveraging all of the previous products enables SharePoint to provide core services as a platform:

- **Database Storage** Provides data and document repository, metadata (information tagging), versioning, and backup
- **Security** Provides rights management (Rights/Roles/Rights trimming), pluggable authentication, per-item security
- **Management** Provides administration, delegation, provisioning, monitoring
- **Topology** Provides configuration management, farm services, feature policies, extranet
- **Application Program Interfaces (APIs)** Provides fields/forms, the object model and SOAP, events and deployment
- **Site Model** Provides rendering, templates, navigation, and a visual blueprint
- **Site Deployment Based on Site Templates** Provides automatic provisioning of sites combined with site templates that can support collaboration, publishing, blog, and wiki sites

Windows SharePoint Services is a dual-natured product; on the one hand, it is a storage application for lists, render views, libraries, versioning, security, and, in many ways, it's an application database tool. On the other, WSS is a complete web application interface that can render pages and web parts, manage security, provide web services, and so on. In fact, SharePoint can be used as a behind the scenes list and storage engine for integration with custom applications.

As for where everything goes in SharePoint, again the key is the Data Store. Everything from list data to documents to the site layouts themselves is stored there. The other components SharePoint uses are template files (ASP.NET pages); XML files that define lists and site templates along with compiled assemblies that represent the functional object model behind the user interface. Other than holding storage, nothing is stored on the server itself, as shown in Figure 1-3.

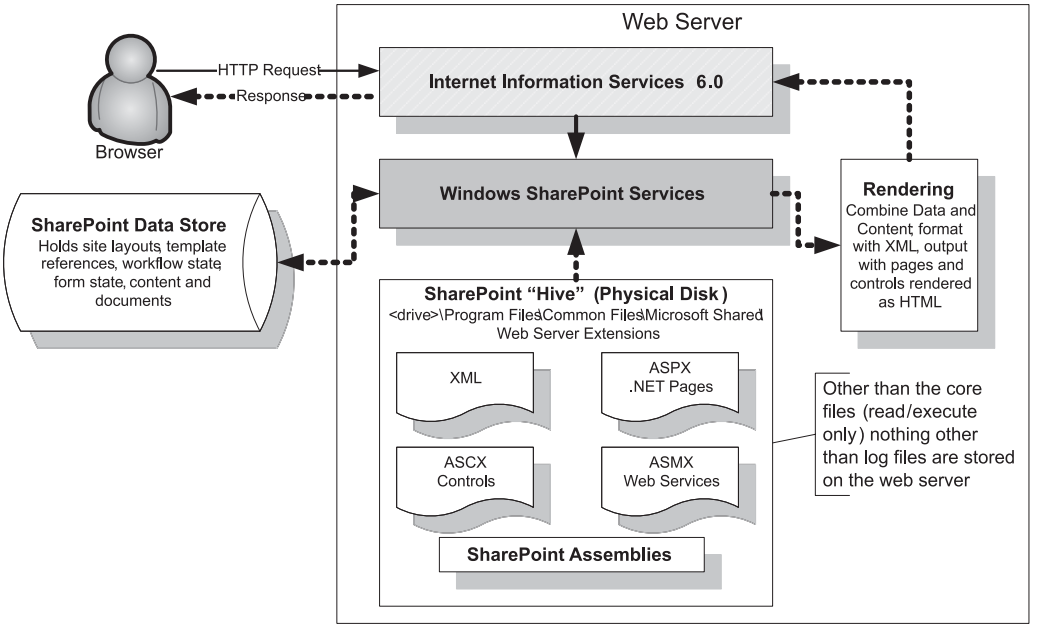


FIGURE 1-3 Windows SharePoint® services rendering

Aside from the technical platform and product integration, SharePoint provides the User Interface, which includes Central Administration (global management), Applications Management, Site Collections, Sites, and Shared Services. As you might imagine, applications are basically the same as an Internet Information Services (IIS) virtual directory. The real core of SharePoint is the Site collection (root site) which hosts sites providing common management and security. Sites themselves are definitions; they only exist subject to how they are defined in XML and combined with data in the database. There are three fundamental principals for WSS sites:

- **SharePoint Sites are completely made up of data and have no physical presence** These sites are rendered using a combination of XML, ASP.NET templates and data from the database (with the exception of pages in the data store, such as web part pages, and so on). WSS handles the rendering of definition using referenced assemblies and similar.
- **Components of SharePoint site definitions are**
  - Defined in XML (ONET.XML to be exact) and loaded into SharePoint
  - May include Master Pages and Page Layouts
  - Represented by a collection of features (capabilities), such as basic collaboration, document libraries, slide libraries, and so on
  - Featured by reference components (assemblies, event receivers, web parts, and so on) and these components may talk to features and functions of Shared Services
- **SharePoint Shared Services (Microsoft Office SharePoint® Server only)**
  - Provides a central place for Features that can be used farm-wide

- Can be managed through the Central Administration
- Provides management of enterprise features like the Business Data Catalog
- Allows management of services by delegated administration

Built completely on top of the Windows SharePoint® Services platform, Microsoft Office SharePoint Server (MOSS) brings a number of additional features and services to the mix:

- **Expanded Collaboration** Discussions, calendars, e-mail integration, user presence, project management light (Task lists w/Gantt features), and offline storage using the Groove Server.
- **Enterprise Content Management** Authoring, approval processing, web publishing, policy/auditing, rights management, retention, multilingual support, and content staging.
- **Personalization** My sites, content targeting, audiences, user profiles, and social sites.
- **Search** Indexing, relevance, Metadata, Alerts, and a customizable user experience.
- **Business Process Integration** Rich forms, web forms, business data, catalog, data in lists, line of business operations, Single Sign On, and BizTalk integration.
- **Business Intelligence** Server calculations, web, rendering, key performance indicators (KPIs), dashboards, and direct integration with SQL Reports.
- **Integration with Microsoft Project** Tasks, schedules, resources, budgets, deliverables, document storage, and reporting.

On the development side of things, SharePoint provides a rich platform for extending the product itself and for using it for core services in a custom application. This includes:

- **Master Pages** SharePoint pages are based on the .NET Framework master pages; in SharePoint, these define the look and feel of the rendered pages.
- **Custom Site Definitions** These provide the capability for building custom site templates that can include lists, libraries, web parts, and custom pages.
- **Custom List Definitions** These provide the capability for generating customized lists for custom metadata, custom field definitions, and so on.
- **Features** These are add on assemblies or definitions that extend SharePoint in some way; a custom data query or view or even a custom site definition.
- **Event Receivers** These provide the capability for responding and reacting to various events in SharePoint lists and libraries and can leverage both SharePoint and anything within the .NET Framework.
- **Web Parts** These provide the capability for creating micro-applications and custom functionality within the SharePoint User Interface. Similar to a feature, web parts can also leverage both SharePoint and anything within the .NET Framework.

On the Enterprise side:

- **Business Data Catalog/KPIs** Though used by the end user, the BDC represents a way to provide direct access to enterprise/external data for use within Features, Event Receivers, and Web Parts. Key Performance Indicator parts enable you to present prebuilt views of custom data.

All of the previous elements come together to provide the user experience. To the end user, SharePoint consists of

- **Portals** Portals provide a common navigation across multiple sites and include some customized site templates, including a Document Center, Site Directory, and the Search Center. These are either collaboration based (intranet) or publishing sites (extranet).
- **Site Templates** Pre-defined site layouts that include certain web parts and pages; these layouts can be selected by the user when creating a site (a Portal by the way is just another template type).
- **Sites** Individual sites are one page sites that can have any number of web parts added to them. The home page can be designed using web part views.
- **My Sites** These are personal sites that provide the standard SharePoint features as well as access to personal calendars, tasks, and so on.
- **Pages** Pages are aspx (ASP.NET) pages created by the end user that are stored as documents in a Pages library. Pages can be content-based or web part pages that include web part zones. (Note: Master Pages used to render the actual User Interface within SharePoint are not accessible to users.)
- **Web Parts** Out-of-the-box lists and libraries, several specialty parts (summaries, filters, and so on) and custom web parts installed as solutions within SharePoint.

Perhaps the most deceiving thing about SharePoint from the user perspective is that the user experience makes SharePoint feel like any other web site—this is because the interactive side of it (collaboration and workflow) is as integrated as changing fonts in Microsoft Word.

Summary

SharePoint has come a long way over the last few years, and while it might not be the end-all in information management, it sure comes close in its presentation. On the business side, it's no longer necessary to enlist IT to build web applications in dozens of products only to achieve the same goal. Using SharePoint, the Internet, Extranet, and Intranet can now all "play nice," using the same platform and all of it integrating with the Office desktop. For developers and IT, SharePoint provides a real edge in applications development and provides new and useful tools. In addition, the concentration in development shifts from building foundations to building functions, meaning more usage and less reinventing of the wheel. It also cuts down on application maintenance, because SharePoint is as close to "set it and forget it" as possible. Once up and running, it's the users that run the sites.

Of course, the biggest beneficiary is the user—no more struggling with inconsistencies in interfaces, shifting gears between applications, or inability to search for things consistently. Plus users no longer have to run to IT to do everything; SharePoint really does empower the end user to do what they want, from creating their own sites to managing their own content.

As you begin to work with SharePoint, I encourage you to look at SharePoint as more than an application, because it's truly a robust development platform that happens to have some excellent tools already built in. As you review the various features, try to keep notes on ways in which SharePoint's out-of-the-box tools can eliminate the need for development, and, at the same time, make note of the features that are now available to you when developing your next custom application.

This page intentionally left blank

CHAPTER 2

SharePoint Architecture

As you will come to find, SharePoint as both a platform and as a product is a pretty large subject. (I know from personal experience; I had difficulties keeping this book this size!) Regardless of what your particular use or role with SharePoint happens to be, it's worth knowing at least a little bit about how everything fits together. Some might prefer the "black box" view, but developers in particular need to know what they are touching when they are working with and developing on top of the SharePoint platform.

Even as a user, it is helpful to have a general idea of what SharePoint is made up of, what it can do, and its basic terminology. Learning how the components all fit together, particularly features like the Business Data Catalog (BDC), takes time. There's also a practical (and financial) need for this knowledge; knowing about these features might save you from buying something you don't need.

As you will soon learn, SharePoint technologies are now a very broad part of Microsoft's technical platforms and are integrated into many of their other products. The actual mapping of the SharePoint Platform is quite complex—naturally, it uses the many components that make up the Microsoft Development Platform such as the .NET Framework, C#, ADO.NET, and Microsoft Windows, as well as Server Applications such as Internet Information Server (IIS), Active Directory, SQL Server, and Exchange Server.

SharePoint also borrows some technology from other applications. For example, content page layouts and management come from Microsoft Content Management Server, which has been integrated to provide SharePoint publishing features; workflow tools are provided via the Windows Workflow Foundation (now part of the .NET 3.0 Framework); web parts (originally part of SharePoint) are now .NET web parts; and the Business Data Catalog is an advancement of tools from the Business Portal. They've even upgraded some products—SharePoint Designer has replaced the previous functionality provided by FrontPage and added tremendous development (and even deployment) capabilities on top of the platform.

SharePoint itself uses all of the features of the Windows Platform and Microsoft Server and Client Applications and also makes this available to you for development. That's right—you have total control over nearly every aspect of SharePoint and since it uses all the same tools you have, there's no reason to fear digging right into it. Virtually any part of SharePoint can be extended—from the pages it uses to render content to the actual user interface itself.

When designing the platform, Microsoft put a great deal of emphasis on making everything customizable for both users and developers alike. Users can control things like navigation, control menus, and even manage branding at all site levels. Developers can change anything using XML and Collaborative Application Markup Language (CAML),

including menus, navigation, site definitions, page layouts, and so on, and of course can also use Web Parts and the new Event Receivers (different from Event Handlers as Receivers can trap any list/library event). They even made the user interface itself completely customizable to the very lowest level; nearly any type of data display within SharePoint (List views, and similar) can be modified using XSLT.

I would say that those familiar with the previous versions of SharePoint might be relieved to know that it is pretty hard to break it. At the very basic level, the simple feature Recycle Bin reduces that "Oops, I just deleted my Library's phone call" problem. At the user level, site management is easier and role-based security prevents unauthorized access; better page management has also eliminated many of the "Page cannot be displayed" problems. Designers (using SharePoint Designer) and developers (using Visual Studio) now have tools like Publishing, Features, and Solutions to install Site Definitions, Workflows, Receivers, and Web Parts, which can be turned on and off in the user interface. They've even added the ability to *restore* a site to its original configuration and page layout.

Breakdown of the Architecture

As I mentioned previously, the actual composition of Windows SharePoint Services (WSS) is quite complex. This is even more the case with Microsoft Office SharePoint Server (MOSS), which adds Portal-like site types and enterprise features like Excel Web Services. A key concept here is that MOSS is simply an application built using the WSS Framework, much the same way a web application is built using the .NET Framework.

While it's a good idea to know the actual composition of the components, in practice most of this is abstracted away from the developers, designers, and most certainly the users. For example, SharePoint uses (actually hosts) the Windows Workflow Foundation Services from the .NET Framework. Through SharePoint, users have the ability to use out-of-the-box workflows and create them with custom forms, designers can create custom workflows in SharePoint Designer using a Wizard, and developers can create advanced workflows using InfoPath and the API (Object Model) in Visual Studio. In all cases, the underlying technology of WWF is abstracted. Users don't need to know anything about it and Designer's Wizard uses no code. Even developers don't need to know how WWF itself works, because the SharePoint Object Model (API) provides all of the tools necessary.

Still, it can be helpful to see a global view so you can see the complete model of the components found in WSS, so I've included this in Figure 2-1.

For reference, Figure 2-1 is based on a Microsoft diagram (author unknown) available during pre-release. While this certainly looks impressive, this really doesn't tell you much about how SharePoint itself works (I'll spare you the MOSS view). How the information moves in the background is not nearly as important as understanding how the SharePoint Platform provides components to create the application functionality. The component approach actually carries all the way up to the topology, with each component riding on top of another. For example, Application | Site | Page | Content | Web Part is a natural hierarchy of components. This will become much clearer to you as you begin to work with SharePoint and begin to look at the components as objects. Like the objects on your desk, they can be created in one place, but moved around wherever you want them.

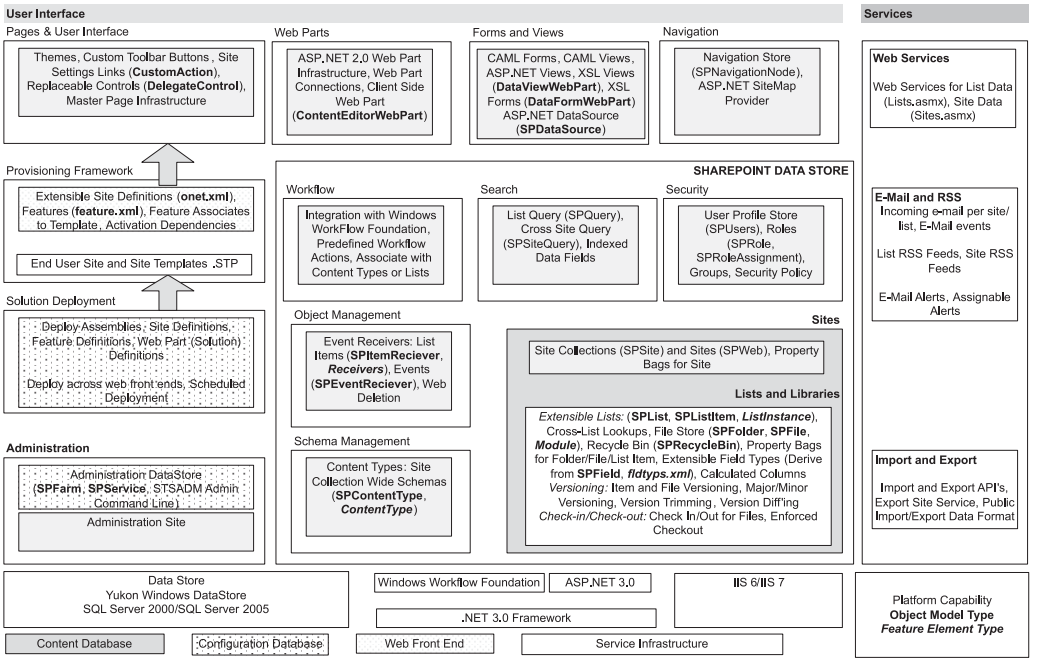


FIGURE 2-1 Windows SharePoint Services 3.0 Information Architecture

So to cover the architecture of SharePoint, I'm going to save both of us a lot of time and take a more practical-use approach, based on the components you will actually deal with, such as Internet Information Server and the services provided by (or through) SharePoint. I'll glaze over the underlying technologies or tools used if they're not significant (this should help if you aren't quite technical or up on Microsoft, because no specific knowledge of the technology is required).

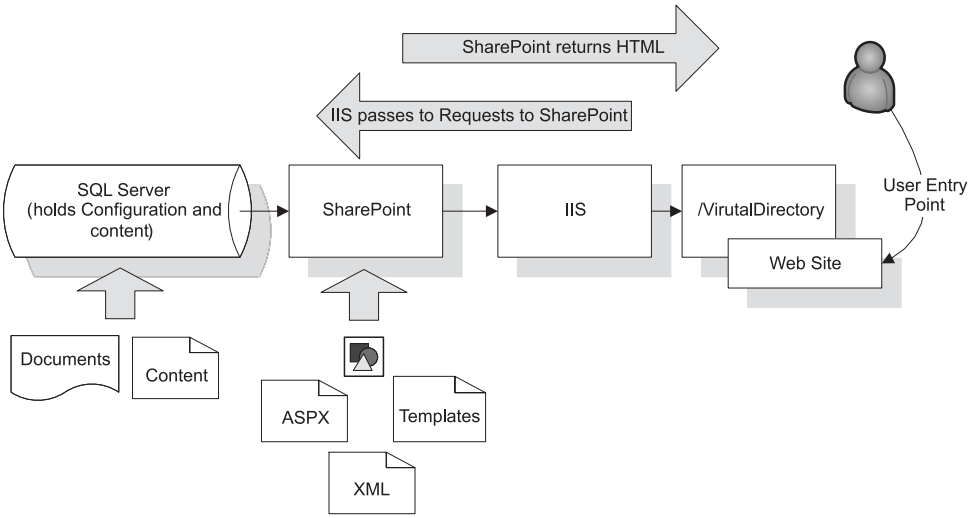
SharePoint Core

So what the heck is SharePoint anyway? Is it really a technology or simply an application? The answer is that it is both. It started out as an application that advanced into a method of developing applications. At the very core, SharePoint itself can be summed up into three Microsoft technologies, including the .NET Framework, SQL Server, and Internet Information Services:

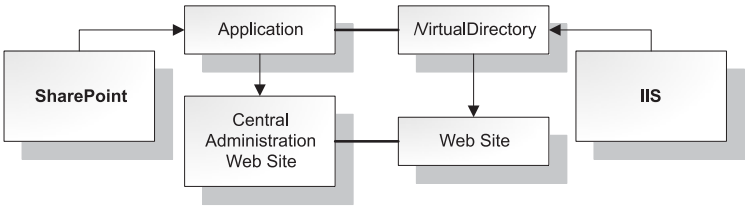
- **SQL Server** Everything in SharePoint is based on tables stored in the database; with the exception of some standard files and XML definition files that must be stored locally (and other assets like images), the entire composition of application, site, site collection, page, documents, and even web parts are defined by data stored in the database.

- **.NET** SharePoint is built on the Microsoft .NET Framework and leverages all of the features and tools built into it (including web parts starting with .NET 2.0). Using this framework, SharePoint provides a web-based application that includes an administrative management interface, an extensive Application Program Interface (referred to as the SharePoint Object Model), and an application end-user interface. It also leverages .NET in providing several web services that can be used to interface with SharePoint. It also hosts the .NET 3.0 Windows Workflow Foundation as an instance that is used to provide workflow functionality inside of SharePoint.
- **IIS** SharePoint has a built-in interface that leverages Internet Information Services (IIS) not as an ISAPI filter (something that simply captures requests), but as a co-process that provides SharePoint with application and site self-provisioning features, as the interface to the web presentation engine used to render web pages. This interface enables SharePoint to manage its own infrastructure from a web application point of view, handling the creating of virtual directories and web sites within IIS.

Conceptually, the core of SharePoint looks something like this:



As you can see, there's not much to it at the high level, because the real guts of the application reside within SharePoint itself, in the way it manages applications and sites starting with the Central Administration Site. The Central Administration site is created when SharePoint is first installed and provides the administrative interface needed to enable services, create applications, sites, and so on. It is here that you can see the one-to-one relationship between SharePoint and IIS:



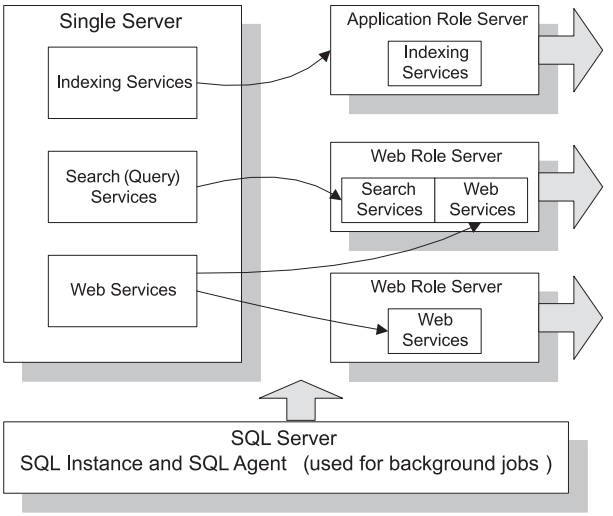
With the SharePoint interface to IIS, it is from this Central Administration site and not IIS where additional applications and sites are created. In fact all settings, including such basics as time zone and application pool accounts are set in Central Administration.

SharePoint Server Roles

In WSS and MOSS 3.0, multiple server capability is now available across the entire platform (WSS V2 did not support this), and with the increased number of features and services available has seen the introduction of a new concept for server roles. Instead of functionality being tied to a specific server, they are tied to Server roles instead. A single server system can host one or more roles, and those roles can be enabled or disabled at will. Server roles include:

- **Web Server Role** This handles the rendering of web pages and the response to users requests
- **Application Server Role** This handles background job processing and indexing
- **Search Server Role** This handles search query requests (and usually runs on web servers)
- **SQL Server Role** This handles the database operations (unlike the others, this role can only be assigned once and cannot be changed via the user interface)

Server roles provide a huge amount of flexibility when configuring SharePoint, since the configuration can be easily changed to adapt to actual usage. They also aid operational areas such as maintenance and disaster recovery, since services can be swapped in real time:



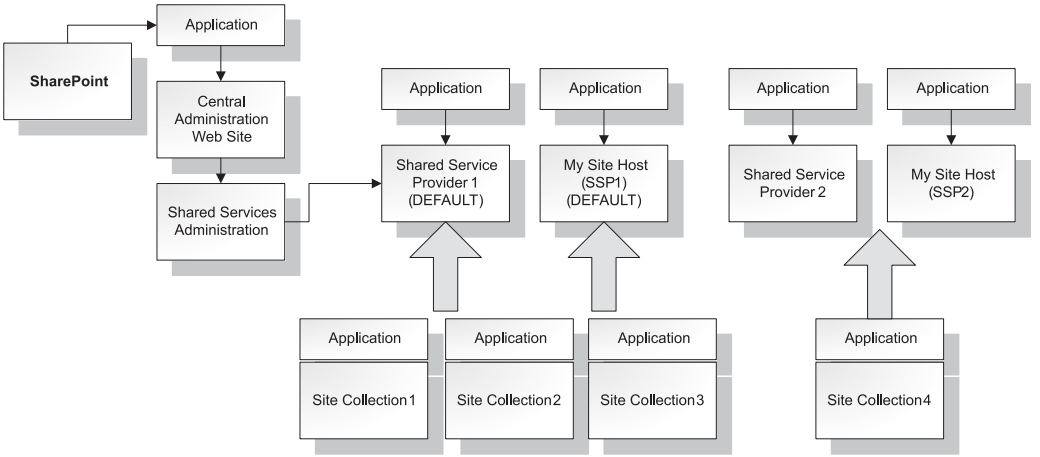
While all of the roles can be designated on a single server, they can be moved to individual servers with the exception that a Search Service must run with a Web Service and the SQL Server service cannot be changed to real time. Additional servers can be added in either the Application role or the Web Server Role at any time, meaning the server farm can be expanded as needed to accommodate use and performance.

SharePoint Shared Services

MOSS introduced a new support site like Central Administration, which is designated as the Shared Services Administration site. Shared Services Providers are designed to provide common services for one (or more) Site Collections including Search, InfoPath Form Services, and My Site Hosting features. Site Collections associate with a Shared Service Provider (SSP) to use the services it provides. As needed, Site Collections can even change associations from one SSP to another (thus changing the services available). As a key element of the SharePoint architecture, at least one Shared Service Provider must be created (before any sites) to support Search and it is required in the Enterprise Edition for the Business Data Catalog, Forms, and Excel Web Services.

The real purpose for SSPs is the simplification of administration and overall maintenance; SSPs can be used to enable or disable central functions, which spares doing this in individual site collections. An SSP also provides centralized search features across multiple sites. Designed to be extensible like the rest of SharePoint, the SSPs are also used for developing custom services; developers use the framework and then let SharePoint handle the deployment.

The My Site Host capabilities provide a number of features—for one, they can be used to distribute the My Sites within a large organization, in order to spread out storage and custom purposes, such as defining My Sites for specific languages. The conceptual view of Shared Services looks like this:



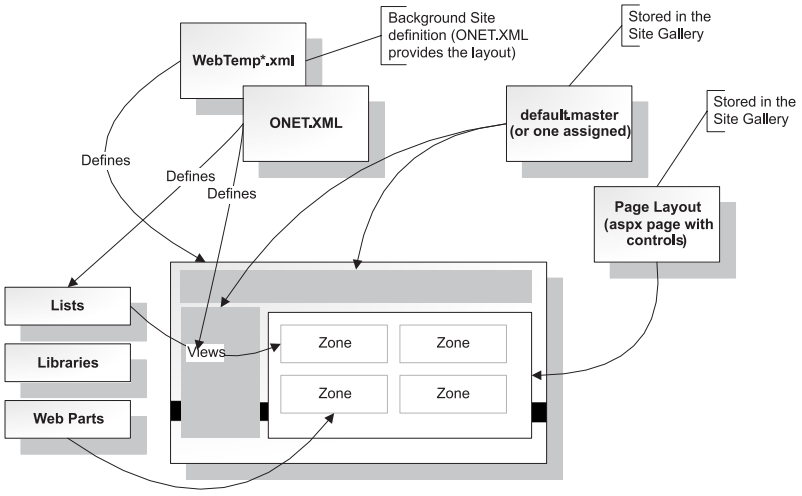
As you can see, Shared Services runs directly from the Central Administration site but it is hosted in its own application. Any number of Site Collections can be associated with any SSP, but one is always designated as the default (all new site collections are assigned to the default).

SharePoint Sites and Pages

Not much has changed in the composition of WSS sites and their layouts over time, but what has changed is the underlying technology used for site designs. Previously based on ASP.NET pages, WSS and MOSS are based on ASP.NET Master Pages introduced in .NET 2.0. Instead of physical page templates, Master Pages, and Page Layouts, along with their content, is stored in the database. In fact, virtually everything is stored in SharePoint, including information about how a site is displayed, the page layout and user security,

items shown on the page, the lists and libraries (and even documents) are all stored there. This means that the physical file footprint of SharePoint is pretty small. This footprint includes the assets that make up much of the SharePoint user interface (images, icons, JavaScript files and some Aspx files) and what SharePoint uses to create sites, lists, and libraries. This includes site definitions, list definitions, features, and so on, all derived from XML and CAML files. All of what is exposed can be used by designers and developers to extend and modify SharePoint while keeping the system and database integrity intact.

For Content Management, MOSS now supports Publishing Pages (replacing MCMS). These are predefined page layouts (used for both content entry and display) that users can use to create and publish pages to the site on the fly. Without any programming, a user can create a page, select the layout, add content (such as text or images), and publish it to the site. The assembly of a site is based on its Site Definition; this is a combination of a Template Definition and an XML file which defines the contents of the template itself (lists, libraries, and what have you). Pages that "appear" in the site are comprised of the Master Page defined for the site and a Page Layout that displays the content. Conceptually, this looks like:



Site Definition and Page Assembly Model

While the Site Definition is static, the Master Page and Page Layouts are completely unrestricted, so that the actual interface presented to the user can look like anything. Sites within a site collection can choose to use a common Master Page or they can be designated at the site level.

Aside from the actual composition, it is helpful to understand the differences between WSS and MOSS sites. While all sites in SharePoint are sites, only certain site templates can be used in MOSS, specifically the Enterprise Templates (including the Document Center site, the Search Center site, and so on) and the Collaboration sites, Collaboration and Publishing. Underneath the skin, there is no difference in the object model, because sites (also referred to as webs) are identical. The difference in the actual site definitions is that the MOSS templates are set up to use Enterprise level services such as Office Server Search. Site organization is different as well—WSS site collections are set up in a hierarchy of top to bottom, which cannot be moved and the content is unique to each site. Subsites can inherit the parent security or have their own membership and rights.

MOSS sites are enterprise level and so are set to be free form, and are not attached to any kind of a hierarchy. This is most suited for Portal site usage, typical in public facing Internet sites and collaboration intranets. Sites in MOSS (as well as pages) can easily be moved around to modify the hierarchy as well as change navigation. Search is already enabled Site Collection–wide, and sites can have their own security settings; however, in MOSS, they can use different methods of authentication.

Pages are a way to create page-like functionality (more in line with a typical web) in SharePoint. Pages have always been a part of SharePoint (though some folks didn't use them), and they include a basic Content Page intended for basic content (including text, pictures, and otherwise static content) and the Web Part Page intended to provide Web Part Zones (fixed layouts) which can be used to organize the display of List Views and Web Parts.

These pages are still created the same way; the user can choose the type (and if working with a web part page, the layout as well) from a set of templates already installed. When the page is saved, it is stored into a document library (in WSS, this is often the Shared Documents Library, whereas in MOSS, the default library created is simply called Documents). Unfortunately, there isn't an automatic way to link to these pages to make them available for users (unless using the one Multipage Meeting site template) other than creating a link list and either adding them or creating a list view of the Document Library. Of course, the links can be used anywhere, including inside of content.

MOSS supports those pages as well as the Publishing Pages used to support content management; these are similar to the pages mentioned previously, but use Master Pages and Page Layouts from the Site Gallery. The page layouts used in publishing provide more than just a zone layout or content space: they can include Publishing Controls, which allow users to create pages and add content directly (they also use Site Content Types, as you will learn in Chapter 14) and even publish content directly to SharePoint from Microsoft Office. In addition, Publishing Pages are stored in a custom library called Pages and can be included in the SharePoint Navigation.

NOTE *Depending on the source, Publishing Pages may use the Document Conversion Service as covered next.*

Features and Solutions

As I've mentioned throughout, a major purpose for developing Windows SharePoint Services as a platform is to make it extensible; that is, to make WSS and MOSS functionality extendable and customizable, but also to provide this same platform to anyone who wants to create custom applications on top of it.

In the past, adding anything to SharePoint took some effort and even creating web parts could be a very long and painful process. Fortunately, Microsoft had this same problem and decided that with WSS and MOSS it would be a great idea to develop some sort of a framework that designers and developers could use to add features to SharePoint. So maybe they did it for themselves or for us, but either way, the Feature and Solution Framework is built into WSS to directly support deploying functionality in SharePoint.

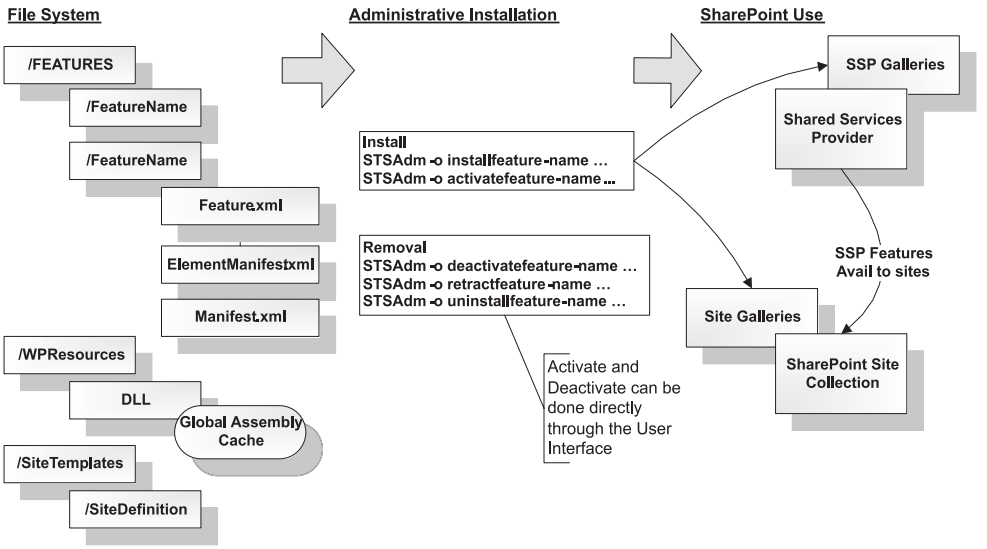
Features are either services or applications, made up of components and/or pages that can be deployed to a SharePoint instance. Using a few XML definition files, Features are installed in SharePoint and can then be activated on a Shared Service Provider, Site Collection, or even Site level. Examples include user-defined (from Designer) or custom workflows, custom site definitions, master pages and styles, and so on. Once installed, site administrators can enable or disable Features at any time, putting control in the hands of the user instead of the developer.

Solutions are just like features, with the exception that they are intended to provide solutions other than add-on functionality. While this might seem like splitting hairs, there is a difference: the intention is that Solutions provide a specific use and are not necessarily SSP or site wide. For example, all web parts are installed as Solutions—it would be common to install a web part on a specific site and but not as a Shared Service. And installation is a snap—Visual Studio 2005 now provides automatic installation of Solutions and debugging!

As I mentioned, Features and Solutions can include a lot of things, from DLLs to pages; the framework is intended to accomplish the following:

- Copy the Feature or Solution definition files into the SharePoint Features folder
- Copy the support files from a folder (aka package) and place them in the correct locations in the SharePoint Hive (folder structure)
- Install the Feature which loads the name and description as well as the definition into SharePoint (pointing it to the right assets defined in Elements Manifest File)
- Activate the feature which makes it available in SharePoint sites

As a repeatable process, this makes deployment to multiple servers much easier. The conceptual view of this looks like this:



Features and Solutions Model

As shown, once files are copied into the right places, the Feature (or Solution) definition handles the rest. Note that for solutions, it is simply a command-line change. Instead of `installfeature` the command is `installsolution`.

Personalization

Windows SharePoint Services pages provide two views for users, a Shared and Personal view. A feature that has been around for most of the versions, these views allow users to modify some settings and views on a page of their choosing. The personal layout is saved as the user's view of the page from then on—if needed, users can always revert back to the Shared view.

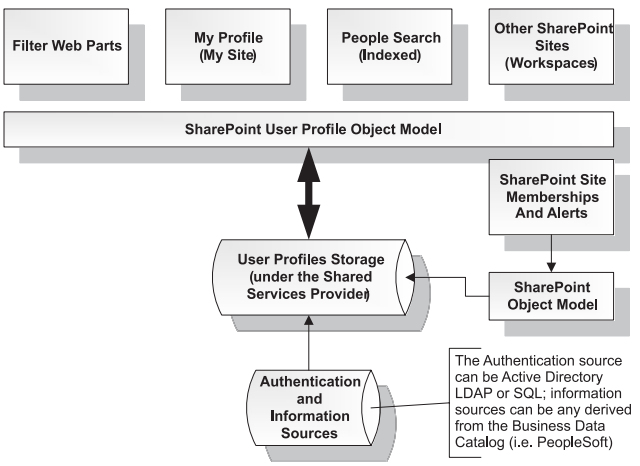
As an enterprise tool, Microsoft Office SharePoint Server provides a host of ways in which information within Site Collections can be personalized and otherwise targeted to specific users. MOSS, of course, supports the personal view but also supports a personalized experience in the content displayed to them, provides lists they can see, and highlights things like news. One method is the personal My Site; there users can centralize links, sites they belong to, and so on. Another is via web parts, such as the Filter Web Parts that can control the content that is shown based on the individual or the groups they belong to. Yet another method is Audiences, used to group users into a specific interest or commonality (set when adding the item, list, or site). When users visit a site, items targeting their Audience(s) will appear.

In a Collaboration site, much of this is enhanced by the My Site since it enables users a single entry point into a wide array of information. With personalized web parts pulling information from sites up to the My Site, the user has a single point of information and content they see is relevant to them or their role in the company. This also means that instead of users hunting for information, it is provided to them.

Other parts of SharePoint, such as web parts and similar, handle personalization based on User Profile information, including the Audiences and Memberships they belong to. These are derived from comparing some part of a user profile to a value that makes them unique (for example Department = Sales). Some of the User Profile information comes from the Authentication Source, usually Active Directory, but a large portion of the profile are SharePoint properties that can be maintained by individual users. With MOSS Enterprise, the Business Data Catalog can be used to import profile items from additional sources such as PeopleSoft or other HR systems. This is important because it helps use the correct system of record and reduces duplication of data. Since Profiles can be enhanced, they can become extensive resource tools. For example, by enhancing the User Profile to include a Resume and specific areas of expertise, users can then be searched for by the kind of work they do.

Be aware, however, that personalization is only skin deep for Search, News, Items, Documents, Tasks, and so on; users are always subject to the security groups they belong to. Audiences do not involve rights, meaning that even if a user is a member of a Target Audience on an item, that does not give them right to view it.

SharePoint's personalization features are composed of the following:

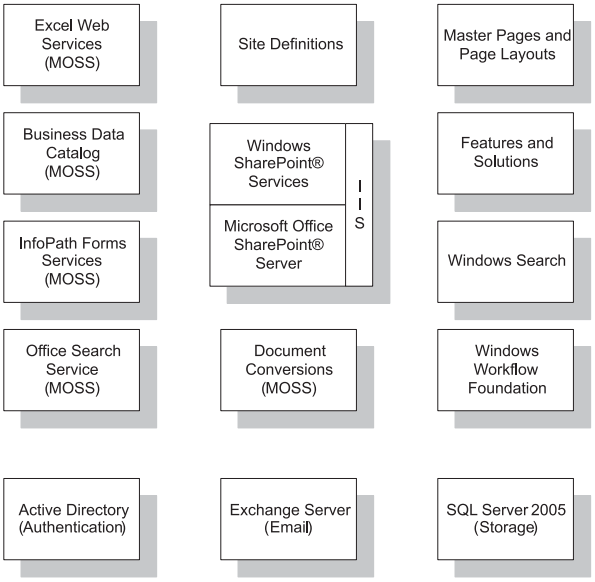


SharePoint Personalization Model

NOTE *On the technical side, SharePoint has a dedicated namespace for User Profiles.*

Services

SharePoint directly depends on some services which are provided by other Microsoft Products. SQL Server, provides the data store and schema needs, Active Directory handles the Authentication (SharePoint only handles authorization), and Exchange Server handles e-mail. There are many other services that are either add-ons, or, as is the case with Excel Web Services, separate products that have been integrated. A high level view looks like this:



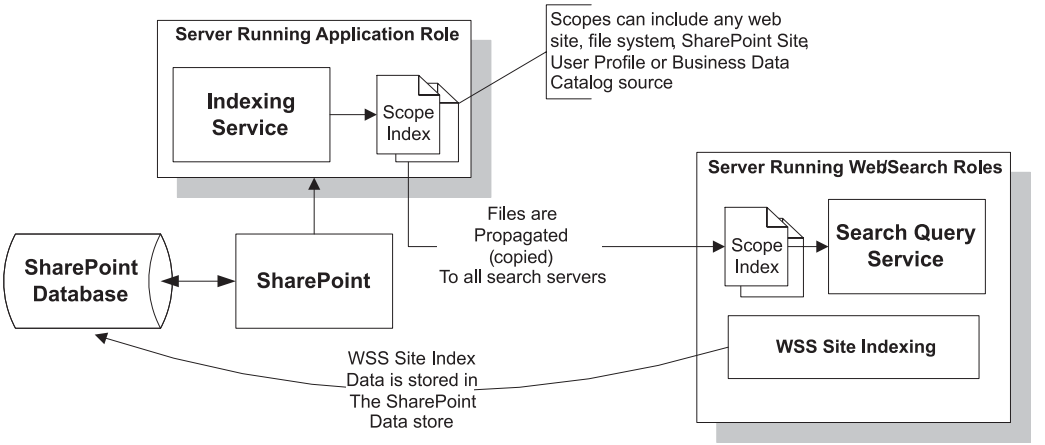
Many of these services are simply a part of SharePoint, such as the Master Pages and the default Windows Search. Several of the MOSS services are actually complete features (or actual products) as explained next.

Office Search Services

Search is probably one of the most important tools in SharePoint and is usually one of the biggest concerns, both on the operations side of where index information is stored and on the user side of getting reliable results. Internally within the SharePoint platform (both WSS and MOSS), it supports Windows Search; this is a limited search that is used to index content (lists, libraries, documents) that is local to a site and is stored in the database along with the other site information. This provides better performance and supports the distributed and separated nature of WSS sites.

MOSS uses Windows Search, but utilizes Office Search Services (a stand-alone product integrated into SharePoint) for enterprise level search. As an enterprise tool, the purpose of OSS is to provide search services across a wide array of information resources, including web sites, file shares, Lotus Notes databases, and—with the Enterprise Edition—data sources (database and web services) in the Business Data Catalog. Not coincidentally, it also knows how to best index SharePoint sites and their content.

Search is based on Search Scopes that define the content source (a URL, file path, and so on), while OSS indexes content on these sources and stores it in local server files (usually a dedicated Application server). This allows the indexing process to run repeatedly while not impacting performance on web services or the database. When indexing is complete, these files are propagated (copied) to all servers that are running the Search Query Service (usually all Web Service servers). As local files on each server (with the properly formatted URL links to items), the Search performance is greatly enhanced since no network connection is needed.



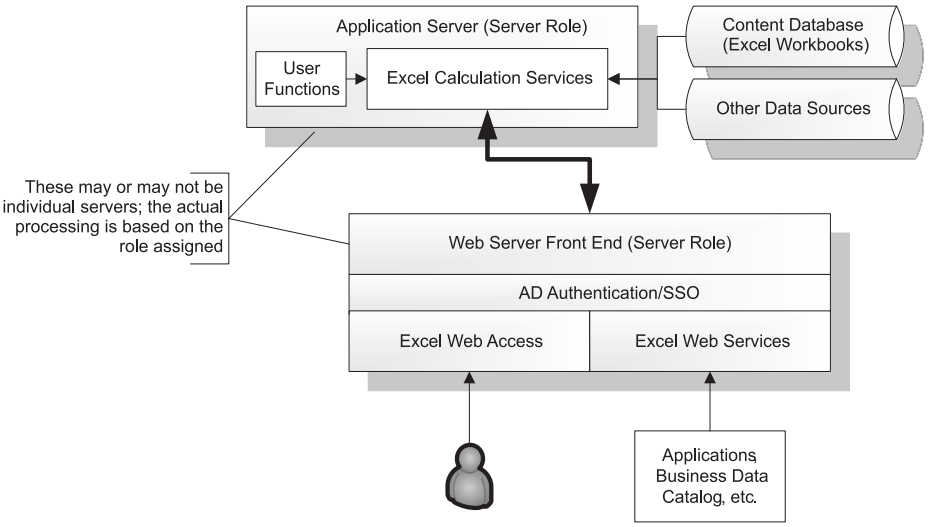
Also note that within SharePoint, many aspects of Search are customizable, including the results presentation, and that there are also several web parts available to provide summarization of results. Note that Office Search is covered in great detail throughout the book.

Excel Web Services

Excel Web Services is a new product that has been released with Microsoft Office Excel 2007 and is included with the MOSS Enterprise Edition. As you may guess, the primary function of this service is to enable Excel functionality via web access—not just viewing, but the execution of macros, calculations, and more, in a fully functioning mode without the need for the Excel Client application.

To accomplish this, Excel Services is really two primary components. The first is a calculation service that handles doing what the Excel client does (calculations, macros, and so on), but that can also can pull information from programs other than Excel. The other component is a presentation control that provides Excel Workbook-like functionality such as inserting and deleting rows, adding data, and so on.

In SharePoint terms, the functionality provided by Excel Services is the same, just housed differently so SharePoint can apply its own security. The Calculation Service is run on the Application Server (isolated from Web Services) and provides much better performance. On the presentation side, it provides both Workbook Services which allows stored spreadsheets to be presented within a SharePoint page and a Web Service that provides access to the data from external services. As an added bonus, SharePoint's authentication and Single Sign On (SSO) can be used for controlling access. This implementation looks something like this:



Excel Web Services Model

Remember that these services are shown separately based on a server farm; this can easily be run in a single server environment as well.

Business Data Catalog Service

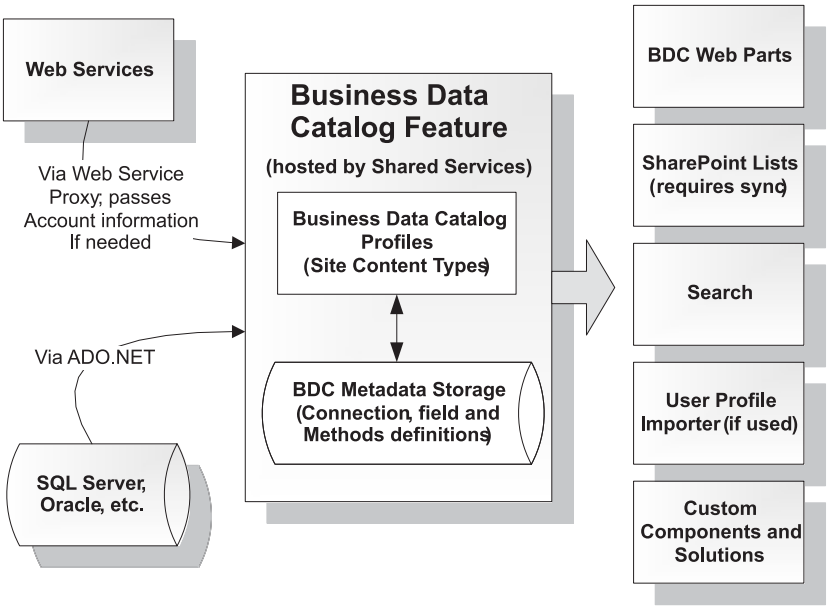
The Business Data Catalog Service is the way in which external data sources, such as search, query, view and even import and update, can be incorporated into SharePoint. While Data Views in SharePoint Designer can provide a query view of data from any data source, the BDC opens the door to nearly all enterprise data, regardless of the format and makes it searchable. (The BDC is covered in detail in the online Web Chapter 1.)

As the name implies, the BDC is a catalog of data sources, which can include databases using SQL Queries and Stored Procedures or Web Services that can be practically anything. The BDC is based on a profile of the information source, which includes connection information (supporting both account/password and Single Sign On), the definition of the data it produces, and a series of methods that enable interaction with the data including direct queries, enabling search of enterprise data and can do backward updates. These profiles automatically become Site Content Types in SharePoint and can be used for a host of things, including importing of the data into a SharePoint list. As Content Types, BDC data sources can be enabled for Search and are indexed with the same ranking criteria as other SharePoint data.

A number of web parts are also provided for enabling the Methods defined in the profile. From a simple display, these web parts can communicate with each other to create various views of information with no programming effort. If enabled in the profile, data can be updated and posted back to the original data source. In addition, all aspects of the BDC are readily available through the Object Model and can be used in other applications, including web parts, web components, and web services.

In addition to the BDC functions, this feature also includes the Key Performance Indicator lists and Web Parts used in the User Interface. KPI lists are used to hold metrics data that can be pulled from the BDC, which then provides some form of measurement of

a particular activity. KPI web parts are indicators that provide dashboard-like indicators (red light, yellow light, and so on), based on the values obtained in a KPI list. A general view of this service is as shown:

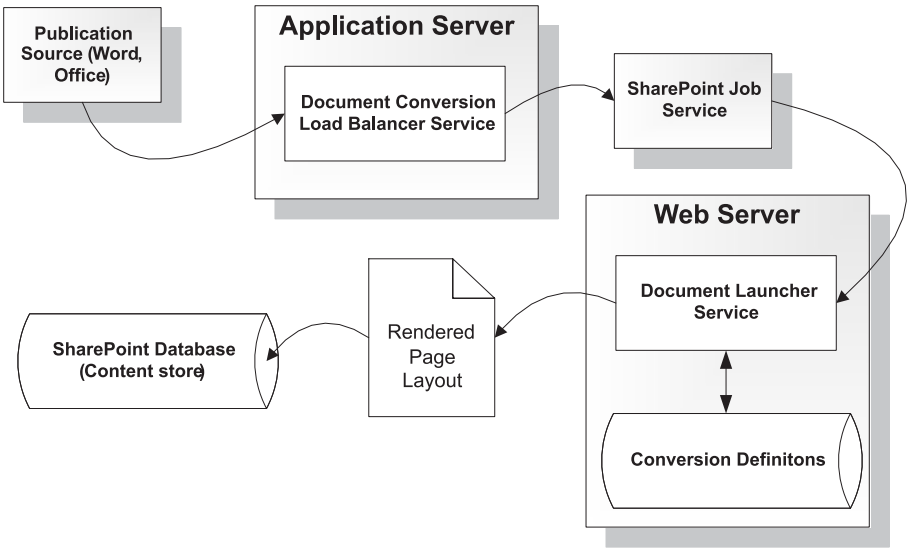


Document Conversion Service

The Document Conversion Service is an integrated part of the Content Management features within MOSS. As the name implies, its specific purpose is to convert a document from one form to another, like a Microsoft Word Document into HTML. The primary use of this is in page publication using Publishing Pages with content controls. Publishing pages can only support HTML or XML formatting and cannot interpret the types of markup included in Word, PowerPoint, or other such documents. The Conversion Service is used to call a background application that can read the document in its native form and convert it to HTML or XML. The converted content then uses a SharePoint publishing page to create a new page on the site.

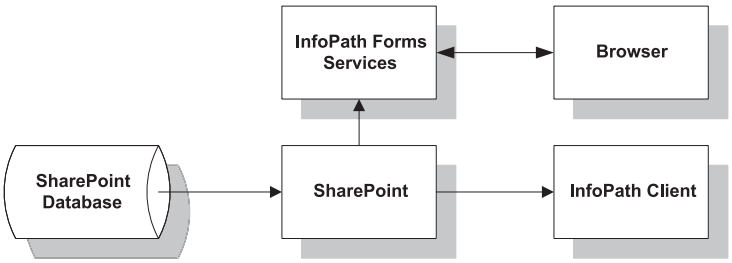
As you might imagine, conversions for publishing type Office products are included, and the service is set up to easily add additional converters available from third-party sources or custom developed. Converters themselves are not as complicated as they may sound; the only effort is converting the document to HTML or XML—the converter service takes care of the rest.

Since some document conversions can take some time and may require a bump up in system resources, the Document Conversion Service is broken up into two parts: the Document Conversion Load Balancer service runs on an Application Server and is responsible for handling requests for conversions (such as a publishing request from Word). The Load Balancer starts a background job on any one of the Document Launcher services running on the web front end(s). The Document Launcher handles initiating the document conversion program and dealing with the HTML or XML retrieved. Conceptually, the functional components are as shown here:



InfoPath Forms Services

In MOSS, InfoPath Forms Services provide a complete rendering engine for forms designed in InfoPath (referred to as "XSN" files). In MOSS, Forms Services enables InfoPath forms to be opened and used automatically by any browser with full functionality without needing the InfoPath 2007 Client software. This enables both SharePoint and SharePoint Designer use on the fly. InfoPath forms, such as the Document Information Panel (DIC) used in workflow inside of Office Applications, is a custom InfoPath form. In Designer, InfoPath forms are used to provide prompts and inputs for users. There is not too much to know in the way of architecture here, since SharePoint handles the request, and with the tight integration with Office, SharePoint knows when and how to open the form, as shown next:



There has been one big enhancement in the use of the Forms Libraries within SharePoint; Site Content Types can now provide profiles of InfoPath forms, which can be used across multiple form libraries and provide multiple form types in the same library.

Summary

As you have seen in this chapter, the overall architecture of SharePoint is quite extensive, providing users, designers, and developers with tools needed to extend it as needed. From this introduction, you should now have a good understanding of all the components involved, including those that are add-ins to SharePoint-like Excel Services. You should also have a better understanding of how SharePoint provides a platform for development in addition to a functional application.

Although all of these tools are great, how you use them is just as important. Knowing if a component in SharePoint will be beneficial depends greatly on how it will be used and who will use it. This goes far beyond development and into what I call "The Business Side of SharePoint," as you learn in the next chapter.

CHAPTER 3

The Business Side of SharePoint

While SharePoint is a great product technically, there is a crucial business aspect to both using and implementing it in an organization. An understanding of the business needs that SharePoint will address, as well as a grasp of SharePoint's capabilities, will determine whether there is a return on investment. A key driver is that SharePoint is a user-based system, meaning that once installed, the users control the system. Aside from systems work (such as backups), IT is simply a different kind of user.

There are two aspects to the business side: the first is capturing the problem that you are trying to solve, without limiting yourself to what SharePoint is capable of (you want to solve the problem, not box yourself in). Focusing on exact needs (for example, document management), then qualifying them (why and how much management) is the guide to the second aspect. In other words, you should be mapping the needs into what SharePoint covers out of the box or can be configured to handle. This will also drive out what SharePoint won't be able to do, so you can easily identify features that might be a "build or buy" solution.

From a planning perspective, you also have to work out a roadmap of how you intend to implement SharePoint, including the ways you will handle things like governance (user use and security), topology planning, metadata use, and so on. You also have to figure out which users or groups will get access when (and be sure to set this expectation with the users!), as it is rare that everyone will get access from day one. I can assure you of one thing—you *must* have a plan with SharePoint and consider everything from who can do what to how you will keep the system backed up. If you simply stand it up and let the users have at it, you'll spend most of your time trying to get it under control.

While many organizations have project planning methodologies, most will find that these don't work with SharePoint, because it isn't a website kind of tool. Instead, it is an enterprise tool that will eventually be used everywhere in the organization (if you're getting the most out of it). The biggest difference is that there are a lot of starts and stops, and most of the effort is in the configuration, not in development. This is further confused when folks realize there's no need for a traditional Development and QA environment (you can't QA SharePoint and then move it unless using a Publishing site).

In answer to this problem, I've developed a proven process that is more focused on a Portal implementation project designed to be easily repeatable regardless of the size or

scope. Whether implementing for a small team (even a project) or an internationally distributed organization, following the same steps without cutting corners won't *guarantee* success, but it will certainly give you a better shot at it.

After doing a few hundred implementations of SharePoint, I have a few tips that might help:

- **Keep the scope** When you have begun to solidify what you will be doing with SharePoint, lock in down to a very specific scope and stick to it (I cannot stress this enough). SharePoint is extendable and expandable at any time, so adding on is easy and very prone to scope creep. My suggestion: hold your horses and go slow.
- **Get buy-in** You must have commitment from the users, the organization, and IT. Any pushback should be addressed before you begin, as you will have to accommodate for it in your planning.
- **Get a (reliable) business sponsor** Nothing is worse than a limp handshake. You need someone (maybe you) as a point person between the project and business. This includes helping set expectations.
- **Do thorough business analyses** Don't think you know what users need. SharePoint should be thought of as both vertical (focused on a specific need) and horizontal (ways to share between vertical needs). For example, I worked with a company to build a Knowledge Management system for tier one and tier two support. They were fine on the vertical (the needs for each group), but almost missed the fact that one automatically feeds to the other (the horizontal workflow). Cross-sharing information and the horizontal workflow have turned out to be their biggest ROI!
- **Try to avoid piloting** I can't tell you the number of times a pilot became production overnight. If you must pilot, keep it focused and *very* limited; keep the number of decision makers small, too. One person with a "misunderstanding" of how SharePoint works can be a disaster (I know, believe me!).
- **Create a project road map** Much of implementing SharePoint is based on configuration and training, as well as some development. Unlike the traditional development model (one of the reasons for my methodology), all of this can be done concurrently and rolled out faster than any custom development project (as an FYI, if SharePoint takes longer than two months to roll out, you didn't plan it correctly!).

SharePoint Governance

One important aspect of a user-based system like SharePoint is governance; that is, determining who can do what and where. Part of this is simply based on the management of a site and topology structure, as you need some level of control. There's also security: you need to determine which users have rights to create items, use workflows, and so on, and site (rights) groups must be assigned. Another aspect is the information architecture, which covers how users will be targeted (audiences) and the taxonomy of content. Last is the information itself—unlike a website that is more regulated, users (if given the ability) can upload and add content anywhere and make it immediately visible to others. This is good in collaboration, but a bit tricky if misused.

Site Governance

SharePoint itself is fully intended to be a user-managed tool across the board, and thus the user is encouraged to create their own environment. However, as with anything that is of a common purpose across the enterprise, it is not completely care-free in management and users must have some limitations set.

The information architecture of a SharePoint site is designed so that users can find information quickly through search. However, the navigational aspect is up to you. It's important because it controls both organization and usability, and it ensures ease in both finding information and in the visibility of sites. For usability, a good rule for navigation is the "two-click rule" (reaching desired content within two clicks), and this must usually be controlled centrally. My recommendation is to restrict top level sites, common enterprise sites (Corporate News site, Document Center, and Reports Center), and top level navigation to a very limited number of users.

Site Policies

To ensure consistency of use, regardless of the site purpose, Site Policies should be created for specific site types that limit the amount of storage in any single site. The purpose of this is two-fold: first, it provides better backup and restore capabilities, since the time to restore can be minimized; second, it maintains a consistency in the use of the sites.

Site Templates

For consistency, Site Definitions that are specific to the intended use should be created. For example, you might create a common Project Site template that already includes all of the necessary lists and documents needed. Specific templates should be required in order to create consistency across the entire site structure down to the lowest level. This should be done primarily for usability reasons, but will also help when users transition between areas of the company and when new employees are introduced to the system.

Content Governance and Education of Users

With the use of My Sites and the general ability to publish pages, it is extremely important that end users be educated on both appropriate material and appropriate use. While it should go without saying, I usually provide the following to new users:

- Maintain a professional tone in content at all times.
- Content that is secured, proprietary, or that may have legal implications should always use an approval process for publication.
- Documents and other content loaded or published in any site must always be of a business nature; personal documents should be kept within My Sites. (This includes any kind of content.)
- Copyrighted information or images should not be used without explicit authorization; when direct reference to external content is needed, links are suggested.
- Pornography in any form is unacceptable.
- All content, including content stored in My Sites should be considered business confidential and not personally confidential.

Methodology Overview

Development of enterprise information systems (or portals) is much more akin to integration of an ERP or SCM system than a website, because a portal touches (or should) every part of the organization and its usefulness depends on its participation. In general, a web site can be viewed as a "window of information," whereas information systems should provide a two-way street, providing users with both a reason to come to the portal and a way in which to contribute to it. As an application versus a website, SharePoint must be configured to fit the information architecture, business model, and workflow, based on the principals of driving usage and facilitating collaboration on content, projects, documents, and so on.

The processes involved in a portal type deployment are very different from most applications. The typical cascade approach to a web application (Requirements, Design, Development, QA, and Deploy) does not apply; much more has to be done with the business and configuration of SharePoint versus development. Taking all of this into consideration, this methodology provides a proven formula that covers an implementation end to end, with unique emphasis on key uses of SharePoint:

- Potential users and the tools they may need (with emphasis on commonality)
- Content produced, content needed, and Metadata (for search/sort)
- Vertical usage and points of intersection with other groups (the horizontal workflow)

Breakdown of the Steps

The methodology is broken down into a number of phases, which, in some cases, overlap on purpose. Figure 3-1 provides an overview of the entire process, and Figures 3-2 through 3-8 detail the individual phases. The point of the multilayered approach is to provide the fastest rollout with the most chance of success.

Phase 1: Initial portal project planning

- Heavily focused on the overall intended use of the portal for capturing the end-use and accommodating for future features in initial development
- Define elements that will drive users to the sites (content, tools, workflow)
- Source and methods of producing relevant content to the user (targeting, audiences, memberships)
- Methods to encouraging daily use (for example, My Site, E-mail, Alerts)
- Types of content to be used
- General tools available out of the box

Phase 2: Business analysis and branding

- Constituent Group Identification including Audiences and Memberships
- Identification of who will use the portal and for what purpose
- Clear identification of potential user security requirements (Intranet, Extranet, Anonymous)

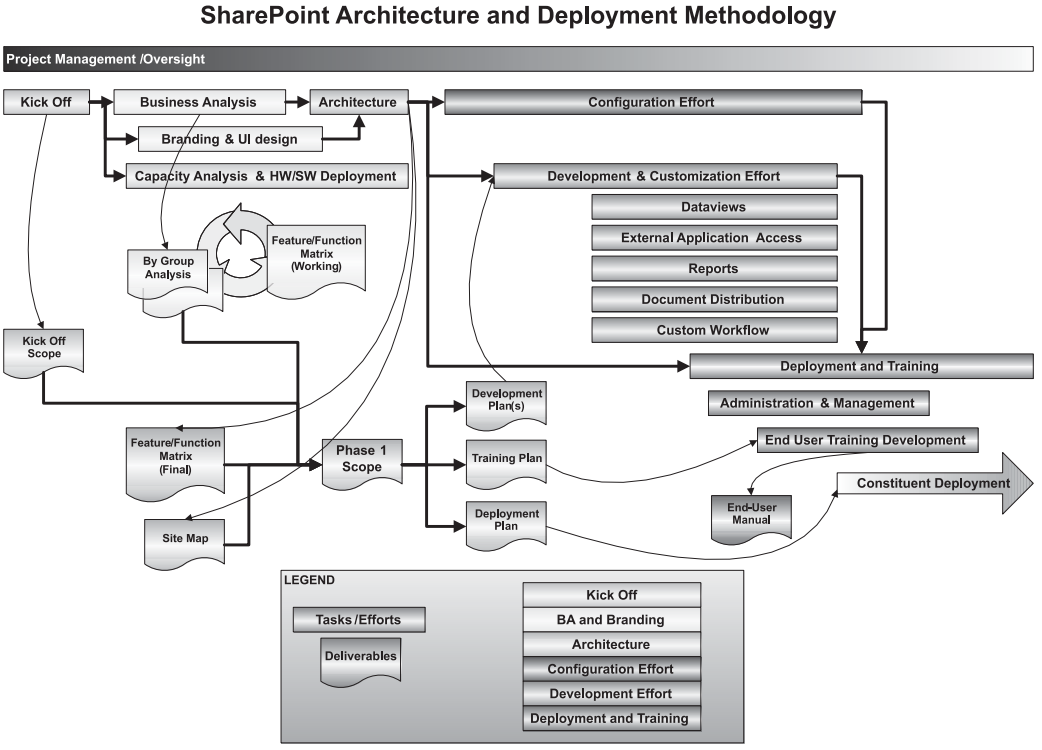


FIGURE 3-1 Methodology high level overview

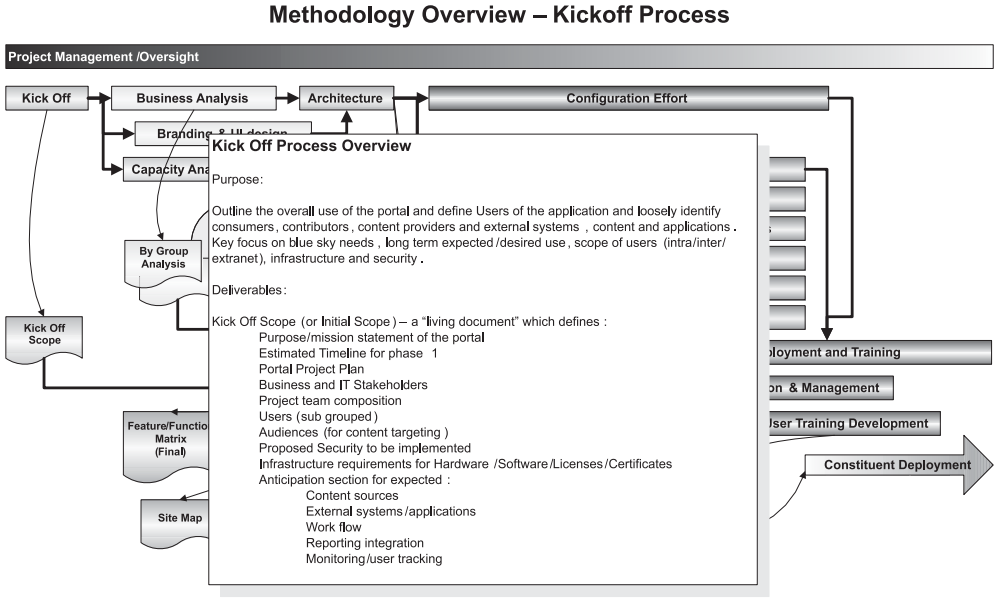


FIGURE 3-2 Methodology kickoff process

Review each Constituent group to develop the following:

- Collect wish lists of functionality
- Insight to expected use
- Documentation of the "ins and outs" of content movement
- Documentation of the business processes for potential workflow
- Documentation of the internal interactions to identify potential areas of collaboration
- Identification of the potential metadata candidates (including Content Types and Site Columns)
- Identification of the current workflow and functionality needs
- Maps of existing interaction and points of collaboration
- Identification of the new areas of possible integration or sharing
- Setting of CLEAR expectations to users

- Define branding and image ideas; set a scope on how much time to dedicate to the look

Phase 3: Devise a formal information architecture and topology

- Methods to drive content (Publishing, Word, Workflow, Business Data Catalog)
- Content contributors
- Audiences and Memberships
- Groups (security)
- Development of the feature-functionality matrix
- Identify out of box versus custom development

Phase 4: Map topology to intended use

- Development of purpose sites
- Areas of intersection for functionality
- Integration of Keywords, Groups, Audiences, and Search
- Setup of Metadata (Site Content Types, Site Columns, Document, and list item sub-classification)

Phase 5: Set scope on requirements and delivery

- Establish (and publish) the scope of the first implementation
- Define a change control process to handle modifications or updates needed and define an acceptable amount of impact to the timeline that will be acceptable

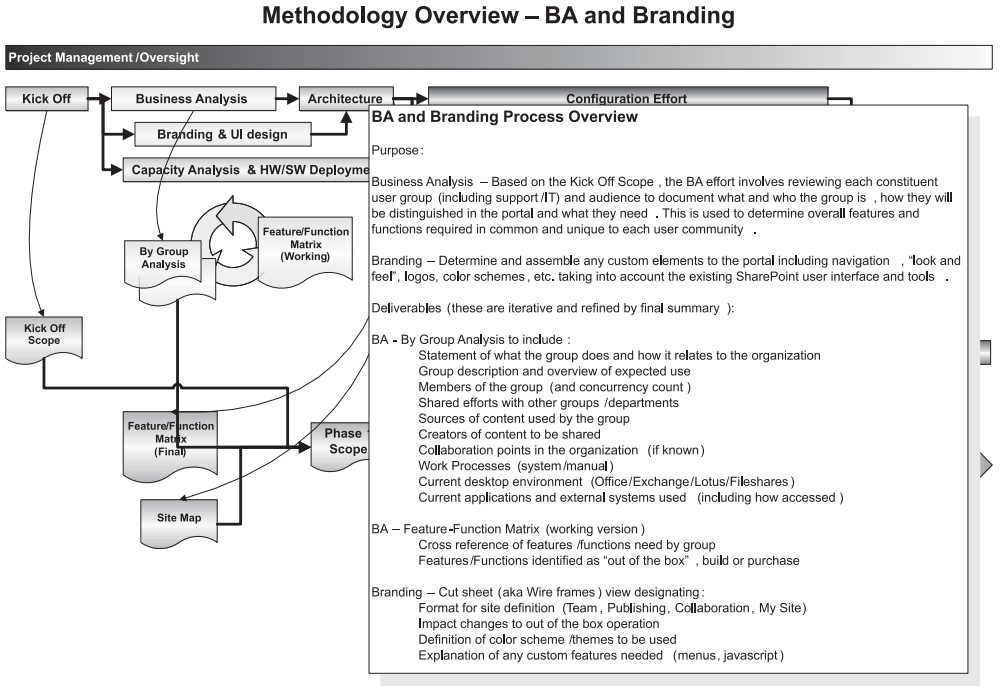


FIGURE 3-3 Business analysis and branding

- Set a realistic expectation delivery plan with users (including training) on approximate dates

Phase 6: Set up the necessary infrastructure

- Determine interfacing systems and applications
- Define access methods
- Provide technical map of ports to use, naming conventions for databases, and so on.

Phase 7: Develop the architecture

- Finalize the feature function matrix on configuration and features
- Lock down the final scope of the development
- Provide the overall final scope and subproject plans for concurrent work in configuration and development

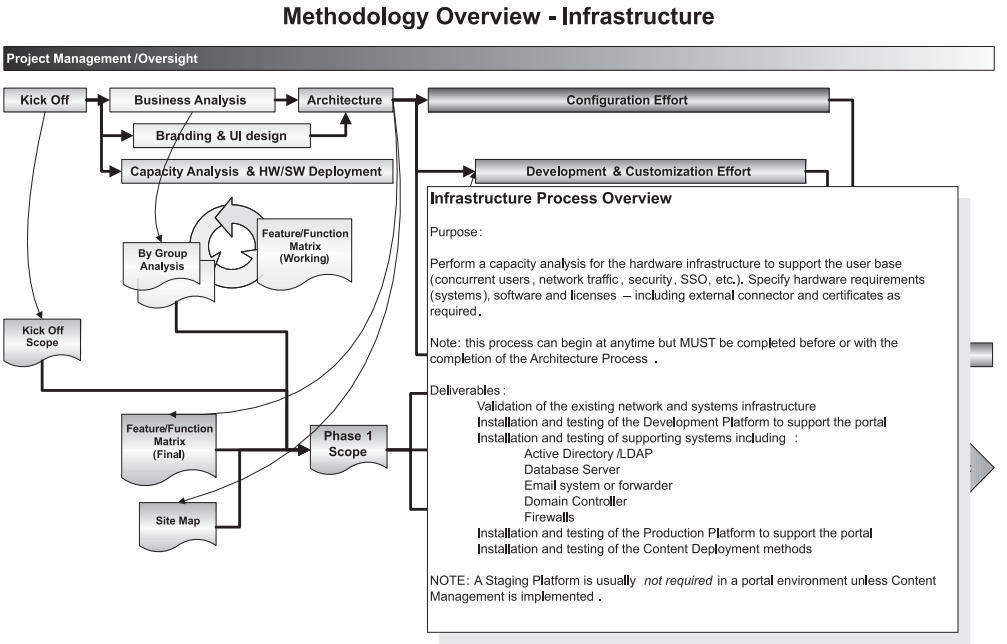


FIGURE 3-4 Infrastructure planning

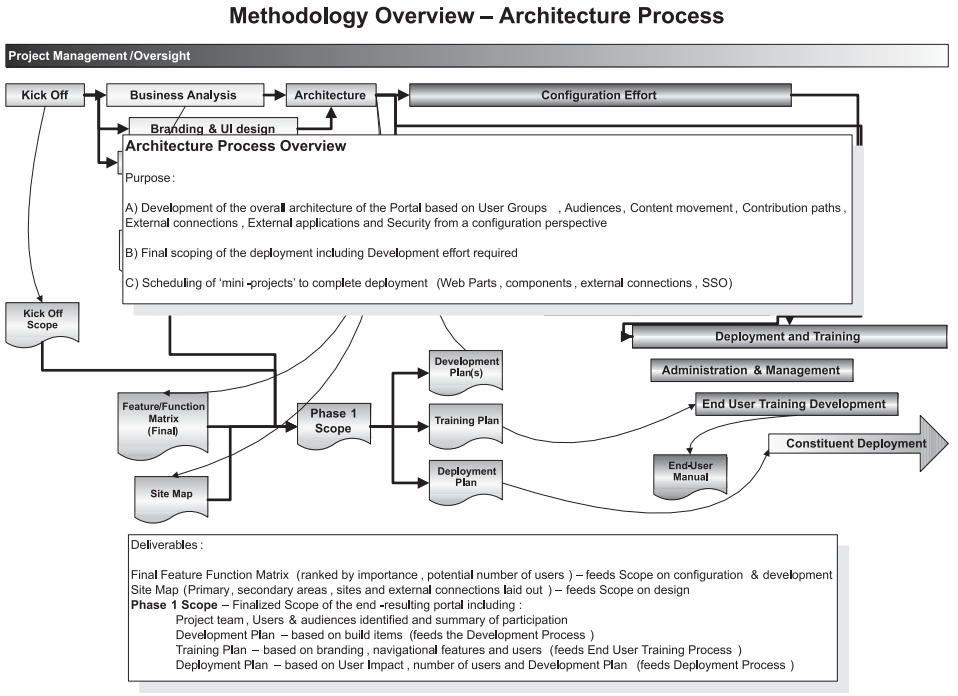


FIGURE 3-5 Architecture and topology design

Phase 8: Concurrent rollout of configuration, development, and training

- Complete primary configuration of topology (as provided in the scope)
- Configure the navigational structure
- Develop web parts and receivers as needed
- Package for deployment or combine into features
- Begin staged rollout to selected users
- Begin all training efforts

Phase 9: Finalize user deployment strategy

- Designate content owners for production system
- Plan knowledge transfer to groups as needed
- Plan out constituent group adoption based on level of content contributions (heavy contributors are first)
- Publish deployment schedule

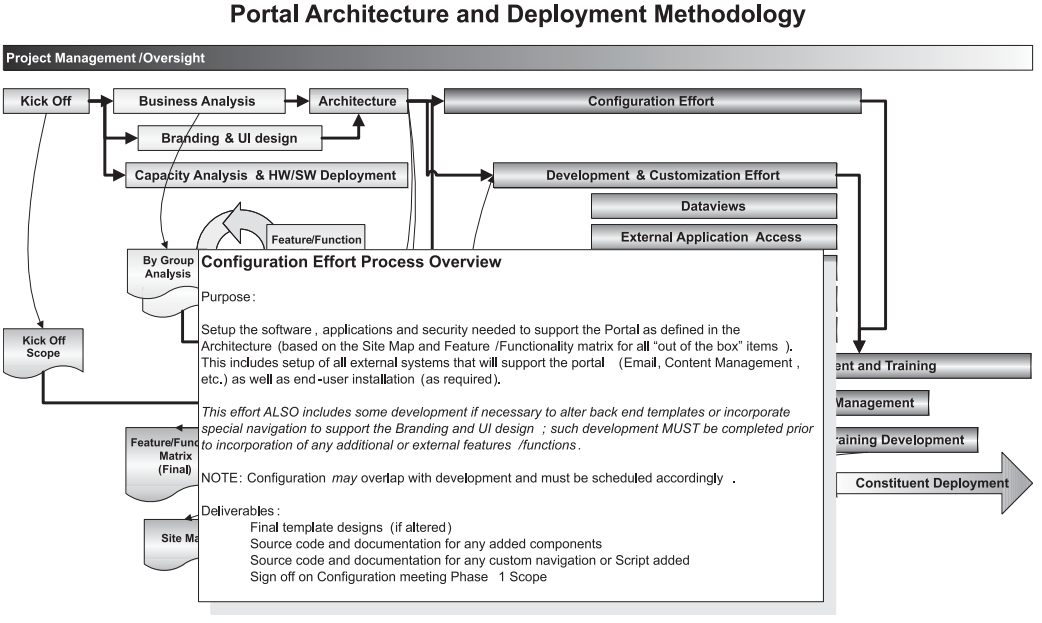


FIGURE 3-6 Configuration

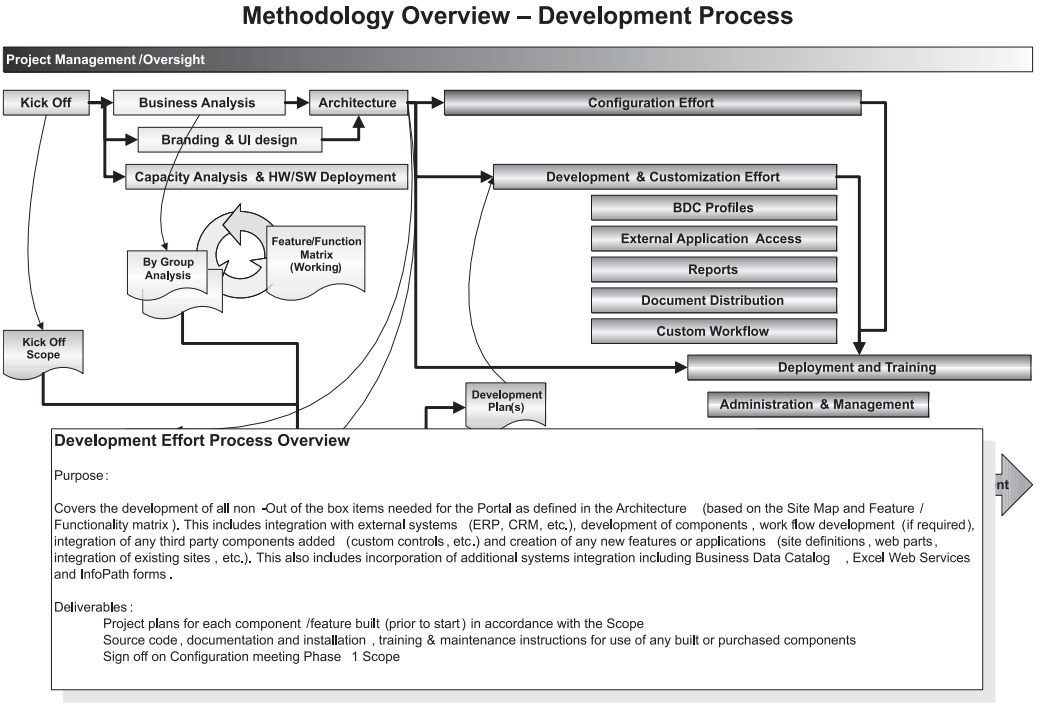


FIGURE 3-7 Development process

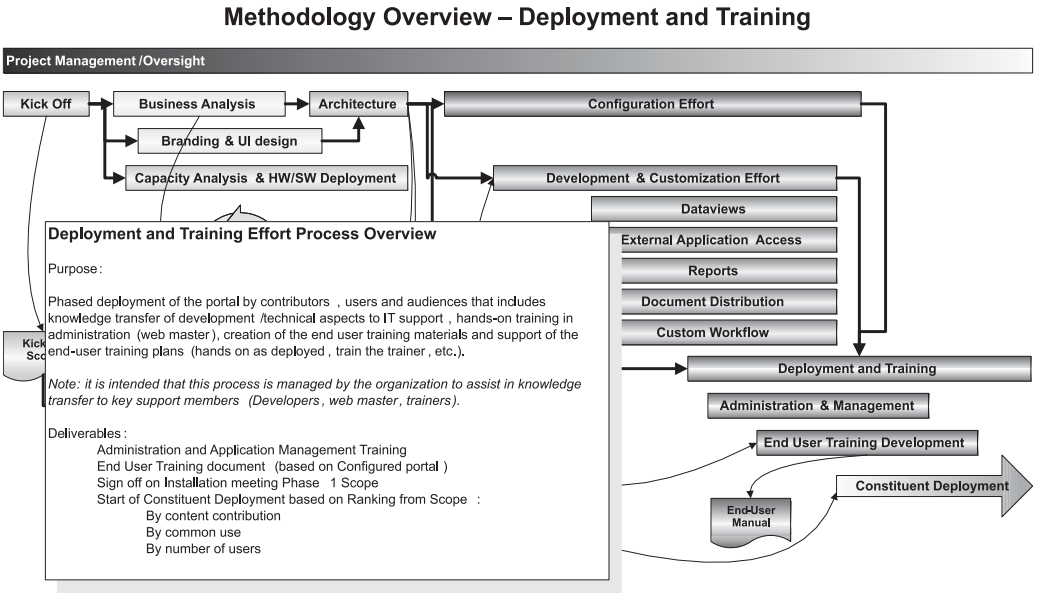
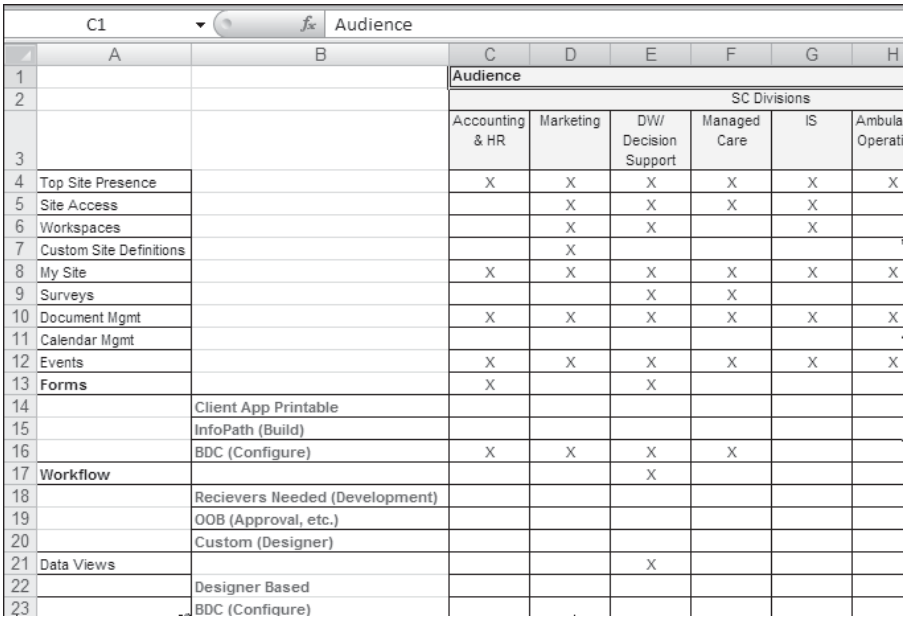


FIGURE 3-8 Deployment and training

Example Feature Function Matrix

The feature function matrix is basically designed to help you consolidate the particular user needs by the type of features provided in SharePoint. This also allows you to make important decisions regarding whether something is a configuration issue, requires some development (such as a receiver or custom workflow), or may require a third-party purchase. In practice, I usually use a spreadsheet to manage the process similar to this.



C1 fx Audience

	A	B	C	D	E	F	G	H
1			Audience					
2						SC Divisions		
3			Accounting & HR	Marketing	DW/ Decision Support	Managed Care	IS	Ambula Operati
4	Top Site Presence		X	X	X	X	X	X
5	Site Access			X	X	X	X	
6	Workspaces			X	X		X	
7	Custom Site Definitions			X				
8	My Site		X	X	X	X	X	X
9	Surveys				X	X		
10	Document Mgmt		X	X	X	X	X	X
11	Calendar Mgmt							
12	Events		X	X	X	X	X	X
13	**Forms**		X		X			
14		Client App Printable						
15		InfoPath (Build)						
16		BDC (Configure)	X	X	X	X		
17	**Workflow**				X			
18		Recievers Needed (Development)						
19		OOB (Approval, etc.)						
20		Custom (Designer)						
21	Data Views				X			
22		Designer Based						
23		BDC (Configure)						

Summary

While every organization is different, you must have a plan—even if you are particularly bad at it, simply hoping you will catch all the details is not a good approach. Of course, following every single step and deliverable outlined is not always necessary, but following the general path will save you a lot of time regardless of whether implementing for a department or the global corporation. As you move into the next chapter on the actual planning of the SharePoint install, you will see many of the elements of this methodology in play. Key decisions to be made for capacity, users, and so on, are all driven from the analysis phase. You'll see even more of this when working in post-install configuration.

This page intentionally left blank

CHAPTER 4

Installation Planning

The Windows SharePoint Services platform supports a number of configurations, from development on a single system to a full-blown, multi-tiered server farm. The idea behind the configuration options is quite ingenious—you can start from a small configuration and expand as needed to enhance features, increase performance, and so on. In fact, this is one of the very few enterprise server products where throwing hardware at the solution does increase scalability and performance markedly.

As the core component of SharePoint, Windows SharePoint Services is designed to support a single server or small, medium, and large server farms with services spread across the hardware. In the bare configuration, SharePoint Services supports using SQLExpress and WSS across a server farm expandable up to the full server farm configuration, using SQL Clustering, Load Balancing, and similar. Although this is covered in the next chapter, it is worth mentioning here that there are two types of installation:

- **Basic** All options are installed for you, including the database; you cannot create a server farm nor change the database. This option is usually used for a single server environment.
- **Advanced** You can pick and choose the options you want to install, create a server farm, and choose the database. This option is usually used for any multisystem configuration.

Planning Your Install

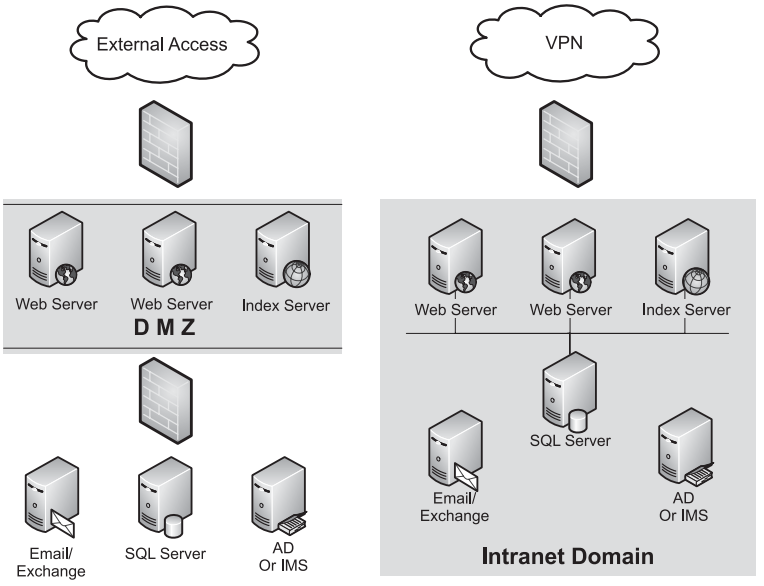
Like most anything you do with SharePoint, installation requires a bit of planning. If you are following my recommendations, a good business analysis and the resulting *Feature Function Matrix* will provide the best guide to what you need to install and how you will use it. For most environments, the most expensive element is hardware, so the rule of thumb is to size servers based on assumptions, but you don't want to overdo it. For example, if you suspect that there will be a large volume of storage, put the money into the SQL Server and

beef up the web servers only if needed. If you suspect that searching for things will be the emphasis, use a dedicated index server. Since SharePoint is intentionally expandable in nearly every way, you can get away with staging deployment and adding on only when needed. Of course, if you have the budget, go for it and spare yourself from having to upgrade later. Some key things to consider:

- **Plan and verify network connectivity** Plan the way your servers will communicate; both with each other and with the users, and then verify that there are no breaks in connectivity.
- **Determine your authentication methods** Choose the best method for authenticating your users, in order to support the services they need and to leverage your existing systems.
- **Determine the account mode** If using Active Directory and WSS, you must decide whether to turn on auto-account creation.
- **Size your server(s) and server farm** Plan on the minimum server setup required to support your intended use, number of users, and minimum capacities needed. During this step, you should also commit to the service, hardware, and licenses needed for the Disaster Recovery level you intend to support (acceptable time limits for recoveries, acceptable downtime, and so on).
- **Determine server roles and software options** Decide on the roles that servers will play and the additional features that will be available (such as Enterprise services, for example).

Planning the Network and Validating Connectivity

Setting up the network connectivity between servers can easily fill its own book, so I won't dwell on the various options here. In most cases, requirements for SharePoint are similar to that of any other web application. Most of the decisions are based on your security and access needs. An intranet needs less security, but also needs a way to ensure that internal users outside of the network still have access. An extranet, on the other hand, needs more security and also needs a way to authenticate. However, for the most common, medium-server farm setup, SharePoint might typically look something like this:



Extranet/Internet *Internet/Local network*

Obviously, this leaves out a lot of detail and your specific configuration will depend on your use. Conceptually, using SharePoint for an extranet or internet environment will usually use a DMZ to secure the web front ends and place the back end SQL Server in the internal network. An intranet, on the other hand, may make use of a subnet contained within the internal domain. Regardless of your network, there are some basic validations you should perform to ensure trouble-free setup and operation:

- **DNS** DNS is crucial to the overall operation and vital to server-to-server communication within a farm; often, production DNS environments have authentication problems or "moments of silence," when the DNS server does not respond in a timely manner. (The result is usually a failure to respond to a user request.) Verify your DNS setup *before* attempting to install any of the servers to be used for SharePoint and make sure that no latency exists. As recommended in the installation process, consider adding HOSTS file entries for all servers involved.

- **SQL Server communication** SharePoint really lives in the database, so communication between the servers and back-end SQL Server is critical. Consider the best possible connection here for both speed and security and please (!) use a dedicated SQL Server—you'll thank me later.
- **General server to server communications** Before any install, verify that all servers can cross communicate via both IP address and DNS name.
- **IPs and network placement** If you are setting up SharePoint for a production environment, set it up in the correct locations before installing something such as web servers in the DMZ, or similar. Having been there myself, I highly suggest you *do not* attempt to move SharePoint servers or change IP addresses after the fact, unless you feel like wasting a great deal of time.

Choosing the Authentication Method

To support the capabilities providing Wiki and Publishing sites and organizations that do not use Active Directory, SharePoint supports different methods of user authentication. The default is Windows Integrated (or NTLM), but it also supports any LDAP service and Forms Authentication. When you are considering server and farm configurations, you should decide on the authentication method first, as it will have an impact on your individual servers. If you expect to use more than one method, you should incorporate all of them up front; trying to add on an authentication method on a production system can be a dangerous undertaking. The type of authentication you decide to use is largely based on the end-use of SharePoint. Consider the following uses:

- **Internet/publishing site** Users will generally be limited to read-only access, which means you will likely use anonymous access. However, to support the benefit of targeting content, you can incorporate either an Identity Management System (IMS) or use ASP.NET Forms Authentication to manage logins and user roles.
- **Extranet/client site** In an internal extranet used internally, such as divisional sites, project sites, remote sites and the like, you have the options of using VPN/NTLM or basic authentication. For external entities, such as partners or clients, you can configure a separate zone for each, and then use Web Single Sign On to authenticate the user against the partner or client's identity management system (this allows user management on the third-party side). Alternately, you can create an "external" Active Directory instance to support external users, providing flow through access for internal users (via NTLM) in a one-way trust.
- **Internal (intranet)** By far, Windows Authentication is the preferred method for any kind of internal site, or, if using some kind of other IMS, for ensuring that user credentials are not passed through clear text. In any case, the best method to use is the one currently supported in the environment (for example, if non-AD, don't implement AD for the sake of WSS).

The different kinds of authentication, along with their ups and downs, are shown in Table 4-1.

Authentication	Upsides	Downsides
Windows	Uses existing Active Directory setup and groups (AD stays as system of record) User and Security Group management all handled through AD Best for use in existing internal LAN environments	VPN required for external access (local LAN authentication required) Not all browsers support IIS authentication protocols Not easily extendable for external entities
Web Single Sign-On	Can use federated authentication to secure authentication across organizations Able to use any LDAP authority (including AD) Can take advantage of using ADFS Best for use in external user environments (partners, clients, customers)	Requires altering the web.config file in the site SSL required to support ADFS An existing Federated Authentication System must be in place
ASP.NET Forms	AD not required Can work with any IMS system Authenticate Internet users using SQL Best for use in internet environments and publishing sites	Requires altering the web.config file in the site Requires SSL/TLS to prevent access attacks

TABLE 4-1 Authentication Methods

Domain Account and Active Directory Creation Modes

When using the default authentication method via Active Directory as the authentication mechanism, WSS supports two user account modes for adding users:

- **Active Directory Account Creation Mode** (ADACM) enables accounts to be automatically created in AD by WS. This enables users to be created using an e-mail address instead of a qualified domain account, the method most often used in Extranet or Hosted environments.
- **Domain Account Mode** (the default) enables the standard method of adding users to WSS by individual and/or security group. Users are first created in Active Directory, and then added to a WSS Site individually or by group, which is most often used for intranets.

The main difference in the modes is a Domain vs. E-mail-based account. In ADACM, users are creating using only the e-mail account supplied, which is handy when users are not part of the domain hosting WSS. Domain accounts are created as typical AD Accounts, that is, name, password, e-mail, and so on. Technically there is no real difference, because accounts are always maintained in Active Directory and in either mode, you can add users and cross site groups as normal.

A few points to consider:

- ADACM is only available for Windows SharePoint® Services V3, whereas it is not available for MOSS.
- Setting of the mode is done at installation time; after WSS is installed, an Advanced Option is displayed which enables you to define the SharePoint Organizational Unit to use in AD.
- You cannot change modes.
- You cannot use ADACM if installing on a Domain Controller.
- When using ADACM, creation of a user checks for an existing e-mail address in AD; if found, a new account is not created and the existing account is used.
- Some administrative tasks are removed when ADACM is enabled. You cannot add a user to a site, enable Self-Service Site Creation, or create a top-level site via the WSS Central Administration; this can be done programmatically or by using the SharePoint Command Line Tool (STSAdm).
- If ADACM is used, you must change the Domain Controller Security Policy on passwords (the Minimum Password Age must be set to 0; otherwise, nonadministrative users will not be able to change their own account passwords).

SharePoint Server Configurations

The first place to get started, of course, is to decide what kind of configuration you need. SharePoint's flexible topology supports everything from a single server to a completely distributed server farm. In addition, servers in server farm roles can be added for performance or capacity. Server farms are the way in which SharePoint supports multitiered processing based on the role a particular server takes on. Roles in server farms include:

- **The Database Server role** A dedicated server running either Microsoft SQL Server 2005 or Microsoft SQL Server 2000 with SP3 or later (the data layer). In large farms, this is typically a SQL Server clustered configuration with two or more servers.
- **The Web Server role** A front-end server that provides the web services used to render applications (the presentation layer). If more than one web server is used, you implement network load balancing to share the load.
- **The Application Server role** A front-end server that provides applications services such as search and index (the services layer); multiple Application Servers are also load balanced.
- **The Shared Services Server role** The same as the application server role, but dedicated only to running Shared Services (My Sites, and so on).

Of course, in a single server setup, all of the previous roles reside on the same machine. While there can be any number of variations and reasons for why you would use them (we'll get into the why a little later), Microsoft recommends the same basic configurations:

- Single Server (1 PC or Server)
- Small Server Farm (2 Servers)
- Medium Server Farm (4 or more servers)
- Large Server Farm (6 or more servers)

Single Server

For development, you would usually use a beefed up PC (with at least 2GB memory) or for multiuser use, a reasonable dual processor server (with at least 2GB). This single system will run IIS, SQL, and WSS/MOSS. In this configuration, all features and services of SharePoint and SQL Express or SQL Server are installed on the same system. If the *Basic* type of installation is selected, this system will be self-contained, meaning it cannot be expanded to create a server farm. The Basic installation also installs (and uses) Windows Internal Database/SQLExpress (SQL Server can be used but must be configured after the install). Since this configuration is quite limited, the primary use of this configuration is for Developers, QA Testers (for limited system test), and content producers. Other excellent uses might include remote teams, small organizations, and other situations where a server farm is not feasible.

Who Should Use It and Why

- **Developers** This is the minimum configuration needed for a developer to adequately build event receivers and web parts, as well as develop workflows.
- **Content developers** Using workflow, content producers can use multiple single server configurations to provide remote content development.
- **System testing for QA** This provides at least a basic level of system testing capabilities.
- **Remote teams** This provides consistent and secured collaboration.
- **Small organizations** This provides a collaborative environment for limited numbers of users.

Disaster Recovery

A single server presents a very high risk of loss, because a simple disk failure could wipe out the entire installation. If ever using this configuration for any type of production environment, you must set up a very vigilant backup process to ensure that you can recover from a system failure.

Small Server Farm

A small server farm is the smallest but scalable option in which you use two tiers to host SharePoint. This is typically:

- **One backend server hosting SQL Server** This is usually a dual-processor server with at least 2GB memory.
- **One front-end web server hosting IIS and WSS/MOSS** This is usually a dual-processor server with at least 2GB memory (more preferred).

The small server farm provides a bare bones tiered environment using separate servers for the web and SharePoint functions and another to host the SQL Server. While limited in bandwidth and the number of concurrent users it can handle, this is excellent for content producers, QA testers, remote teams, and small organizations where the number of concurrent users will be less than 1000 and storage will be moderate (under 500gb). This configuration can easily grow as needed.

Who Should Use It and Why

- **Content developers** Using workflow, content producers can collectively use it for staging of content on a portal wide scale.
- **QA system test** This provides testing of distributed functions (although load testing is not practical).
- **Project teams** This configuration provides an excellent platform for dedicated purposes such as centralizing projects.
- **Divisions or small organizations (< 1000 users)** Small organizations will get much better performance over a single server and still be able to expand later if needed.
- **Limited internet/extranet (< 500 users)** Provides a good foundation for limited extranet or internet where the number of users is low.
- **Prototype for a large organization** The small server configuration provides a low-cost starting point for deployment within a large organization.

Disaster Recovery

A small server farm presents a pretty high risk of loss in the event of a system failure; if you lose the SQL Server, for example, SharePoint is useless. You should set up a vigorous backup schedule (even if performing a full backup every day) to handle system failures.

Medium Server Farm

A medium server farm is the most common configuration used by enterprise organizations, since it can handle a large number of users, provide for a high amount of storage, and deliver good performance overall. Even for uses that will not have a high demand, you will still get added benefits in performance, uptime, and ability to handle light disaster recovery. This is typically:

- **One backend server hosting SQL Server** This is usually a dual-processor server with at least 2GB memory.
- **Two or more front-end web servers hosting IIS and WSS/MOSS** This is usually a dual-processor server with at least 2GB memory (more preferred).
- **One or more front-end web servers hosting IIS and WSS/MOSS to be used for indexing** This is usually a dual-processor server with at least 2GB memory (more preferred).

The medium server farm provides a true multitiered environment in which two or more load-balanced web servers, one or more index/job servers, and a single SQL Server are deployed. Like any farm, servers can be added as needed to expand bandwidth and to adjust

to future needs. For example, web servers can be added to support additional concurrent users and index servers can be added to enhance search.

Who Should Use It and Why

- **QA system and performance testing** This provides full testing of distributed functions and load utilization.
- **Medium to large organizations (> 500 users)** Small organizations will get much better performance over a single server and still be able to expand later if needed.
- **Distributed organizations (> 500 users)** Large organizations can spread out feature functionality across several farms; this enables multi-purpose sites and provides the performance needed for most site types.
- **Internet/extranet (unlimited users)** This provides the performance and distributed processing power needed for a typical mid-sized extranet or internet.

Disaster Recovery

A medium server farm provides the bare minimum setup for true disaster recovery, since the distributed systems enable quick replacement in the event of a failure without too much risk of losing data. The biggest benefit is up time—a server can be brought down for maintenance while still allowing SharePoint to operate at full capacity. Note the only exception here is SQL Server; if the database is running on single server in the network (not clustered), it still represents a single (and major) point of failure. Regular database and full SharePoint backups are a must.

Large Server Farm

A large server farm is the most robust of distributed configurations since it provides the best for performance, search, and failure recovery. An expensive option, this configuration is usually used by large organizations, mission critical environments, and large internets/extranets (unlimited users). While offering the same front-end performance as a medium farm, the large farm provides better protection of the SQL Server configuration, since a Cluster configuration is used. The typical setup for a large farm is:

- **Two or more backend servers hosting SQL Server in a clustered configuration** Each server is usually a dual-processor server with at least 2GB memory; communications in the cluster varies from high-speed LAN to fiber optics.
- **Two or more front-end web servers hosting IIS and WSS/MOSS** This is usually a dual-processor server with at least 2GB memory (more preferred).
- **One or more front-end web servers hosting IIS and WSS/MOSS specifically to run Shared Services** This is usually a dual-processor server with at least 2GB memory (more preferred).
- **Two or more front-end servers to be used for indexing** This is usually a dual-processor server with at least 2GB memory (more preferred).

NOTE *It is common to use multiple index servers for high-volume search environments as they can be configured to provide Search Query.*

The large server farm provides a true multi-tiered environment in which two or more load-balanced web servers, one or more Shared Services servers,[1] one or more index/job servers, and a SQL Server cluster are deployed. Intended for large organizations and/or high availability, this configuration will support a very large number of users and supports a high amount of storage. Like any farm, servers can be added as needed to expand bandwidth; web servers can be added to support additional concurrent users and index servers can be added to enhance search.

Who Should Use It and Why

- **Large organizations (> 5000 users)** Shear volume of concurrent users dictates that more front-end servers must be provided to handle the user requests.
- **High availability or mission critical applications** With all services distributed across the farm, there are no single points of failure enabling guaranteed up time or full 24×7 operation.
- **Intranet high availability** This provides the performance and storage capacity needed for a large intranet, where users exceeding 5000 concurrent and content storage is high.
- **Internet/extranet (unlimited users)** This provides the performance and distributed processing power needed for a large extranet or internet, which is common for commercial use.

Disaster Recovery

A large server farm provides the basic setup for true disaster recovery, since the distributed systems enable quick replacement in the event of a failure with a low risk of losing data. In addition to higher capacity across the board, better guarantee of uptime and high availability, this provides the best user and search performance.

SharePoint Backup and Disaster Recovery Planning

One thing to understand in recovery and backup is what constitutes a full backup in SharePoint. SharePoint is really composed of the database and the many settings it uses in the User Interface. Since you need to be able to selectively back up and restore things, SharePoint includes the Backup and Restore options in SharePoint Central Administration and only this utility creates a true backup of SharePoint. While maintaining database backups is also a great idea (because you could technically recover a disaster), a DB backup alone is not sufficient to restore a SharePoint instance.

In addition to backups, SharePoint is often a mission critical system so actual disaster recovery planning can be a crucial need. Depending on the size of the farm, the ability to purchase additional servers will dictate how much of a real plan you can achieve. Preventing a failure in the first place is, of course, the least expensive route; since SharePoint is always dependent on SQL Server, the more power in the back-end the better. SAN Storage and SQL Cluster are better investments than the front-end servers, which can be easily replaced or added. For the front end, hot standby servers are the best for a quick recovery (preconfigured to simply join the farm as needed). As is the normal case, true disaster recovery requires redundancy across the board and doubles licensing, hardware, and storage costs.

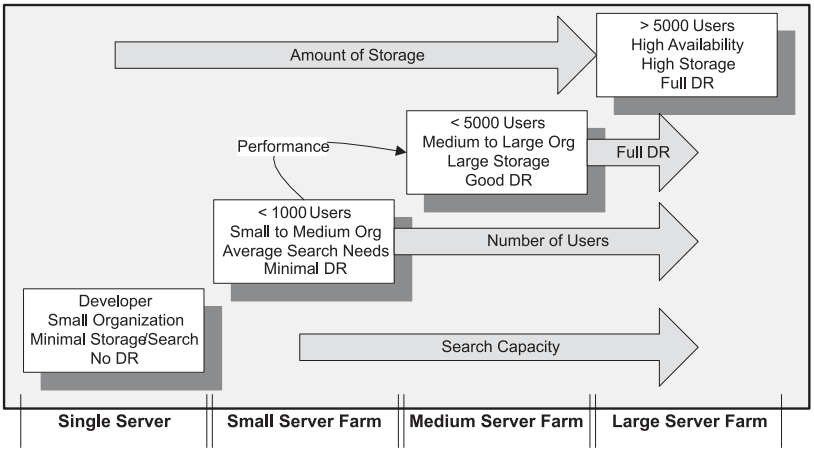
---

[1] **Note that Shared Services can be separated in a Medium Server Farm configuration but that is not a typical configuration.**

Server Farm Summary

In nearly any configuration (except for the *Basic* install), SharePoint's footprint can be expanded to fit the needs of the organization. From a single server configuration, organizations can expand the farm to suit the actual needs as the organization begins adoption. This expansion allows issues to be easily addressed by additional servers. For example, if search performance degrades, additional index server(s) can be added. If a large organization has user performance issues, an additional Shared Service can be added to break up the load. If web response is slow, additional web servers can be added, and so on.

There is a direct upgrade path from each farm, as indicated by the boxes showing typical requirements. In addition, the arrows indicate reasons to move to the next level. For example, a higher amount of storage may require moving from a small farm to a medium farm; need for full disaster recovery may require moving from a medium to a large farm, and so on.



Hardware Requirements

The hardware requirements are directly related to what is needed by the organization. Since both WSS and MOSS can be expanded after installation, it is easy to upgrade in order to handle additional users, increase performance, and so on. Fortunately for developers, WSS and MOSS can both be self-contained on a laptop, desktop, Virtual PC, or Virtual Server, and can run on nearly any system with enough memory and disk space. By the way, the whole environment is a must to do any development.

As a base line, servers should be at least a dual-processor system running at 2.5GHz or higher and have at least 2GB RAM, though more is suggested (SharePoint, particularly when caching, can be very memory intensive). The server must be running the NTFS file system (FAT cannot be used); if needed, a conversion utility called convert.exe converts a FAT file system to NTFS in place without affecting data or programs.

For small and medium farms, the SQL Server 2005 (or SQL Server 2000 with SP3) server should be configured with at least the same as the previous example. In a large farm or where additional features such as SQL Clustering are to be used, you should consult SQL Server installation documentation to determine the best setup for your specific configuration.

Using Virtual Machines

Whether you use Microsoft Virtual PC, Microsoft Virtual Server, or VMWare, virtual machines provide a new and increasingly common way of providing the same kind of performance and functionality while reducing the overall hardware costs and physical server foot print. In addition, the ability to completely capture the environment via disk copy provides the most secure and flexible disaster recovery capabilities available. Using virtual servers for WSS/MOSS is supported by Microsoft. Because of this, many organizations are now adopting this practice as an effective way to get the most out of their hardware, replacing standard servers with blade servers and disk arrays that can support inexpensive swappable drives. However, there are two major considerations:

- **Memory** Plan on needing a lot of it; the server hosting the virtual servers must be robust enough to handle the load. For decent performance, plan to allocate at least 2GB or more to each server. With the overhead of the virtual itself, a medium virtual server farm means at least 6GB or more.
- **Licensing** A virtual in a production environment is the same as a stand-alone server in terms of OS license, connector licenses, and so on.

CAUTION *A common practice with Virtual Servers is to create one and then simply make duplicate copies and rename them. However, renaming a system does not make it unique, because the original System ID that is generated is the same for each copy. This is bad news when joining a domain, because systems can be registered with duplicate SIDs causing all sorts of network oddities and problems. To fix this, you can use the NewSid utility to create a unique ID on the VS before joining a domain or installing Active Directory. The NewSid utility is available for download here:*

http://www.microsoft.com/technet/sysinternals/Utilities/NewSid.mspx

Be aware, however, that sometimes even when NewSid is used, WSS and MOSS installations can fail. In production, I recommend creating each Virtual Server from scratch.

Software Requirements

Depending on the role a particular server will play, several software packages and add-ons are required as a prerequisite to installing SharePoint. While all require some basic tools (like the .NET Framework), the role can dictate the type of OS and options enabled.

Web and Application Servers

- Windows Server 2008 or any Windows 2003 with at least Service Pack 1 (SP1); Web servers (only) can run Web Edition (WSS V3 in single server mode cannot run Web Edition)

- File system must be NTFS
- Windows SharePoint® Services V3 (WSS) or Microsoft Office SharePoint® Server 2007 (MOSS)
- .NET 2.0 Framework and .NET 3.0 Framework
- Internet Information Services 6.0 or higher running in worker process isolation mode

SQL Server

The SQL Server configuration will obviously be based on your environment and existing investments. For most uses, SQL Server Standard Edition is sufficient, although advanced high availability and analytics requires the Enterprise Edition. For reporting and statistics, the SQL Server 2005 Reporting Services Add-in for Microsoft SharePoint Technologies is available for download from the Microsoft site. If SQL Server 2005 Standard or Enterprise is not available, both WSS and MOSS will install the Windows Internal Database locally. If you wish to install SQL Server separately in a farm environment using SQ Express, you must either install SQLExpress stand-alone or you will have to install WSS or MOSS on that server to install SQLExpress (SQLExpress is also available on the Visual Studio 2005 Installation media).

Development Systems

Developers must have the full environment available when working with SharePoint, testing Master Pages and Page Layouts, and interacting with the SharePoint Object Model. This is critical when debugging, because remote debugging is difficult at best and is not worth the additional effort required to set up. For development and QA testing stand-alone (or virtual) server, the following is suggested:

- Windows Server 2008 or Windows 2003 with at least SP1 (except Web Edition)
- Windows SharePoint® Services V3 (WSS) or Microsoft Office SharePoint® Server 2007 (MOSS)
- NTFS file system required
- .NET 2.0 Framework and SDK, .NET 3.0 Framework and, optionally, the SDKs
- Internet Information Services 6.0 or higher running in worker process isolation mode
- Visual Studio 2005 (Express, Team, or Enterprise)
- SharePoint Designer
- Microsoft Office (at least Word, Excel, and Outlook)

Summary

As you have seen here, installation planning with SharePoint is really a matter of determining which configuration you will use, the type of authentication to set for users, and the hardware and network infrastructure needed to support long-term use. Because SharePoint provides complete flexibility and expansion, planning doesn't have to be exact, but it should account for all the needs in the first implementation. The planning you've done in this chapter is the guide you will use in the next.

CHAPTER 5

Installing SharePoint

As I've mentioned throughout, Windows SharePoint Services (WSS) represents the SharePoint technologies *platform*, whereas Microsoft Office SharePoint Server (MOSS) is just one *application* that runs on that platform. Because installing SharePoint is, in all cases, based on WSS, the steps and process involved in the installation is identical for the core installation - the real differences are the post-install configuration. Of course, when MOSS is installed, some additional background steps are done (copying sample data for example), but this is invisible to the user. As mentioned in Chapter 4, all versions of SharePoint (WSS, MOSS Standard, and MOSS Enterprise) have the same installation steps with two methods of installation. The *Basic* method pre-sets all options (including installing the database) so that a farm cannot be created, whereas the *Advanced* method sets up a farm and everything is configurable.

The Basic option has obvious limitations and, in most cases, is of little use to anyone other than developers and very small organizations. The Advanced option, even on a single server, enables the administrative tools, allowing you to change back-end databases, re-create them if needed, add servers to create a farm, and so on. For either the WSS installation or MOSS, the only notable differences happen after the install is completed and you are configuring SharePoint to use it.

NOTE *Throughout the book, I refer to the SharePoint-installed database as SQLExpress. Microsoft calls the WSS version the Windows Internal Database, whereas in MOSS it is called SQLExpress.*

CAUTION *The Basic install does not allow changing of authentication modes since the applications are provisioned automatically. If you wish to use any method other than Windows Integrated, you must use the Advanced method.*

Installation Issues to Avoid

While system planning for servers and software is a vital part of a SharePoint install, an often overlooked part of the process is making sure that the entire environment (including people, accounts, and access) is actually ready to do the installation. This includes not only having the information you need, but also ensuring that the organization is ready to support it. Installing SharePoint is a relatively painless exercise and many people, particularly IT types, think that if the software installs correctly then everything is all set. This is definitely not the case. Because of the "one click install" nature of software these days, people often overlook details, cutting a corner here or there, just to get it done.

On the organizational front, the most common issue is an inability to overcome fiefdoms. For example, the database group might not want to give other departments access to the server or a "password king" might feel that no one (but he or she) should know account passwords. The problems that these situations can cause are invented and really have no place in business (the cost alone should be enough reason to not fall for these traps).

On the system side, you'll have the typical issues around any software installation, but be *very* aware that SharePoint likes to be done right the first time. While a very stable product, there are a lot of moving parts—and missing a key detail could cause you big problems down the road. A successful install takes good coordination and cooperation (after all, it is a collaboration system right?). While I've seen many kinds of problems, here is a list of some of the major issues to avoid:

- **Physical access** You must have physical access to all servers during installation (both Visual Studio and SharePoint have problems installing remotely).
- **Admin Rights** For the initial system setup, you must have full Administrator rights, as well as the right to add accounts locally (once the installation of the software has begun, you will use a different account, which is specifically set up for that purpose, as shown in Table 5-1).
- **Active Directory** If using Active Directory, get your AD house in order first with regard to security groups, information in profiles, and so on. SharePoint pulls information from AD, so if AD is screwed up, plan on SharePoint replicating the mess.
- **Create individual accounts** You MUST create *all* of the individual accounts required for SharePoint, because these are dedicated background accounts that control how search and sites operate, and each as a specific role. The accounts are mandatory for correct operation so do not cut this corner!
- **Configure before install** Install and configure everything in the actual environment it will be used. Don't change IP addresses, system names, or similar after the install. Instead, set them up as you will be using them right from the start.
- **Verify** Make sure to verify DNS, IP addressing, system names, e-mail server access, AD access, and SQL Server connections before attempting an install.
- **Don't share the database** Sharing the database (or the DB Server for that matter) with any other application (with the exception of SQL Reports) is not a good idea; since SharePoint is completely dependent on the database for everything, any other application can effect performance as well as cause issues in handling restores, backups, and so on.
- **Allow SharePoint to create databases** While the DBA can create databases for SharePoint, it is much better to allow SharePoint to create them. The DBA can lock things down and even change passwords after the installation.
- **Don't skip steps or wing it** Follow the guidelines in the order presented here; the order of the steps have been tested in nearly 100 installations so there's no need to vary from the program.
- **Don't install. . .until you're ready** Finally, if you are not ready, forget about deadlines and push back if you need to. If you breeze through the install, you will pay the price later. If the installation is incorrect, don't try to fix it; uninstall and start over.

Set Up Windows Network and Windows Load Balancing

Before installing SharePoint, you should ensure that you have completely validated the network environment between the front-end web servers, up to and including the setup of Windows Load Balancing. You should also fully test any firewalls that are to be used (particularly important in handling port access).

Testing the Network

To test the network settings, all of the servers must be in the same place that they will be for the post-install configuration. Verify that all systems can ping each other using the default internal IP addresses and by system names in order to verify the local DNS. As previously suggested, add the information to the Host files for each of the servers to guarantee connections. Once you have verified connectivity on the internal network, repeat this process on the front-end network. Once you have verified the network connectivity on both sides, you can go ahead and set up Windows Load Balancing between the front-end servers.

Some strong advice here: once you have verified the network, don't move it (I can't count the number of times I've been called in to resolve the problems caused by this). This goes for DNS and Active Directory settings too! Also, once you have verified Windows Load Balancing, disable it for the duration of the installation and configuration of SharePoint. Attempting to troubleshoot installation issues with Load Balancing active is nearly impossible, and it's not worth the headache or extra time.

Setting Up Load Balancing

Setting up load balancing is a pretty easy procedure but be forewarned that you should have some good network knowledge before attempting it. Incorrect setups can cause problems that can prove very hard to track down. Some advice: if you are unsure, solicit help from a network engineer. To set up load balancing, you'll need to have the following information gathered together before you start:

- The IP address to be used as the front-end IP (outward facing) for the front-end network card
- The subnet mask used for the front-end IP
- The default gateway's IP address for use on the front-end IP
- The Virtual IP address (the load-balanced IP address) used as the entry point into the server farm
- The subnet mask used for the Virtual IP
- The Cluster IP address (virtual IP) that is the entry point to the server farm (this is the IP that must be mapped to a DNS entry)

Now, to set up load balancing on a front-end server (web or application role) do the following on each of the front-end servers (web and application). Be sure to set the Priority correctly for each:

1. Log in to the server with an account that is a member of the local Administrators group.

2. Next, set a static IP address on the front-end network interface card (NIC) as follows:
   a. Select Start | Control Panel | Network Connections.
   b. Right-click on the front-end NIC and select Properties.
   c. In the This connection uses the following items section, select the Internet Protocol (TCP/IP) check box.
   d. Click the Properties button. When the IP Properties Dialog window displays, click the radio button to select Use the following IP address.
   e. In the IP address box, enter the front-end static IP address you are using.
   f. In the Subnet mask box, enter the subnet mask for your network.
   g. In the Default gateway box, enter the IP address of the front-end gateway.
   h. Click the radio button to select Use the following DNS server addresses.
   i. In the Preferred DNS server box, enter the IP address for your primary DNS server.
   j. In the Alternate DNS server box, enter the IP address for your alternate DNS server (if you have one).
   k. Click the Advanced button to open the Advanced TCP/IP Settings window.
   l. Under the IP Address section, click the Add button.
   m. When the Add IP Address window displays, in the IP address box, enter in the virtual IP address.
   n. In the Subnet mask box, enter in the subnet mask if needed.
   o. Click the Add button to close.
   p. Click OK to close the Advanced TCP/IP Settings window.
3. Next, back on the LAN Connection Properties dialog box in the This connection uses the following items section, click the check box next to Network Load Balancing check box to select it.
4. Click the Properties button.
5. Click the Cluster parameters tab.
   a. In the IP address box, enter the cluster IP address.
   b. Enter in the Subnet mask (this must be the same mask as shown by the IPCONFIG/ALL command for this NIC).
   c. Enter the DNS name for this installation in the Full Internet name box (the format is server.domain.com unless using Wins, then it is just the server name).
   d. In the Cluster operation section, click Unicast (if not already checked).
   e. Verify that Allow Remote control check box is not checked.
6. Click the Host Parameters tab.
   a. In the Priority box, enter **1** for the first web then a unique number for each server added.
   b. Under the Dedicated IP Configuration area in the IP address box, enter in the IP address originally assigned to the front-end NIC (be aware that you are assigning two IP addresses to the same front-end NIC).

   c. In the Subnet mask box, enter the subnet mask originally assigned to the front-end NIC.
   d. Leave all other settings the same.
7. Click the Port Rules tab to select it.
   a. Make sure the default rule is selected and click the Edit button.
8. In the Filtering mode section, click to select Multiple host.
9. In the Affinity section, click to select None (this is preferred, however SharePoint can work with affinity if a third-party application requires it).
10. Click the OK button to save and close the edit page.
11. Click OK again to close the Properties window.
12. If, when closing the Properties, you get a message that you must enter the new IP address in the TCP/IP component, you have not completed all of the previous steps. To correct this:
    a. Click the OK button to close the dialog box.
    b. In the Properties window for the front-end NIC, click the General tab.
    c. Click Internet Protocol (TCP), and then click the Properties button.
    d. Click the Advanced button.
    e. Click the IP Settings tab, and then, in the IP Address section, click the Add button.
    f. In the IP Address box, enter the Virtual IP address.
    g. In the Subnet mask box, enter in the appropriate subnet mask.
    h. Click the Add button to save, and then click OK to close the Properties window.
    i. Shut down, restart the server, and then continue.

Testing Load Balancing

After you've completed the settings for the IP addresses on all of the servers, you need to test load balancing to validate all is well. To test LBS, do the following:

1. Log in to a PC or some other system that is not part of the server farm.
2. Open a new command prompt.
3. First, verify the virtual name of the server farm is known by entering:

   ```
   ping <server farm/DNS name>
   ```
4. Verify a response; if you don't get a reply, try using the virtual IP address. If that fails, you have a problem with the IP addresses and you must revisit the setup process from the beginning.
5. Next, verify the actual LBS operation. In the command window, enter in the ping command but include the -t switch (this will cause the ping to repeat until you stop it):

   ```
   ping -t <server farm/DNS name>
   ```
6. While this is still running:
   a. Log in to the first front-end server and select Start | Control Panel | Network Connections.

b. Right-click on the Local Area Connection for the NIC assigned to load balancing and select Disable (ping should continue unabated).
c. Repeat a. and b. on all front-end servers; if the ping stops before the last server, the settings are not correct and you must review the IP/NIC settings starting with the servers that you have not disabled.
d. When you have completed your testing, type Control-C to stop pinging and close the window.

Stopping Load Balancing

Before you install SharePoint, I highly suggest stopping load balancing until you have completed configuration. Because network connectivity is already validated, any issues that may arise later will be easier to identify. The easiest way to stop Windows Load Balancing is from the command line, but you can also use the Network Load Balancing Manager utility (NLBMgr). To stop from the command line:

1. Log in to the first web server and open a command window (Run | cmd | OK).
2. Enter in the command:

```
wlbs stop <server farm/DNS name or IP Address>
```

3. To restart later, simply enter **wlbs start.**

To use the Network Load Balancing Manager utility:

1. Log in to the first web server and select Start | Run. Next, enter in **NLBMgr** and click OK**.**
2. When the NLB Manager window appears, connect to the cluster that you want to stop/start.
3. Under the cluster container, click the host that you want to stop.
4. On the Host menu, point to Control Host, and then click Stop.

Step-by-Step Installation Checklists

Since it's always nice to know the proper order of things to be done when doing an install, I've set up checklists showing what to install and in what order for each Server Role. There are really just three basic configurations for installation: the SQL Server as a standalone box, a SharePoint Server base installation for a server farm (SharePoint servers can assume different roles), and the Single Server setup which is used both for basic installations and development systems. For the OS and related services, the installation is the same for all three. These checklists can be used as a quick reference list to make sure you've covered each task or, even better, can be easily added to a Project Plan (add them in the order shown). At the beginning of each step, I will refer back to the exact task item(s) covered by that step.

NOTE *The particular order of the installation shown here was tested extensively to eliminate backtracking; I suggest you follow the exact order and not skip around.*

As you prepare for the installation, there are a few decisions to be made beforehand:

- If installing WSS, decide whether your organization will be using the Active Directory Account Creation Mode feature (this allows SharePoint to create unique users automatically in Active Directory). If so, you will need to designate an Active Directory Organizational Unit to hold users. The name of this unit should fit in line with your organization and be added during the install process.
- Decide on the type of Authentication that will be used to access SharePoint, Windows Integrated, Forms Based, and so on, as the method may require additional post installation tasks (the various authentication methods and how they are set up is covered in Appendix B).
- Be sure to review Chapter 4 (Installation Planning) to be sure you are prepared to select the correct options during the installation process.

CAUTION *When building front-end servers, make sure that the servers are configured identically. Under load balancing, heavy usage, or a system failure, any front-end server should be capable of assuming additional roles, even replacing a whole server if necessary.*

Active Directory Accounts (Server Farm) Installation Checklist

Active Directory accounts are the first thing needed in setup since the installation account is required before installing any application software.

- Create a new Active Directory Organizational Unit for SharePoint Users (optional for MOSS and required for WSS using auto-account creation)
- Create new domain accounts and group for installation

NOTE *Account setup is the first step for setting up a server farm; for a development server with a stand-alone Active Directory, account setup is done after AD is installed.*

SQL Server (Server Farm) Installation Checklist

SQL Server is the basis for all installations because the Configuration Database creation is the first part of the SharePoint install. Here is the checklist:

- Install the Windows OS (2003 or 2008)
- Run Windows Update to update all service packs and IE
- For Development/QA systems (before reboot):
  - Turn off IE Enhanced Security
  - Reduce system log sizes
  - Turn off shutdown tracker
- Reboot the system (required by the updates)
- Install IIS and IIS-related components (Internet Data Connector, and so on—optional)
- Install .NET Frameworks (2.0 and 3.0)

- Join the existing domain
- Run Windows Update
- Reboot the system
- Log on as SharePoint Installation Account
- Install SQL Server 2005 and Service Packs
- Add SQL Server Logins

NOTE *Installation of advanced SQL Server configurations (such as Clustering) is out of the scope of this book. Refer to the SQL Server documentation regarding clusters, failover, and similar.*

Web/Application Servers (Server Farm) Installation Checklist

Web and application servers are added to the server farm once the SQL Database is available. At the same time, all web servers and application servers are installed before the actual configuration of the software.

- Install the Windows OS (2003 or 2008)
- Run Windows Update to update all service packs and IE
- For Development/QA systems (before reboot):
  - Turn off IE Enhanced Security
  - Reduce system log sizes
  - Turn off shutdown tracker
- Reboot the system (required by the updates)
- Install IIS and IIS-elated components (Internet Data Connector, and so on—optional)
- Configure IIS if required (If you upgrade an existing IIS 5.0 installation to 6.0, you must turn off IIS 5.0 Isolation Mode in IIS after the install.)
- Install .NET Frameworks (2.0 and 3.0)
- For Development/QA systems:
- Install POP3 for Development/QA e-E-mail testing
- Run Windows Update
- Reboot the system
- Log on as SharePoint Installation Account
- Dev/QA only: Install Microsoft Office Applications
- Install WSS/MOSS

Complete Installation (Single Server) Installation Checklist

If you are installing for a Single Server environment, most of the previous tasks are needed—however, the order is varied slightly:

- Install the Windows OS (2003 or 2008)

- Run Windows Update to update all service packs and IE
- For Development/QA systems (before reboot):
  - Turn off IE Enhanced Security
  - Reduce system log sizes
  - Turn off shutdown tracker
- Reboot the system (required by the updates)
- Install IIS and optional IIS-related components (Internet Data Connector and so on)
- Configure IIS if required (If you upgrade an existing IIS 5.0 installation to 6.0, you must turn off IIS 5.0 Isolation Mode in IIS after the install.)
- Install .NET Frameworks (2.0 and 3.0)
- Install Active Directory, DNS, and DHCP or join an existing domain
- Create new Active Directory Organizational Unit for SharePoint Users (Optional)
- Create new domain accounts and group for installation
- For Development/QA systems:
  - Install POP3 for Development/QA e-mail testing
- Run Windows Update
- Logoff/On as SharePoint Installation Account
- Optional: Install SQL Server 2005 and Service Packs
- Add SQL Server Logins
- Dev/QA only: Install Visual Studio (SQL Express may be installed)
- Dev/QA only: Set up Personalization Database
- Dev/QA only: Install Workflow Extensions for Visual Studio
- Dev only: Install SharePoint Extensions for Visual Studio
- Dev/QA only: Install Microsoft Office Applications
- Install WSS/MOSS (if SQL installation skipped the Windows Internal Database will be installed)

Step 1: Defining SharePoint Domain Accounts

Installation tasks covered (check off when completed):

- Create new Active Directory Organizational Unit for SharePoint Users (Optional)
- Create new domain accounts and groups for installation

Project Task These tasks cover the preparation of Active Directory and definition of all of the Domain User accounts needed by SharePoint, its services, and its applications pools. In an existing AD environment, these tasks are first; in a development environment, these are

installed after a development Domain has been created. These tasks must be completed prior to installing any application software in a farm.

Creating Accounts and SharePoint Administrators Group

For SharePoint to operate correctly, a number of accounts must be set up prior to beginning the installation if you are building in an existing domain or added during the installation process if using a Single Server/Domain Controller. In WSS, whether you are using Active Directory Account Creation mode or segmenting SharePoint users, you must also define an Organizational Unit in Active Directory before adding accounts. Note that if you are installing for Development, this task will follow step 5.

SharePoint AD Organizational Unit and Using Account Auto-Creation

When using Active Directory as your authentication mode under WSS, AD is simply the authentication agent handling user logins. In MOSS, it handles not only the authentication but also imports AD profile information to create user profiles in SharePoint. In Active Directory, user accounts are usually added to the standard AD container called Users. For most cases this default is typical, because an intranet, for example, would already have the users and security groups defined. However, there are a few cases when you might define a new container (called an Organizational Unit or OU) to be used by specifically for/by SharePoint:

- **You wish to use Active Directory Account Creation mode in WSS (not available in MOSS).** When accounts are created in WSS, these accounts are created under the OU, which allows them to be easily separated from other accounts on the system. This is typical on hosted WSS systems.
- **You want to segment SharePoint users from the AD structure**. It makes sense to the Active Directory administrator to segment SharePoint users from the rest of the AD structure (this applies to both WSS and MOSS). The only change is that when created, users are added under the OU instead of Users. Note that this is a best practice if you have the luxury of starting out with a new Active Directory.
- **You wish to segment SharePoint users by functional boundaries.** These boundaries include region, language, and so on.

To give you some background, AD uses a database to hold user and group information for the domain based on Lightweight Directory Access Protocol or LDAP, which is specifically designed for fast read access and is vital in supporting thousands of users. LDAP is hierarchy based, much like an organization chart. At the top is the domain name followed by folders used to hold (and keep separated) the definitions of users, printers, computers, and so on. OUs in AD are just another kind of folder. Creating Users and Groups under an OU allows them to be easily segregated from system users and printers, for example, and provides a unique LDAP path that is used for the Active Directory Profile Import process in SharePoint. Having a distinct OU enables importing *only* the SharePoint users and not the entire domain user community. In addition, using multiple OUs enables using multiple SharePoint User Groups to support localization (language support), host multiple sites (such as an intranet and extranet), and more.

Queries against AD are done using the LDAP Path (synonymous with a SQL Select) using either a Common Name (CN) to find a specific item (such as a user, a group, or similar) or an OU (Organizational Unit). You can use the Domain Content (DC) path to find a segregated group of users, groups followed by the path to the domain. For example, on Domain.com the default LDAP path for all standard Users is:

```
LDAP://CN=Users,DC=Domain,DC=com
```

Using this path, the SharePoint Shared Service Provider will import all users, regardless of type. If using a SharePoint OU named SharePointOU, for example, the LDAP path points only to users in that group:

```
LDAP://OU=SharePointOU,DC=Domain,DC=com
```

If multiple OUs are used, this enables access to the appropriate group. For example, if broken down by region, you might define an OU for each region name, such as East, West, or North. A query to return all users in the East region would be similar to:

```
LDAP://OU=SharePointOU,OU=East,DC=Domain,DC=com
```

SharePointAccounts

In a server farm environment, Microsoft recommends using six separate accounts for the installation and operation of SharePoint (I recommend an additional one as well). Though it may seem unnecessary, I advise you to set up and use the correct accounts regardless of the type of installation (even a single server). The accounts, what they are used for, and the rights they need are detailed in Table 5-1 (I'll also list a suggested account name). A few notes on accounts:

- Base setting for all accounts must consist of domain accounts, not be a member of any particular Security Group, and must be a member of the IIS_WPG and WSSAdmin_WPG groups on all servers (except SQL).
- All accounts should be created under the SharePoint OU if used.
- You can't add the VS Developers group until after Visual Studio is installed.
- I don't suggest you use the Install account. Microsoft recommends that this be used for the install only, and then disabled. Since the service account needs to be running everything, I suggest you use that account instead (including the SQL Server install if possible).
- I suggest you create a master administration account, which should be the secondary owner in all sites. This allows back door access to sites if needed.

Setting the Account Password Policies

On a production platform or domain, policies are likely already in place with regard to password length, complexity, and so on, and everyone, including developers will have to conform to the firm's security policies. While this is expected in production, and even QA, having complex passwords in development is a considerable burden and costs time and

Account/Group	Purpose	Group Rights	Domain Rights	SQL Access
SharePoint Server Administrators Group	Security group for Farm Administrators	Domain Users Member of IIS_WPG, WSS_WPG and WSSAdmin_WPG	Impersonate a client after authentication	SQL Rights: Public (read/ exec)
Install Account (SPInstall)	Temporary account used for installing SharePoint (suggested but optional)	Domain Users Domain Admins (suggested) Local Administrator on each front-end server Remote Desktop Users (suggested) VS Developers (Development system only)*	Log on locally Impersonate a client after authentication Log on as a service	SQL Rights: DBCreator Sysadmin (suggested) DB Rights: dbo
Services Account (SPServices)	Used to run the SQL and SharePoint Services (not search)	Domain Users Domain Admins (suggested) Local Administrator on each front-end server Member of IIS_WPG, WSS_WPG and WSSAdmin_WPG	Log on locally Impersonate a client after authentication Log on as a service	SQL Rights: Public DBCreator DB Rights: db_datareader db_datawriter db_ddladmin
Search Service Account (SPSearch)	Runs the SharePoint Search Service	Domain Users Member of IIS_WPG, WSS_WPG and WSSAdmin_WPG Local Administrator on each front-end server (if Search cannot be started)	Impersonate a client after authentication Log on as a service	SQL Rights: Public DB Rights: db_datareader db_datawriter db_ddladmin
Content Account (SPContent)	Used when crawling content for indexing	Domain Users	Must be granted access to any content source to be crawled	SQL Rights: Public DB Rights: db_datareader db_datawriter db_ddladmin

Table 5-1 SharePoint Accounts

Account/Group	Purpose	Group Rights	Domain Rights	SQL Access
Central Admin Application Pool Account (SPCAAppPool)	Run the CA Site Application Pool (in IIS)	Domain Users Member of IIS_WPG, WSS_WPG and WSSAdmin_WPG	Impersonate a client after authentication Log on as a service	SQL Rights: Public DB Rights: WSS_Content_ Application_Pools
Application Pool Account (SPAppPool)	Runs Site Application Pool(s) (in IIS)	Domain Users Member of IIS_WPG, WSS_WPG and WSSAdmin_WPG	Impersonate a client after authentication Log on as a service	SQL Rights: Public DB Rights: WSS_Content_ Application_Pools
Administrator Account (SPAdmin)	Master Administrator account (suggested)	Domain Users Member of IIS_WPG, WSS_WPG and WSSAdmin_WPG Domain Admins (suggested) Remote Desktop Users (suggested) VS Developers (Development system only)*	Impersonate a client after authentication	SQL Rights: Public DBCreator DB Rights: db_datareader db_datawriter db_ddladmin

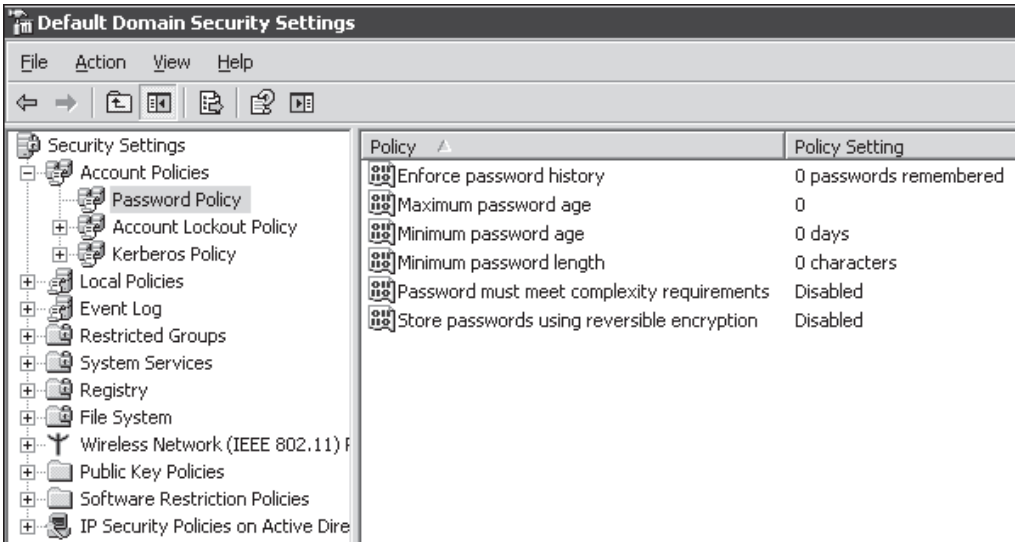
TABLE 5-1 SharePoint Accounts *(continued)*

effort; I recommend setting the Password Policies to permit any password. To reduce the password restrictions, do the following:

1. Log in to the Domain Controller as a Domain Administrator.
2. Open Start | All Programs | Administrative Tools | Domain Security Policy.
3. Click + next to Account Policies to expand it.
4. Click on Password Policies to select it.
5. Modify the rights in the right-hand pane:
   a. Right-click on Enforce password history and select properties. Leave Define this policy checked, change the value from 24 to 0, and click OK.
   b. Right-click on Maximum password age and select properties. Leave Define this policy checked, change the value from 24 to 0, and click OK.
   c. Right-click on Minimum password age and select properties. Leave Define this policy checked, change the value from 24 to 0, and click OK.

d. Right-click on Password must meet complexity requirements and select properties. Leave Define this policy checked, check the box next to Disabled, and click OK.

e. When complete, the final display should look something like what's shown here:



6. Close the Domain Security Policy Settings (changes are not applied until closed). Note that it can take several minutes (even on a single server setup) for the policy to be applied.

Creating the Organizational Unit Admin Group and Users in Active Directory

To create the OU for Groups and Users, perform the following steps. Note that if you are using WSS in Active Directory Account Creation Mode or simply using a designated OU to contain SharePoint Users, you'll need to create it first:

1. Log in to the Domain Controller hosting Active Directory as a Domain Administrator.
2. Open Start | All Programs | Administrative Tools | Active Directory Users and Computers.
3. In the left pane, right-click on the Domain Name, and then select New | Organizational Unit.
4. Enter the desired OU Name and click OK to close.
5. Right-click on the OU Name and select New | Group.
6. In the Group name field, enter the name of the SharePoint Administrators group. Under Group Scope, make sure Global is checked; while under Group Type, make sure Security is checked, and then click OK to create.
7. Again, right-click on the OU Name and select New | User.
8. Enter the user First/Last name and specify the login name for the SharePoint Installation account (for example, First name: SP, Last name: Install, Login: SPInstall).

9. Enter the desired password and confirmation password, uncheck all options, and click Next > then Finish to create the user.
10. Repeat steps 7 through 9 for the Services, Search, Content, Application Pool, and Administrators accounts.
11. Back in AD's right-hand pane, double-click on the SharePoint Administrators Group.
12. On the Properties window, click the Members tab, and then click the Add button.
13. Enter (or search) for the SharePoint Administrator account and click OK to add it.
14. On the Properties window, click the Member Of tab and click the Add button.
15. Enter Domain Users and click OK.
16. Again in the right-hand pane of AD, double-click on the SharePoint Services account.
17. Close out Active Directory Users and Computers.

NOTE *On a development or QA system, you should add several test user accounts to represent various roles in SharePoint; this will help when testing security and SharePoint features.*

Setting Domain Rights

All SharePoint accounts are domain accounts and need specific rights granted for SharePoint to operate correctly. Specifically, those that require the impersonate client right have a direct impact on SharePoint operation. These rights must be set prior to beginning the installation.

1. Log in to the Domain Controller hosting Active Directory as a Domain Administrator.
2. Open Start | All Programs | Administrative Tools | Domain Security Policy.
3. Under Security Settings, click + next to Local Polices to expand it.
4. In the left pane, click on User Rights Assignment to select it (policies should open on right).
5. In the right pane, locate the Allow Log On Locally policy. Then, right-click on it and select Properties.
6. If the policy has not been defined, click the Define this policy settings check box.
7. Click the Add User or Group button. In the pop-up names field, enter the SharePoint installation and services accounts (SPInstall and SPServices). Also, if defining the policy for the first time, also add the Administrators group.
8. Click OK to close when done.
9. In the right pane, locate the Impersonate a client after authentication policy, right-click on it, and then select Properties. Repeat the adding of the accounts/group as done in step 7, and click OK to close when done (it is vital that the services, search, and content accounts have this correct).
10. In the right pane, locate the Logon as a service policy, right-click on it, and select Properties.
11. If the policy has not been defined, *do not* define this policy (click OK to close). If the policy has been defined, repeat adding the accounts/group as done in step 7 and also add the Search, Administration, and Application Pool accounts. Click OK to close when finished.

Step 2: Installing the Operating System

Installation tasks covered (check off when completed):

- Install Windows Server 2003 Standard, Enterprise, or Datacenter
- Run Windows Update to update all service packs and IE
- For Development/QA systems (before reboot):
  - Turn off IE Enhanced Security
  - Reduce system log sizes
  - Turn off shutdown tracker (using Run | MMC | Add Snap-In | Group Policy Editor, select Local Computer | Administrative Templates | System and select the *Shutdown tracker*)
- Reboot the system (required by the updates)

Project Tasks These tasks cover the setup of the actual operating systems on the servers. These tasks should be completed on all servers that will be used in the farm before beginning the software installation. This is also to help ensure that network connectivity and system identification issues are resolved and are separate from software installation issues. Note that the tasks outlined here are based on Windows Server 2003; these settings may be different in Windows Server 2008.

Install the OS (All Server Types)

1. Prepare the server with the Windows Server 2003 Media loaded. Any version of 2003 may be used for any server role except for Windows Server 2003 Web Edition, which can only be used as a front-end web server for MOSS and WSS only in a farm configuration.
2. Begin the OS install and when prompted, be sure to specify the NTFS file system. In addition, if setting up a new system, a disk Quick Format is sufficient. If recycling a previously installed system, use the Full Format option instead.
3. Enter your serial number to begin the installation. During the process, use all default—do not install additional server roles at this time. If available, specify the Network information (such as IP address) when prompted during the installation.
4. After the installation, you will be presented with the installation update screen. Click Update to connect to the Windows Update Site (when first installed, all inbound connections to the server are blocked except access to Windows Update). When prompted, accept the installation of the Active X Controls to run the update, then download and apply the latest patches and service packs. You may be required to specify acceptance of installing a new version of Internet Explorer; accept this and allow the installation to complete. Once finished, you will be required to reboot the system.
5. Upon reboot, you will be presented with the installation update screen again; click the Turn On Automatic Updates link (if desired) and turn on the update.
6. Click Finish to close the installation update screen and enable inbound connections to the server.

Post Install (Servers Joining an Existing Domain)

If you are using an existing domain (the usual server farm environment), you can join the domain once the OS has been installed and updated. When you join the domain, the server will be registered with Active Directory and domain accounts can then be used to log in to the server. When you join, you will be prompted for a Domain Account that has permission to update AD—you should have this information before trying to join.

CAUTION *When Virtual Servers are copied, the System ID is the same for both copies; this confuses DNS since it uses the system ID to identify a system (not its name) uniquely. Download the NewSid utility from Microsoft onto the system copy and run it to generate a new system ID before attempting to join the domain.*

To join an existing domain:

1. Log in to the system as an Administrator.
2. Click Start from the menu, or if the icon is on the desktop, right-click on My Computer and select Properties.
3. Click the Computer Name tab.
4. Click the Change button.
5. Click the Domain radio button, and then enter in the domain name in the box provided (such as domain.com).
6. Click OK to join the domain.
7. If prompted, enter the account and password of the domain account that has permission to access AD. Click OK.
8. Once successful, you will be presented with a message welcoming you to the new domain.
9. Shut down and reboot the system.
10. Log in as the Administrator and make sure that Log On To: shows the domain name (if needed, click the Options button to show the Log On To option).

Add Accounts to Local Administrators group

The SharePoint Installation account must be added to the local administrators group on each of the front-end servers (this is required). To do this:

1. Log in to the server as a local administrator.
2. Open Start | All Programs | Administrative Tools | Computer Management.
3. In the left-hand pane, click + next to System Tools to expand it.
4. Click + next to Local Users and Groups to expand it.
5. Click on the Groups folder to open it.
6. In the right-hand pane, right-click on the Administrators group and select Add to group.
7. When the Administrators Properties window is shown, click the Add button, and then click Advanced.

8. Use the search feature to search for the Installation account. Click to select it, and then click OK. Search again to select the Services, and then click OK to add it. Next, close out each window until you return to Computer Management.

Update the Server Hosts file

SharePoint and SQL Server depend heavily on solid connections in the network. While DNS naming and addressing is usually correct, drops in the connection can cause a variety of errors, even temporary site failure. Having seen this happen in more than half of the larger installations, I recommend adding the servers to all HOSTS files (located in c:\windows\system32\drivers\etc). Edit this file and add the fully qualified names and IP addresses for the servers. For example:

```
# Copyright (c) 1993-1999 Microsoft Corp.
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
…
127.0.0.1         localhost
192.168.2.100     sicgsql.sicg.com
192.168.2.101     sicgweb1.sicg.com
192.168.2.102     sicgweb2.sicg.com
192.168.2.103     sicgindex1.sicg.com
```

Post Install (Development and QA Systems Only)

These post installation steps apply to QA and development environments only, and these settings should not be done on a production server. To perform these steps, you must be logged in on the server as the system administrator; you can do these steps any time after the OS has been installed.

Turn off Internet Explorer Enhanced Security Windows Server 2003 comes with Internet Explorer Enhanced Security installed and activated. This requires that access to any site through Internet Explorer requires the site be trusted; otherwise, it is blocked (including SharePoint sites). In development and QA environments, this can quickly become very cumbersome, so I recommend turning this off unless there is some specific reason your organization needs this. To turn off Internet Explorer Enhanced Security:

1. Open Start | Control Panel.
2. Double-click on Add or Remove Programs.
3. From the left menu, click on Add/Remove Windows Components.
4. Scroll down to Internet Explorer Enhanced Security Configuration and clear the check box.
5. Click Next | to apply the change.
6. Close the Add or Remove Programs window.

Reduce Windows Server 2003 Event Log Sizes By default, the Windows Event Logs are set to various sizes. While this is fine for production, they take up needless space in development and QA. In fact, the smaller logs make debugging easier by keeping only the most recent events. To do this:

1. Open Start | All Programs | Administrative Tools | Event Viewer.

2. For each of the Log types:
   a. Right-click on the name and select Properties.
   b. On the Properties page, change the Maximum log size to 512.
   c. For the When maximum log size is reached option, check Overwrite events as needed.
   d. Click Apply. You will see a pop-up that indicates the log has been reduced but must be cleared; click OK to accept.
   e. Click the Clear Log button when prompted to save before clearing, and then click No.
   f. Click OK to close the properties page.

Disable the Shutdown Event Tracker The Shutdown Event Tracker is used to track why a particular server is being shut down or restarted. By default, this means you must enter a description and click OK each time you need to reboot or shut a server down. In development, rebooting is very common and recording the reason is of little or no use. To speed up the process, I recommend you turn this option off. To turn this option off (it can be turned back on at any time), do the following:

1. Select Start | Run…; type in MMC and click OK.
2. When the Microsoft Management Console opens, select File | Add/Remove Snap-in.
3. From the Add/Remove Snap-In window, make sure the Standalone tab is selected and click the Add… button.
4. From the Add Standalone Snap-in window, scroll down the list to find the Group Policy Editor, click the name to select it, and then click the Add button.
5. On the Select Group Policy window, click the Finish button.
6. Click Close, then OK to return to the MMC. You should see Local Computer Policy shown in the left-hand pane.
7. In the left-hand pane, click + next to Local Computer Policy to expand it.
8. Click + next to Computer Configuration to expand it.
9. Click + next to Administrative Templates to expand it.
10. Click + next to System to expand it.
11. In the right-hand pane, scroll the window to locate Display Shutdown Tracker and double-click the name to open the policy.
12. On the Properties window, make sure the Setting tab is selected.
13. Click the Disabled radio button, and then click OK to close the properties window.
14. Select File | Exit to close the MMC.
15. When you are prompted to save the console settings, click the No button.

Post Install (Remote Access)

For development and virtual servers, it is necessary to turn on Remote Access to the system to enable you to use Remote Desktop. You will also find this useful in Production while

working with the server farm setup since it can be disabled at a later time if needed. To turn on Remote Access:

1. Select Start, and then right-click on My Computer and select Properties.
2. From the System Properties window, select the Remote tab.
3. Click to select the check box next to Allow users to connect remotely to this computer.
4. Click the Select Remote Users button.
5. When the Remote Desktop Users window opens, click the Add button.
6. Enter the name of the Installation account (or other account if you intend to not use this account for connection), click OK, and then click OK again.
7. Click the OK button to close the properties window.

Step 3: Installing Internet Information Services

Installation tasks covered (check off when completed):

- Install IIS and IIS-related components (Internet Data Connector and so on)
- Configure IIS if required

Project Task These tasks cover the preparation of Internet Information Server for those servers that will provide web services or, in the case of SQL Server, provide SQL Reports. These tasks must be completed prior to installing any application software in a Farm setup.

After Windows is installed, you must install Internet Information Services on all of the front-end servers. You can also install IIS on the SQL Server system for SQL Reports and the SQL Browser if desired. Before you begin this, you should have the Windows Server 2003 Installation Media handy.

Install IIS Using the Wizard

1. Open Start | All Programs | Administrative Tools | Configure Your Server Wizard.
2. Click the Add or remove a role link.
3. Click Next > and Next > again. The Wizard will scan the system to determine the current setup.
4. On the Server Role selection page, select Application server (IIS, ASP.NET), and then click Next >.

CAUTION *Do not select SharePoint Services as this will install WSS Version 2.*

5. On the Application Server Options page, ensure that *FrontPage Server Extensions* is not checked and *Enable ASP.NET* is checked, and then click Next >.
6. On the Summary of Selections page, you should see the following listed:
   - Install Internet Information Services (IIS)

- Enable COM+ for remote transactions
- Enable Microsoft Distributed Transaction Coordinator (DTC) for remote access
- Enable ASP.NET

7. Click Next > to begin the installation; when complete, a new page showing "This Server is Now an Application Server" will appear.
8. Click Finish to complete the installation.

All necessary options for SharePoint will be installed using this configuration. However, if you're using POP3 for development, the SMTP component will need to be added (see Adding POP3 for Development later in this section).

Install IIS Using Manual Configuration

1. Open Start | Control Panel | Add/Remove Programs.
2. Click Add/Remove Windows Components.
3. From the Windows Components window, check Application Server, and then click Details…Ensure the following are checked:
   - Application Server Console
   - ASP.NET
   - Enable network COM+ Access
   - Enable network DTC Access
4. Check IIS, and then click Details… Ensure the following are checked:
   - Common Files
   - Internet Information Services Manager
   - SMTP (for front-end web servers)
   - World Wide Web Services
5. Click OK until back at the Windows Component Wizard, and then click Next > to install.
6. Click Finish to complete the installation.

Upgrading from an Existing Windows 2000 Server

If you upgraded the server from a Windows 2000 installation, IIS may be set to run in IIS 5.0 Isolation Mode. To verify the setting:

1. Open Start | All Programs | Administrative Tools | Internet Information Services (IIS) Manager.
2. In the IIS Manager tree, click + next to the server name.
3. Right-click the Web Sites folder and select Properties.
4. Click the Service tab.
5. In the Isolation Mode section, make sure the Run WWW Service in IIS 5.0 isolation mode check box is cleared and click OK to close.

Step 4: Install .NET Frameworks

Installation tasks covered (check off when completed):

- Install .NET Frameworks (2.0 and 3.0)

Project Task These tasks cover the setup of the .NET Framework; since .NET is required for all software installs, these tasks must be completed after the Operating System and IIS have been installed (IIS must be installed first) and before attempting to install any software (regardless of setup type).

You will need to download the latest installation package from the Microsoft site. Using your browser, navigate to the Microsoft Download Center Web Site and download both the .NET 2.0 and .NET 3.0 Redistributable Packages and the Microsoft Windows SDK for .NET 3.0 (the SDK is optional but suggested). Be sure when downloading that you select the appropriate version for your processor (x86 or x64). To install the frameworks:

1. Run Microsoft Windows Update to verify that all updates necessary have been applied (including any service packs for .NET 1.1, etc.). If the option to download the .NET 2.0 Framework is shown, select it—if you want to download it separately, download it from here (this should all be on one line without spaces):

   ```
   http://www.microsoft.com/downloads/details.aspx?
    FamilyID=0856eacb-4362-4b0d-8edd-aab15c5e04f5&displaylang=en
   ```

2. After the .NET 2.0 Framework has been installed, you need to download the .NET 3.0 Framework Redistributable Package and SDK for .NET 3.0. The Framework is available here:

   ```
   http://www.microsoft.com/downloads/details.aspx?
     FamilyID=10cc340b-f857-4a14-83f5-25634c3bf043&DisplayLang=en
   ```

3. Download the SDK from here:

   ```
   http://www.microsoft.com/downloads/details.aspx?
     FamilyId=C2B1E300-F358-4523-B479-F53D234CDCCF&displaylang=en
   ```

4. Once downloaded, double-click on the .NET 3.0 Framework Redistributable Package to install it. Repeat this for the Microsoft Windows SDK for .NET 3.0.

CAUTION *Do not attempt to install any applications before the .NET Frameworks have been installed (some cause you to uninstall and re-install if you don't!).*

Set Up IIS to Use the Correct Framework

Once these installations have been completed, you need to configure IIS to use the .NET 2.0 Framework (by default, it is set to .NET 1.1 by the Windows Server 2003 installation). To set this:

1. Open Start | All Programs | Administrative Tools | Internet Information Services (IIS) Manager.
2. In the IIS Manager tree, click + next to the server name, and then click the Web Server Extensions folder.

3. In the right-hand pane, look for ASP.NET 2.0.50727. If it shows *Prohibited*, click the name to select it, and then click the Allow button.
4. In the IIS Manager tree, click + next to the Web Sites folder to expand it.
5. For each web site:
   a. Right-click the Site and select Properties.
   b. On the Properties window, click the ASP.NET tab.
   c. In the ASP.NET Version drop-down list, select 2.0.50727.
   d. Click OK to close the window.
6. Click File | Exit to close the IIS Manager.

Step 5: Installing AD, DNS, and DHCP (Development only)

Installation tasks covered (check off when completed):

- Install Active Directory, DNS, and DHCP *or* join an existing domain.
- Create new Active Directory Organizational Unit for SharePoint Users (optional).
- Revisit step 1: Create new domain accounts and group for installation.

Project Task These tasks cover the preparation of Active Directory and setup of a Domain (DNS) to use for development purposes. These steps include the creation of the accounts outlined in step 1, and this step must be completed prior to installing any application software (including Visual Studio) in a Development setup.

In a development environment, you can always work off an existing domain, however, there are many cases in which this is either prohibited or may cause conflicts or security issues. In this case, WSS and MOSS will happily work on a Domain Controller setup to run Active Directory. Also, it is always a good idea to try to use as close to a production setup as you can, including the installation of any custom master pages, web components, and similar. This will save you a lot of time and avoid conflicts that might arise moving from development to QA to production. You should also follow the best practice rule that all customization of SharePoint be done using Solutions (see Chapter 17 for more details).

NOTE *A few MOSS features are disabled if running on a domain controller; you cannot, for example, use the remote publishing features. If you can use an existing domain, I'd suggest that to enable full capabilities in the development environment.*

Before you begin, you should make sure that the Windows Server 2003 installation media is loaded (or accessible via the network), as this may be required during the install. To install these services, you must be logged in as the Administrator of the server first, and then follow these steps:

1. Open Start | All Programs | Administrative Tools | Configure Your Server Wizard.
2. Click Next > and Next > again. The Wizard will scan the system to determine the current setup.
3. On the left-hand pane, find Domain Controller (Active Directory), click it to select it, and then click Next >.

4. On the Summary of Selections page, under Summary, it should list Run the Active Directory Installation Wizard to Set Up This Server as a Domain Controller. Click Next >.
5. When the Active Directory Installation window displays, click Next >.
6. On the Operating System Compatibility page, click Next >.
7. On the Domain Controller Type page, make sure the Domain Controller for a new domain is checked, and then click Next >.
8. On the Create New Domain page, make sure the Domain in a new forest is checked, and then click Next >.
9. On the New Domain Name page, enter the full DNS name for the new domain (for example, wssdev.com or wssdev.local), and then click Next >.
10. On the NetBIOS Domain Name page, leave the default value and click Next >.
11. On the Database and Log Folders page, leave the default value and click Next >, unless you need to use a different drive letter.
12. On the Shared System Volume page, unless you need to use a different drive letter, leave the default value and click Next >.
13. On the DNS Registration Diagnostics Page, you will receive a Diagnostic Failed message (this is normal). Click the radio button to select Install, and then configure the DNS server on this computer. Next, set this computer to use this DNS server as its preferred server, and then click Next >.
14. On the Permissions page, leave the Permissions compatible only with Windows 2000 or Windows Server 2003 operating systems checked, and then click Next >.
15. On the Directory Services Restore Mode Administrator Password page, enter your choice of password in the Restore Mode Password and Confirm Password fields, and then click Next >.
16. On the Summary page, click Next >.
17. If your system does not have a set IP address, you will receive a message during the installation that the computer has at least one dynamically assigned IP address; if you have a set IP you wish to use, you can set it at this time, however for most development environments, using a dynamic IP address is fine; simply click OK to close this box, click OK to close the Local Area Connection Properties, and then click OK again when you receive the next warning box.
18. Click Finish when the Active Directory Installation Wizard has completed.
19. Click Restart Now to reboot the system; on reboot, you'll be notified that the server has been upgraded to be a domain controller.

Adding Domain Accounts

When the system has rebooted after installing Active Directory, you must create the accounts and security group as outlined in step 1. Note that in addition to setting the Domain Security Policies, you must also add the SharePoint Installation account Log on locally to the Domain Controller Security Policies.

1. Open Start | All Programs | Administrative Tools | Domain Security Policy.

2. Under Security Settings, click + next to Local Policies to expand it.
3. In the left pane, click on User Rights Assignment to select it.
4. In the right-hand pane, locate the Allow Log On Locally policy, right-click on it, and select Properties.
5. If this policy has not been defined, click the Define these policy settings check box.
6. Click the Add User or Group button.
7. In the Add User or Group Popup names field, enter the Installation account name created previously; if you are defining this policy for the first time, be sure to add Administrator and the Administrators group as well.
8. Click OK to close.

Step 6: Install E-mail Services (Development only)

Installation tasks covered (check off when completed):

For Development/QA systems:

- Install POP3 for Development/QA E-mail testing

Project Task This task covers the setup of a POP3 E-mail, service which permits setup of mailboxes and allows SharePoint to send e-mail through SMTP. This can be used to simulate e-mail features without using an Exchange Server (you cannot, however, use the My Site Features in a POP3 e-mail setup). This should be completed prior to installing SharePoint since e-mail accounts are needed as part of the install.

During development you often want to use the same notification services that will be available in production, including e-mail. If you have an Exchange server available, you will be able to use that for sending and receiving mail. However, in many development environments, using a production Exchange server is not desirable (or allowed). To enable this functionality, you can use the POP3 Services local on the server itself.

CAUTION *Do NOT attempt to install E-mail Services until after you have already installed AD (in single server/development) or have joined a domain. If you do it will have to be uninstalled and re-installed after you have joined.*

To install E-mail Services and SMTP:

1. Select Start | Control Panel | Add/Remove Programs.
2. Click Add/Remove Windows Components.
3. From the Windows Components window, scroll down and check E-mail Services.
4. From the Windows Components window, check Application Server, and then click Details.
5. Check IIS, and then click Details.
6. Ensure that SMTP is checked.
7. Click OK until back at the Windows Component Wizard.
8. Click Next > to install and when done, click Finish.

After the installation is complete, you will have to create a domain within the POP 3 service. You can then create mailboxes for each of the domain accounts you created. To do this:

1. Select Start | All Programs | Administrative Tools | POP3 Service.
2. When the POP3 Service opens, right-click on the Server name in the left-hand pane and then, from the drop-down menu, select New | Domain.
3. Enter the domain name to use (I usually use the one defined in AD). This becomes the second part of the e-mail address, for example, @customdomain.com. Click OK when done.
4. After the domain is created, click on the name in the left pane and you can begin adding mailboxes:
   - Either right-click on the Domain name and then from the drop-down, select New | Mailbox, or click the New Mailbox link in the right-hand pane.
   - On the Add Mailbox dialog box, enter the Mailbox Name, for example, **SPAdmin.**
   - Clear the Create associated user for this mailbox check box, and then click OK to create.
   - The confirmation dialog box will show you the details about the mailbox that was created. Save this information for use in Outlook later.
   - Repeat for as many accounts as you wish; I suggest that you create a mailbox for each of the six (or more) accounts used for SharePoint.
5. When you have completed all accounts, select File | Exit to exit the POP 3 Service.

Step 7: Run Windows Update

Installation tasks covered (check off when completed):

- Run Windows Update

Project Task This task is the first of several runs of Windows Update (the Microsoft Service), used to search for any new security patches or service packs. Even if this has already been run as a part of the installation of the operating system, this task must be run again before installing any application software.

After you have completed the installing IIS and, if necessary, E-mail Services, Active Directory, DNS, and DHCP, you must run Windows Update to get any patches, security releases, and so on before you begin installing either SQL Server or SharePoint. To run the update:

1. Open Internet Explorer and from the Tools menu, select Windows Update; this will start and determine any updates that are required.
2. Click the Express button.
3. If any updates are found, click the *Install These Updates* link, and then confirm it.
4. If necessary, accept any EULA checks.
5. After the patches have been applied, you may be asked to reboot the system. If so, click OK and let the system reboot automatically.

Step 8: Install SQL Server (SQL Server in farm or Development)

Installation tasks covered (check off when completed):

- Install SQL Server 2005 and Service Packs
- Add SQL Server Logins

Project Task These tasks prepare the SQL Server to host SQL Server 2005 and add the logins necessary for SharePoint. These tasks must be completed before attempting to install any other application, and the logins must be active and enabled for use during the SharePoint installation for creation of the SharePoint Configuration Database (and other databases during configuration).

Installing SQL Server 2005

Installing SQL Server can be anything from simple to complex. In most installations, the simple installation is just fine, but to use advanced features like SQL Clustering, the installation has many additional steps. Since it is not possible to reproduce SQL Server's installation here, you should consult the SQL Server installation instructions for advanced features before proceeding.

CAUTION *When specifying accounts, be sure that you use the account format of domain\name and not the e-mail format, specifically on SQL Server services. During the system startup process, the services are started before Active Directory can respond to requests and cannot handle the lookup for the account in time for the service to use it; thus, the service will fail.*

Simple installation procedure:

1. Log in to the SQL Server or Development server as the SharePoint Installation account.
2. Load the SQL Server 2005 installation disc.
3. On the End User Agreement page, click the check box to accept it, and then OK to begin the installation.
4. On the Installing Perquisites page, a list of items it needs to continue will be presented. Click the Install button and when complete, click Next >.
5. Setup will run a system configuration check and then display the Welcome to the Microsoft SQL Server Installation Wizard page. Click Next >.
6. Setup will run a second configuration check, and if there are any issues with the server configuration, setup will notify you. If necessary, stop the install, correct the issues, and restart setup; otherwise, click Next >.
7. Setup will copy the necessary installation files to disk and then present the Registration Information page. Enter your name, company name, and the serial number, and then click Next >.
8. Choose the services you want to have installed (all are suggested, however, Workstation components should not be installed on a production server). Then, click Next >.
9. On the Instance Name window, you can use either the Default Instance (checked by default) or create a Named Instance if desired. If this SQL Server is to be dedicated

to SharePoint, the Default Instance is best. Make your selection, and then click Next >. Warning: If you don't use the default instance or something other than the default ports, you will need to accommodate for it. See Chapter 6 (SQL Settings and Correcting Common Errors).

10. On the Service Account window, leave the Use a Domain User Account box checked, and then enter the SharePoint Services account under the Username, Password, and Domain fields (note that this will grant that account Logon as a service right on the server).
11. Under the Start services at the end of setup section, make sure all items are checked, and then click Next >.
12. On the Authentication Mode window, check Mixed Mode (Windows Authentication and SQL Server Authentication).

NOTE *Windows Integrated Authentication is the default setting. While this is the preferred method, there are many instances in which SharePoint Features and Web Parts must utilize databases on the server. Many network configurations cause a "double hop" in which user connection authentication is lost between the web server and SQL Server. A common work around is to use Mixed Mode so that SQL Authentication can be used if necessary.*

13. Enter the password for the sa account, confirm it, and click Next >.
14. You may receive a popup window that indicates that the sa account requires a strong password; if so, click the OK button and reenter a strong password.
15. On the Collation Settings page, leave the default settings and click Next >.
16. If you selected to install SQL Reports, the Report Server Installation Options window is displayed, and unless you have a reason to use custom settings, leave the default settings and click Next >.
17. On the Error and Usage Report Settings window, check both options (unless you don't wish to participate), and then click Next >.
18. On the Ready to Install window, verify that the components match what you wanted to install and click the Install button.
19. SQL Server setup will then begin installing the components and will notify you if any errors occur.
20. When setup has completed, click Next >.
21. On the Completed SQL Server 2005 Setup window, click Finish.
22. Verify that the SQL Server Agent service is set to start automatically on startup (this is mandatory).

Account Setup in SQL

After SQL is installed, you must add the SharePoint accounts and set the roles and permissions for each before attempting to install SharePoint. To do this:

1. Open Start | All Programs | Microsoft SQL Server 2005 | SQL Server Management Studio.

2. When the Connect to Server window appears, leave the Server type as Database Engine, make sure the server name is the local server and that Authentication is set to Windows Authentication.
3. Click the Connect button.
4. In the Object Explorer, click + next to the security folder to expand it.
5. Right-click the Logins folder and select New Login.
6. In the Login New window, under Select a page, click General.
7. Enter the Login name for the SharePoint Installation account (or you can click Search... to find it using AD).
8. Make sure that the Windows Authentication radio button is selected.
9. Verify that the Default database drop-down list is set to master.
10. Under Select a page, click Server Roles.
11. Click to select both dbcreator and securityadmin roles, and then click the OK button.
12. Under Select a page, click User Mapping.
13. In the right-hand pane in the Map check box column, select master.
14. If SQL Reports is installed, you must also select ReportServer and ReportServerDB as well. This is mandatory for applying the SQL Server service packs.
15. Continue adding the rest of the SharePoint accounts that need access, such as SPServices, SPAdmin, SPSearch, and SPContent, but note that you will have to return here to give the accounts access to site databases created by the installation.
16. Select File | Exit to close out the management studio.

Apply SQL Server Service Pack 1

While in many installations of SQL Server there is a tendency to wait to apply service packs (a good thing), the 2005 Service packs are required for SharePoint. To date, there are two available, but they must be applied in order (Service Pack 2 is not cumulative of Service Pack 1). To apply SQL Server 2005 Service Pack 1 (SP1), you should be logged in as the SharePoint Installation account. Once you have logged in you will need to prepare the system before SP1 can be applied.

Prepare for SP1

1. Obtain the service pack either from DVD Media if available or download it from the Microsoft Download site.
2. All SQL Server services, except for the Database Service (Microsoft SQL Server), must be shut down manually before beginning the install. This includes SQL Analysis, SQL Browser, SQL Reports, etc. (you will not restart them as a reboot is required after install). To do this:
   a. Select Start | All Programs | Administrative Tools | Services.
   b. Scroll through the list of services and for each SQL service installed (except for the *Microsoft SQL Server* service), right-click on the name and select Stop.

Install SP1

To install the Service Pack, simply double-click on the Service Pack 1 .EXE file to begin the installation process. Once the installation package has loaded, simply follow the directions. During the installation, you may encounter a few issues:

- During the startup of the Service Pack install, the package will check authorization against the database (and reports database). This may return an error message that one or more databases could not be validated and gives you the option to connect via different connections. To get around this, verify that the *Microsoft SQL Server* service is running; if it isn't, start it and if it is, stop/restart it and try again.
- During SP1, you may be prompted that there is an installation pending file operation indicating you will have to reboot (this is fine in this instance as we do not have a database established at this time). Simply click Yes to continue the update.
- Another known problem has to do with a file locked message occurring when updating the Database Services, even though the SQL Services are down. This can happen on a clean installation, but there are a few other possible causes for this:
  - The Master Database file has been moved from its original location and the Resource Database is not in the same location.
  - SQL Server has been reinstalled on the same system.

In all cases, you will have to choose Cancel and the remainder of the update process will continue through anyway. The file locked condition will likely repeat again when attempting to update Analysis Services—again, click Cancel and let the process complete. On completion, if the Master Database was moved, you should follow the instructions covered here:

```
http://support.microsoft.com/default.aspx/kb/918695/EN-US/
```

You will also need a utility for this called procexp.exe, which is available for download here:

```
http://www.microsoft.com/technet/sysinternals/Utilities/ProcessExplorer.mspx
```

You can also use Filemon utility to determine the process locking the file available here:

```
http://www.microsoft.com/technet/sysinternals/utilities/Filemon.mspx
```

Once you have resolved the issue, rerun SP1. If either a new install or SQL was reinstalled, allow the process to complete, stop/start the Microsoft SQL Server service, and then re-run SP1.

Post Install of SP1

After SP1 has been applied, reboot the system. On restart, common side effects include SQL Agent, Integration Services, or other SQL service not starting on reboot. If this happens, simply select Administrative Tools | Services, locate the offending service, and start it.

Apply SQL Server Service Pack 2

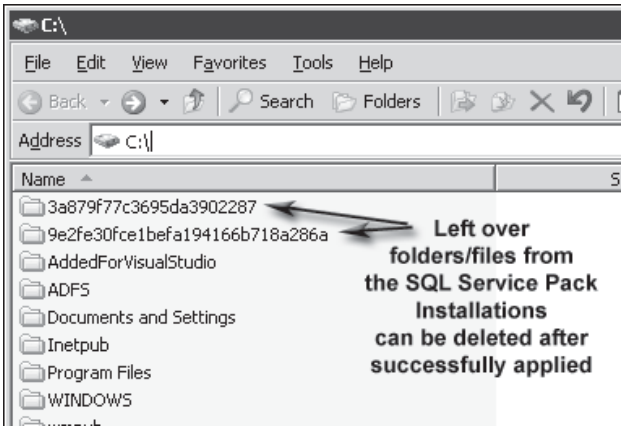
Unlike SP1, SQL Service Pack 2 (SP2) does not require shutting down services as the process handles this operation. However, you can avoid a reboot if you do so and the process will be a bit faster, because the installation will not have to wait while it shuts down the services itself. So, while a reboot is not required, it is suggested to ensure all services have been cycled.

NOTE *During the Authentication step of installing Service Pack 2, a Test Authentication button is shown. Be aware that this test is not accurate and usually reports (erroneously) that there is a login failure using Windows Authentication.*

Upon restart of the SP2 installation, common side effects include SQL Agent, Integration Services, or other SQL services not starting; simply select Start | All Programs | Administrative Tools | Services, locate the offending service, and start it.

Cleanup after Service Pack Installs

After the SQL Server 2005 service packs are installed, it will leave a few remnant folders on the default system drive which you should delete. While some contain small log files, the patches themselves take up 300+ MB of space.



Step 9: Install Visual Studio (Development or QA)

Installation tasks covered (check off when completed):

- Dev/QA only: Install Visual Studio (SQL Express may be installed)
- Dev/QA only: Setup Personalization Database
- Dev/QA only: Install Workflow Extensions for Visual Studio

Project Task These tasks cover the setup of the development environment needed for a development system or to supplement a QA environment. These tasks must be completed after the OS and SQL server is installed (this can precede or follow installing Microsoft Office).

Software Installation

Since there are multiple versions of Visual Studio, installation should be based on the options you have available. Consult the Visual Studio installation guide for setting up the options and features you want based on what is available. Also be aware that SQL Express can also be installed (included with VS); see the post installation instructions to determine if you should install this.

Post Install

After Visual Studio has been installed, it must be set up to enable the personalization feature needed to allow the use of ASP.NET web parts. This requires that a database be created for Visual Studio. It can be used to store this information using either SQL Express (installed with Visual Studio if that option is selected) or any SQL Server instance.

If SQL Express is used, the database needed is created during the installation process. This will automatically create and provision the database when you attempt to use the services that require it (note that the connection string for the database is automatically created in the web.config file). If SQL Server 2000 or SQL Server 2005 is used, you must create this database and configure Visual Studio to use it.

To create the database, follow these steps:

1. Log in to SQL Server and create a blank database instance called AppServiceDB.
2. Provision the new database with the ASP.NET Schemas using the **aspnet_reqsql.exe** utility. This utility can be found in the current .NET framework directory (for .NET 2.0, this is found in c:\windows\Microsoft.NET\Framework\v2.0.50727, though you may have a newer version; use the highest version installed).
3. Double-click on the .exe to run it in GUI mode (this utility also has a command line interface but the GUI mode is easier to configure).
4. From the Welcome to the ASP.NET SQL Server Setup Wizard splash screen, click Next >.
5. On the Select a Setup Option page, make sure the Configure SQL Server for application services is checked, and then click Next >.
6. On the Select the Server and Database page, specify the SQL Server name, and make sure Windows authentication is checked. Then, in the Database drop-down list, select the name of the database you created in step 1.
7. On the Confirm Your Settings page, verify that the entries are correct, and then click Next >.
8. Providing all went well, you will receive a confirmation page that the database was created or modified. Click the Finish button to complete the process.
9. To work with the newly created database, you must set a connection string for Visual Studio to use. The easiest way to accomplish this is by using the ASP.NET IIS Snap-in to set the connection string on a per site or per server basis. To set up the connection string using the Internet Information Services (IIS) Manger, do the following:
   a. Open IIS Manager via Start | All Programs | Administrative Tools | Internet Information Services (IIS) Manager.
   b. From the IIS Manager tree, click + next to the server name to expand it.
   c. Click + next to the Web Sites folder to expand it.
   d. Right-click on the web site you wish to modify and select Properties.
   e. On the properties window, click the ASP.NET tab.
   f. Verify that the ASP.NET Version is set to 2.0.50727 (or higher version); if not, change it in the drop-down list and click the Apply button.

g. To set the connection string for the server, click the Edit Global Configuration… button—to modify just for the site, click the Edit Configuration… button.
h. On the ASP.NET Configuration Settings window, click the General tab.
i. In the Connection String Manager pane, you should see the default connection string (LocalSqlServer); click on the row to select it and click the Edit button, modify the Connection parameter string accordingly, and then click OK to save.
j. If LocalSqlServer is not shown, create a new entry by clicking the Add… button, then enter Name as LocalSqlServer and the connection string to use in the Connection parameter field, and click OK to save.
k. Click OK to close the settings window.
l. Click OK to close the Properties window.
m. Exit IIS Manager via File | Exit.

NOTE *These configuration settings can also be set manually. Consult the Visual Studio installation instructions for the XML needed.*

Adding Workflow Extensions

Workflow Extensions (part of .NET 3.0) are required for SharePoint and are necessary to enable using work flow tools within Visual Studio. The extensions provide you with tools to work directly with the Windows Workflow Foundation and include the ability to create complex workflows in Visual Studio. Assuming you already installed the 3.0 Framework and SDK, you must also download the latest version from the Microsoft site and install the Visual Studio 2005 Extensions for Windows Workflow for .NET Framework 3.0.

Apply the Visual Studio Service Packs 1 & 2

There is nothing remarkable about the service packs, but I do have a few suggestions for working with them:

- Copy the entire service pack installation package to the PC or server you want to update; *do not* try to apply the service pack over the network or use a file share.
- During the installation (which can take quite some time), you may encounter Error 1718, File <file> was rejected by digital signature policy message and the installation will quit. In order to correct this issue, refer to the instructions in Microsoft KB925336, which can be found below (after this fix, reboot and reapply).

```
http://support.microsoft.com/kb/925336
```

Prepare for SharePoint

To work with SharePoint, you will need to download the Visual Studio 2005 Extensions for Windows SharePoint Services 3.0 from the Microsoft Download site to ensure you obtain the latest version. Note that you cannot install these until after WSS/MOSS and Visual Studio have been installed.

Step 10: Install Microsoft Office (Development or QA)

Installation tasks covered (check off when completed):

- Install Microsoft Office Applications

Project Task This task covers the installation of the Microsoft Office Products needed for both development and testing; this task follows installation of the server and Operating System software and should be accomplished before installing any SharePoint software.

While this step can be considered optional, much of SharePoint's functionality is extended through the tight integration with Office applications and in both development and QA, it is highly likely that testing features will need to utilize these applications. While the full suite is suggested, the minimal applications should include Word 2007, Excel 2007, PowerPoint 2007, and Outlook 2007.

Step 11: Installing SharePoint 3.0

Installation tasks covered (check off when completed):

- Install WSS/MOSS

Project Task This task covers the installation of the software packages for both Windows SharePoint Services platform and Microsoft Office SharePoint Server. This task follows installation of the server and operating system software.

The installation Windows SharePoint Services and Microsoft Office SharePoint Server is nearly identical using either the Basic or the Advanced methods. Again, this is because the install process in either WSS or MOSS is installing the platform. In fact, the only noticeable difference in the installation is that MOSS requires you to enter a serial number to install (this is usually printed on the DVD jewel case or available from the MSDN Site for MSDN subscribers); the real differences between the two are not evident until configuring the products for use.

Troubleshooting Note In some installations, it is possible that when attempting to install Office 12 products, you may receive an error that says *The language of this installation package is not supported by your system*. There are two possible fixes for this issue. The first is caused if using an ISO image and the files have been extracted; if using ISO Buster, you must be sure to select the normal extract indicated by the >\>> button to extract all of the files individually; do not extract the files as an ISO. The second is by adding a few keys and a string value to the registry, which can be done by hand or by using a text file. If using a file, create a new text file and enter the following:

```
 [HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Shared Tools\Web Server
Extensions\12.0\ServerLanguage]
"1033"=""
```

Save this file as FixLang.reg, then double-click it and click OK to accept it to the registry.

NOTE *Modifying the Registry can have disastrous results, including bringing a server down; always make a backup before making any changes. If unsure, seek assistance.*

Basic Install Overview (Single Server)

This option installs WSS with all associated services on the same system. The option to change services or manage farm settings is disabled and *cannot* be changed after the fact and you cannot add additional servers to create a server farm. The intent of the single server system is for limited applications since there is no way to expand beyond the capabilities of that system. In most cases, this configuration is ideal for:

- Development systems
- Team-based portals
- Small department portals
- Remote portals

Basic Install: Step by Step

1. Log on to the server using the SharePoint Installation account.
2. Load the media or download the setup.exe file from the Microsoft Download site. Double-click the WSS .exe file (for MOSS this is Setup.cmd) to begin the installation. After the setup program loads, it will check to verify that necessary components are installed (you will be notified if there are any problems). If you are installing Windows SharePoint Services, no serial number is required and the Windows SharePoint Services splash page is displayed. Microsoft Office SharePoint Server will present you with a serial number page. If needed, enter your serial number (including dashes). This will be validated automatically, and when correct, a green check mark will appear.
3. Click the Continue button when ready: this will begin the actual install. When the install starts, the End User Agreement License page is displayed. Assuming that you accept the License agreement, click the I Accept check box, and then click the Continue button. This will open the Installation selection page:



4. From this page, you can choose the type of install you would like—click the Basic button to continue. The setup program will then begin installing the WSS core files

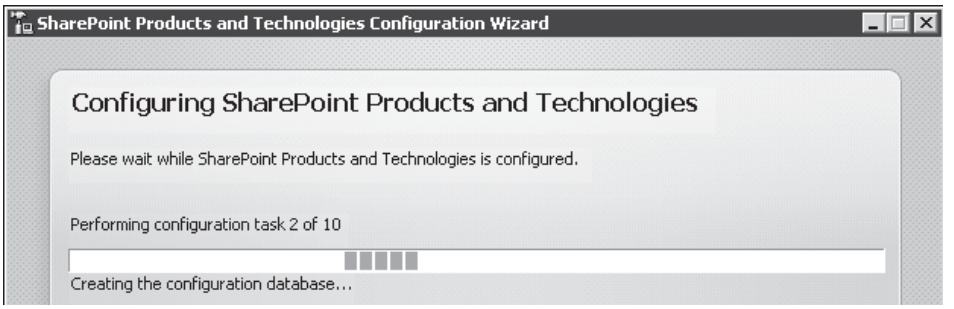
and show you a progress bar. When this is done, the installation complete page is shown.



5. Leave the Run the SharePoint Products and Technologies Configuration Wizard now checked and click the Close button. This will open up the configuration application.



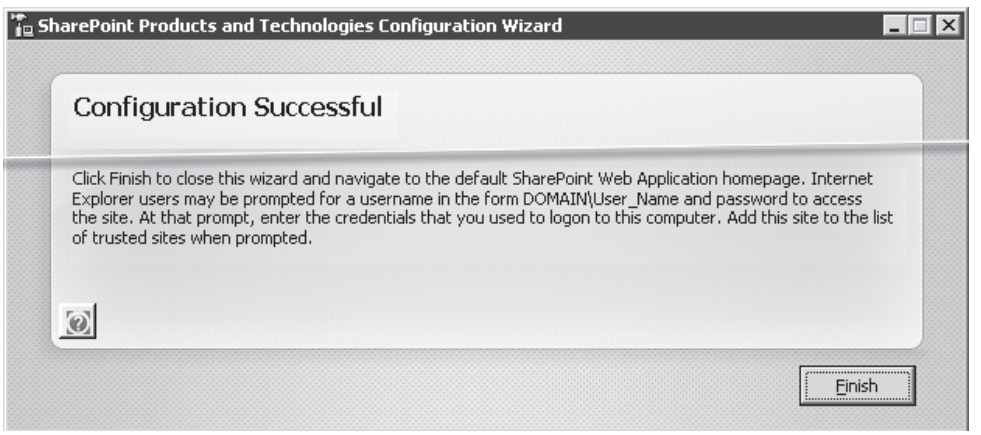
6. Click Next > to continue. A warning message will be displayed indicating that services will be stopped during configuration.
7. Click the Yes button to continue. The configuration application will run the installation process, showing you each step being performed.



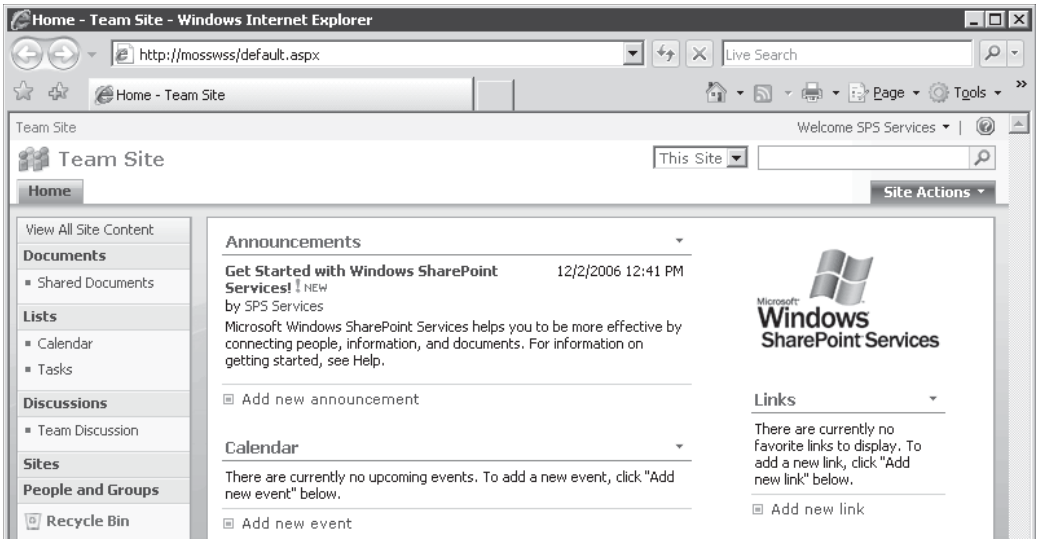
During the installation, the following steps occur:

1. **Install SQL and Initialize Setup** This installs SQL Express and setup files.
2. **Create the Configuration Database** This creates the SharePoint Configuration database in SQL Express and installs the schema.
3. **Install Help** This installs the compiled help files.
4. **Install Resources** This installs assemblies, templates, and images.
5. **Register SharePoint Services** This installs the WSS Web Services.
6. **Register SharePoint Features** This registers the assemblies for features.
7. **Provision and create the Central Administration Site** This creates a new virtual directory in IIS, creates a new application pool, and then installs SharePoint Central Administration on that site.
8. **Create Sample Data.**
9. **Install Application Content Files** This stores the templates on the server, creates a new virtual directory, creates a new default web site, and creates the top level site.
10. **Finalize** This completes the install and restarts services.

When completed, the Configuration Successful window is displayed.



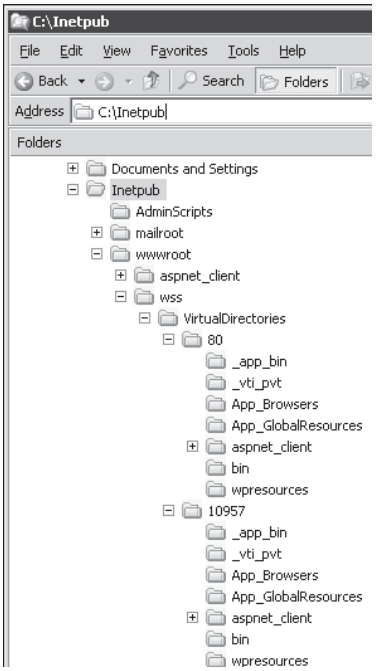
At this point, the databases have been created, Central Administration installed, and the default site has been created at port 80. Click the Finish button to complete the installation and the top-level site will be opened in the browser.



NOTE *It's common to see an error page when first loading the site. This is annoying, but typical on startup (and sometimes after running IISReset). Simply click the Return to Site link to retry; if several attempts are not successful, a reboot may be required.*

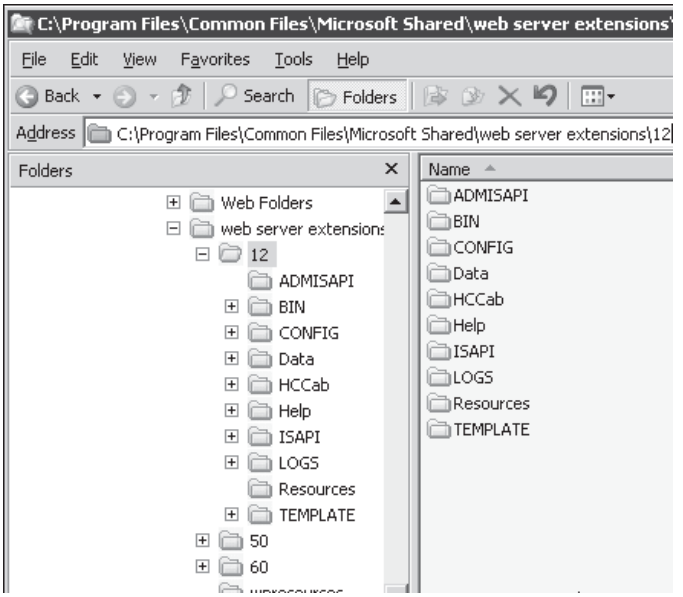
SharePoint File Footprint

After the installation has completed, the installation will have created several directories within the file system. In support of both the Central Administration and default top-level site, you will find the supporting directories have been created in the c:\Inetpub\wwwroot\ directory under a new path of wss\VirtualDirectories.



NOTE *As you might suspect, the port number shown in the previous example as 10957 is randomly generated on a particular server; yours will be different.*

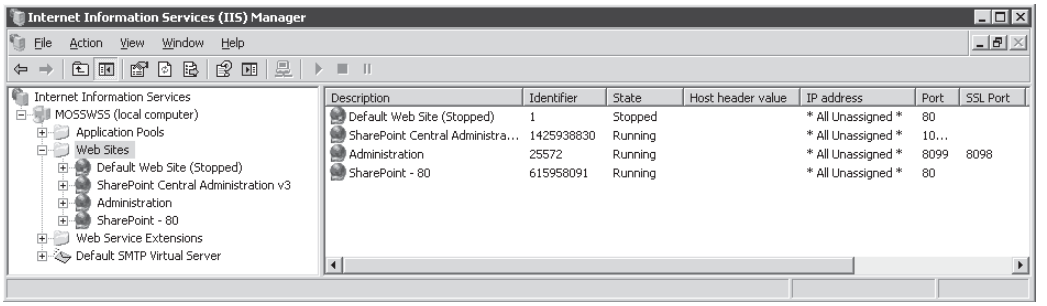
The SharePoint program files, DLLs, templates, XML files, and so on, are installed under the SharePoint Hive that is (by default) C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\ (though you may have installed on a different drive).



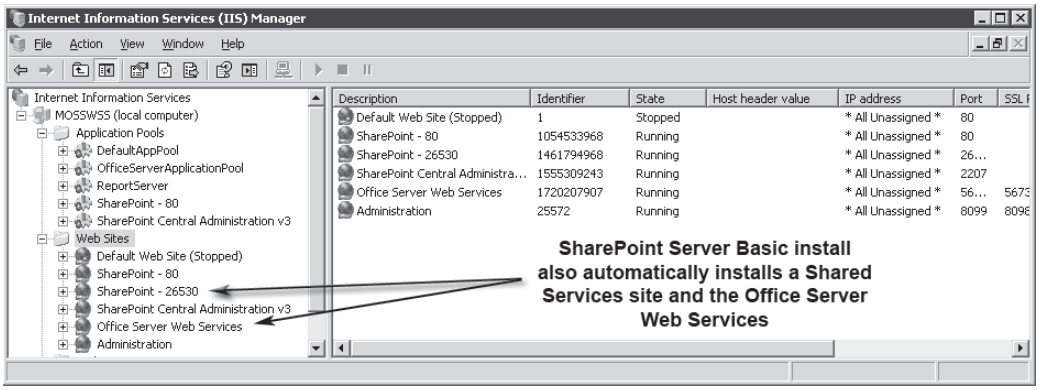
Under the 12 folder is where you will find all of the site definitions, images, features folder, SharePoint Admin tools, and similar.

Post Install Adjustments

If you open the IIS Manager via Start | All Programs | Administrative Tools | Internet Information Services (IIS) Manager, you'll see that the Default Web Site installed with IIS has been stopped and a new site called SharePoint - 80 has been created. In the WSS install, this will look like this:



In the MOSS install, a few additional items—Shared Services and Office Web Services—are installed as well. In that is case, it will look similar to this next screen.

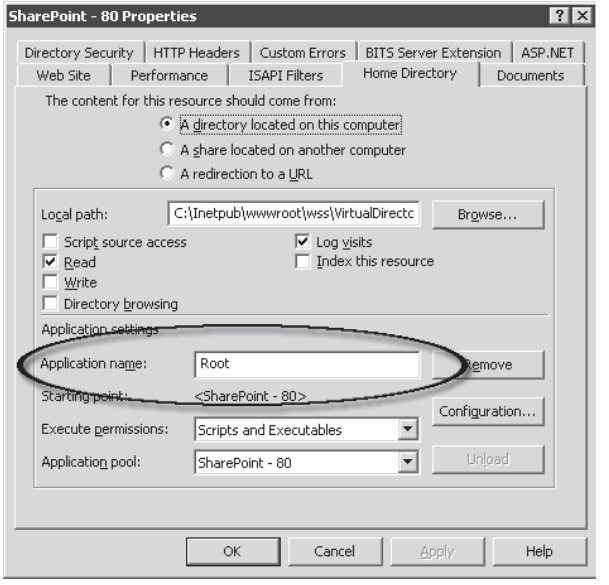


While it is great that everything has been done for us so far, it has also created a few issues:

- The application pools are not assigned to the correct users, which can potentially cause SharePoint to fail to load.
- If SQL Reports is installed, we have just blocked access to the site by disabling the default IIS site.

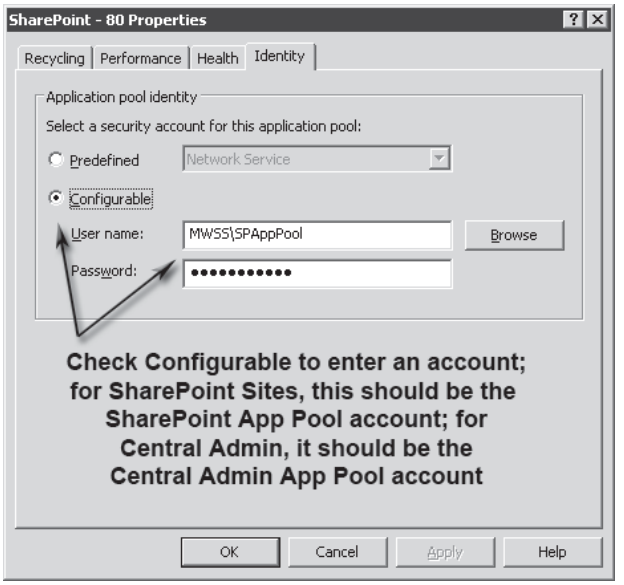
Correct Application Names and Application Pool Identities The first item to correct is the application names assigned to the web sites; by default, the SharePoint sites are all called Root which makes it pretty hard to determine which application is running under which application pool. You want to change each site application name to match the site it refers to, so do the following:

1. Open IIS Manager (if it's not open already).
2. Click + next to Web Sites to expand it.
3. Start with the first site, right-click on the name, and select Properties.
4. When the Properties window opens, click on the Home Directory tab. This displays the Application Name (as far as IIS is concerned).
5. Change the application name shown (for example Root) to the site name (for example, SharePoint - 80) and verify that the Application Pool is set correctly. If it is a SharePoint site, it should be pointing to a matching SharePoint application pool; if this is the Central Administration site, it should be pointing to SharePoint Central Administration V3. Repeat this for all web sites.



Now you need to correct the application pool account settings since they are set to run as Network Service and not the SharePoint accounts they need to be. If you are j WSS, you are only concerned about the SharePoint - 80 site and the SharePoint Central Administration V3 site. If you are installing MOSS, you will also have to set the application pools for the Shared Services and Office Web Services sites. To do this:

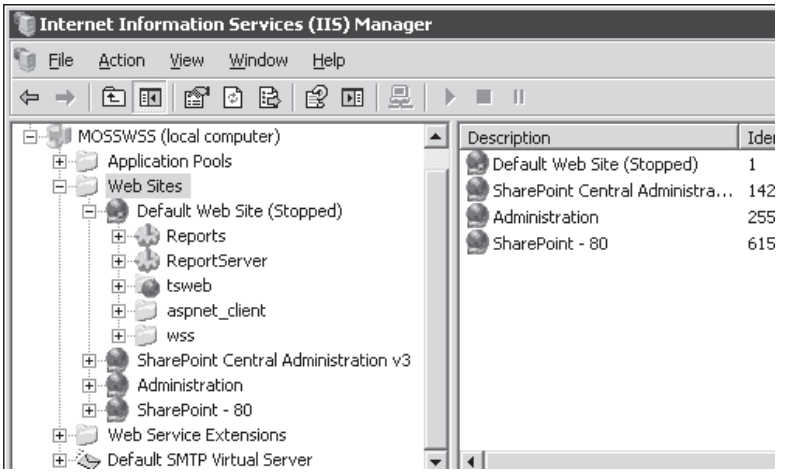
1. In IIS Manager, click + next to the Application Pool to expand it.
2. Right-click on a SharePoint application pool (for example, SharePoint - 80) and select Properties.
3. On the Properties window, click the Identity tab.
4. Click the Configurable radio button and enter in an account and password. For SharePoint sites this should be the SharePoint Application Pool account (SPAppPool); for SharePoint Central Administration, this should be the SharePoint Central Administration Application Pool account (SPCAAppPool), as in the following example:



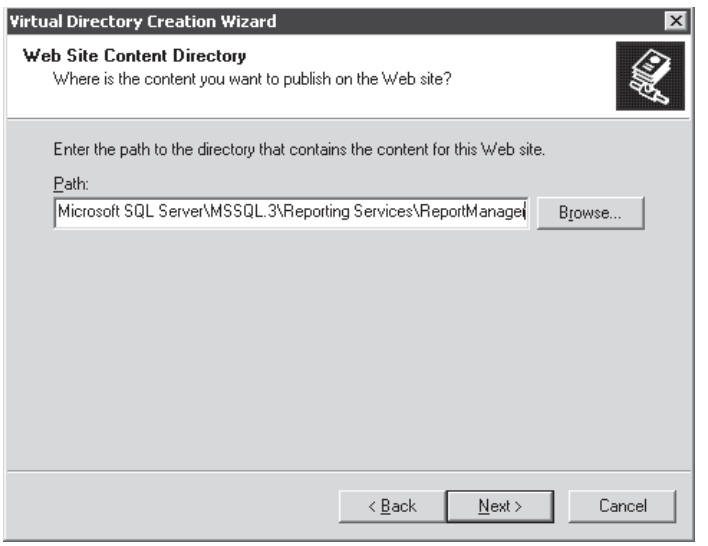
5. Repeat for all Application Pools as necessary. When you are done, leave IIS Manager open and run an IISReset. After the reset has completed, refresh IIS Manager and verify that all Application Pools are running. (If there is a problem, it will be indicated by a small red x on the gear next to the application name. If this occurs, the account set for the identity again is incorrect or does not have "impersonate a client after authentication" rights; reenter the account information and try again.

Correcting SQL Reports after WSS Install After SharePoint has been installed, the virtual directories for SQL Reports that were created under the Default Web Site will now have a problem because the site has been stopped (and replaced) by SharePoint. For SQL Reports to be accessible, the Virtual Directories specified must be redefined under the new default site. Note that this only applies if SQL Reports is installed on the same server. Open the IIS Manager via Start | All Programs | Administrative

Tools | Internet Information Services (IIS) Manager and click + next to the Web Sites folder to expand it.

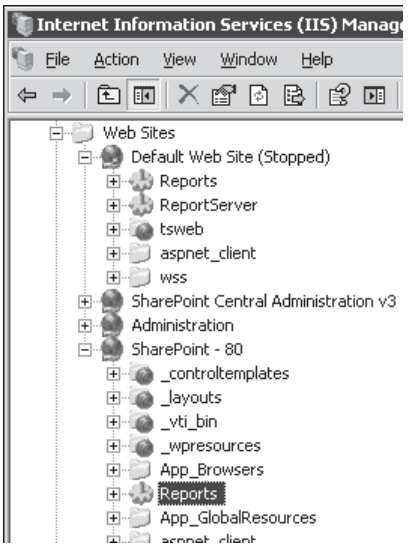


Correcting this is a matter of re-creating these two virtual directories underneath the new site created by SharePoint. First, right-click on the Reports web and select Properties, and then click on the Virtual Directory tab. Select and copy the value of the Local path (paste it into notepad if you like), as you need this path to create a new virtual directory in the site created by SharePoint. Click OK to close the Properties window. In the IIS Management tree, right-click on the SharePoint - 80 site and select New | Virtual Directory. This will open the Virtual Directory Creation Wizard and display the Virtual Directory Alias selection page. Enter the Alias as Reports and click Next >; this will open the Web Site Content Directory page.



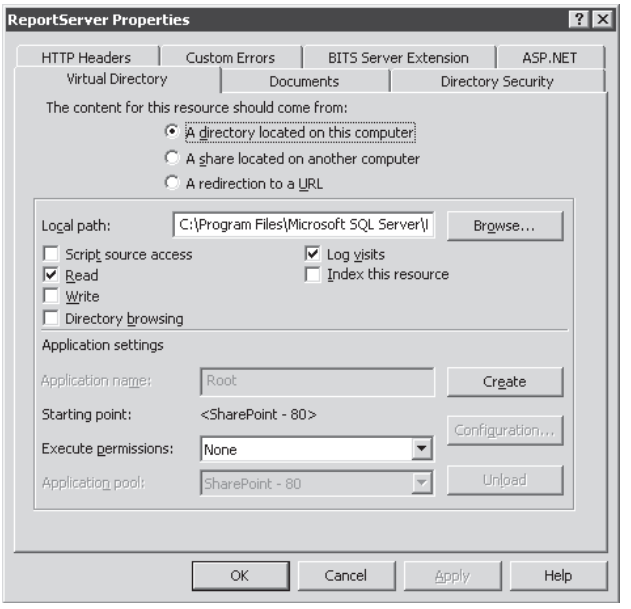
Paste in the path copied from the original virtual directory, and then click Next >. This opens the Virtual Directory Access Permissions page.

Be sure to check Read, Run Scripts (such as ASP) and Execute (such as ISAPI applications or CGI), and then click Next >. The new virtual path will be created under the site as shown.



Repeat the previous process for adding the ReportServer virtual directory; however, when the Virtual Directory Access Permissions page is displayed, you must select *only* Read access.

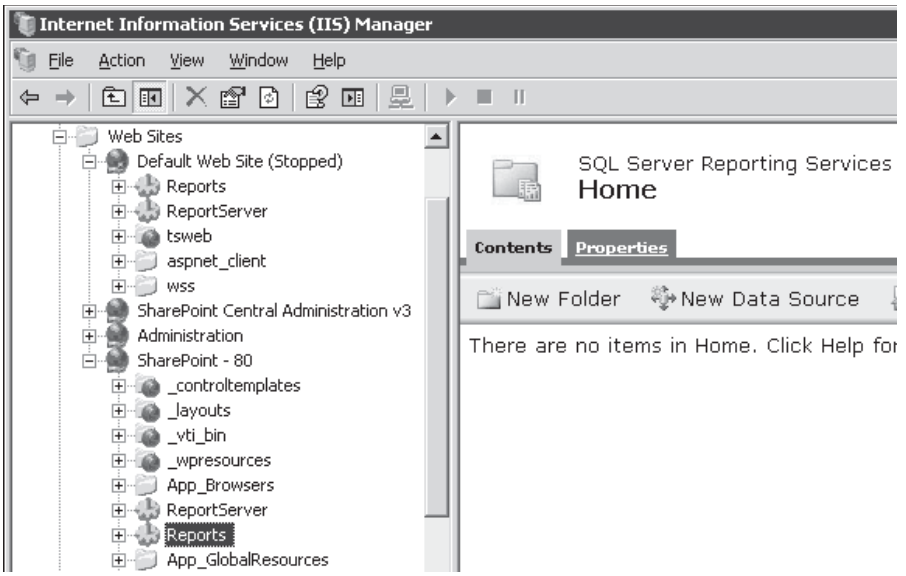
After the virtual directory is created, you'll notice that it was created without an application; you will need to correct this as well as make a few more changes. Right-click on the ReportServer virtual directory and select Properties, and then click the Virtual Directory tab.



Click the Create button to create an application, enter the new Application name as Report Server, and verify that the Application Pool is set to ReportServer. Click OK to close out the Properties page. Then, in the IIS Manage tree, right-click on the Reports virtual direction and select Properties. Click the Virtual Directory tab and in the Application Pool drop-down list selection, change it from DefaultAppPool to ReportServer.

Next, click on the Documents tab. In the Enable default content page box, click to select each of the existing page names (index.htm, default.aspx, and so on), and then click the Remove button. When all have been removed, click the Add… button and when the Add Content Page popup appears, enter the new page name as **Home.aspx**.

Click OK to save this page name, and then click OK again to return to IIS Manager. You should verify that SQL Reports is now accessible by right-clicking on the Reports virtual directory. Next, select Browse. This will open the SQL Reports Home page in the right pane, as shown.



Moving on to SharePoint Configuration

At this stage, you must configure a few additional settings in Central Administration site via Start | All Programs | Administrative Tools | SharePoint 3.0 Central Administration outlined in the Administrator Tasks list. Configuration in its entirety is covered in Chapter 6.

Advanced Install Overview (Server Farm Members)

The Advanced installation option enables you to install WSS/MOSS in server farm mode, where you can selectively configure services and webs across one or more servers. Individual roles, such as a Web Server, can be selected by default. This is the more typical installation, as it applies to all installs and is even best for a Single Server configuration, enabling you to avoid installing SQL Express and the sample site. With the exception of a development system, if there is any potential that the system will need to be expanded later, the Advanced method will give you the most flexibility.

NOTE *I won't cover the steps in great detail in this section as they are exactly the same as shown previously with the Basic install (I will, however, highlight the differences).*

Advanced Install: Step by Step

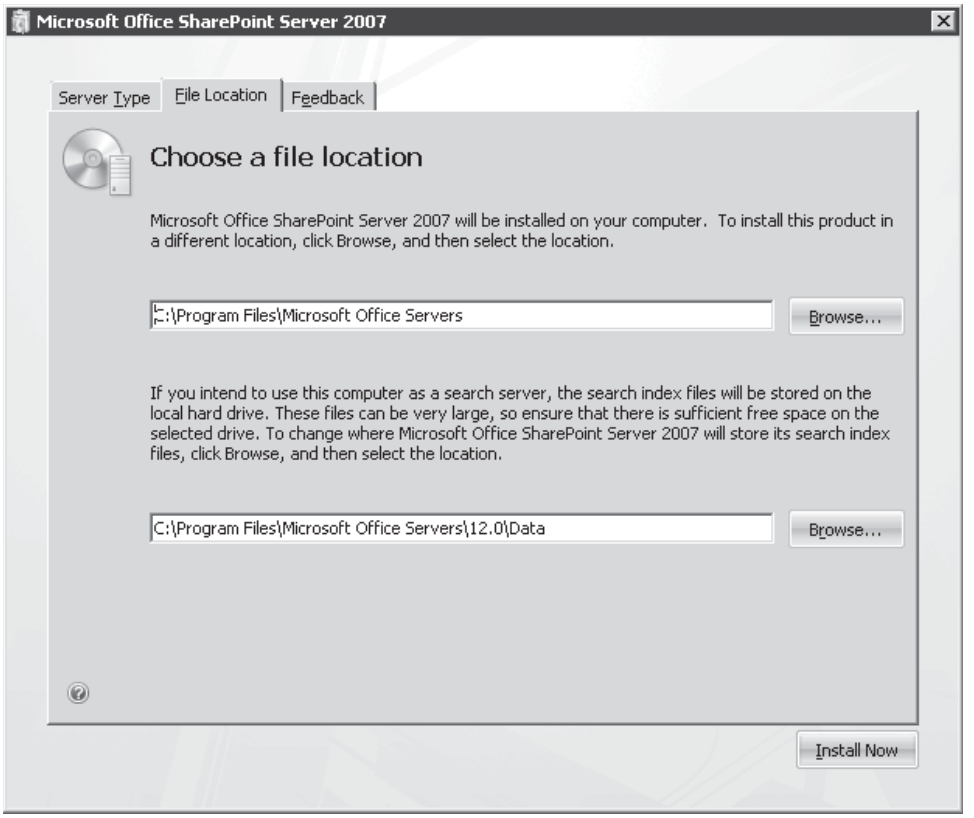
As you would do with the Basic install, load up the media to begin the installation and then double-click on the WSS .exe file (for MOSS this is Setup.cmd) to begin. When the setup begins, the splash screen will be displayed. Click Next > to continue. On the License page, click the I Accept check box and then click Next >, which will display the installation options page.



Click the Advanced button (highlighted) to choose the settings for a SharePoint farm. The setup program will load, and will then display the Server Type page. This is where you can specify the type of server you are installing as well as the data location, and then set the Customer Feedback options. By default, the Standalone option will be set. This is basically exactly the same option as using the Basic install method (you won't be able to set any options). To access the advanced options, you must check either Complete (for MOSS) or Web Front-End option (MOSS and WSS).

The MOSS Complete option installs all of the services on the server including Shared Services, Search, and so on, and is the setting to use for setting up a Single Server environment (this is the installation you should use for setting up Application/Index servers). The Web Server option installs only SharePoint, with the services for a web front end (you can, however, easily add services after the install). Note that the previous image only shows WSS options, and that the Complete option is only available in a MOSS install.

Next, click the Data Location tab and set the location for SharePoint's files.

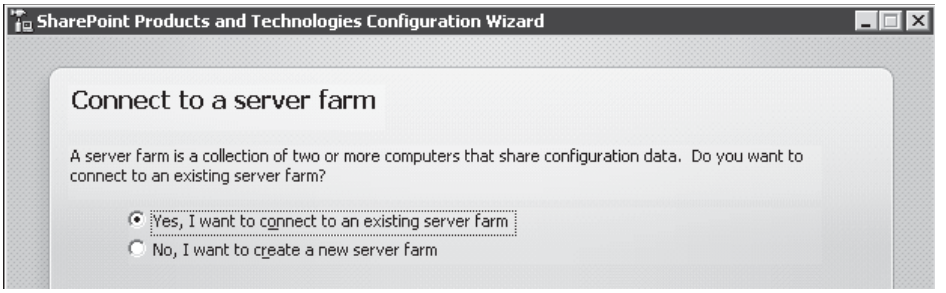


As you can see, the default drive specified is the C: drive (or system drive). If possible, this should be changed, unless you have plenty of space on the main drive (particularly the Index files as they can grow quite large).

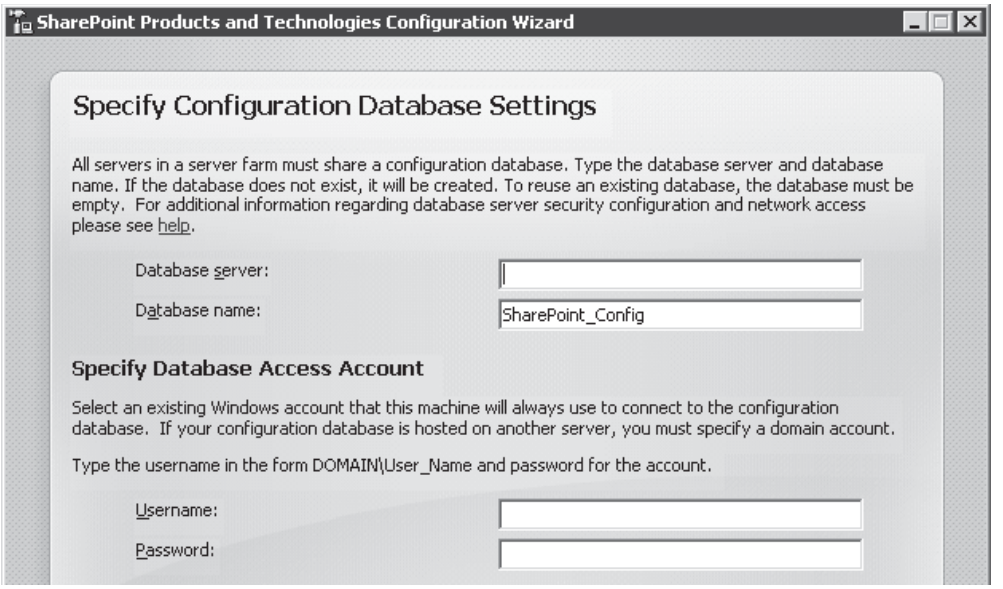
On the last tab, you can select customer feedback options (these are optional). When ready, click the Install Now button to begin the install. This will open the Installation Progress window and show the progress bar. Unlike the Basic Install, the progress records no steps, because only the core SharePoint application components are being installed. When the install is complete, you are presented with the splash page with the option to Run the SharePoint Products and Technologies Configuration Wizard. For several reasons, you may *not* wish to run the configuration steps at this time (to choose this option, uncheck the check box before clicking Close). If you are preparing several servers and have not determined individual roles yet, you can leave the system in this un-configured state and run the configuration wizard later. Note that this is also the ideal state for a disaster recovery spare.

To continue the configuration, leave the check box checked and click the Close button. This will start the Configuration Wizard where you are presented with the splash page.

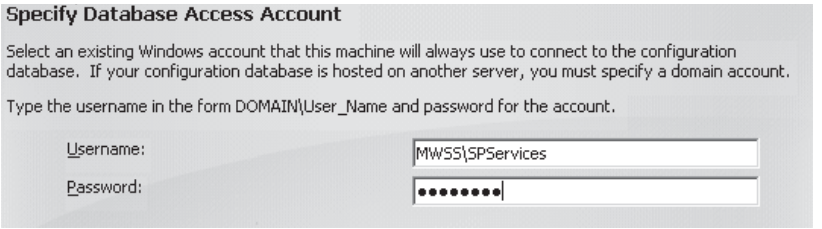
This lets you know that you will need to supply the database server name and the database name for the database settings, as well as the username and password of the database access account that will administer the farm. Click Next > to continue. You will then receive a warning that some services will be stopped during configuration. Click the Yes button to continue; this will display the Connect to a Server Farm page.



If This is the First Server in the Farm To complete the installation of SharePoint for the first server in the farm, the SharePoint Configuration Database (which is needed for the central administration and site management features) must be created. This applies regardless of the role a particular server will play, because the configuration database must be created before any other servers can join the farm. Select the No, I want to create a new server farm option, and then click the Next > button, which opens the Specify Configuration Database Settings page.



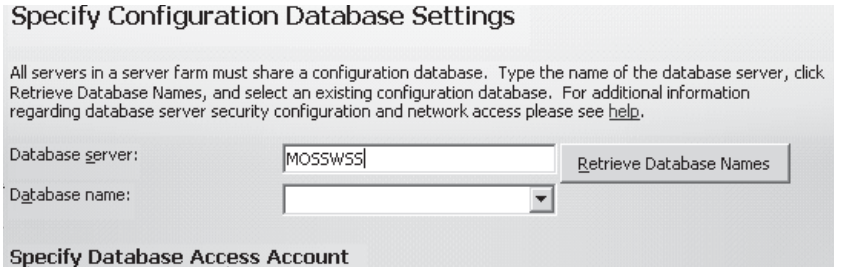
The default name for the configuration database is SharePoint_Config, but in real life, you would change this to make it easier to identify in SQL and to avoid having multiple configuration databases down the road. Specify the name you want, and then specify the Database Access account as the SharePoint Services account (SPService).



NOTE *The account you specify must already be a member of the SQL Logins and granted both Database Creator and Security Administration rights on the database.*

Click Next > to continue. The setup process will connect to the database and if successful, will create the database, make the account you specify a member of the SQL Logins, and grant it Database Creator and Security Administration rights on the database.

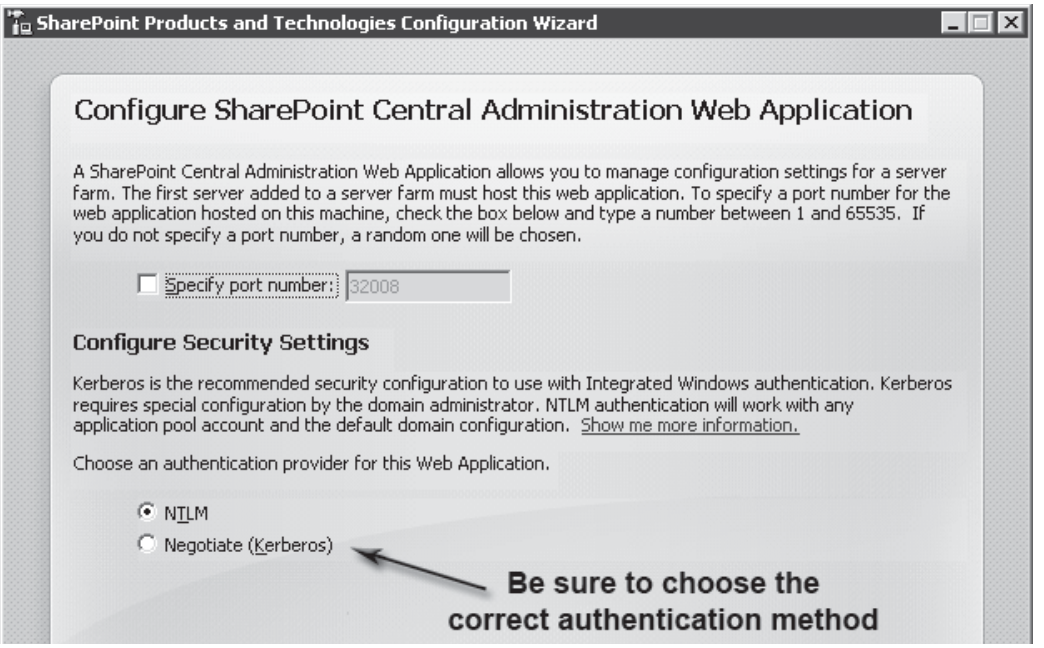
If This is NOT the First Server in the Farm When you are installing a server to join a farm, you will want to select the Yes, I want to connect to an existing server farm box—this will allow you to join an already created configuration database. Make sure that option is checked and then click Next > to display the Specify Configuration Database Settings page, which looks slightly different.



Notice that on this page (since you are connecting) you select the database instead of entering the name. That said, you do have to specify the same account information (the SPService account) as you do when creating the configuration for the first time.

NOTE *As with create, the account specified must already be a member of the SQL Logins and granted both Database Creator and Security Administration rights on the database.*

Make your selections, enter the account and password, and then click Next > to continue. Setup will verify the connection to the database at this time and will let you know if there is any problem; otherwise, it will bring you to the Configure SharePoint Central Administration Web Application page. From this page, you can specify the Port number you wish to have Central Administration running on and then specify the authentication method.



As you can see, a port number is assigned by default and you do not have to change it. It's fine for a first time/development installation! However, if you are joining a farm, you should ensure that this port matches the others in the farm (all servers should match). You must also choose your authentication method: NTLM (Windows Integrated) or Kerberos. For most, NTLM is the correct choice; Kerberos is a special authentication method and requires several extra steps, including setting up Service Principal Names for SharePoint. If using Kerberos, use the link on this page to view up-to-date information on setup or refer to the Microsoft Knowledge Base article KB832769. When you've selected the port and

authentication type, click the Next > button to continue; this will display the Confirmation Page, which will list the settings used for the install.



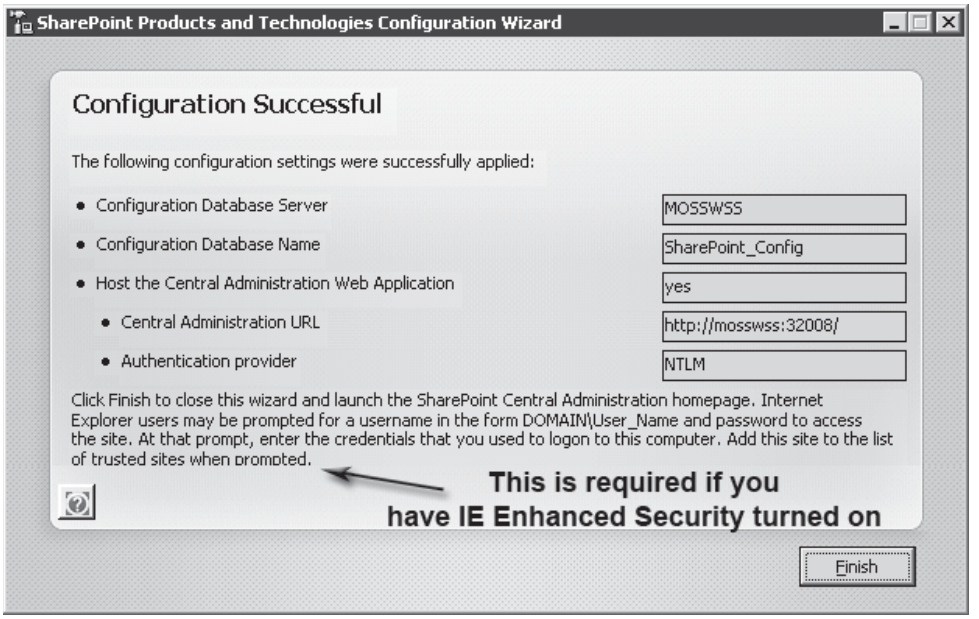
If you are installing MOSS, you'll notice that the Advanced Settings button is disabled but this option is available in WSS. The Advanced Settings button allows you to set the Active Directory settings and then specify a specific Organizational Unit to hold users. To do this, click on the button to display the Advanced Settings Page. Enter the information and click OK to continue. This will return you to the confirmation page, where you will notice that the Active Directory information you specified is shown.

Back on the Confirmation Page, verify that the settings are correct (most importantly, the account information), then click Next > to continue (use the < Back button to go back if you need to change anything). Make a note of the Central Administration URL, as you will need it for post-installation tasks. The configuration process will then show the status of the install. During this time, the following steps occur:

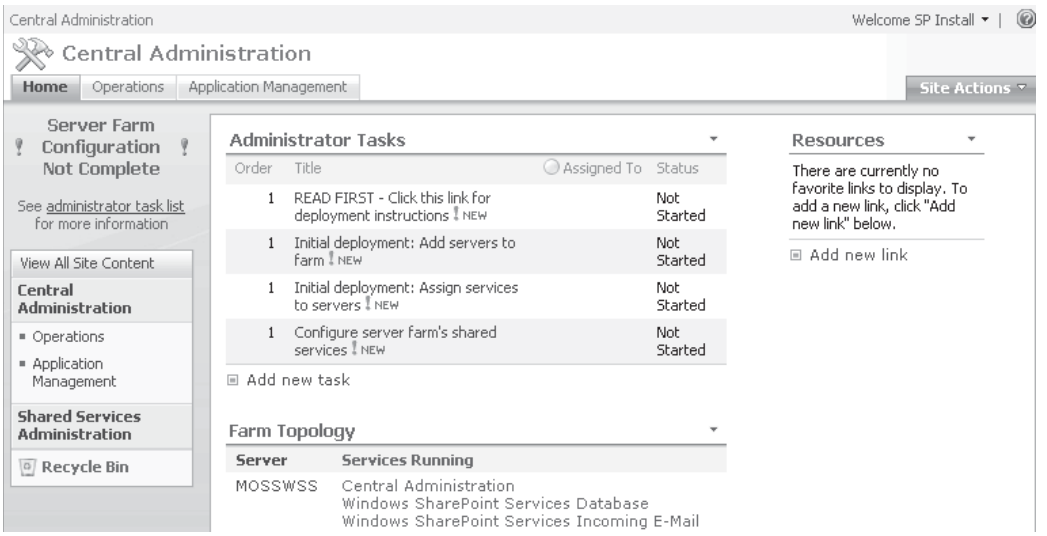
- **Start install** This loads the necessary setup files.
- **Create the Configuration Database** This creates the SharePoint Configuration database and installs the schema.
- **Create Help Collections** This installs the compiled help files.
- **Create Resources** This installs assemblies, templates, and images.
- **Register SharePoint Services** Installs the WSS Web Services (MOSS installs Search services as well).
- **Register SharePoint Features** This registers the assemblies for features (MOSS includes several additional including SSO).
- **Provision and create the Central Administration Site** This creates a new virtual directory in IIS, creates a new application pool, and then installs SharePoint Central Administration on that site.

- **Create Application Content Files** This stores the templates on the server, creates a new virtual directory, creates a new default web site, and then creates the top level site.
- **Finalize** This completes the install and restarts services.

When the process is complete, the Configuration Successful page is displayed with all of the settings used.



To complete the installation, click the Finish button. You will automatically be redirected to the SharePoint Central Administration page (note this will take some time to load).



From here you can begin working on the configuration tasks, such as setting up services. You'll be doing these same tasks in Chapter 6 as well (notice the Administrative Task list has several configuration tasks already listed).

Post Installation (Basic/Advanced)

After the installation, if you are in a QA or Production environment and have the Internet Explorer Enhanced Security option turned on, you will have to configure Internet Explorer for the Central Administration site and configure user authentication for trusted sites. To configure the IE for the Central Administration site and user authentication settings for trusted sites, do the following:

1. Open Internet Explorer. Under the Tools menu, select Options.
2. Click the Security tab in the Select a Web Content zone, in order to specify its security setting box. Click on the Trusted Sites icon, and then click Sites.
3. Clear the Require server verification (https:) for all sites in this zone check box.
4. In the Add this web site to the zone text box, enter in the URL of the SharePoint Central Administration site, and then click the Add button.
5. Click to select the Require server verification (https:) for all sites in this zone check box, and then click Close.
6. Under the Security Level for this zone section, click the Custom Level button.
7. In the Settings box under User Authentication, click Automatic Logon with current username and password, and then click OK to close.
8. Click OK to close the options window.
9. If you do not wish to add the Central Administration site to the Trusted Sites zone, you can optionally add it to the Intranet zone in the same manner shown previously and enable Automatic Logon only in the Intranet Zone.

Settings for Using a Proxy Server

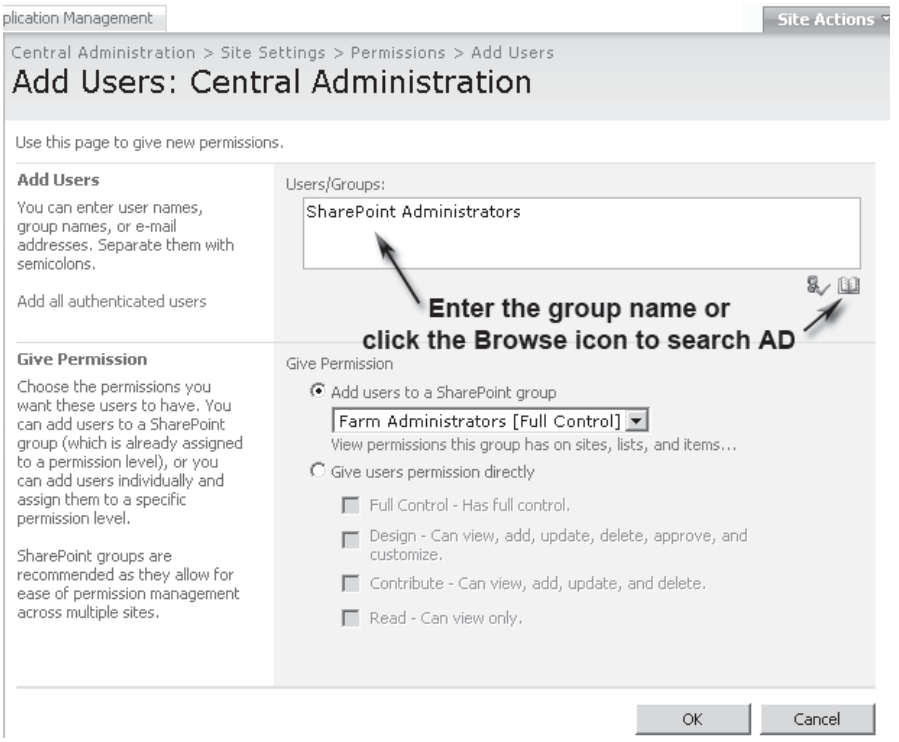
If you have a proxy server, you must configure it to bypass local addresses. To do this:

1. Open Internet Explorer. Then, select Options under the Tools menu.
2. Click the Connections tab then, in the Local Area Network (LAN) Settings section, click LAN Settings.
3. Under the Proxy Server section, select the Bypass proxy server for local addresses check box.
4. Click OK to close the LAN Settings window.
5. Clock OK to close the options window.

Setting the SharePoint Administrator's Group

SharePoint allows any Active Directory Security Group (or user) to be added as a SharePoint Farm Administrator group, although I typically use only one, to be consistent. Users within this group will have access to the server farm settings, shared services administration, and so on. After you have completed installing SharePoint, you must designate the name of the group in SharePoint via Central Administration.

1. Open Central Administration via Start | All Programs | Administrative Tools | SharePoint 3.0 Central Administration.
2. Click the Operations tab.
3. Under the Security section, click the Update farm administrator's group link.
4. On the Enter the People and Groups: Farm Administrators page, click the New button.
5. Enter the name of the SharePoint Administrators group created in Active Directory (you can also click the Browse Icon to search for the group using AD).
6. Under the Add Users to a SharePoint group drop-down list, make sure Farm Administrators [Full Control] is selected:



7. Click OK to save.

Step 12: Install SharePoint Extensions for Visual Studio

Installation tasks covered (check off when completed):

- Dev only: Install SharePoint Extensions for Visual Studio

Project Task This task covers the installation of the SharePoint Extensions for Visual Studio (only required for development); these extensions provide the VS Project types needed for SharePoint Site Definitions, Web Parts, and Lists. This task can be done at any time after Visual Studio is installed.

After you have successfully installed SharePoint, you can install the SharePoint-related extensions and project template mentioned earlier in step 9. If you have not already done so, you must go to the Microsoft site and download Windows SharePoint Services 3.0 for Extensions for Visual Studio. Simply double-click on the MSI file to open and run the install. After the installation, open Visual Studio 2005 and select New | Project, and you should see the SharePoint category listed.

Summary

This chapter covered the entire installation from a software standpoint. If you followed this completely, you are now in a position to actually begin the configuration of SharePoint. You will have also verified all of the presettings so you can be confident the systems are ready. As the next logical step, the following chapter covers the actual configuration on top of the setup done here. Since the configuration is heavily dependent on how you use SharePoint, you should read about the configuration even if not doing an install.

CHAPTER 6

Post-Install Configuration

Welcome to the most important part of the SharePoint installation, namely the configuration of the platform! I say most important, because this is where you will be laying the ground work for SharePoint to support your business needs, from user authentication to setting up services and deciding where they will run. I'm sure you agree that the software installation is a pretty easy task. The configuration, on the other hand, requires a bit more preparation. To complete it, you will need to have the following information available and have made some decisions:

- Accounts to be used for the WSS and MOSS Search Services
- Mail server to be used for incoming e-mail
- Mail server to be used for outgoing e-mail
- Type of Authentication to be used on Sites
- Type of Authentication to be used on Shared Services (My Site)
- Trusted file locations (on the servers) to be used for Excel Calculation Services
- Determining the servers that will run Document Conversion services
- Initial Applications and Sites to be set up (and which Site Templates)

This list is helpful, but if you don't have all of these items right up front, that's ok; I will be explaining the background for each of the steps and explain what's required to make a decision (if needed) as we go. Note that I am presenting each of the tasks here in the order in which you would usually do them, but you may need to skip around in your own setup. There is no real order, but suffice it to say, you should complete all of the tasks before using any sites.

Administrative Tasks

Regardless of the way SharePoint is installed (Basic or Advanced), there are several tasks that must be completed to fully configure the environment for real use. While the Basic install goes all the way up to provisioning the first site, there are still a few settings like e-mail services that

must be assigned. The Advanced install leaves the SharePoint server in an non-configured, raw state with only the SharePoint 3.0 Central Administration site provisioned (this is true for the first server, though servers joining the farm may or may not host a SharePoint 3.0 Central Administration site). For the first server in a farm, additional configuration includes assigning server roles, setting up Shared Services (MOSS), establishing applications, setting up sites, and so on. For subsequent (farm) servers, configuration includes joining the server farm and assuming role(s) and services.

Fortunately, SharePoint gives you quite a bit of help here with the Administrative Task list in SharePoint 3.0 Central Administration. Based on the product (WSS or MOSS) and the type of installation, SharePoint will add the appropriate tasks necessary to complete the installation, including dynamically adding tasks if needed. For example, in a Basic install, SharePoint knows to add a task for assigning the e-mail server; in an Advanced install, it adds assign servers, setup shared services, and so on. If SharePoint detects setup issues that must be dealt with, it creates a task to perform the required action; for example, if SharePoint detects that the SharePoint 3.0 Central Administration application pool is not running on a dedicated account, a task is added to instruct you to correct the issue.

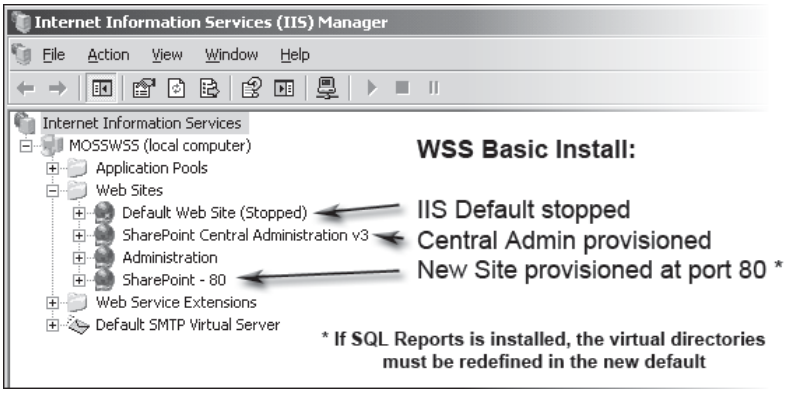
Note that in this chapter, the focus is on the tasks needed to complete the configuration of SharePoint and to get it up and running. As such, I will only cover the administrative tasks from the Administrative Task list here (and the setup of Shared Services), because those other parts of SharePoint 3.0 Central Administration are covered separately. Also, note that for the sake of saving some space since many of the tasks are the same regardless of install type, I'll cover all of the tasks required in the initial configuration of both WSS and MOSS and indicate if the task is product specific.

Where to Start

Once SharePoint has been installed, regardless if WSS or MOSS, it is not ready to turn loose on users because a few settings must be done beforehand. To start, let's review the state of the server when the installation completes.

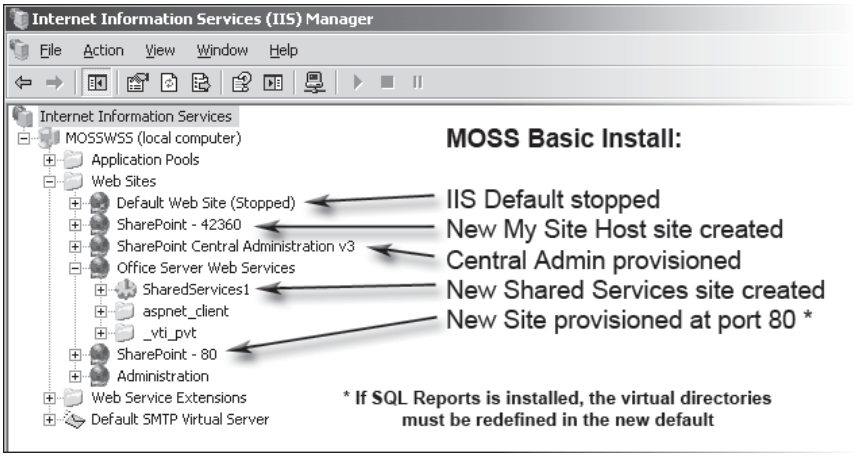
If you used the Basic install method, the following has happened:

- SharePoint has been installed.
- The existing Default website installed with IIS has been disabled.
- The Configuration Database has been created.
- The SharePoint 3.0 Central Administration site has been provisioned with a dedicated application pool. (Keep in mind that the account will be incorrect as you will see in tasks that follow. For reference, the *application pool* is a service that is used to run a website, whereas the account given to this service is who the service will run as.)
- A new Site Collection is created and provisioned at port 80 (also with a dedicated application pool). If it's WSS, a Team Site template is used; if MOSS, the Collaboration Site template is used.
- Administrative Tasks are added to the task list in SharePoint 3.0 Central Administration.



If you are working with a MOSS installation, the following has also been done:

- A new Shared Services Administration site has been provisioned and assigned as the default provider (port number is assigned randomly).
- A new My Site Host site has been provisioned (port number is assigned randomly) and associated to the default Shared Services site:



If you used the Advanced install method, the following has occurred on the first server:

- SharePoint has been installed.
- The existing Default website installed with IIS has been disabled (WSS).
- The Configuration Database has been created.
- The SharePoint 3.0 Central Administration site has been provisioned with a dedicated application pool.
- No applications or sites have been provisioned.

On subsequent servers

- SharePoint has been installed.
- Server has attached to an existing Configuration Database.

NOTE *If you are setting up a server for stand-by or disaster recovery operation, you should leave it in the raw state it's in after the installation. In fact, you may opt to not join an existing configuration; this will allow the server to be configured on the fly to play any role needed.*

Adding the Application Pool Accounts to SQL

To add these accounts, you first have to do a little utility work before you start configuration. This utility work is to add the Application Pool accounts to the SharePoint configuration database. While Microsoft hasn't documented this, there can be an issue with the application pools not starting unless the accounts (SPCAAppPool and SPAppPool) are added as users in SQL. To do this:

1. On the SQL Server, open the SQL Server Management Studio (or Enterprise Manager).
2. Click + next to the Server name, and then the + next to Security to expand them.
3. Right-click on the Logins folder and select New Login.
4. Add the SharePoint 3.0 Central Administration Application Pool account (this is a domain account name in the form of <domain>\<account>).
5. Under User Mappings, add this user to the SharePoint Configuration database.
6. Add the additional right of WSS_Content_Application_Pools.
7. Click OK to save the user.
8. Repeat this for the SharePoint Application Pool account(s).
9. Close out the Management Studio (or Enterprise Manager).

Checking for Install Errors

After you have installed SharePoint, you should review the system Event log to determine if any errors have occurred or are occurring. This is extremely important, as any service failures, login failures, or the like will come back to bite you in the configuration. The most important log to check is the Application Event Log (via Start | All Programs | Administrative Tools | Event Viewer). Any errors you see there must be resolved before trying to configure SharePoint.

SQL Settings and Correcting Common Errors

Specifying an Instance and or Port Number for SQL When specifying the database name, you must use the full instance name and port (if different than the default), for example:

- **MySQLServer** This specifies a single server (no instance) using default ports.
- **MySQLServer\InstanceName** This specifies a single SQL instance using default ports.
- **MySQLServer\InstanceName, 1450** This specifies a single SQL instance using port 1450.

Note that when you do use different port(s), the server is usually not aware of this. To enable this, open Administrative Tools | Data Sources, and when it opens, click the Next button. On the next page, click Client Configuration, disable auto-discovery, and then specify the correct port. Alternately, you can install the SQL Client Tools to set this (though many installations will not permit this).

Changing SQL Servers Use the STSAdm command for this (see Appendix A under renameserver).

Using Already-Created Databases Follow the Microsoft TechNet article that explains this in detail available from: http://technet2.microsoft.com/windowsserver/WSS/en/library/c7647e52-2178-4d3d-9376-84b2c9a35a1e1033.mspx?mfr=true.

Fix: Cannot Connect to SQL Server or Central Administration Won't Start The most common errors in a SharePoint install occur when you are trying to connect to SQL to perform the installation or when Central Administration and/or the Shared Services sites cannot be opened. This is usually due to the application pool account you are using. Errors here can include services failing to start, SQL Server login failures, and sites not opening. The fix usually lies in one of the following solutions:

- Validate that all SharePoint Services are running as the correct SharePoint Service account and that the account has Run as a service and Impersonate a client after authentication rights.
- The SQL Instance Name or Port is incorrect (try the ODBC connection method shown previously).
- Validate that the SharePoint Install and SharePoint Services account have the correct access in SQL Server. Often the account is added but does not have Read and Execute on the Stored Procedures in the database (also true for the Application Pool, Search, and Content Accounts).
- If in a server farm, shut down all servers, bring up SQL Server first, login and verify SQL is running, and then bring up the first web server and then all others (this corrects a situation where SQL Server is not ready when the SharePoint server makes its first request—doing this one time often is a permanent fix).
- The SQL server itself is not known or DNS is not working correctly; try pinging the SQL Server and if necessary, add an entry to the Server HOSTS file.

Fix: SQL Reports No Longer Accessible Another common error situation occurs when you install SharePoint on the same server as Microsoft SQL Reports:

- The Default website is stopped so the Reports web is also stopped; this may result in logging startup errors. To correct, review the Installation section regarding moving the Virtual Directories.
- You may see SQL Reports logging errors in the Application Event Log, specifically telling you that it cannot connect to the database and repeats roughly every minute. This error happens because the account specified in SQL Reports does not have access to the Report Server database. This is usually because the SharePoint Services account is running the actual SQL Reports Service, but the account used defaults to Local System. This must be corrected by opening the SQL Reports Configuration,

and then on the Database Setup tab, the correct account and connection method are specified. More information on this issue can be found at http://msdn2.microsoft.com/en-us/library/aa337324.aspx.

CAUTION *SQL Reports is installed with its own database and runs in native mode. Configuring SQL Reports for SharePoint Integration requires selecting a SharePoint Site (content) Database and updating it to load the SQL Reports schema. However, verify your installation first and do not configure SQL Reports until the SharePoint installation and configuration has been completed! See the SQL Reports Installation documentation to set up the integration.*

Where to Now?

So where to go from here? If you are new to all of this, you should be sure to read the overviews on the Command-Line tool and SharePoint 3.0 Central Administration. If you are ready to begin setting a single server setup or installation of the first server in the farm, you should now go and install the rest of the servers and have them join the farm. After that you can begin (on the first server) the Post-Install Administrative tasks (using the Administrative Task list in SharePoint 3.0 Central Administration). If you are installing MOSS, all the base server services (WSS Help Search and so on) for all servers in farm should be started before you begin the setup of Shared Services.

If you are adding the server to join an existing farm, you should review the process at the end of this chapter and review "Joining an Existing Farm" and "Starting Windows Load Balancing" before starting configuration.

Overview of the SharePoint Administration Command-Line Tool

While you will spend most of your time in the SharePoint 3.0 Central Administration site when it comes to working with applications and sites, you will also need to use the SharePoint Administration Command-line tool on occasion. This tool is called STSAdm and is located in the SharePoint Hive Bin directory (c:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\bin). If you read the overview at the start of this book, you might have guessed that the name is a holdover from SharePoint Team Services.

With STSAdm, you can do basically anything with SharePoint that you can do with the user interface and even more (such as deleting a configuration database). In fact, if you are a developer, you'll find that you'll use this tool much more than the UI, particularly when you are building features, solutions, and web parts. Of course, nearly everyone will find that there are some things that the Command-Line tool can do much better; for example, creating a batch file to pre-build 20 sites takes but a few minutes, whereas building 20 sites using the user interface would take quite a while longer.

I'm introducing STSAdm at this point as it is a vital tool for working with SharePoint and you should be familiar with what it can do. Also be aware that most of the configuration you will do when using UI is actually using this tool behind the scenes.

The command format for this tool is always

```
STSAdm -o <operation/command name>
```

For example, to see a list of the solutions that are installed, the command would be

```
STSAdm - o enumsolutions
```

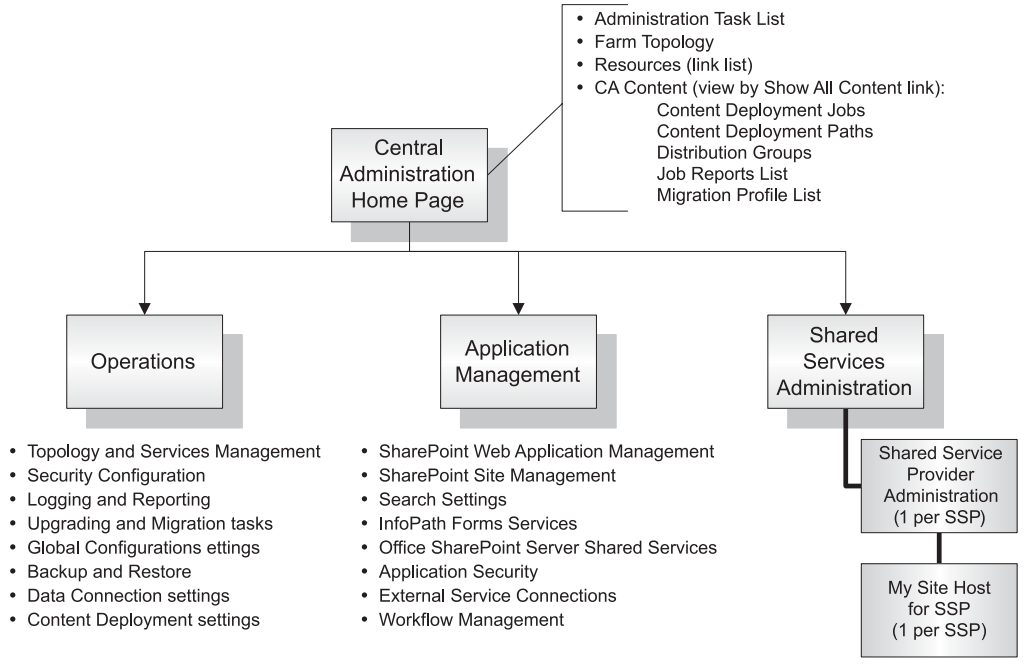
The command list for this tool is quite extensive. Refer to Appendix A for the full description of the STSAdm commands as well as how you can extend STSAdm to include your own commands.

Overview of SharePoint 3.0 Central Administration

The SharePoint 3.0 Central Administration site is the central management site for services on services, the farm topology, general settings, and job management. In a Single Server setup, the server hosting SharePoint is where the SharePoint 3.0 Central Administration site will be provisioned. In a farm, the first server will host the SharePoint 3.0 Central Administration site by default. Servers that join the farm have the option of hosting this site or simply pointing to the first server site. Regardless of how it is configured, the SharePoint 3.0 Central Administration settings are applied to all servers in the farm.

As an FYI, when a farm server is not hosting SharePoint 3.0 Central Administration the link to it on the local server (in Administrative Tools) will not work. This is because the local SharePoint 3.0 Central Administration site will be stopped, and in order to access SharePoint 3.0 Central Administration from these servers, you should use the correct URL to the hosting server. Also note that if you do start SharePoint 3.0 Central Administration on the local server, you can only see the server itself. You cannot view or change other servers in the farm.

The SharePoint 3.0 Central Administration site is nicely organized into three sections: Operations, Application Management, and Shared Services management. An overview of the content of this site is shown:



It is from SharePoint 3.0 Central Administration that everything global is managed. The Operations area covers server management through backup and restore, Application Management manages applications and sites collections, and Shared Services is where you

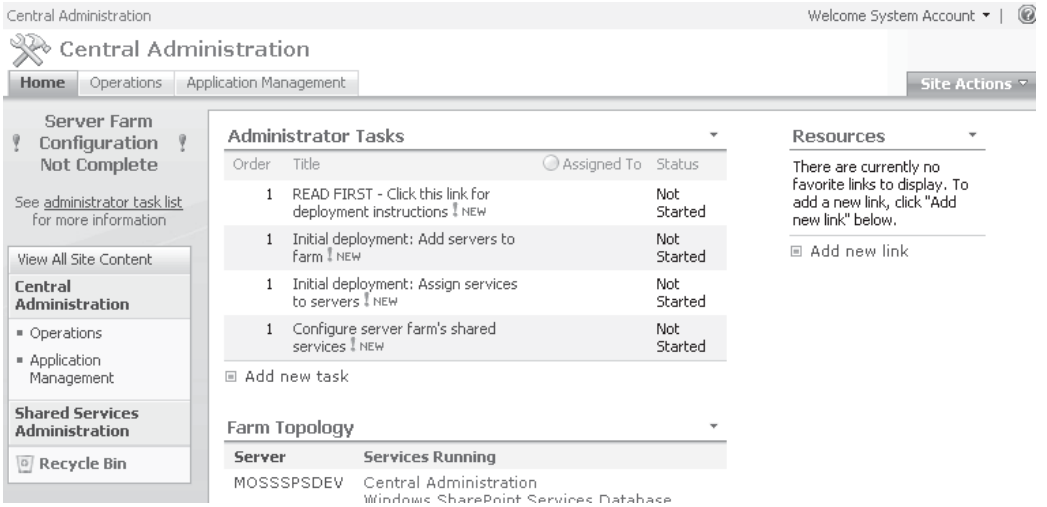
can create MOSS Shared Service Providers, manage User Profiles, and control the My Sites Host (a single site that hosts all a user's My Sites independently of any particular site). The SharePoint 3.0 Central Administration site is also home to the Administrative Task List, which you will begin working with next. This dynamic task list is designed to help you complete your configuration successfully.

SharePoint 3.0 Central Administration is accessible either from Start | All Programs | Administrative Tools | SharePoint 3.0 Central Administration or Start | All Programs | Microsoft Office Server | SharePoint 3.0 Central Administration on a server hosting it.

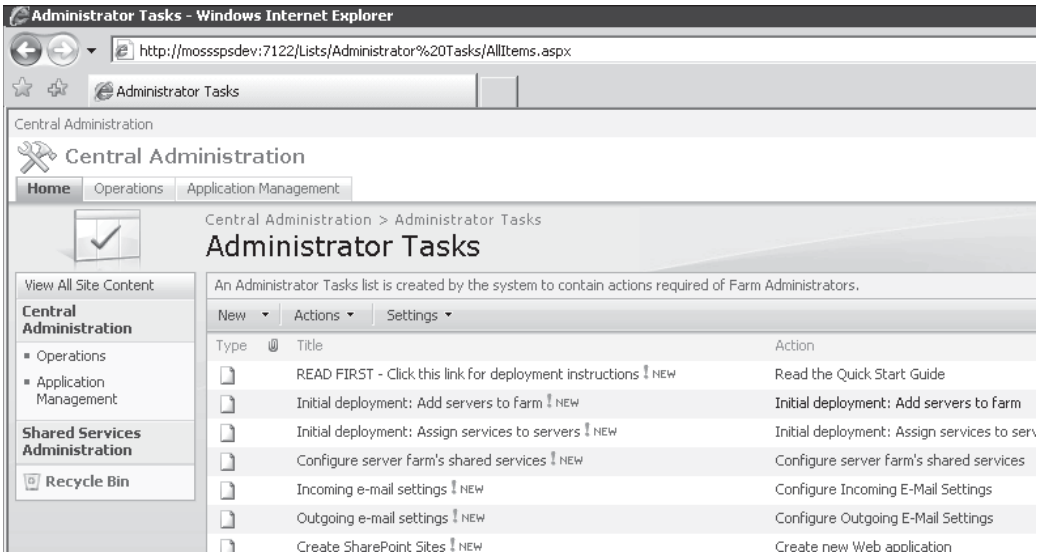
Post Install Administrative Tasks

If you are ready to start the configuration process you'll need to login to the server using the account you used for the installation (or I suggest you use the SPAdmin account). To perform the configuration, you must have full privileges in SharePoint, be a member of the Local Administrators group on each front-end server, and be a domain administrator with administrative access to Active Directory and SQL Server.

Once on the server, select Start | All Programs | Administrative Tools | SharePoint 3.0 Central Administration. This will display the SharePoint 3.0 Central Administration site:



If you used the Basic install, when you first open SharePoint 3.0 Central Administration you will simply see the Administrative Task list. If you used Advanced (and this is the first server) you will also see a warning message indicating that the Server Farm Configuration is not complete (this is because not all of the required server roles have been assigned yet). To get a better view of the administrative tasks, click either the title of the task list or click the link in the left hand navigation. This will display the current list of tasks (this list is from a MOSS installation; tasks vary by product and services installed):



Let's start out with the Read Me task—view this by clicking on the task title to see the details.

This first task gives you an overview of what needs to be done to complete the installation and it also tells you about the Quick Start Guide. If you click on the Action link, this will open the guide, which provides a very general overview of installing SharePoint and some of the deployment scenarios. It also includes some links to other resources that are useful in the install. To return, you must hit the Back button or link on your browser (if you close the window, it will close Central Admin).

If the Basic install was used, there is only a little to do administratively as almost everything (including deploying the first site collection), has been done for you. If the Advanced method was used, there a few more upfront tasks required to complete the install. The Basic install tasks are as follows:

- Decide on incoming e-mail settings
- Decide on outgoing e-mail settings
- Create SharePoint Sites*
- Configure Workflow Settings
- Decide on diagnostic logging settings (optional)
- Add antivirus protection (optional)

For MOSS, additional tasks include:

- Configure Shared Services
- Configure Session Throttles for InfoPath Forms Service **
- Add/Change Excel Services Trusted Locations **

*** The first site is already provisioned by the Basic install.**
**** These services apply only to the MOSS Enterprise Edition.**

- Service level settings for Shared Services (SharedServices1 by default)
- Create Alternate Access Mapping
- Check services enabled in this farm
- Enable SSO in the farm (optional)

NOTE *In the Basic install you have SQL Reports installed on the server, and you will have to create new virtual directories under the newly provisioned site at port 80 since the IIS Default website will be disabled (refer to the Basic Installation instructions in Chapter 5 for how to correct this specific issue).*

If the Advanced install was used, two more tasks appear (shown here as #1 and #2, as they must be done first):

1. **Initial Deployment** Adds Servers to the farm
2. **Initial Deployment** Assigns services to servers

Two other common tasks that may appear:

- The SharePoint 3.0 Central Administration application pool account should be unique (this is normal when installed since the install will assign the application pool to the account you used when installing).
- Configure Alternate Access Mapping (AAM).

As you work through these previously outlined tasks, some of them (designated as System Tasks) will be automatically updated by SharePoint. For those not in this category, it is very important to go back and update the status of the tasks in the Task List. This is, after all, just a list. Some tasks will be automatically removed (for example, once the e-mail services are configured, those tasks go away), but others (like server services assignment) do not. It's important to keep updating as you go, because these tasks should be done (loosely) in order. Besides, if you need to stop for some reason, you won't lose your place in the configuration or want to hand it off to someone else to complete.

In the next section, I will cover all of the configuration tasks required in a MOSS Server Farm setup, covering the tasks for both WSS and MOSS in either the Basic or Advanced installations. Be aware that this may not be all-inclusive; you may have additional configuration tasks to perform depending on your environment.

CAUTION *If you are setting a server farm environment and intend to use load balancing, make sure that the network and load balancing setup has been fully tested—then stopped—while configuring SharePoint (see Chapter 5).*

Step-by-Step Configuration

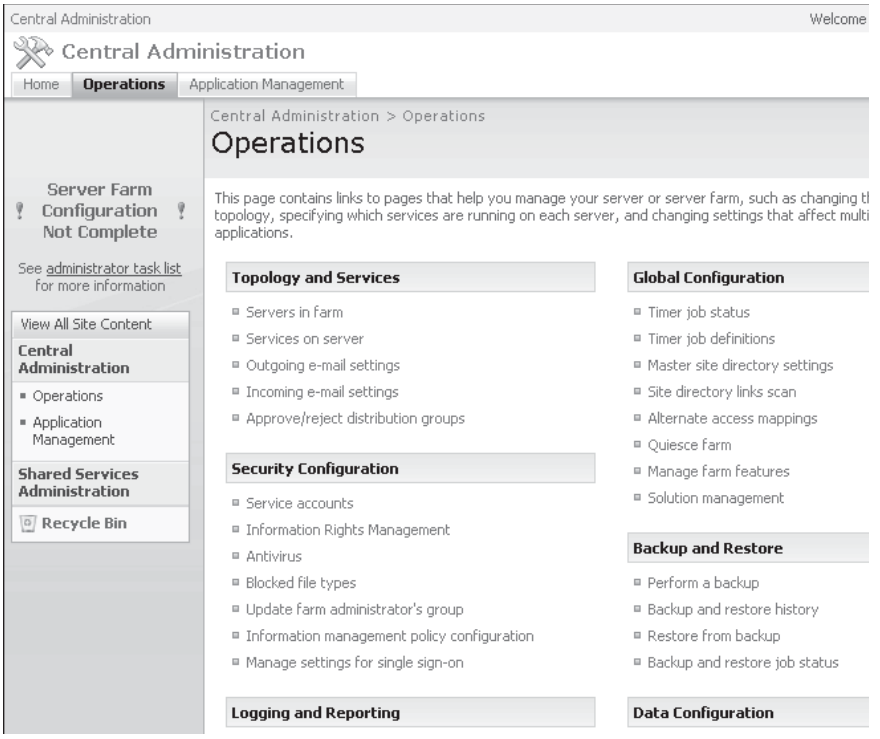
As I mentioned previously, this section will go step by step through each one of the installation/configuration tasks in the order specified by SharePoint. This includes every task for both WSS and MOSS in both Basic and Advanced configurations; where necessary, product specific tasks (like those for MOSS) are also indicated.

Adding Servers to the Farm

In a server farm environment, the setup of servers is based on joining the farm by connecting to the common configuration database. Servers can join and leave the farm at any time since all of SharePoint is contained in the database and other than Features or Templates that have been created, nothing is stored locally on the server. In both WSS and MOSS (Advanced install only), when the first server of the farm is installed (and subsequently the SharePoint Configuration Database created), the base services for running SharePoint 3.0 Central Administration are assigned to the first server. This is the same setup when installing MOSS and selecting Web Front-end as the installation target (If you selected Complete, then all services will be assigned to the first server, which is the option you should use when setting up an Application Server.) As other servers are loaded and then join the farm (though the existing configuration database), services can be assigned or changed between servers. While this is listed as the first task, you can actually skip this for the first server.

NOTE *If you use the MOSS Complete option on installation Windows SharePoint Services, Web Application services is not enabled by default. This service must be started manually if you intend it to provide web front-end services.*

Viewing servers in the farm (and the services assigned to them) is done through SharePoint 3.0 Central Administration Operations. To access this setting, open SharePoint 3.0 Central Administration and then click the Operations link or tab. This will open the Operations page, as shown:



For this task, on the Operations page under the Topology and Services group, click the Servers in farm link. This will open the list of servers; on initial install, only the installing server (and SQL Server in a farm) is shown, but once all servers have attached to the same configuration database they are all shown along with the services each is running.



NOTE *For reference, using the Basic install method assigns all basic services to the installing server automatically.*

Again, there's really nothing to do for the first server. For reference though, when additional servers join (as you will see at the end of this chapter), they will appear in this list. As a best practice, after you have created the Configuration Database, you should install all servers and join the farm before starting the individual configuration steps (see the section "Join an Existing Server Farm" at the end of this chapter).

Assigning Services to Servers

SharePoint services are assigned on a per server basis and depend on the role each will be playing in the farm. Front-end servers can host webs, search, the document load-balancing service, e-mail services, and so on. Application servers host search, indexing, and more. You can mix and match the service assignments to suit your needs, however, all services must be assigned for SharePoint to operate correctly. Be aware, however, that the ability to assign specific roles is limited in WSS; when you have performed a WSS Advanced install, you can add services, but the service assignments are defined as Custom and you cannot change them (see the next illustration).

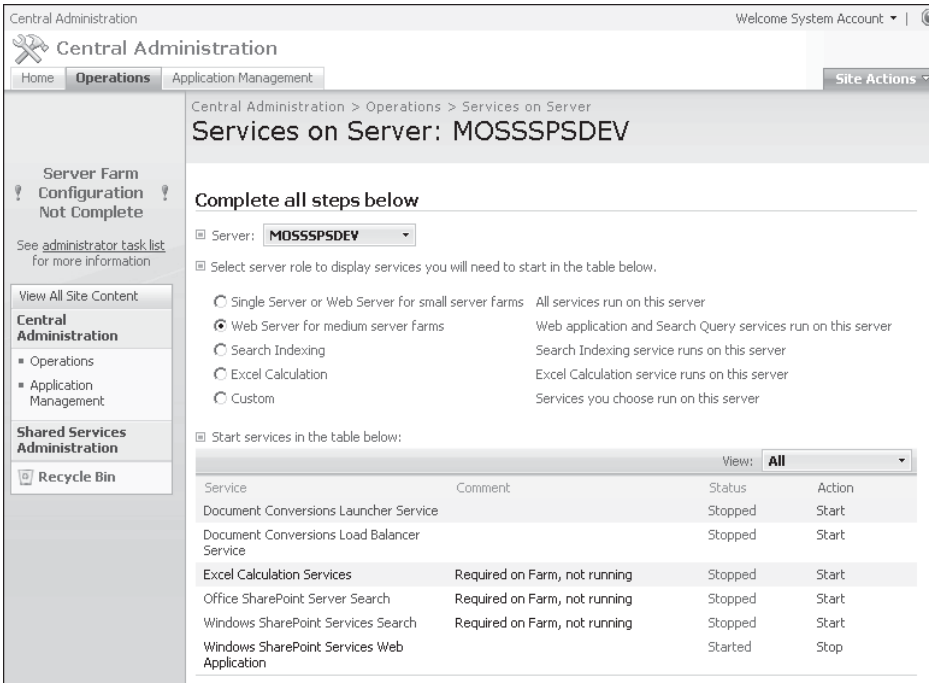
So how do you know what to assign? How services are assigned depends on the role the server will take (with the sole exception of the database, all services run on the front-end servers).

- In a Single Server setup, all services are assigned to the one server by default. If the Basic install was used, this cannot be changed (and the option is not available).

- In a farm environment (providing you have more than one front-end server), individual servers can be designated as Web host, Application Server, Search server, or all three. Web hosts support the sites and IIS; Application Servers support services and features; and Search servers are dedicated to indexing content.
- In a three system configuration (two front and one back), web hosting and search should be assigned to both front-end servers to balance the load.
- In a normal, medium-sized farm, search should be dedicated on a single server or running on all front-ends.

NOTE *The Windows SharePoint Search Services must be running on at least one front-end server. (Search can run on any front-end server, Web, or Application.)*

Of course, SharePoint does provide granular assignment of services if you wish to customize them. However, other than the previous configurations, the custom settings are primarily for reassignment of services for maintenance, disaster recovery, and the like. To assign services to servers, open SharePoint 3.0 Central Administration, and then click the Operations link or tab. When the Operations page opens, click the Services on server link, which will display the Services on Server page. In MOSS, the first display would look something like:



If you have a WSS installation, there are a few less services than shown (notice the role option is disabled for WSS). As you can see, this page allows you to administer the services

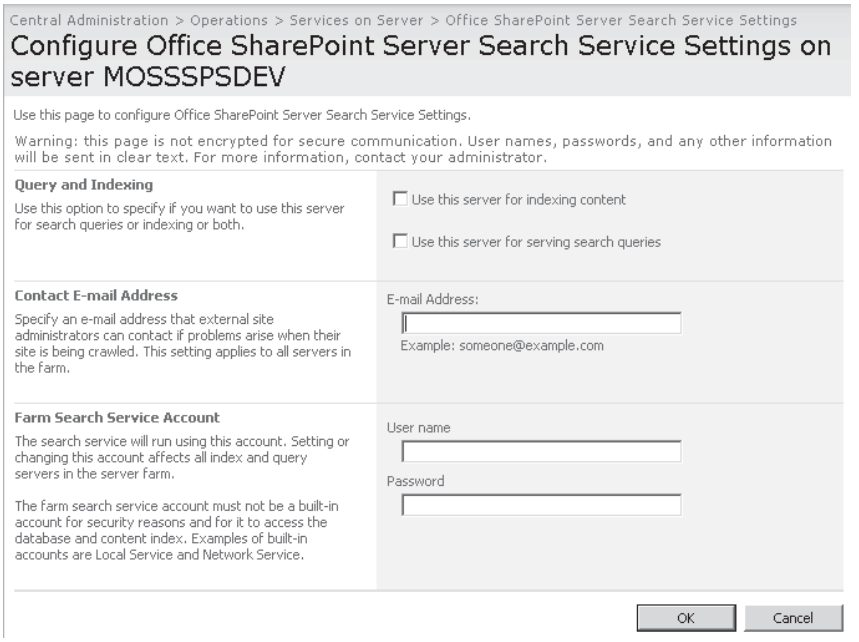
(and role) for any servers in the farm. You can easily alternate between servers using the drop-down list selection at the top of the page:



You can also select which services you want to see in the list, whether all or configurable (non-configurable services have no settings).

Setting Up the Office SharePoint Search Service

If you are installing MOSS (not WSS), you need to set up the SharePoint Search service first. The Office SharePoint Search Service provides the farm-wide search capabilities in MOSS and provides the indexing for Shared Services. On the Services for Server page, click on the Office SharePoint Search Service title. This will open the Search Settings page:



At the top of this page, you can see two options for the server, Indexing Content and Serving Up Search Queries. If indexing content, the server will crawl content and maintain search indexes locally. If serving queries, it will handle the search requests to look up indexed content. Either or both of these may be selected and additional options will be shown, depending on which is checked (options are explained next).

NOTE *If after assigning, you remove either of these options from a server, you will get a warning message that any indexes maintained by that server will be dropped. This doesn't have an impact during installation, but on a production system be aware that dropping indexes can a) reduce the reliability of searches until the indexes are rebuilt and b) have a severe impact on performance. Also, be aware that if the service account is ever changed, the pertinent indexes are deleted and rebuilt.*

To set up the service, you need to first specify an e-mail account that SharePoint will use to send e-mail messages if there are problems with crawling sites. This one account is used for the entire farm, so I suggest using the SharePoint Administrator account (SPAdmin) if you do not have a particular person responsible for this. Next, you will need to specify the service's account, which should be the SharePoint Search account (SPSearch), as outlined in the installation account setup. Remember that this account must have sufficient rights to crawl all site content in the farm and can be different than the Windows SharePoint Search service if needed.

CAUTION *Built-in accounts like NETWORK SERVICE or Local System should not be used to run the search service. You should always use a dedicated (non-expiring) account (such as the SPSearch account explained in Chapter 5).*

If you select to use this server for indexing content, the additional indexing options will be shown:

- Index File Location specifies where the physical files will be stored; I highly suggest that you make sure that the drive used here has plenty of space (more than 10gb) and when at all possible, do not use the system drive (usually C:, wherever Windows itself is installed). While this is sometimes not an option in development, production should never use the system drive.
- Indexer performance setting determines how aggressively the content will be crawled:
  - Reduced means that other services will take priority over indexing and thus, content may not always be indexed in a timely manner. If the content is not particularly volatile or relatively static (such as with a company internet site), this is the best setting.
  - Partly reduced (the default setting) means indexing services will be balanced out with other services on equal priority. This is the typical setting when search will be spread across more than one front end.
  - Maximum means that all resources will be dedicated to crawling and indexing. This should only be used if this server will be dedicated to search services only.
- The Web Front End and Crawling options give you the option of either dedicating a single server for search services or running the service across all front-end servers. In most installations, search should be enabled on all front-ends to benefit from the built-in Windows load balancing. However, for medium farms with heavy content search requirements or large farms, you should have a dedicated server to run search services.

If you enable this server to serve queries, the additional query options will be shown:

- **Query Server Index File Location** This option is where you can specify the physical location to store index files that are propagated between servers (another

case in which you should NOT use the system drive). This is created when installing the first server and used by all servers in the farm; this also creates a network file share called searchindexpropagation, visible to all servers in the farm. The Search service account (specified previously) is granted write access to this share.

You have three options available: If this is the first server (or setting up search for the first time), you will select Configure the share automatically. Then you must specify a local system account (like Administrator), which will be used to set up the share and grant write access to the Search Service Account. Alternately, if you can choose the "I will configure the share with STSAdm" option (the same as Do Nothing), SharePoint assumes you will use STSAdm (SharePoint's administration Command-Line tool) to configure the share. If you have manually set up the share, you will select the Do Nothing option.

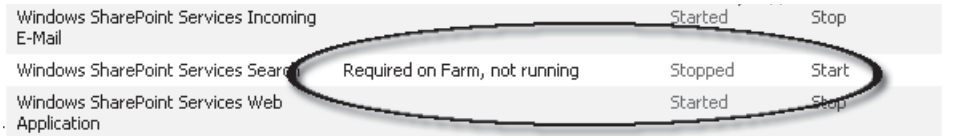
When you have completed your settings, click the OK button. On returning to the Services on Server page, you should now see the Office SharePoint Search Service in the running state (you should start the Search Service on all servers that will run it in the farm).

Setting Up the Windows SharePoint Search Service/WSS Search Help Service

A required service, the Windows SharePoint Search Service is the core search engine that supports the localized search for Sites in WSS. In MOSS, this is a helper service (as denoted in the title) since the Office SharePoint Server Search is doing most of the work. This one is responsible for the indexing of local content within the sites themselves, providing the site wide search. Unlike the OSS, indexes for this search are stored in the database and do not use the file system (no propagation is done). In the Advanced installation, this service must be set up manually and accounts must be assigned. In a Basic install, this service is predefined and assigned to the Local Service account and must be changed.

To set up the service:

1. Login to the server and open SharePoint 3.0 Central Administration.
2. When it opens, click the Operations tab or link to open the Operations page.
3. When the Operations page opens, click the Services on server link.
4. In the Services display, you'll see that the Windows SharePoint Search Service is not running and also indicates a message that the service is required:



5. Click the Start link (or if already started in a Basic install, click the name of the service) to open the Configuration page.
6. If you are using the Advanced install, you must specify both the Service Account (SPService), which Search Service will run as, and the Content Account (SPContent), which will own the search content stored in the database. If you're using the Basic install, you should change the Local System account to the correct accounts.
7. Scroll down to the Search Database section. If you are installing only WSS using a Basic install or you installed MOSS with SQL Express, you should leave these settings as they are (accepting the defaults). Otherwise, you will specify the name of the

Database Server and specify the name to use for the Search Index database (the default is always WSS_Search_<Server>). As shown with the Basic install, a GUID is included automatically. You can change this to a more descriptive name if you like.

8. Just below that, you will also set the Indexing Schedule. This determines how often the local content is crawled and how quickly an added item should appear in search results. This service can consume a lot of resources depending on your content, so here are some basic guidelines. The default setting is 20 (MOSS) or 5 minutes (WSS), and for most that will be sufficient, but consider the following:
   - The more time specified, the more time it will take added items to be indexed.
   - If you have a low volume of items added, you should increase the time to reduce useless crawling.
   - If you experience performance problems, particularly during heavy-user load, you should set the schedule to crawl overnight instead.
9. Enter your settings (or accept the default) and click the Start button.
10. As the search service starts, a working page will be displayed. When you return to the Services on Server page, you should see that the Search Service is now started.

NOTE *You should start the WSS Search service on all front-end servers in the farm.*

Troubleshooting WSS Search Once the search service is running, local items will be indexing and the results are stored in the database. When first started on an installation, this will be very quick as there is no content. However, as users begin to use the site, you may experience some problems. Here's a quick checklist of things to try if you run into trouble:

Search returns nothing:

- Verify the service is running.
- Verify that the content account specified is a valid account and has permissions to crawl the content.
- Verify whether it is a site issue or common across the board; the most likely cause if a single site is permissions (check that the Content account has access).

Performance degrades when running:

- If a continual degradation occurs, verify that the database is not experiencing growth problems or running out of space.
- If a sharp degrade occurs and then performance returns to normal:
  - Reduce the indexing time (from 20 min to 5). Wait for at least 30 minutes.
  - If performance improves, change the schedule back to 20 minutes.
  - If performance is slow but steady, increase indexing to 60 minutes or make it daily.
- Reduce logging on both SharePoint and SQL Server.
- Verify that all front-end servers have sufficient space on the System Drive.
- Verify that none of the servers are storing index files on a System Drive.

Starting the Document Conversion Services

The Document Conversions Launcher and the Document Conversions Load Balancer Services are a major part of SharePoint's Enterprise Content Management features, providing content producers with the ability to convert documents from one form to another. For users, document converters enable viewing documents even if the required client application is not available. There are several included with MOSS: for example, the From Word Document to Web Page converter converts Word .doc files into web page .html files. Additional converters can be added, either custom built or created by a third party.

These services are used when there is a request or workflow step used for converting a document from one format to another, such as publishing a Word document to a web page. Conversions are invoked either through the user interface such as through a web request or via programming code, which can provide a custom display or initiate a conversion inside of a workflow.

For conversions to operate in a farm, both services must be running on servers in the Application Role. At minimum, one Load Balancer service must be running with any number of servers running the Launcher service (including the same one running the Load Balancer). That said, at least one service must be running in the farm. In a single server setup, both of these services run on the same server. The actual conversion process is pretty straightforward:

1. A web request is made for a particular document or the timer service requests a document conversion (as part of workflow).
2. The web application sends a conversion request to the Load Balancer service.
3. The Load Balancer returns a ticket to the request, which tells it which Launcher is available.
4. The web application sends a request to the Launcher to perform the conversion.
5. The Launcher does the conversion, and then returns the result to the web application.
6. Upon receipt, the web application notifies the Load Balancer service that the request is complete.

At your option, you can configure these services later, but both services must be up and running in the farm for conversions (publishing) to work. The Load Balancer service manages the document converters and handles requests, whereas the Launcher service actually launches the conversions on the Application server. Scaling out the service is easily done by simply adding additional application servers to run the Launcher service (only one Load Balancer is needed). If you used the Basic install for MOSS, these services are already assigned to the installing server and started. When setting up the farm, you should check the properties of the services to ensure that the port numbers used are the same for all servers (and the ports used are not blocked). Also, if custom conversion programs are used, such as PDF to Text, they must be installed on all servers running the Launcher service.

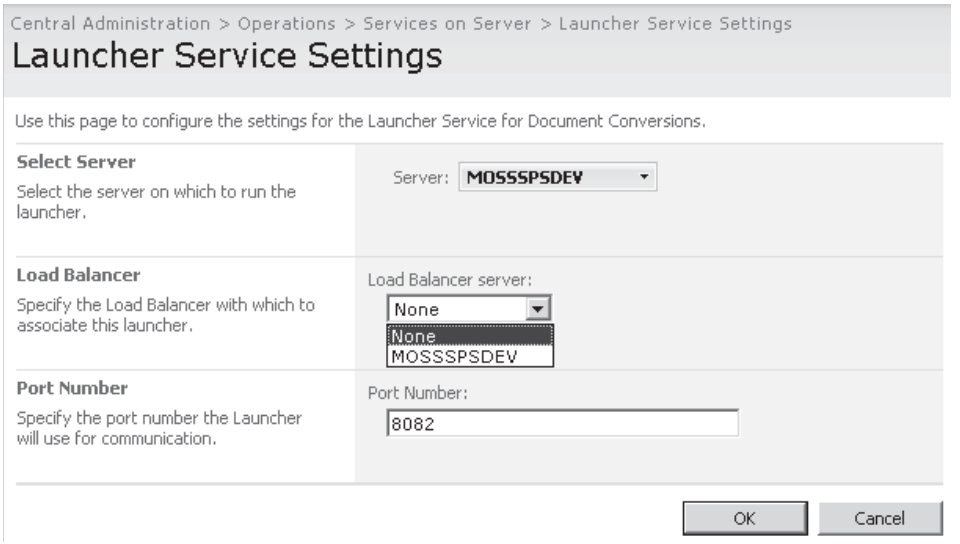
To start up the services:

1. If you are not on the Services on Server page, get there by logging in to the server that will running the Load Balancing service (this must be in the Application Server role).
2. Select Start | All Programs | Administrative Tools | SharePoint 3.0 Central Administration.
3. Click on the Operations tab or link to open the Operations page.

4. Under the Topology and Services group, click the Services on server link.
5. On the Services on Server page, find the Document Conversion services:



6. A Load Balancer service must be running for the Launcher Service to start, so this must be done first. The Load Balancer default settings are set as using the communication scheme http (you can choose between http or https) and the port is set at 8093; if you need to change these, click the Document Conversions Load Balancing Service link to open the properties page. Make any changes necessary and click OK to save. Then, on the Services on Server page, click the Start link to start the service.
7. Next, you can (optionally) start the Launcher Service. Click the Document Conversions Launcher Service link to open the properties page:



8. Select the Load Balancer from the drop-down list, and then click OK to save the changes. On the Services on Server page, click the Start link to start the service.
9. When you return to the Services on Server page, you should see both services (if you started both) now in the running state.

NOTE *Any server in a farm environment that is running the Launcher Service, but is not hosting the Central Administration site, will have a service status indicating that it is Starting.*

Handling Document Conversion Services Errors You may encounter errors when attempting to start the Document Conversions Load Balancing Service. There are three common issues:

1. If the service will not start, it is possible the account running the service isn't the right one (for example, using the Local System account instead of the SharePoint Services account) or that the account specified does not have "Logon as a service"

rights on the system. To change the account, open the Central Administration site and click the Operations link in the menu bar. Then, under the Security Configuration group, click Service Accounts. From the Service Accounts page, you can select the service using the drop-down list, and then under the account, specify configurable and enter the account name (Domain\Account format). If this does not correct the issue, verify (and if necessary, grant) Logon service rights to the account.

2. When you attempt to start the server, SharePoint displays an error page that shows DCLoadBalancer and nothing else. This can be caused either by account problems (as shown previously), or by an error caused internally within SharePoint. If you verify that the service is running the correct account with the correct rights, you may need to run an installation repair (see the section on Repairing/Uninstalling SharePoint at the end of this chapter).
3. If when attempting to start the service, SharePoint displays a Page Not Found error, this indicates a problem with the physical installation—and running a Repair operation will not correct the issue. The fix is to remove SharePoint and then re-install it on the server where the error is occurring (see the section on Repairing/Uninstalling SharePoint at the end of this chapter).

Starting the Excel Calculation Service

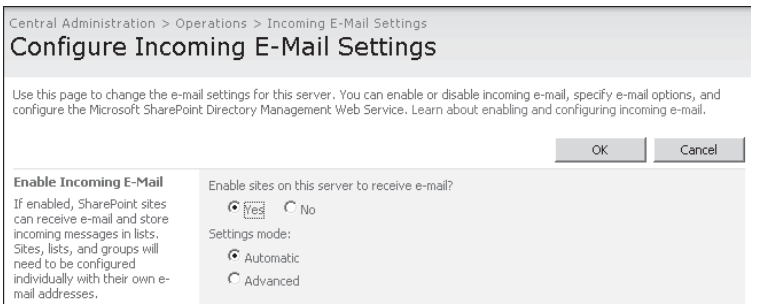
Excel Services is an integrated part of MOSS and is used to provide Excel web services and the ability to work with Workbooks directly in SharePoint. There are three basic components to this service: the Excel Calculation Services that run on an Application Server, the Excel Web Services, and the Web Access Web Part, which run on front-end servers. Actual configuration of these services occurs after you have installed a Shared Service Provider, but they must be started to be available (by default, this is stopped after the installation).

To start the service:

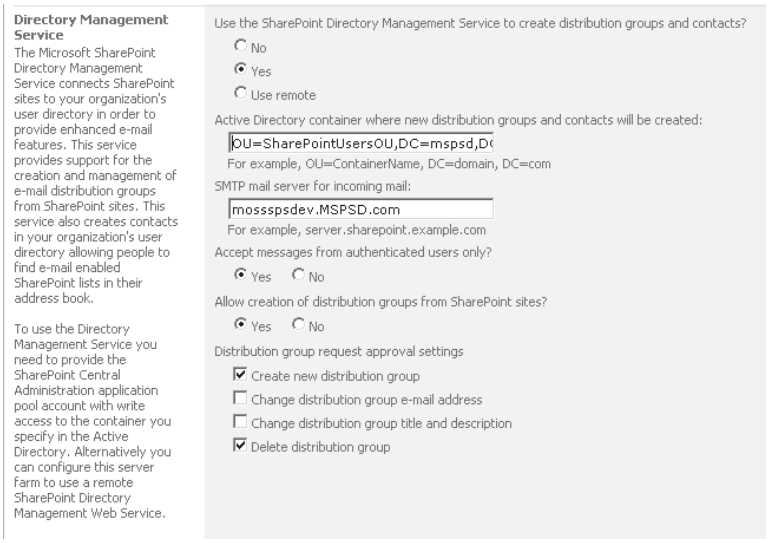
1. If you are not already on the Services on Server page, select Start | All Programs | Administrative Tools | SharePoint 3.0 Central Administration.
2. Click on the Operations tab or link to open the Operations page.
3. Under the Topology and Services group, click the Services on server link.
4. On the Services on Server page, find Excel Calculation Services and click the Start link to start the service.

Setting Incoming E-mail Settings

The Incoming E-mail server is used by SharePoint to support e-mail services for sites, lists, and libraries that are e-mail enabled. The default setting for this service is enabled (allow sites to receive e-mail) with automatic detection of settings enabled. Note that these settings are not available if you used the WSS Basic install.



- **Enable Incoming E-mail** This enables or disables the capability of sites, lists, and so on to receive mail. It can be set to Automatic, meaning e-mail will use the default SMTP server or Advanced indicating that you are using another SMTP server or e-mail system (you must specify the mail drop folder next if this settings is used).
- **Directory Management Service** If enabled, this connects SharePoint sites to the organization's user directory in Active Directory, which enables enhanced e-mail features such as address books and so on. If Remote is set, a remote directory service can be specified.



When enabled (set to Yes), you specify the Active Directory container that will store new distribution groups and contacts, such as the SharePointUsersOU shown previously. You will also specify the SMTP server for incoming mail. This is usually filled in for you and is the current web server (the service is already there by default). As for actual AD interaction, you can select whether e-mail will be accepted by anyone or only by authenticated users, enable or disable the creation of distribution groups, and set the group request approval settings (Note that the distribution group requests and approvals are managed from the SharePoint 3.0 Central Administration Operations page, under the Server and Topology group, and directly below the Outgoing E-mail settings link).

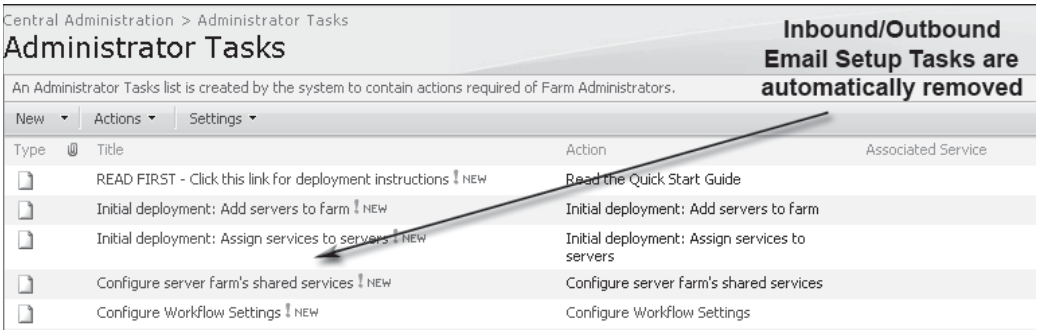
- **Incoming E-mail Server Display Address** This sets the e-mail address format for lists and sites to provide a more natural name.
- **E-mail Drop Folder** If the Advanced setting is used on the Enable Incoming E-mail option, you must enter the full path to the actual mail drop folder or network share here. Otherwise, this remains blank.

Setting Outgoing E-mail Settings

The outgoing e-mail settings are used by SharePoint when it is sending mail out (alerts, errors, workflow triggers, and so on). For outbound mail, all that is needed is the e-mail server address and an account name to use as the sender and reply-to address. Any account

can be used, however I strongly suggest that you enable an e-mail account for the SharePoint Administrator account and use it as the common e-mail address (this will allow users to easily tag SharePoint mail in Outlook). As with Inbound, these settings are not available if the WSS Basic install was used. To enter the settings, simply specify the SMTP server name (usually the fully qualified name), e-mail account to use, and set the Character set most appropriate for your installation: the default and typical setting is 65001 (Unicode UTF-8).

Simply click OK to accept the settings and return to the Operations page. If you have completed setting up the e-mail settings, you'll now notice that they've been removed from the Administrative Task List:



The e-mail settings are example of how some tasks are updated automatically in the Administrative Tasks lists (tasks are also added when setting up Shared Services).

Creating SharePoint Sites and Setting Up Shared Services

Once the basic services are running (such as search and e-mail) you can actually begin to create applications and sites within SharePoint. This of course depends on what you have installed; if you installed WSS with the Basic installation method a default site has already been created for you. If you're installing MOSS using the Basic method, both the default site and a Shared Services Provider have been created. Using the WSS Advanced install, you will need to setup your first site. If you've installed MOSS Advanced, you will need to setup a Shared Services provider first, create the Shared Services sites (Admin and My Site Host), and then you can create a site.

For most installations you will not create any sites until after all servers have joined the farm. However, in order to keep this step by step consistent with the order of the installation, I will cover creating the first site here. Many of the post install settings are application specific (such as workflow settings) as you will see. So, if you are not ready to install a site, you should stop when you get to that point in the configuration and skip ahead a chapter to review sites and site templates. This will help you choose which site type is best for you.

NOTE *At this point and for the rest of the configuration, if you are using the SPInstall account, you should logout and back in using the SPServices account.*

Shared Services Overview (MOSS Only)

Shared Service Providers are a new concept in this release, and they add a new tier of information management in a server farm environment by taking full advantage of distributed application architecture. For those that may have used this in the SharePoint

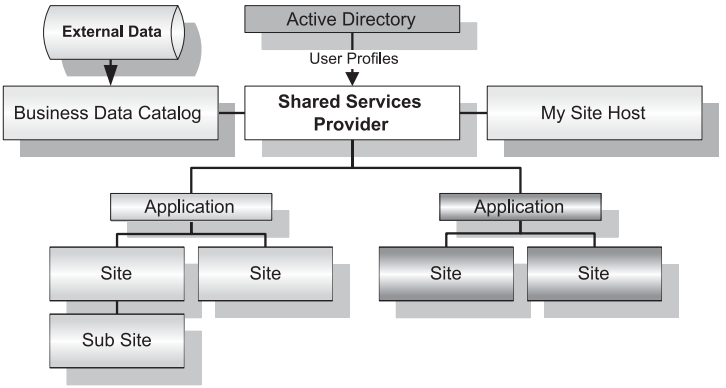
Portal Server 2003 edition, be aware that while the name is the same, the functionality isn't (optional in 2003, Shared Services is a required component for installing MOSS).

The purpose of Shared Services Providers is to act as a component of MOSS that enables grouping of common functionality (features used on every site). This centralized configuration is very convenient because you only have to set things up once to serve multiple sites (such as with business data), and you also don't have to keep reinventing the wheel each time you need search, a My Site Host, or similar. Features provided within Shared Services include the following:

- Office Server Search and Indexing
- External Site Search and Indexing
- User Profile Management
- My Site Host Management
- Business Data Catalog Management (Enterprise only)
- Excel Services Management (Enterprise only)

Within MOSS, there must be at least one Shared Service Provider per installation (Single or Farm), though there can be several (you may need language localization, different authentication methods, and so on.). One is always assigned as the default. As you create new applications, they are assigned to a specific Shared Service Provider (SSP); the applications, in turn, provide the sites below them access to the services the SSP provides. A general view of this functionality appears next.

The composition of Shared Services includes the Shared Services application, which runs the Shared Services Administration site, and the My Sites Host application, which hosts the My Site sites. These are the same as any site (where you have both an application and a site), except that Shared Services is only accessible through SharePoint 3.0 Central Administration and the My Sites site has no interface (My Sites are accessed per user via Site Navigation).



From a user's perspective, Shared Services provide a very seamless experience, allowing use between different sites while also allowing access to common features in the background such as search and My Sites. From a developer's perspective, Shared Services is much more than just SharePoint functions; it is intended to be a central repository for custom applications and features. With a centralized capability already provided out of the box, you can easily

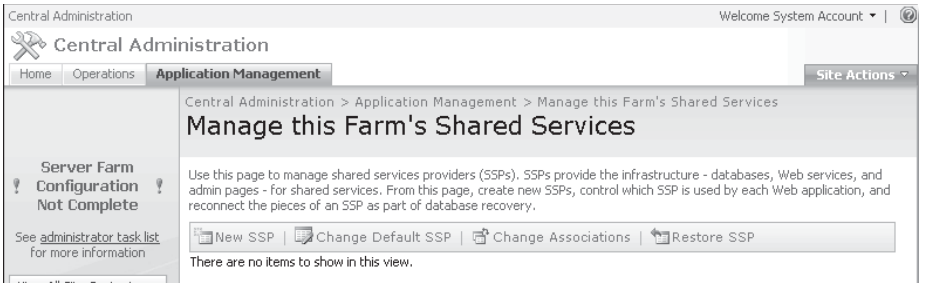
create applications and features without having to worry about the mechanics of deploying it to multiple sites.

Creating a Shared Services Provider requires three primary steps, as you will see next; two of these involve the creation of two applications (one for the SSP and one for the My Site Host), while the third is actually creating the SSP. The creation process handles the rest of the actual work, including provisioning both sites and establishing the SSP Administration site.

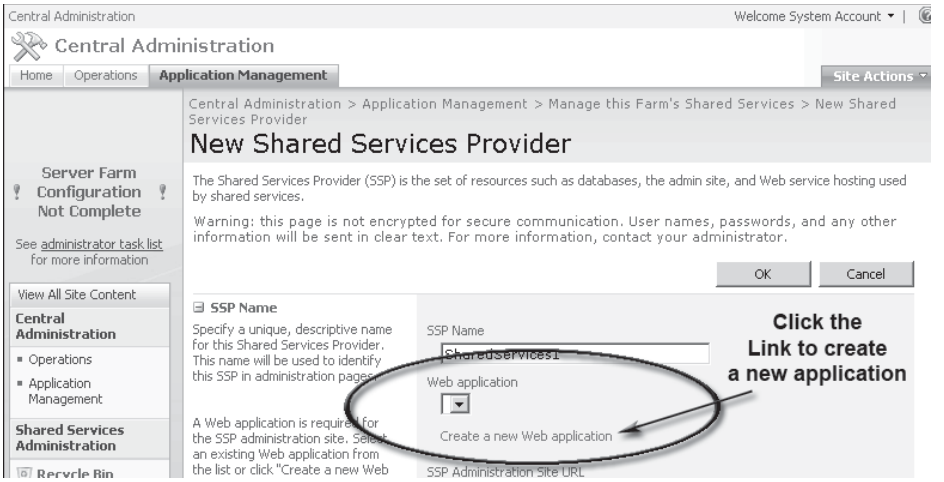
You must have at least one default SSP, but you can have any number of Shared Services Providers supporting any number of sites. This enables you to use the localization features of the My Sites, limit features, and so on. Whenever a new application is created, the default SSP is the one it will be associated with; of course, associations can be changed using the Manage Farm Shared Services.

Starting Shared Services Setup (MOSS Only)

In a MOSS installation, the SharePoint 3.0 Central Administration site automatically checks for the status of Shared Services in the farm (or server). If it detects that Shared Services is not yet installed (or if something required is not configured), it indicates this with a warning message on the home page (Server Farm Configuration Not Complete). This message will be displayed until Shared Services has been configured and services assigned to servers. To create a Shared Service Provider, in SharePoint 3.0 Central Administration, clicking the Shared Services Administration link will open up the Create Shared Services page:



Click the New SSP link in the menu bar. If this is the first installation, you will see that you cannot create an SSP because no applications have been created yet:



To create an application, you can simply click the link shown. Alternatively (and what you will do most of the time), you can click the Application Management link to access the application features for the farm/server, create the two applications needed, and then return here.

Application Overview

While mentioned throughout, it is worth revisiting the role of an application in SharePoint. Technically speaking, an application is the same as an IIS website running under its own application pool (IIS 6.0 Isolation mode). Under this application, you can have any number of sites, just as you can have multiple sites underneath an IIS web. The purpose of an application in SharePoint is that it provides complete flexibility in how your sites are set up, enabling you to have true localization of sites, different authentication methods, and so on. The fact that applications are separate from one another provides better security and protection from errors (if one SharePoint application fails for some reason, no others are affected).

An application can have a single root site and any number of sites underneath it using different site types (limited by the root site type) and while individual sites can have their own members, the authentication method (and port number of web), as well as the Alternate Access Mappings (AAM) are set at the application level. The steps involved in creating an application are first selecting the authentication method that will be used for users, assigning a port for the application, specifying the database information, and then finally creating the application. Once an application has been created, sites can be created within it. In following along with the Administrative Tasks, you will now set up the two applications needed to support Shared Services: one to host the Shared Services Administration site and the other for the My Site Host. As mentioned, creating applications to host your actual SharePoint user sites is done exactly the same way.

NOTE *Any time you create a new application, you must choose the authentication method for users. Refer to Appendix B for choosing the method you need.*

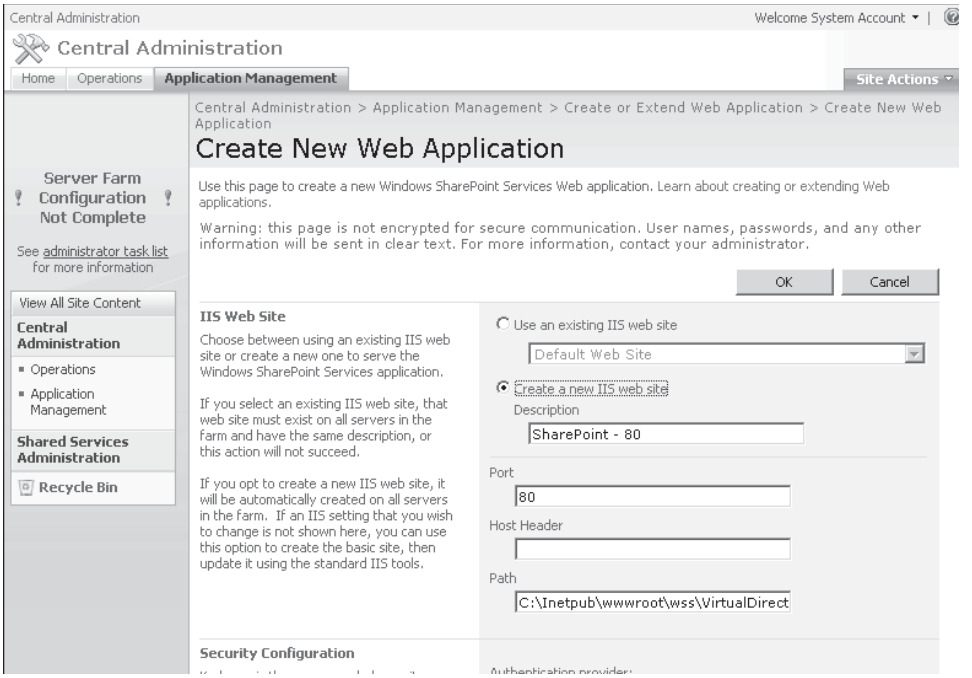
Creating an Application

The process of creating an application is the same regardless of the site type or purpose. As the root, an application is required to manage the overall services that will be provided to the site(s) it hosts—much the same way you might configure an IIS Virtual Directory. For MOSS Shared Services, two separate applications are required: one to run the Shared Services administration and the other to be used for running the My Site Host. While technically both of these sites can use the same application (no problem doing this on development and test systems), keeping them separate (in production) enables you to back them up separately (this is particularly important with My Sites!).

You can create an application through SharePoint 3.0 Central Administration's Application Management page (or in our case, accessible from the Shared Services creation page). When you click the link from the Create page or from Application Management | Create or extend web application, SharePoint will load all virtual directories found that do not already have SharePoint extensions. By default on a new install, the only available virtual is the default site at port 80; an existing virtual can be extended (SharePoint added to it) or a new one created (which you'll see next).

Creating an application is always done the same way; the setup for the first application is an example of how you will set up all of them:

1. Make your way to the Create Application page through one of two methods, either through the link in the Create Shared Services page or through the Application Management page. When the page first opens it should look something like this:

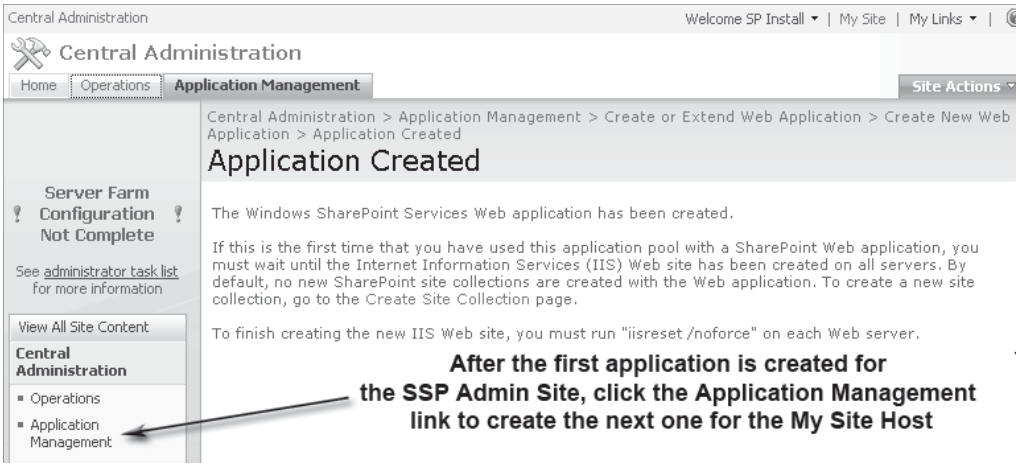


2. It may pre-set the values for Port 80 if it doesn't detect any SharePoint Applications (other than Central Admin) installed. For the Shared Services Application, you'll want to change the name and port number. The port itself is immaterial (as long as not used elsewhere and as long as it's not port 80). My personal preference is to use port 100 for the SSP and 101 for the My Sites Host. You can select a Host Header here, but that is not typical for Shared Services. Be aware that you can have lots of problems here if the port you use is blocked by a DMZ or firewall. Also note, if you intend to make My Sites available for Internet use, these are well designed for NTLM, but you will have authentication problems if identity management is not used because authentication cannot cross Applications.

TIP *If you enter the port number first, it will change it everywhere.*

3. In the next section, you select the security settings for this site. By default this is set to NTLM (Windows Default), with Anonymous access and Security Sockets Layer (SSL) turned off. For most installations, these settings are what you want. Use Kerberos (Kerberos is a network authentication protocol designed to provide strong authentication using secret-key cryptography) only if you really need to and already know what has to be done to make it work. In the meantime, you might wish to turn on SSL if you want to add an extra layer of security.

4. Next, you will specify the Load Balancing URL (this is the URL assigned in a farm environment for all load-balanced front-end systems). Remember that for Shared Services, the default is simply the URL for the site.
5. Now you'll have to specify the application pool account settings. This sets the account that the application will run (equivalent to the IUSR account in IIS), and the account you use here requires special privileges (specifically Run as a service) outlined in Chapter 5. The account designated for this is the SPAppPool account. Be *sure* to use the correct account here (the account designated to run the application pool like SPAppPool), because if you use an account like NETWORK SERVICE or Local System, you will break SharePoint. This is because neither account has the correct access to SharePoint or the database.
6. Next, you simply need to choose whether or not to restart IIS automatically after the application has been created (select Automatic here). Keep in mind that you might receive an Error Page after the application has been created (this is expected); use the Go Back to Site link to return to the SharePoint 3.0 Central Administration home page, and then click on the Application Management link to return to the Application Management page.
7. The last sections allow you to define the database information for the application and choose a search server. These tasks include setting up the user account for access and naming the database the application will use. Always change the default name from WSS_Content to something more readily identifiable (like SSPContentDB).
8. You'll also notice that you can't select Search options here because this farm doesn't have multiple search servers installed. In fact, this is the case most of the time and especially when doing the first install. When ready, click the OK button to create the application. When the application has been created, you are directed to the completion page:



9. Return to the Application Management page, select Create or Extend Web Application again, and this time, set up the application for the My Sites Host. In this example, I'm using 101 to remain consistent with the Shared Services at port 100.

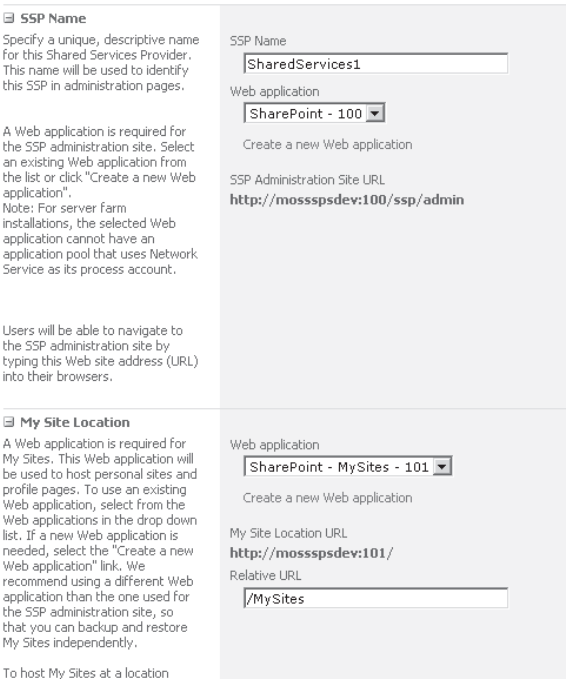
10. Complete the process for creating the second application and click OK to create it. Once again, you may have to refresh the page after the application has been created. Be sure to change the database name from WSS_Content to something else; I usually use WSS_Content_MySites.

Creating a Shared Services Provider (MOSS Only)

Once there is at least one application setup for Shared Services (though I recommend always using two), you can begin to configure Shared Services to set up the Shared Services Administration site and to set up the My Site Host. Note at this point, you are only installing Shared Services; setup and the configuration comes later.

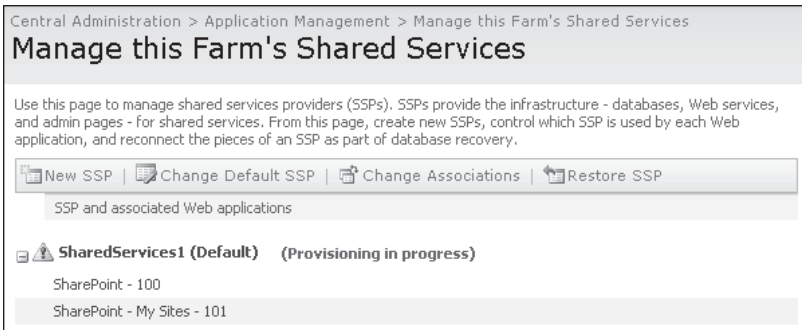
To create the Shared Services provider, do the following:

1. From any page in SharePoint 3.0 Central Administration, click the Shared Services Administration link.
2. When the Manage this Farms Shared Services page opens, click the New SSP link.
3. Enter the SSP Name and then, using the drop-down list, select the web application to the one created at port 100 or the port you set (you will see the URL shown when this is selected). Next, select the My Sites Host application for the My Site Location as shown:

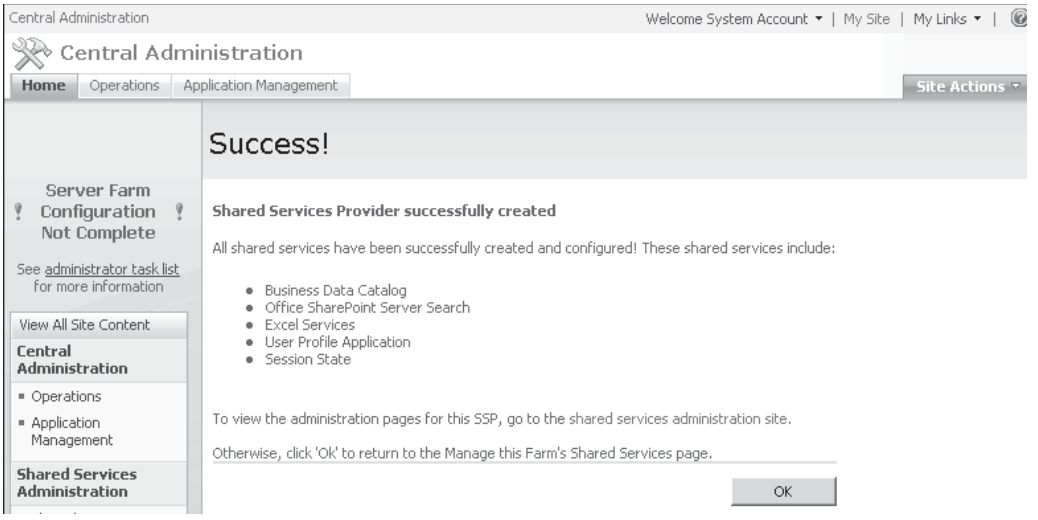


4. Next, you must specify the SSP Service Credentials. The account you specify here will be used by Shared Services to run web services and needs to run as a service access. This should always be a SharePoint Services account (SPService or whatever name you selected).
5. Next, you must specify the database names and the authentication to be used for the Shared Services Provider Administration Site database. The SSP's Search storage names these appropriately so that they can be easily identified in SQL Server (I like to include the SSP Name like SharedServices1_DB and SSP1_Search_DB).
6. In the next section, you can select an Index Server to use for this SSP. In a Single Server setup (as shown), the system itself is set and cannot be changed. In a farm setup, all of the servers with the index service installed (usually all front-ends) are available. You should also specify the file location for holding local index data; this goes to the System drive by default and should be changed to a drive that has sufficient space (this should never be the system drive in a production environment).

7. In the last section, you can select whether the Shared Services Web Services will require SSL; this should be turned on only in a production environment (unless you have certificates for QA/Test/Development, which is unlikely).
8. Click the OK button to create the SSP. The timer page will show while the work is being done. Note that if you did not select different applications for the SSP and the My Site Host, SharePoint will warn you that this is not recommended (as mentioned previously); other than this production is fine.
9. While Shared Services is provisioning, the Manage Farm Services page indicates this:



10. When the new SSP has been created, you should be presented the Success page:

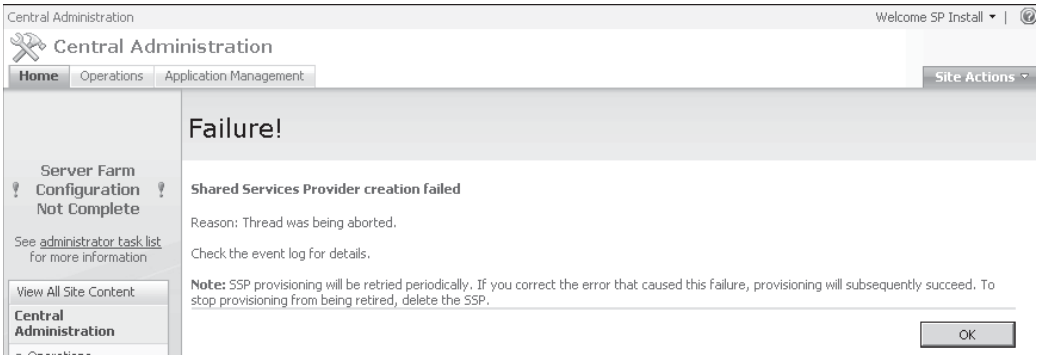


As you can see, several services are installed with the SSP including the Excel Services and Office SharePoint Server Search. All of these services (except the Business Data Catalog) must be configured as shown in the steps that follow. To ensure you don't forget to configure these, SharePoint detects the SSP has been created and adds additional configuration tasks to the Administration Task list. Also, if you intend to use a different authentication method other than NTLM as shown, you'll need to review setting up authentication methods in Appendix B.

Handling Shared Services Installation Errors (MOSS Only)

A number of errors can occur when setting up Shared Services, which can be troublesome because you can't continue the configuration until at least one SSP is operational. Adding to this, it can be a challenge to track down what the exact problem is. The most common issues are privileges granted to the accounts used, but there are many other reasons why the process can fail. Unfortunately, the biggest problem is that if a failure does occur, it can be a bit tricky to get it cleared up, even if you have corrected the problem (this is further complicated by the fact that you cannot delete the Default Shared Service Provider once it's created).

If there are any problems with the installation of the SSP, you will be notified with the error page:



While this doesn't tell you much, it does include a pretty important tidbit of information if the problem is related to provisioning—and that is that the job to handle the provisioning (running under the OWSTIMER server) will keep retrying unless you delete the SSP. If this was the first SSP you defined, you now have a problem because you cannot delete the default SSP (I'll explain how to get around that issue next).

Determining the Failure You have a few places to look to determine what might have caused an error in the SSP creation. The first place to look is in the server Application and System Event logs, because application errors will appear in the Application log and any security issues (bad logins, and so on) will appear under System Events. Remember that you'll have to check all servers in the farms as an error can occur anywhere, including the SQL Server itself.

Another place to look for errors is the SharePoint job log. This log is cycled periodically and is located under the <installed drive> | Program Files | Common Files | Microsoft Shared | web server extensions | 12 | LOGS (the SharePoint Hive). The format is <ServerName>-<Date>-<Time Log Opened>.log. Find the latest log and search the SSP name that failed. Don't be surprised if the errors shown there do not totally explain what the problem is, but you'll usually find that account problems, file locks, and similar failures are shown.

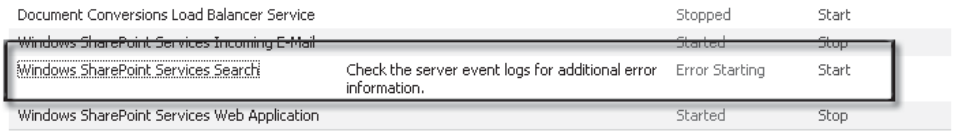
If you find nothing in the previous areas, your best bet is to follow Recovering from a failure. In the worst case, if nothing seems to fix it, uninstalling and reinstalling SharePoint may be required.

Access Failures You may have an issue after installing the first website where the Shared Services becomes unreachable. There is no particular issue that causes this, but you'll find that regardless of the account you use, you can't load the site. It may prompt you to login and then send you to a Not Found page or may simply not display. There is only one best

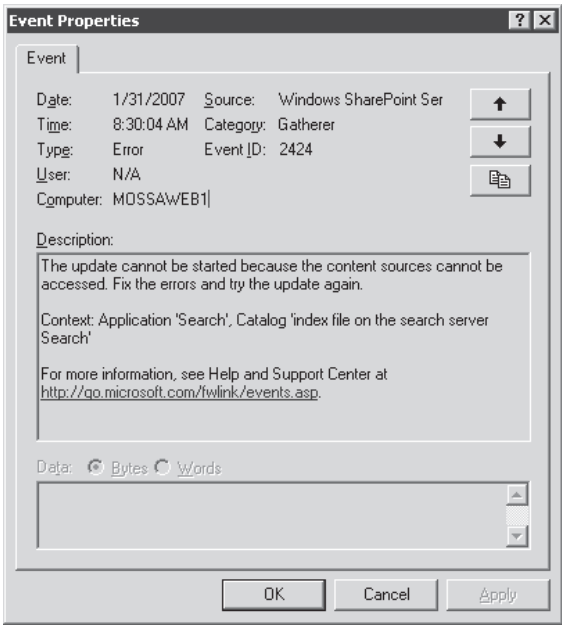
workaround for this issue, which involves creating a new set of applications and SSP, moving things around, deleting the original, and then moving things around again. There are a bunch of steps but none of them are very complicated. To try this workaround, perform the following steps:

1. Go to SharePoint 3.0 Central Administration | Application Management.
2. Create two new Web Applications using port 200 and 201. (These will be deleted later.)
3. Create a new Shared Service Provider and use the two new applications (200 and 201).
4. Return to Shared Services Administration.
5. Use the Change Associations link to move the original sites to the new service.
6. Use Change default SSP to switch to the one you just created.
7. Go to SharePoint 3.0 Central Administration | Application Management.
8. Under the SharePoint Web Application Management section, click Delete Web Application.
9. Make sure you have the correct application set under Application Name (100 for the original SSP site), click to select Yes to Delete Content Databases, and then click Yes to Delete IIS websites.
10. Repeat this for the My Sites Host Web application (101).
11. Go back to Shared Services Administration; there should be nothing listed for the original SSP.
12. Mouse over the name and use the drop-down list menu to select Delete.
13. Select yes to delete Shared Services database, and then click OK.
14. Repeat the process to create a new Shared Service with the original name.
15. Go back to Shared Services Administration.
16. Use Change Associations to move the sites (100 and 101) to the new replacement service.
17. Use Change default SSP to switch to the replacement one.
18. Repeat the process to delete the temporary one created.
19. Under SharePoint Web Application Management section, click on Delete Web Application.
20. Make sure you have the correct one set in the drop-down list, and then select Yes to Delete Content Databases and Yes to Delete IIS websites.
21. Repeat this for the (dummy) My Sites Web Application.
22. Go back to Shared Services Administration; there should be nothing listed for the temporary SSP.
23. Mouse over the name and use the drop-down list menu to select Delete.
24. Select Yes to Delete Shared Services database.
25. Click OK.
26. Run an IISReset and attempt to reconnect.

Service Failures A common issue with an SSP created failing is that one or more services are not available, most often the Windows SharePoint Services (Help) Search Service. If you encounter an error creating the SSP, check each server to verify that the service is running. If not, when you start it, the SSP creation process will detect the correction and should complete normally. If there is some kind of error, this will be indicated under SharePoint 3.0 Central Administration | Operations | Servers and Topology | Server Settings:



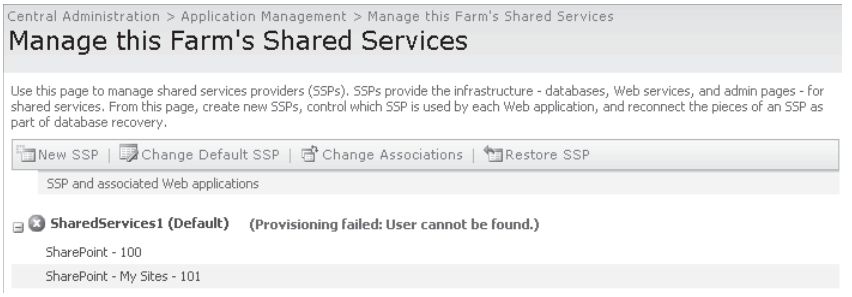
Another indication of a related problem might be found in the Event Log on one of the servers (usually the server with the problem that looks similar to this):



Whatever error exists here (the usual culprit is the account specified does not have SQL access or is the wrong account) must be corrected before an SSP can be created.

CAUTION *The provisioning job will keep retrying; if a service problem cannot be corrected (and this is the first SSP), you may need to stop the OWSTIMER service on the server to prevent excessive errors (stopping a service temporarily can be done via Start | Control Panel | Administrative Tools | Services; locate the OWSTIMER service in the list, right-click on the name, and then from the menu, select Stop).*

Account Failures If there are account problems, this is usually indicated by checking the status under the SharePoint 3.0 Central Administration | Operations | Manager Farm Shared Services page:



Since there are only two accounts associated with an SSP, the Application Pool (SPAppPool) account and the Services Account (SPServices), you need to determine which one might be the problem.

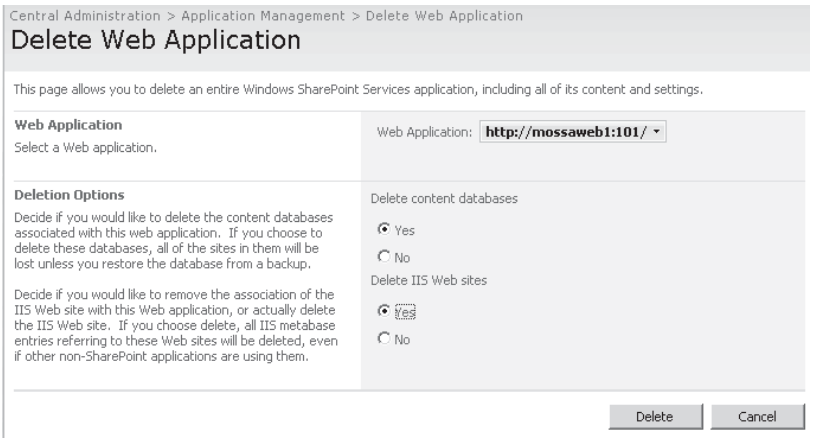
The most common problems are

- The Application Pool account used does not have access to SQL and/or does not have the WSS_Content_Application_Pool access to the SharePoint configuration database.
- One or both of the accounts have not been granted a Log On as a Service right in the domain.
- One or both of the accounts have not been added to the local WSSAdmin_WPG and IIS_WPG groups.
- The Service account is not a member of the Local Administrators group on one or more of the servers.

CAUTION *The provision job will continue to retry and you may need to stop the OWSTimer service (see "Service Failures").*

Recovering from a Failure If none of the issues shown previously correct the problem, recovering from a failed creation process can be a bit interesting. As I've mentioned, because you can't remove an SSP designated as the Default, there are two steps you can use to try to correct the issue:

- First, try creating a new SSP and simply use the My Sites application pool (101 in the examples) as the default for both the SSP and the My Sites Host. This will reallocate the application to the new SSP. If the create is successful, reassociate the SSP Site (100) to the new SSP and then change the new SSP to be the Default (this is the same fix as Access Errors shown previously).
- Second is a drop-kick approach, characterized by deleting the created applications and effectively starting over. Using SharePoint 3.0 Central Administration | Application Management, in the SharePoint Web Application Management section, click the Delete web application. Make sure the application drop-down list indicates the My Sites Host application at port 101 (change it using the drop-down list if necessary), and then make sure that the databases and websites will be deleted as shown:



Click OK to delete the site. Repeat this process to delete the SSP Site on port 100. After you have completed this, navigate to the SharePoint 3.0 Central Administration | Operations | Manager Farm Shared Services where it should look something like this:



Create two new applications, again at 100 and 101 as described previously. These will, by default, be assigned to the bad SSP. Create a new SSP using the correct Application Assignments. Assuming you're successful, under SharePoint 3.0 Central Administration | Operations | Manager Farm Shared Services, change the associations of the sites to the new SSP, make it the Default SSP, and then delete the failed one. If you need to have the specific name and database names you set in the original, you can create another new one using the correct name, DB names, and so on. If you're successful, as previously, reassign the sites, set the default SSP, and delete the one you don't want.

Cleanup SQL Server after Failure If any failure in creating an SSP requires you to drop it and then re-create it, SharePoint leaves a timer job running on SQL Server. This search job is run by the SQL Agent and fires every minute, and subsequently, for every minute the orphaned job fires, you receive an error in the Event log on the server. This isn't a big deal but very annoying to most DBAs.

To correct this, you must log in to the SQL Server. Then, using the SQL Server Management Studio (or Enterprise Manager), locate the timer jobs under the SQL Server Agent | Jobs. The name of the SSP is always part of the job name; right-click on this and then select Delete. Note: The application pool(s) created within IIS on the front-end web servers must also be removed. Open Internet Information Services (IIS) Manager from Administrative tools, right-click on each of the leftover application pools, and then select Delete.

Configuring Shared Services Providers (MOSS Only)

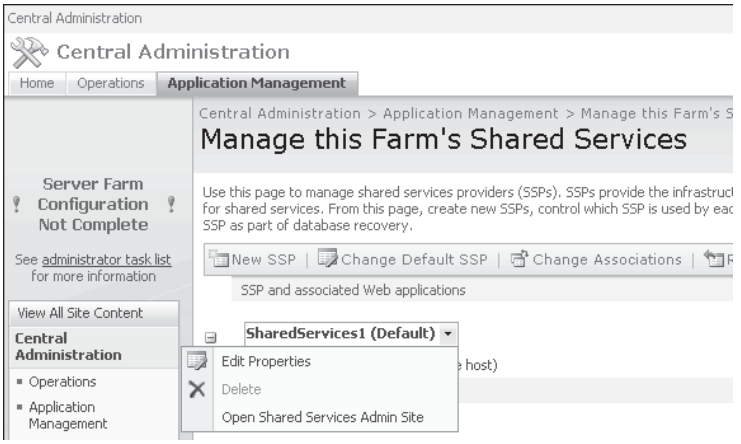
Once an SSP has been installed, some of the services must be configured before it can be used; this includes defining a connection for Active Directory, starting Office SharePoint Search, and so on. Remember, I'm only covering the configuration needed for the initial install; all of the options in the SSP administration are covered in the next chapter.

SSPs have an Administration site that is dedicated to the functions it manages, specifically user profiles, search, and so on. This is a one-for-one, with each SSP having its own, and they are not shared. After the installation, the services the SSP runs should be configured prior to creating any sites.

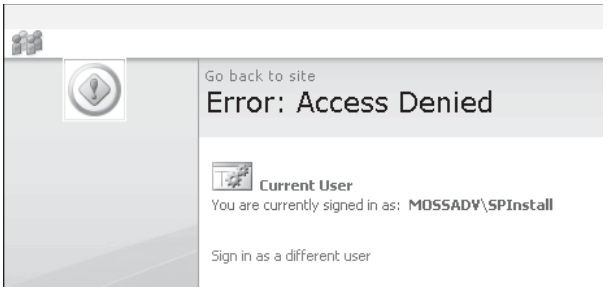
To start configuration of the SSP, do the following:

1. From the SharePoint 3.0 Central Administration site, on the top link bar, click Application Management.

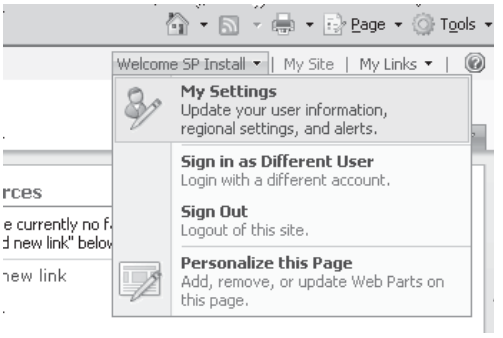
2. On the Application Management page, in the Office SharePoint Server Shared Services section, click Create or configure this farm's shared services.
3. On the Manage this Farm's Shared Services page, double-click or mouse over the name of the SSP and then select Open Shared Services Admin Site as shown:



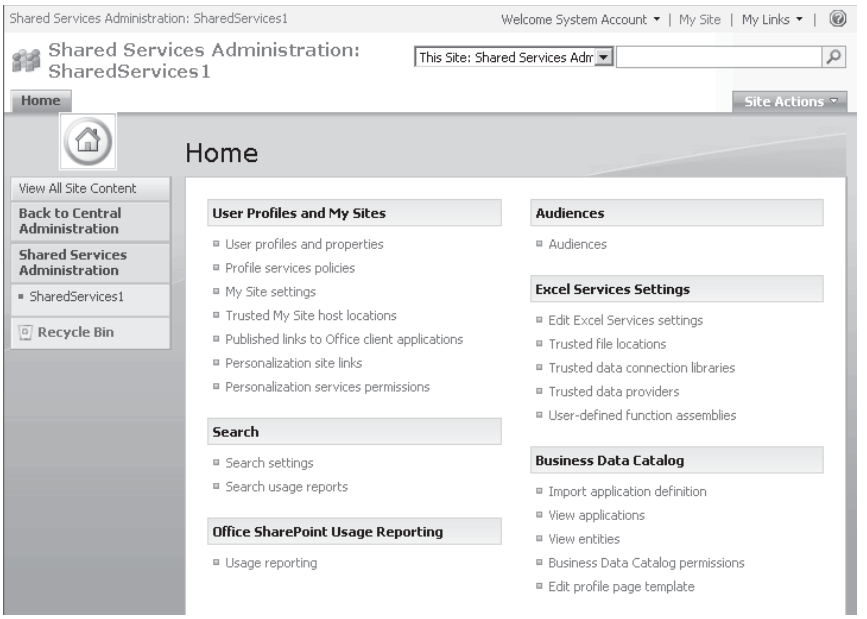
4. There is a good chance you will receive the following:



5. This is because if you are doing the install correctly, you are still logged in as the SharePoint Installation account. However at this point, the only valid user in Shared Services is the SharePoint Services (SPServices) account (designated as the System Admin account). Click the Sign in as a different user link and then log in as the SharePoint Services (you can also use the menu option as shown):



6. Return to Shared Services Administration and open the SSP Admin site (or click the name), and (logged in as the Service account) it should open as shown:



NOTE *You cannot add accounts when the SSP is created. If you want to add additional process accounts to Shared Services at this time, you can do this from SharePoint 3.0 Central Administration | Shared Services Administration. When the list of SSPs is displayed, mouse over the SSP and select Edit Properties; when the properties page is displayed, scroll to the bottom where you can add additional accounts as shown:*



NOTE *After an SSP Administration site has been created, you can also add users via the Site Settings like any other SharePoint site.*

7. If you return to the SharePoint 3.0 Central Administration home page, you will see that by creating the first SSP, the server farm is considered complete so the warning message is no longer shown; in addition, some tasks are removed from the Administration Task list (which depends on the installation type).

Creating the First Site

Once the basic settings for WSS and MOSS have been completed (MOSS including the setup of the SSP), you can begin creating sites. Before you create a site though, you must define an application to host it just like we did previously for the Shared Services Provider setup. Remember that an application can host many sites, but a site is only attached to one application.

If you used the Basic installation method, this was already done for you; a new application was created at port 80 and a new Site Collection (top level site) is created there (you can skip on to the next step if this is the case). In the Advanced installation, it is necessary for you to define these including selecting which port the site will run under.

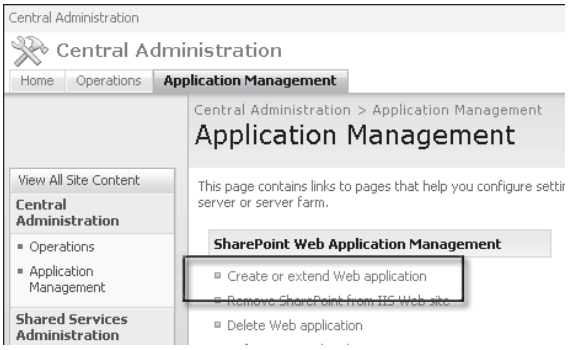
NOTE *As I mentioned at the start of the configuration, you may or may not be ready to create a site. Many of the remaining tasks are application-specific and may or may not be needed for the type of site you will use. If you're not sure, this is another good stopping point.*

To get started, you need to create an application and you must also decide which port it will run under. By default, SharePoint expects to create a new application and will assign a random (unused) port number (sometimes port 80, if it's not in use), but you can also choose an existing virtual, including the Default website installed with IIS. The actual port number you use depends on your environment: firewalls, public IP access, DMZs, and so on, and all play a part in that decision. What does matter is if you use the Default website, you may need to make a few adjustments, specifically around using SQL Reports on the same server (see the Basic Installation section in Chapter 5 for instructions on what needs to be changed). If you wish to use port 80 and not the Default website, SharePoint will stop the site at Default website and create a new Virtual for port 80 (you should verify this in IIS Manager).

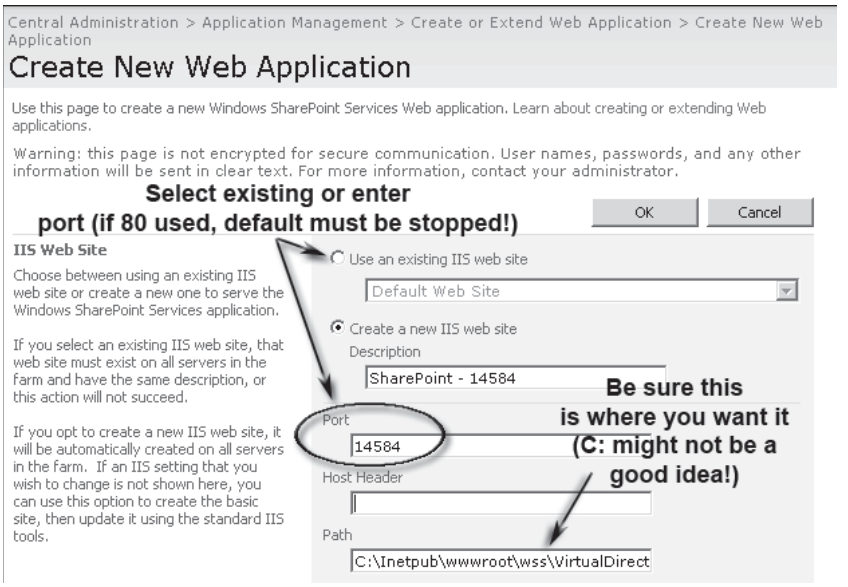
NOTE *For a public-facing site, be aware that you will have to provide port-forwarding through your firewall to enable direct access to non-port 80 sites.*

To create the application for the first site, do the following:

1. Open SharePoint 3.0 Central Administration and click the Application Management link.
2. Under the SharePoint Web Application Management section, click Create or extend Web application:



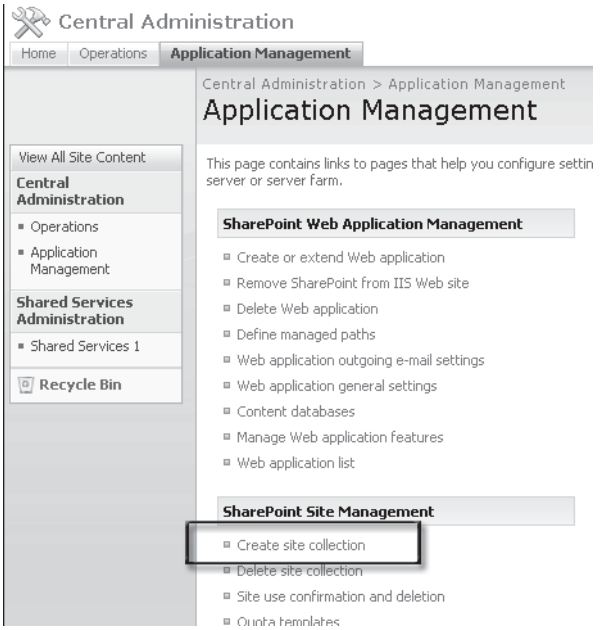
3. From the Create or Extend Web Application page, click the Create a new Web Application link. Then, on the Create New Web Application page, select the port number and the name to be assigned to the application (this is the name that is displayed in the IIS Manager; within SharePoint you always select applications by the Port number):



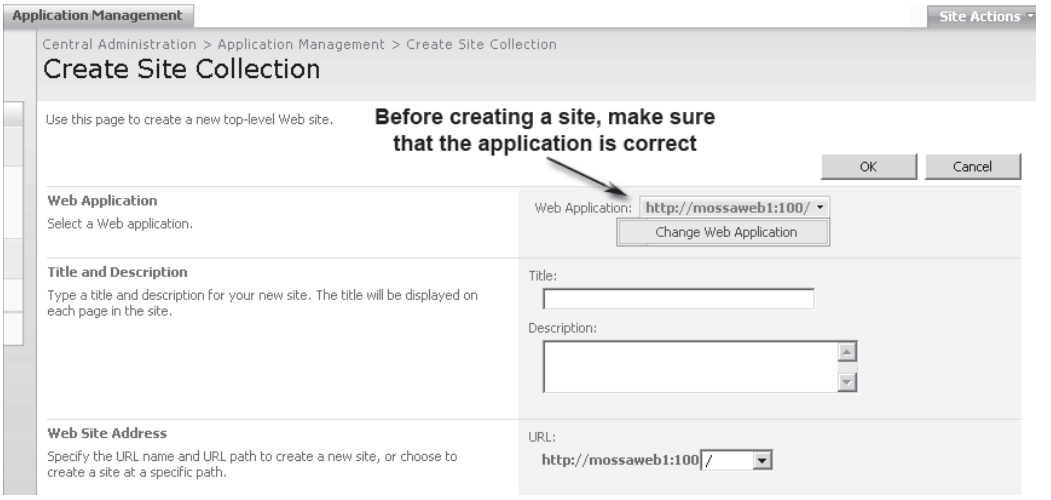
NOTE *In this example, I've set 14584 as the port. If I'd used 80, I would have stopped the Default website first (because otherwise creation will fail).*

4. Scroll down to complete the rest of the sections as you did for the Shared Services Applications previously. Specify the account to be used for the Application Pool (the SharePoint Application Pool account, SPAppPool) and be sure to change the database name from WSS_Content to something else (in the example, I've used WSS_Content_S14584).
5. Click OK to create the application—remember that if you selected Restart IIS Automatically, you'll get an error page and you'll have to return to the SharePoint 3.0 Central Administration home page, and then to the Application Management page.

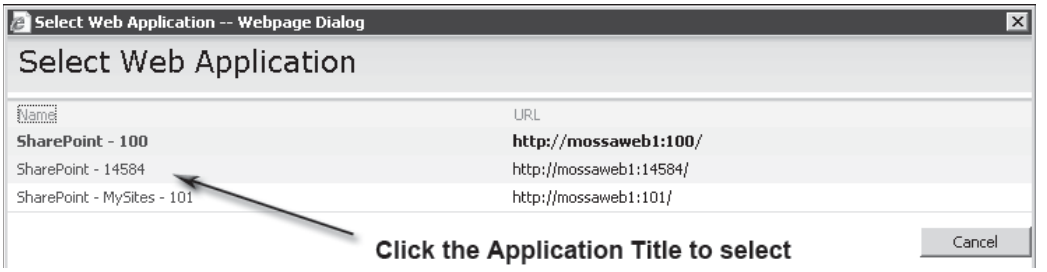
6. In WSS, skip the next steps and go to Creating the first site for WSS Advanced Install. To create the first site (in MOSS) from the Application Management page, click the Create Site Collection link under the SharePoint Site Management group:



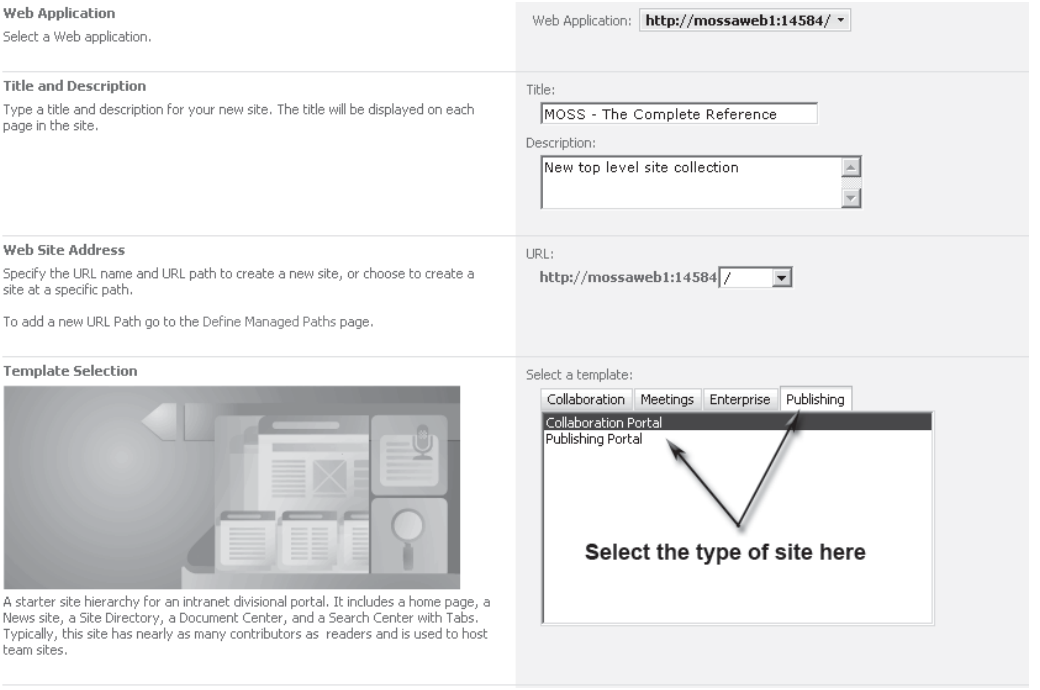
7. This opens the Create Site Collection page:



8. Notice the Web Application setting, which indicates the application currently selected. If the one shown is not the one you just created, you must change it using the drop-down list link as shown previously. This will open up the Select Web Application dialog box:

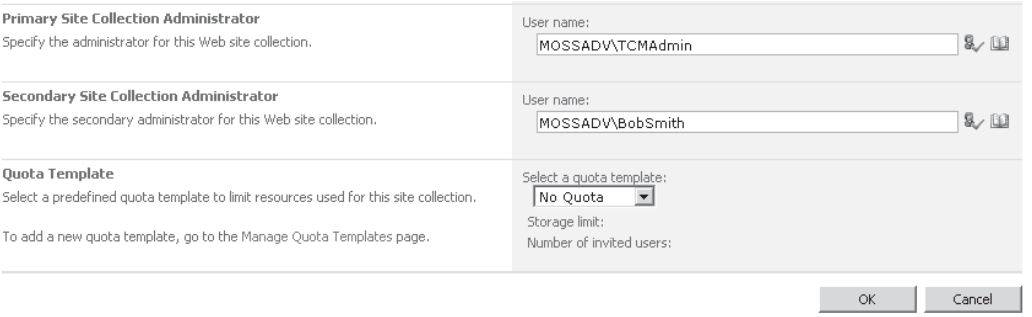


9. Click on the name of the application (in this example, 14584) to select it (you'll see this in the upper right when you return to the Create Site Collection page so you know you are using the correct Application. Enter in the Title and Description of the site (this will appear as the Site Title and blurb that appears on the home page of the site), choose the URL (this is either / for the root folder (the default) or /sites/), and then choose the type of site from the template selections:

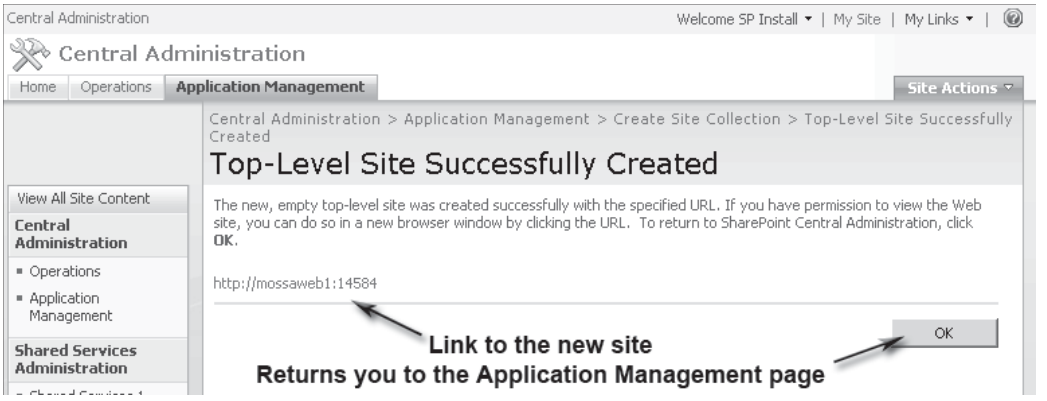


NOTE *When you have additional site templates installed or sites that have been saved as templates, they appear under a new tab called Other. You'll never see this in a new install, however.*

10. Next, you need to specify the Primary and Secondary Site Collection Administrators. The users you designate here are the first Administrators in the site, but you can also set a quota here (a quota enables you to limit the amount of storage a particular site can use, which is explained in detail in Chapter 7):



11. Click OK to create the site. You will now be presented with a ticker page while the site is built, and, when the process is complete, you should see the successful completion page:



12. At this point, you can actually view the site via the link shown or by clicking OK to return to the Application Management page. Since you still have more to configure, you should now return to Application Management. However, you should test the link to the site to view it. For the template selected in the example (Collaboration site), the new site looks like this:



NOTE *You might encounter an error page when you open the site for the first time. If this happens, simply click the Go back to site link and it should render normally.*

As I mentioned, there is still more to configure for the server farm, however, if you want to create additional sites under the collection or additional applications and sites, you can do that now. That said, I suggest you complete the entire configuration for the first site before creating others.

Creating the first site for WSS Advanced Install If you are installing WSS with the Advanced install, after you have created the first application, the Create Site Collection link does not work. Instead, you'll receive a message stating, "The page cannot be displayed because your server's current configuration does not support it. To perform this task, use the Command-Line operations in Stsadm.exe."

To do this, you must go to the command line, set the default to the SharePoint Hive bin folder (<drive>\Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\bin), and run the STSAdm command to create the first site collection. You must also specify a site template in this command covered in detail in Chapter 10. But for now, the WSS templates are shown in Table 6-1.

The site templates designated previously are from the WEBTEMP.XML file in SharePoint, whereas the complete command format for STSAdm can be found in Appendix A. To create a top-level site using the Team Site, the command would be:

```
stsadm.exe -o createsite -url http://<server> -ownerlogin <domain>\<login> -
ownere-mail <name>@<domain>.com -sitetemplate STS#0 -title Site Title –description
Description of the site
```

For example:

```
stsadm.exe -o createsite -url http://wssaweb1 -ownerlogin WSSADV\SiteAdmin -
ownere-mail siteadmin@wssadv.com -sitetemplate STS#0 -title SharePoint - 80 -
description Top Level Site
```

TABLE 6-1 SharePoint Services Standard Site Template Types

Template Name	Site Type
STS#0	Team Site
STS#1	Blank Site
STS#2	Document Workspace
MPS#0	Basic Meeting Workspace
MPS#1	Blank Meeting Workspace
MPS#2	Decision Meeting Workspace
MPS#3	Social Meeting Workspace
MPS#4	Multipage Meeting Workspace
WIKI#0	Wiki Site
BLOG#0	Blog

The command should be on one line (no line breaks), and there are some additional parameters that can be set (such as a secondary owner). You can refer to Appendix A for the complete list of options. Be aware that after the site is created, you will have to log in either as the account you specified previously or as the SharePoint Installation account (SPInstall), because they will be the only allowed users in the site after it is created. If the account was created previously (for example, the last example creates an account called siteadmin), you will also have to set the password for that account before it can be used.

NOTE *If you have Active Directory Account Auto creation enabled, this will create an account but will not automatically create an e-mail account if not using Exchange.*

Correcting Site Access Issues

After the site is created, there are a few things that can cause the site to be inaccessible.

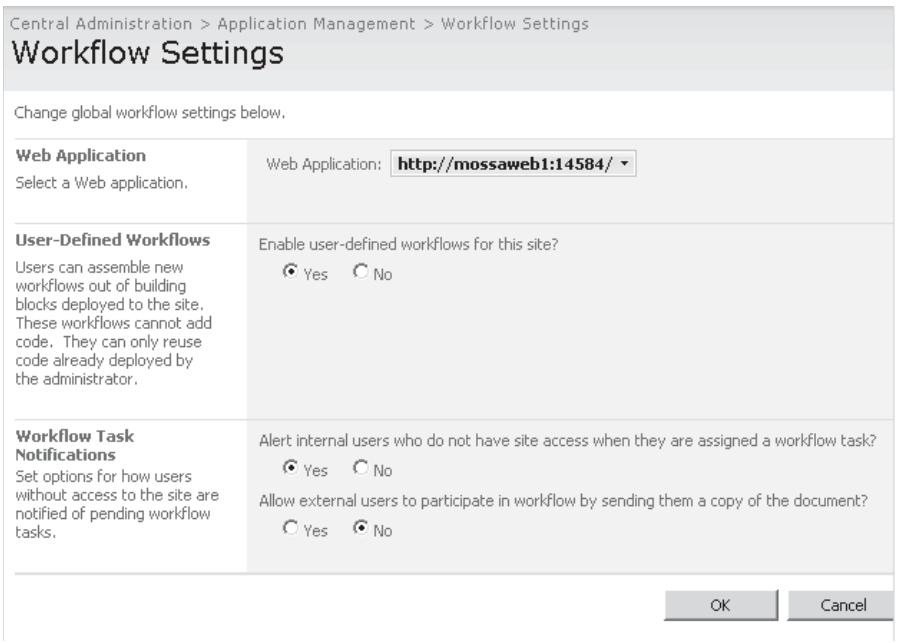
If the Basic installation was used and the top-level site is not accessible, this can be caused by the account used not having sufficient privileges to operate as an administrator of the site. To correct this, verify that the account used has ACL rights to the virtual directories, verify the password and account settings in Active Directory, verify that it is a member of the SharePoint Administrators Security Group, and verify/add rights for Impersonating a client after authentication.

If the Advanced Option for creating accounts automatically is enabled and you use the Advanced installation option, the accounts you specify in the STSAdm command line are created without using existing accounts. This means that if you had an existing account named SiteAdmin, using this account on the command line will create a new account (in the SharePoint OU) with a number added to the name (such as SiteAdmin5). The problem is that you don't know the password to this account and you cannot log in using any other account. A side problem of this is that you cannot change Site Administrator's using Central Administration either, because it simply returns an error page that states "unable to complete this operation." To correct this problem, use Active Directory to reset the password, and then you must assign the account all the rights required (like adding it as a member to the SharePoint Administrators Security Group). Run IISReset after doing this and try to connect to the site again.

Configure Workflow Settings

Once the server farm has been set up (all services enabled, the first application/site built, and in the case of MOSS, the default SSP installed), you can begin finalizing the operational side of the configuration. The next step here is to configure the workflow settings for the Application you just created for the first site. The workflow options here allow you to enable or disable whether user-defined workflows (those created using SharePoint Designer) are enabled or not. If enabled, SharePoint Designer workflows can be published to sites; if disabled, they cannot.

To set these, open the SharePoint 3.0 Central Administration and click the Application Management link. Under the Workflow Management group, click the Workflow Settings link to display the settings page:



As you can see, there's not much to set here. First, make sure that you are looking at the right web application (if not, change it), and then choose the following:

- **User Defined Workflows** When this is enabled, it means Designer built workflows can be published and used in a site. When disabled, it prevents workflows being published to the site (any that are there already will not be disabled). Be aware that the settings here do not affect the out of the box workflows (Approval, and so on) that are provided by default in lists and libraries (see Chapter 16 for more information).
- **Workflow Task Notifications** This is pretty self-explanatory; the Alert Internal users option, if enabled, notifies a user when they've been assigned a workflow task in a site that they do not have access to (thus, they can contact the site owner for access). The Allow external users option, if enabled, will send a copy of the document along with notifications in a workflow (under the assumption that they have no access to the site, and this option is excellent for working with external resources).

NOTE *While you have only set the workflow settings for the first application, you might want to do the same for the Shared Services and particularly the My Site Host applications. While it is common to extend Shared Services with user defined workflow, the My Site Host might be a case in which you want to prevent them.*

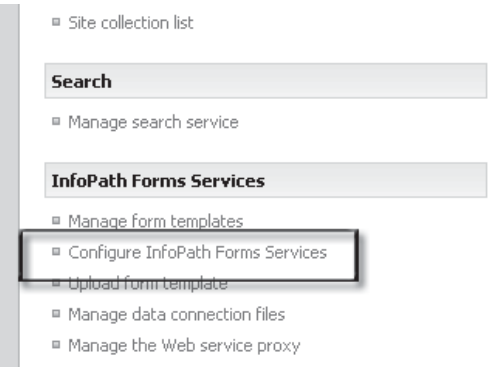
As is true with many of the configuration settings, the workflow settings are on a per-application basis. If you have additional applications or set a new one up, you will need to repeat this process for that application.

Configure Session Throttles for InfoPath Forms Service (MOSS Only)

The InfoPath Forms Service (IFS) is an add-on to SharePoint that provides the ability to render InfoPath forms created with InfoPath 2007 in a web browser. This enables users of SharePoint to view and use InfoPath forms, even if they do not have the InfoPath client installed. As many users of the 2003 version know, requiring a client program is a non-starter in most enterprise environments and with anything that is public facing. The IFS eliminates the need, treating InfoPath forms much the same as the Word Viewer. The user still gets the rich experience, though full capabilities are limited.

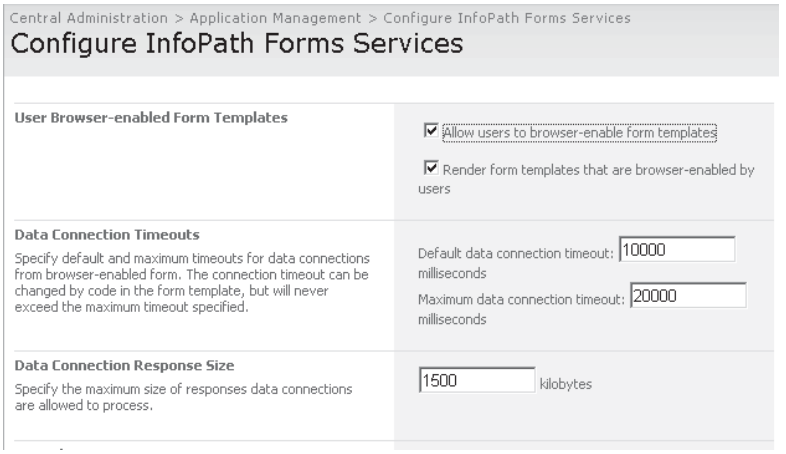
IFS is natively controlled through SharePoint 3.0 Central Administration and the settings are farm wide (not application specific) and provide the following options:

- Management of form templates
- Capability for configuring InfoPath Forms Services
- Uploading of form template
- Management of data connection files
- Management of the web service proxy



You'll use most of these when actually deploying farm wide forms, but for now you're only concerned with the session throttle settings. To configure these, go to the SharePoint 3.0 Central Administration site and click the Application Management link. Then, when the page opens, click the Configure InfoPath Forms Services link under the InfoPath Forms Services group. This will now open the configuration settings:

The throttles (or limits) defined here are used to control the resources that the service will consume and enables you to limit or increase bandwidth if needed. In addition, you can control a little of what users do and don't do. As shown previously, the first few sections cover user controls and data connection physical limits:



- **User Browser-enabled Form Templates** These allow you to enable or disable users to browser-enable form templates and render form templates that are browser-enabled (these are usually both on or both off).
- **Data Connection Timeouts** These set the default and maximum time the service will wait for a connection to a data source (10000 milliseconds = 10 seconds).
- **Data Connection Response Size** This sets the maximum amount of data that can be retrieved (the default is 15kb).

The user form settings enable or disable using forms with browsers (as I indicate, these are usually either both on or both off). The data source settings allow you to control how much and how long to wait for data responses; the more time given the more the system will wait on requests the more impact on performance. The maximum setting will prevent IFS from waiting too long. The next sections all deal with data source access.

- **HTTP data connections** This enables or disables SSL for HTTP connections to data sources (the default is enabled; for development, this will be disabled).
- **Embedded SQL authentication** This enables or disables database connections using embedded SQL connection strings that include the user name and password (the default is disabled; this should only be enabled in intranets and other low security environments).
- **Authentication to data sources** This enables or disables whether users can use authentication embedded in data connection files (the default is disabled; enable this if data connection files are secured).
- **Cross-Domain Access** This enables or disables cross-domain data connections, meaning a form can connect to a data source outside of the local domain (the default is disabled).

The HTTP data connections set controls whether access to internet data sources require SSL or not (keep in mind that in production, they always will). The remaining options are security based; the Embedded SQL selection determines whether forms can connect to data sources using a basic text user ID and password embedded in the connection string. Authentication enables you to embed connections into data connection files, and while not visible to the end user, they are still basic text. Lastly, cross-domain access determines whether a form can reach outside of the current domain. In general, the last three options here are always disabled in a production setting (unless there is a specific need, such as multiple domains in the same organization, and so on).

The last few sections are actually the throttle settings:

- **Thresholds** These set the number of postbacks, as well as the actions permitted per postback allowed on a form. This prevents excessive postbacks to the server, like a denial-of-service attack.
- **Form Session State** This controls how long sessions remain active, the maximum in memory storage for the form, and the location of where to store the form Session State (the default is the Session State Service).

The thresholds are used to limit how much activity is permitted for a single form. The default numbers shown are more than adequate and should not be altered unless there is a

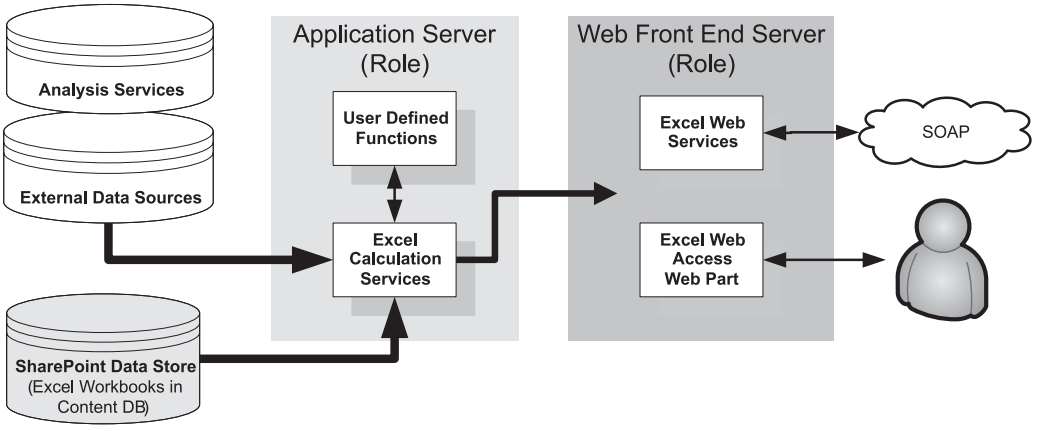
definite need to change them. The Form Session State basically keeps the state of a particular form while it is being used; you have the option of storing it in the Server process (best for low bandwidth users and public facing sites) or in the database (less vulnerable to a server failure but rendering takes extra time).

So now that you know what is there, I can say that for most of you, you won't change a single thing here and will simply accept all of the defaults. Even for those that plan heavy use of InfoPath forms, these settings shouldn't be changed until something in the environment requires it (for example, reducing the amount of connection time allowed or using data connection files). When you have completed reviewing the settings, click the OK button to return to the Application Management page.

Add/Change Excel Services Trusted Locations (MOSS Only)

A new feature provided within MOSS Shared Services is the introduction of Excel Calculation Services (ECS). ECS is a stand-alone product (an add-on to Microsoft Office Excel) that is tightly integrated into SharePoint. ECS provides the capability of rendering full-blown spread sheets directly in a browser while using SharePoint for both interim and permanent storage. The primary purpose of this is to leverage the capabilities of Excel combined with SharePoint storage and external data sources to produce data drill-down capabilities, custom reporting, graphing and charting, and so on. While SharePoint has always supported Excel as documents, ECS exposes the Excel functionality through the SharePoint user interface.

There are a few components involved in the Excel Calculation Service. Part of ECS resides within the Application Role and the other part within the Web Role on the SharePoint Farm, providing access to users through the User Interface and delivering a web service for external systems and other web services. The application side of the functionality handles all of the communication to data sources, while the web side handles the requests from both users and external processes. The conceptual overview looks like this:



ECS is completely server-based, and all calculation operations (and data source access) is handled via an Application server and rendered by a web server. In addition, since Excel data can take some time to generate as well as render, built into the ECS architecture is automatic load balancing when more than one Application Server is available. This enables requests to be balanced between servers, preventing someone loading a large spread sheet from tying up all requests.

For ECS to work within SharePoint, there is a dual setup process. The services must be configured as I will explain next, but for actual operations, document libraries must be created within SharePoint for use as interim storage. I will touch on that process in exploring the user interface later but for now, the base settings (sans the document libraries) can be set at any time.

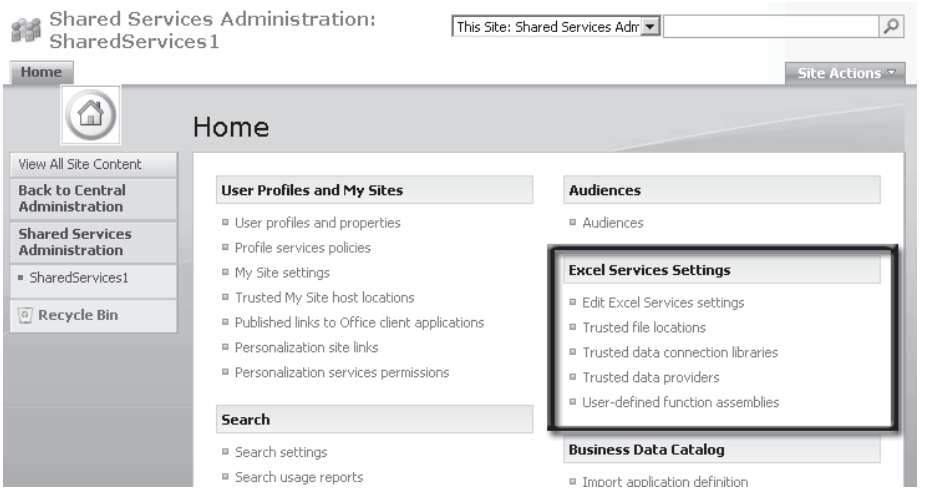
Excel Services Setup in Shared Services

Excel Services runs under a Shared Services Provider since its farm-wide service is available to all sites (it can be enabled and disabled at the site level). There are two requirements for this service to be set up: the Excel Calculation Service itself must be running on at least one of the servers in the farm and the default Shared Service Provider (SSP) must be installed.

First, verify that the Excel Service is running on at least one web server (and should be on all front-end servers). You can check this via SharePoint 3.0 Central Administration | Operations | Services on Server; Excel Calculation Services should be in the Started state.

NOTE *There is a known issue with Excel Services in which the system generates an exception error when the service starts. See the Microsoft article on this at http://support.microsoft.com/default.aspx?scid=kb;en-us;928626.*

To begin the configuration, open the Shared Services Provider Administration site by clicking on the SSP name link in the left navigation of Central Admin:

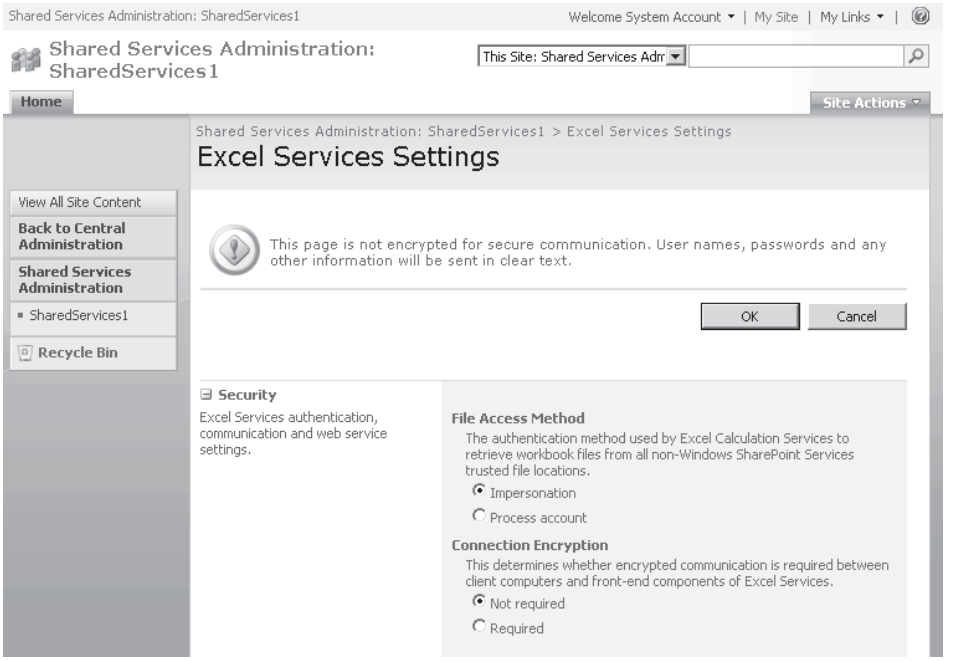


From the SSP Admin home page, you can manage all of the Shared Services, including Excel Services (highlighted). Technically speaking for configuration purposes, the only true requirement for this setup is to define the Trusted File Locations; however, since all of the settings are interrelated, I'll cover each of them below.

NOTE *If you are not logged in as the SharePoint Service account you used when you created the SSP, you will have to logout and back in to access the Admin site.*

Setting Excel Service Settings

These settings are specifically for the management of the service and affect how it operates on a farm wide basis. From the SSP Admin home page, click the Edit Excel Services settings link to open the settings page:



Since the settings here are pretty detailed, I'll briefly cover the each group of settings in a list format. You can follow along with the screen displayed next.

Security File Access Method sets the method to be used when ECS retrieves workbook files from all non-SharePoint trusted locations (file systems and so on). The options include Impersonation, which means a specific user will be used or a Process account indicating it will use a process account (like SPContent) to access files. Note that I recommend using the Impersonation method to enable better control and tracing capabilities.

Connection Encryption sets whether encryption is required when communicating with a client computer (this should be enabled if possible).

Load Balancing There are three schemes for load-balancing operations. Workbook URL (the default) uses the web load balancing; Round Robin rotates requests to each server (linear); and Local keeps requests locally. You can also set the Retry interval (in seconds) for connection failures; the default is 30, which is generally acceptable for most environments.

Session Management This defines the maximum number of sessions a single user can start. A value of -1 means no limit and 25 is the default. I suggest that you set a realistic limit based on typical user usage (the less the better; some users have a tendency to open up lots of windows, but this can tie up connections for other users).

Memory Utilization The maximum number of Private Bytes defines the memory in megabytes (MB) that Excel Services will use. A value of -1 (the default) means no more than 50 percent of available capacity.

Memory Cache Threshold defines the memory that can be allocated to inactive objects in Cache; 0 means no cache.

Maximum Unused Object Age defines how long (in minutes up to 24 days) an unused item can sit in cache, where a value of 1 means no maximum. I suggest you set this to a reasonable number that will reflect your typical work cycle; if things are accessed weekly, extend this time, but if daily, reduce this time.

Workbook Cache Workbook Cache Location can either be blank, which means use a subdirectory in the system temporary folder, or you can specify a specific file location (I highly recommend you specify a location other than the system disk).

Maximum Size of the Workbook Cache sets the total size in megabytes (MB) that the Cache can expand to. The default is 40960 (4 GB); my recommendation is that if you have the space, use it.

Caching of Unused Files enables or disables caching of files that are no longer active. This is enabled by default but I suggest you disable this to prevent clutter.

External Data Connection Lifetime sets the maximum time a connection should be allowed to stay open (in seconds up to 24 days), with a value of -1 indicating never force disconnect. I recommend you keep this to sixty (60) minutes or less.

Unattended Service Account settings are used to set the credentials ECS uses when accessing data sources that require a user name and password to access (like a Site). If the account information is not set here, unattended connections of this type will fail. I suggest you use the SharePoint Content (SPContent) or SharePoint Search (SPSearch) account here if you don't want to designate a specific domain account. Both of these accounts should have access to the sites where a login might be necessary.

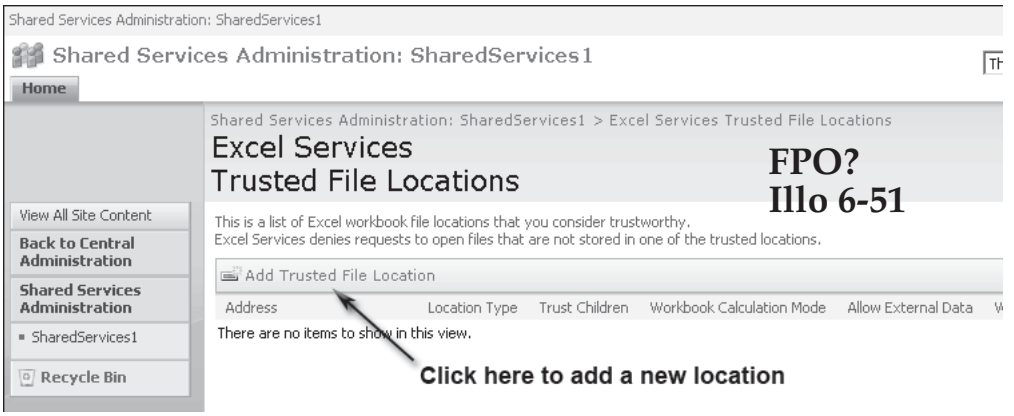
Once you have completed reviewing and updating any settings necessary, click the OK button to save your changes. The settings here are usually "set and forget," and once you've defined them, you won't be coming back to this often.

Setting Trusted File Locations

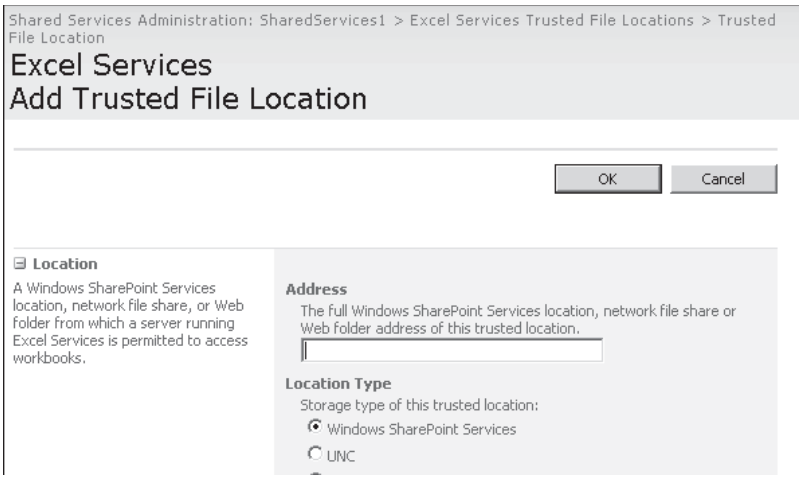
The Trusted File Locations are where Excel Services can store workbooks during the course of calculations and rendering. Any number of trusted locations can be specified for a single SSP. For ECS to work, however, you must define at least one location.

NOTE *As mentioned in the overview, this particular step requires that any library, file share, or web share be created before it can be added here. You will be back here to add additional libraries, shares, and so on as site use expands.*

From the SSP Admin site home page in the Excel Service Settings, click the Trusted file locations link. This opens the Trusted File Locations list page (this is page contains a simple list, like any other SharePoint list):



To add a new location, click the Add Trusted File Location link, which will display the Add Trusted File Location page:



Again, the settings here are pretty detailed, so I'll cover each group of settings separately.

Location Address sets the address of the location and should match the format for the type of location that you are specifying (location type next).

Location Type is one of three types: Windows SharePoint Services specifies a SharePoint site URL; UNC specifies a file share using a UNC address; and HTTP, which specifies an HTTP URL as the trusted file location.

Children Trusted either enables or disables trust of child folders. Note that although you can improve manageability by trusting children, you can also create a potential security issue by enabling sub sites and subdirectories of trusted locations to be automatically trusted as soon as they are created.

Description is simply free text you can use to describe the location and its purpose.

Sessions Management Session Timeout sets the time (in seconds up to 24 days) that an Excel Calculation Services session can be open and inactive before it is shut down, as measured from the end of each Open request. A value of -1 sets no time-out, 0 expires the session at the end of a single Open request.

Short Session Timeout sets the time (in seconds up to 24 days) that an Excel Calculation Services session can be open and inactive, prior to any user interaction, before the session is shut down, and as measured from the end of the initial Open request. A value of -1 sets no time-out, 0 expires the session at the end of a single Open request.

Maximum Request Duration sets the time (in seconds up to 24 days) for the maximum duration of a single request in a session; a value of -1 sets no duration limit. Note, that if you limit the amount of time that a single request can last, you can help reduce the risk of a denial-of-service attack.

Workbook Properties Maximum Workbook Size sets the size in megabytes (MB) for the maximum size of workbook that Excel Calculation Services can open. Note that performance and resource availability can be compromised when users open extremely large workbooks. If you do not control the allowable size that workbooks running in open Excel Calculation Services sessions can have, you risk having users exceed your resource capacity and overloading the server.

Maximum Chart Size sets the size in megabytes (MB) for the maximum size of chart that can be opened by Excel Calculation Services.

Calculation Behavior Volatile Function Cache Lifetime sets the maximum time in seconds that a computed value for a volatile function is cached for automatic recalculations. A value of -1 will calculate the volatile function once on open, and 0 will continuously calculate the function.

Workbook Calculation Mode is set to one of the following:

- **File** Set this option to perform calculations as specified in the file.
- **Manual** Set this option if you want recalculation to occur only when a Calculate request is received.
- **Automatic** Set this option if you want any change to a value to cause the recalculation of all other values that are dependent on that value. Also, volatile functions are called if their time-out has expired.
- **Automatic except data tables** Set this option if you want any change to a value to cause the recalculation of all other values dependent on that value, so long as the values are not in a data table. Also, volatile functions are called if their time-out has expired.

External Data Allow External Data is set to one of the following:

- **None** Set this option to disable all external data connections for the trusted file location.
- **Trusted data connection libraries only** Set this option to only enable connections to data sources that are part of a trusted data connection library. The server will ignore settings embedded in the worksheet.
- **Trusted data connection libraries and embedded** Set this option to instruct the server to try embedded connections to data sources only after the server fails to try connections to data sources that are part of a trusted data connection library.

TIP *If you are managing data connections for a small number of workbook authors, consider selecting this option. This enables workbook authors to embed direct connections to external data sources in their workbooks, but to still have access to trusted data connection libraries if the embedded links fail.*

Warn on Refresh enables or disables display of a warning before refreshing external data for files in this location. When you select this option, you ensure that external data is not automatically refreshed without user interaction.

Stop When Refresh on Open Fails enables or disables automatic stopping of an open operation on a file in this trusted file location under the following circumstances:

- The file contains a Refresh on Open data connection.
- The file cannot be refreshed while it is opening.
- The user does not have an Open user right to the file.

External Data Cache Lifetime in the Automatic refresh (periodic/on-open) sets the maximum time (in seconds up to 24 days) that the system can use external data query results for automatically refreshed external query results. A value of 1 will prevent data refresh after the first query.

External Data Cache Lifetime in the Manual refresh sets the maximum time (in seconds up to 24 days) that the system can use external data query results for automatically refreshed external query results. A value of -1 will prevent data refresh after the first query.

Maximum Concurrent Queries per Session sets the maximum number of queries that can run concurrently during a single session.

User-Defined Functions User-Defined Functions Allowed enables or disables user-defined functions to be called from workbooks that are located in this trusted file location.

Click OK to save the trusted location and return to the Trusted Locations list page. You can continue to add additional locations as needed.

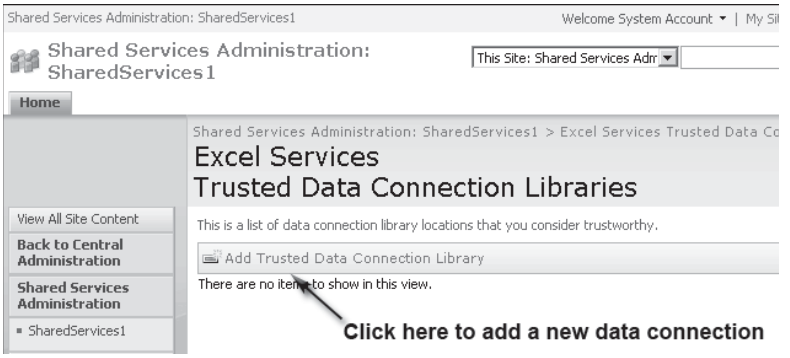
Setting Trusted Data Connection Libraries

Trusted data connection libraries are Microsoft Office SharePoint Server 2007 document libraries containing data connection files that Excel Calculation Services trusts to use to connect to databases. Excel Calculation Services does not use data connection files that are not stored in a trusted data connection library. However, data connection information can be embedded directly into a workbook that is attempting to make a connection. You can create different trusted data connection libraries for different purposes or projects, and you can tailor the settings and permissions to the libraries accordingly.

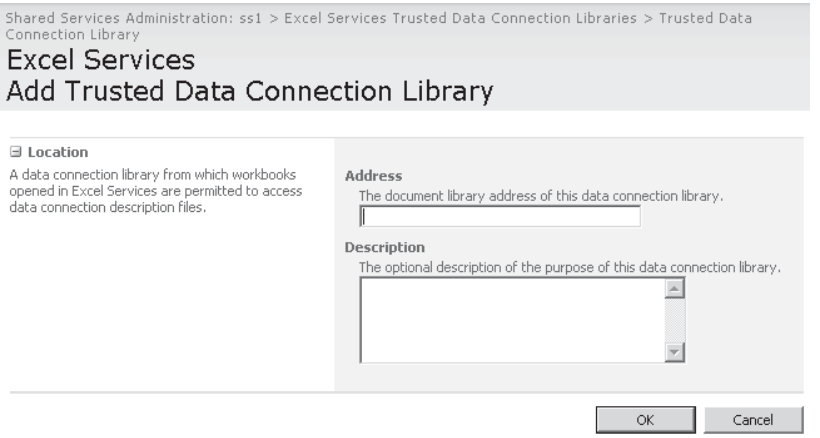
Data connections connect worksheets to data providers. One common use for data connection files stored in data connection libraries is to transfer a worksheet or a set of worksheets from a staging database to a production database. For worksheets that use the same data connection, changing the data connection file is all that is necessary to switch from one database to another, and changing the individual worksheets is not necessary. Initially, there are no trusted data connection libraries (with the exception of Report Center sites, which create their own trusted data connection file), so to store data connection files, you must create at least one to begin with.

NOTE *As mentioned in the overview, this step requires that a site and document library already be set up and available before it can be added here. You will be back here to add additional libraries, shares, and so on as site use expands.*

To manage Excel Services Trusted data connection libraries, navigate to the SSP Admin site home page in the Excel Service Settings and click the Trusted Data Connection Libraries link. This opens the Excel Services Trusted Data Connection Libraries page (this is a list just like the Trusted file locations):



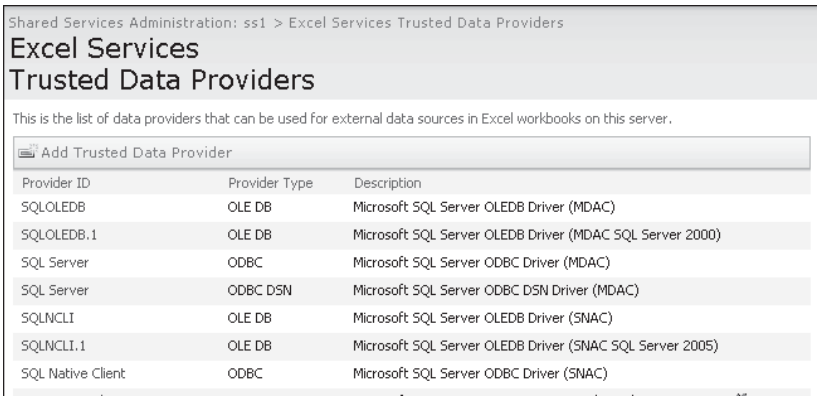
Click the Add Trusted Data Connection Library link to open the Add Connection page:



On the Excel Services Add Trusted Data Connection Library page, in the Location section, type the address of the trusted data connection library in the Address box. In the Description box you can optionally type a description of the purpose for this trusted data connection library. Click the OK button to save the connection entry.

Setting Trusted Data Providers

Trusted data providers define the "types" of database entities that are allowed for connection and the method in which they connect (native, ODBC, and so on.). For most installations, these will not need any maintenance and rarely does one need to be added. To manage Excel Services trusted data providers, navigate to the SSP Admin site home page in the Excel Service Settings and click the Trusted data providers link. This opens the Excel Services Trusted Data Providers page:

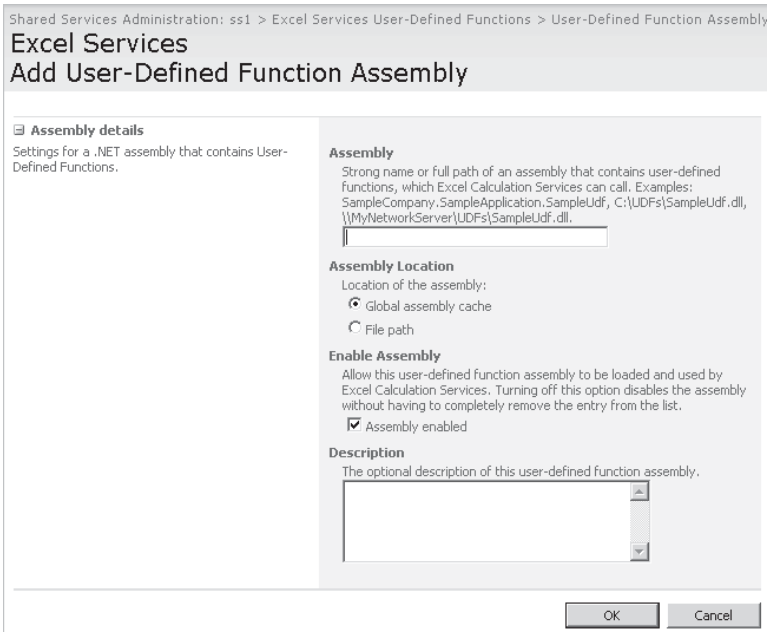


As you can see, the basic definitions are already there for SQL Server 2000, SQL Server 2005, Oracle, and DB2. Unless you have a unique environment as I mentioned, you'll likely never see this again.

User-Defined Function Assemblies

User-defined function assemblies are custom-built classes that are used to extend the capabilities of Excel Services. These functions can be incorporated into spreadsheets to provide extended data calculation and transformation services, and so on. Unlike features and web parts, ECS user-defined assemblies are added/removed and enabled/disabled here.

To manage Excel Services User-defined function assemblies, navigate to the SSP Admin site home page and under the Excel Service Settings, click the User-defined Function Assemblies link. This will open the Excel Services User-defined function assemblies page, which lists any that have been installed. To add a new one, click the Add User Defined Function Assembly link. This displays the definition page:



Adding an assembly only takes a few tasks. In the Assembly details section, under Assembly, add the strong name of the assembly (the format of the strong name is shown) and then set the Assembly location, either with the Global Assembly Cache (I recommend this) or by using a File path. The Enable Assembly check box disables or enables the use of this assembly (in other words, use one, then turn it off if needed). Finally, there is a Description which should be used to describe the function and how it's used.

For reference, this is yet another example of extensibility in SharePoint; using an assembly you have the full capabilities of .NET inside of SharePoint. This includes performing background operations within SharePoint (you have the object model available), updating remote systems and databases, starting transactions, you name it.

Service Level Settings for Shared Services (MOSS Only)

This is an optional task that involves making any final adjustments to the Shared Services settings. This includes connecting up the Active Directory import, and so on. You have the option here of deferring this task, but I suggest you get these out of the way since most of them are one-time settings.

NOTE *Be aware that all of the settings covered here apply to a single SSP; these settings must be made each time a new one is added.*

All of these settings are done from the Shared Services Provider Administration site. I'm going to assume that by now you know how to get there but just in case, let's review. Start by opening the SharePoint 3.0 Central Administration site and then clicking on the SSP name in the left navigation (under Manage Shared Services). We've already covered part of the setup when we set up Excel Services previously; what remains are a number of base items including the Business Data Catalog (we won't cover the BDC here since that is covered in its own chapter). The base settings are as follows:

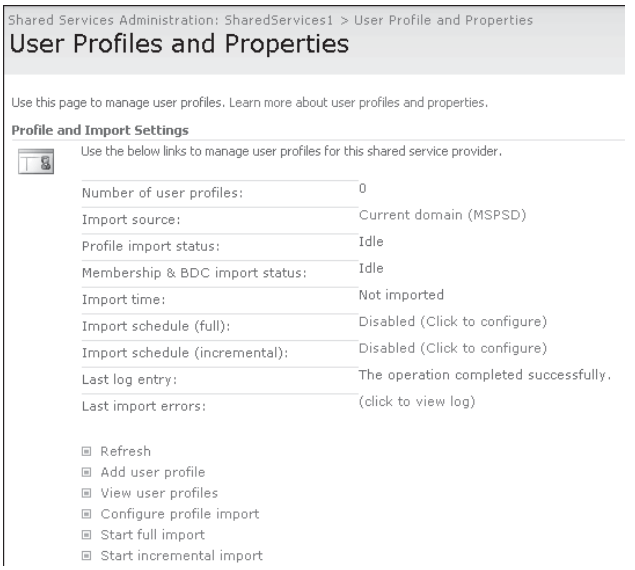
- User Profiles and My Sites Group
  - User profiles and properties
  - Profile services policies
  - My Site settings
  - Trusted My Site host locations
  - Published links to Office client applications
  - Personalization site links
  - Personalization services permissions

- Search Group
  - Search Settings
  - Search usage reports
  - Office SharePoint Usage Reporting
  - Usage Reporting
- Audiences Group
  - Audiences

As I mentioned previously, I assume you are somewhat familiar with moving around this page and know that you will access all of the previous from links on the SSP Administration site home page.

Set Up User Profiles and Properties

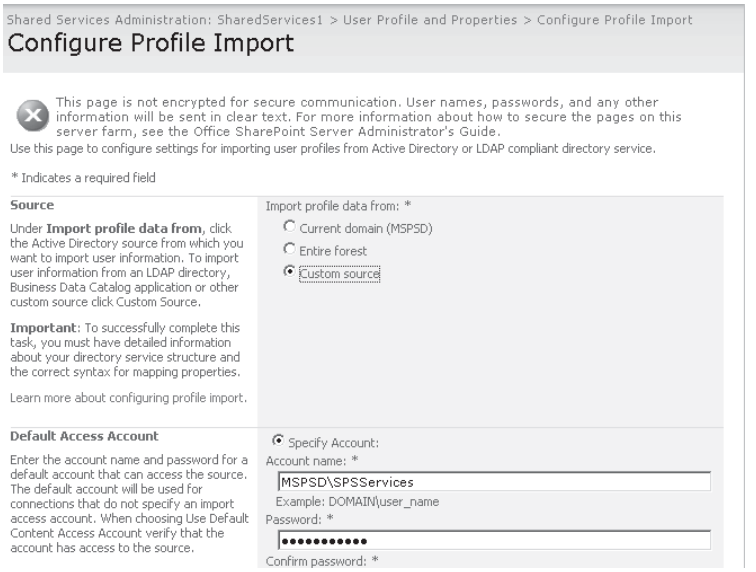
From the SSP Admin site home page, click the User profiles and properties link to open the User profile page:



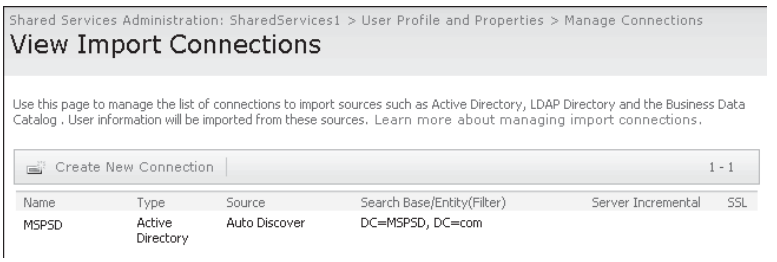
As you can see, only the default information is set (the base domain) and none of the import schedules have been set. If you are using a Single Server setup or have not set up a SharePoint Organizational unit in Active Directory, the default domain as specified does not have to be changed. However, if you have set up an OU or have a large Active Directory community and want to restrict what you import, you must configure the import settings to point to the correct location.

Setting the Active Directory Path To change or set the Active Directory import path, do the following:

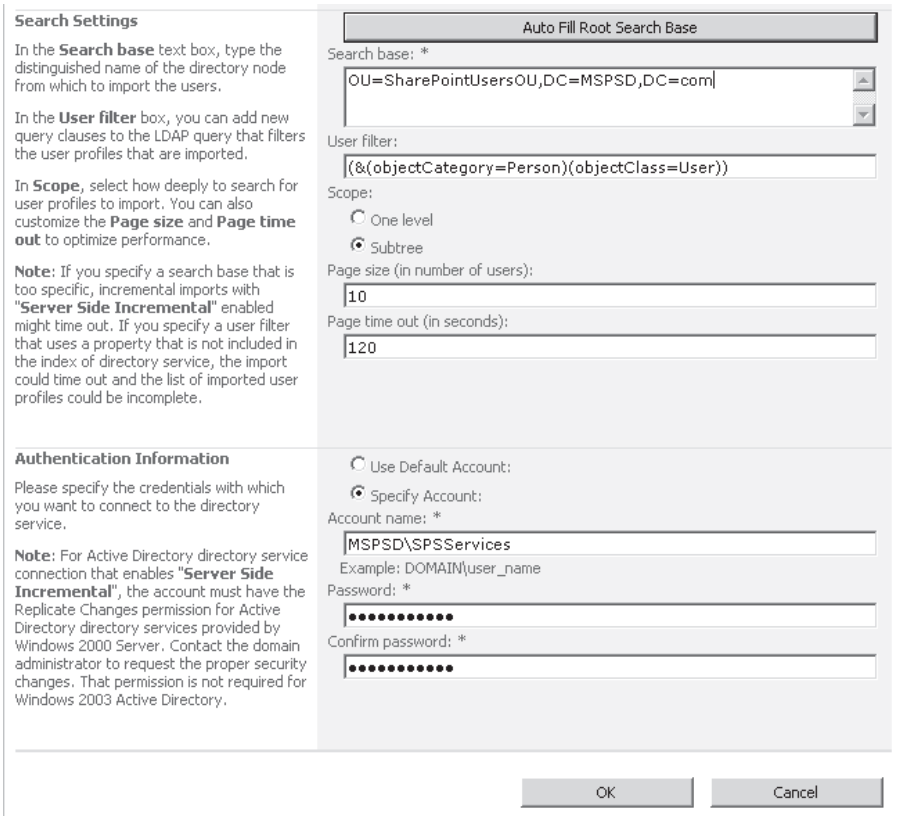
1. Locate Import Source: and click the Current Domain link. This will open the Configure Profile Import page:



2. When it opens, Current domain will be selected. Click the Custom Source radio button. Note that importing from the Entire Forest is potentially dangerous. If you are in a very large organization and are not pointing to the correct level in AD, you could import a whole bunch of accounts you do not want.
3. Under the Default Access Account, specify an account that has permissions to read Active Directory (this is usually the SharePoint Services account, also known as SPServices).
4. Below the account specification you can (optionally) specify the Full and Incremental Import schedules; the frequency depends on your environment and how often accounts are added and removed (a good rule of thumb is a full import once a month and incremental daily). Click the **OK** button, which will then display the View Import Connections page:

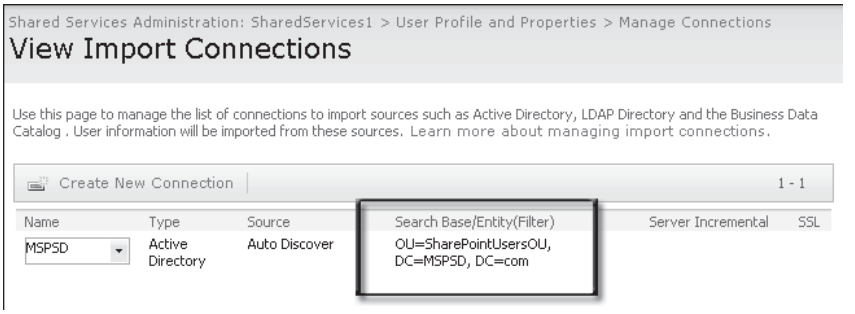


5. Notice that the current Search base is pointing to the base domain, which is MSPSD.COM in this example (this of course means the same as the Entire Forest). The Search base needs to be modified to point to the correct path. Mouse over the name of the connection and from the drop-down list, select Edit. When the Edit Connection page is displayed, scroll down to the Search Settings:

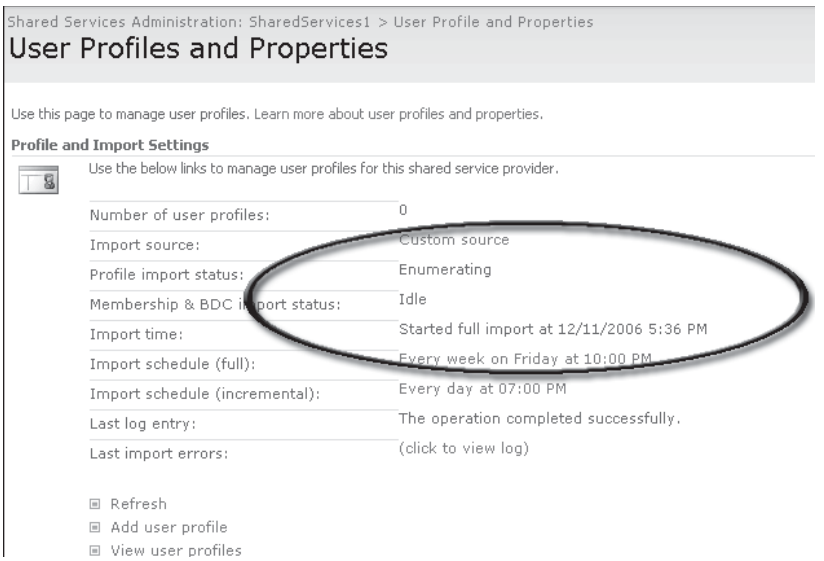


6. In the Search Base field, you need to add the path to the correct location in Active Directory. In this example, I've set up an Organization Unit (SharePointOU), so the combined path is as you see in the example.
7. Scroll down and verify that the correct account is set for Authentication and click the OK button to accept the change.

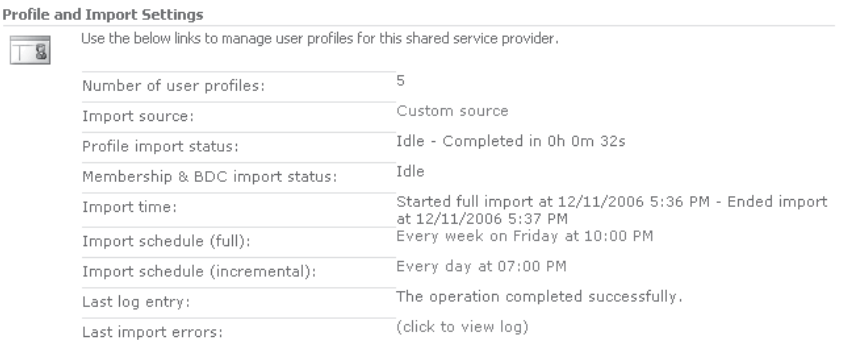
8. When you return to the View Import Profiles page, you should see the change:



9. Return to the User Profiles and Properties page and click the Start Import link. When this begins, you will see that it is loading from Active Directory:



10. When the import has completed (the page will refresh), you will see the results of the import—hopefully, without any errors:

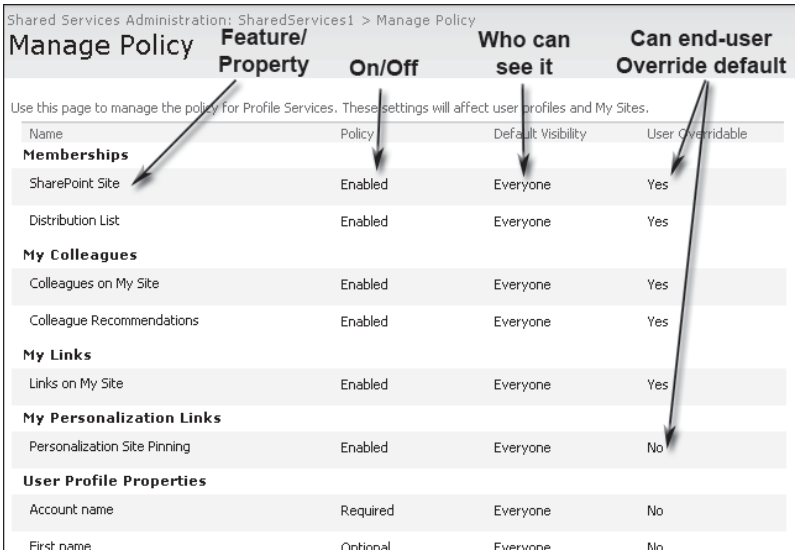


11. If you do see errors, you should check out the issue. The most likely cause for problems is the account being used to access Active Directory (usually because the account does not have rights to read AD).

Set Up Profile Services Policies

The Profile Service Policies allow you to enable/disable and customize profile features that are available to users. This includes setting required fields for profile information (name and so on) and some of the features in the My Sites. These policies are basically nothing more than on/off switches that allow you to set the privacy scope (who can view the item such as Only Me, My Manager, My Colleagues, and so on) and set which properties are end-user modifiable. In addition to some of the features (such as My Links), you can also set the policies for SharePoint's User Profile information, which is profile information maintained by SharePoint alone and not linked to Active Directory.

From the SSP Admin site home page, click the Profile services policies link to open the policies page:



To modify any of the policies, simply mouse over the item and select Edit Properties. For each item you have the following options:

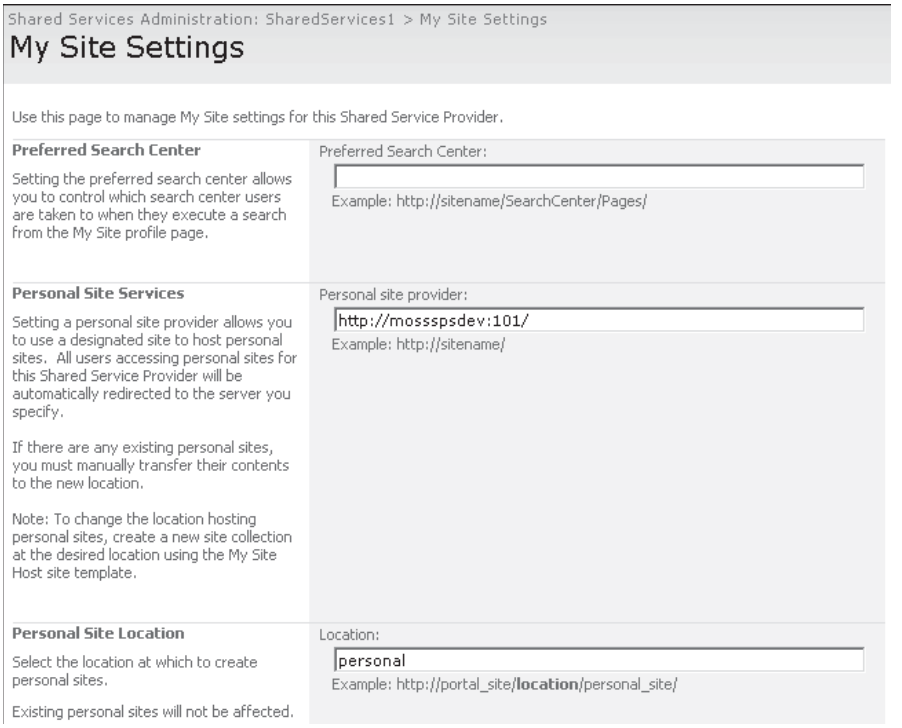
- Enable/Disable the Policy
- Set the Default Privacy: Only Me, My Manager, My Colleagues, Everyone
- Enable/Disable if the end-user can override the default policy setting

You may not wish to make any changes to these at this time (the default settings are typical for most). Down the road, however, if you add any custom user profile properties, you will need to revisit here to set the policy for the item(s) as required. When done, click the Shared Services Administration: <SSP Name> breadcrumb link at the top of the page to return to the home page of the site.

Set Up My Site Settings

The My Site settings set the ground rules for how My Sites operate as well as their scope, both local and global. Particularly important is how these settings are applied; the My Site settings apply only to the SSP that it was created with, and they affect only Sites associated with the SSP. For most organizations with a single SSP, this is not an issue; however, when multiple SSPs are being used with multiple sites, you have to keep in mind that the relationship is User | Site | SSP | My Site. If My Sites are not global (across the farm), it is possible for a user to have multiple My Sites, with one in each SSP.

Next, from the SSP Admin site home page, click the My Site settings link to open the settings page:



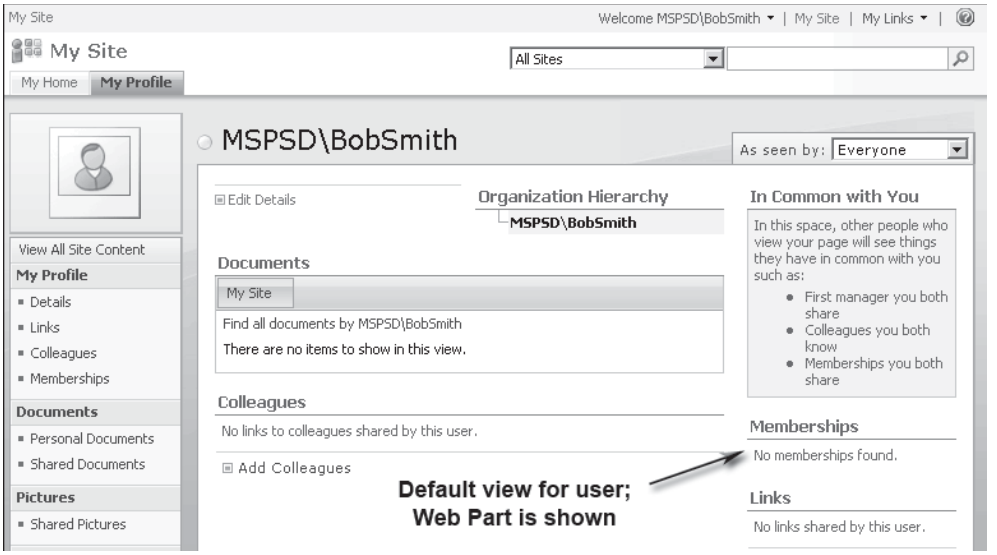
From this you have a number of setting groups as follows:

- **Preferred Search Center** This allows you to set a specific search center for users when they run a search from their My Sites.
- **Personal Site services** Personal site services point the My Site Host of a specific SSP; in my example, you created the My Sites at Port 101 as shown.
- **Personal Site Location** This group sets the path so that it's included as part of the URL to personal sites, with personal as the default. This only affects new sites; existing ones are not changed.
- **Site Naming Format** This group applies to all new sites (any existing ones are not changed) and contains one of the following:

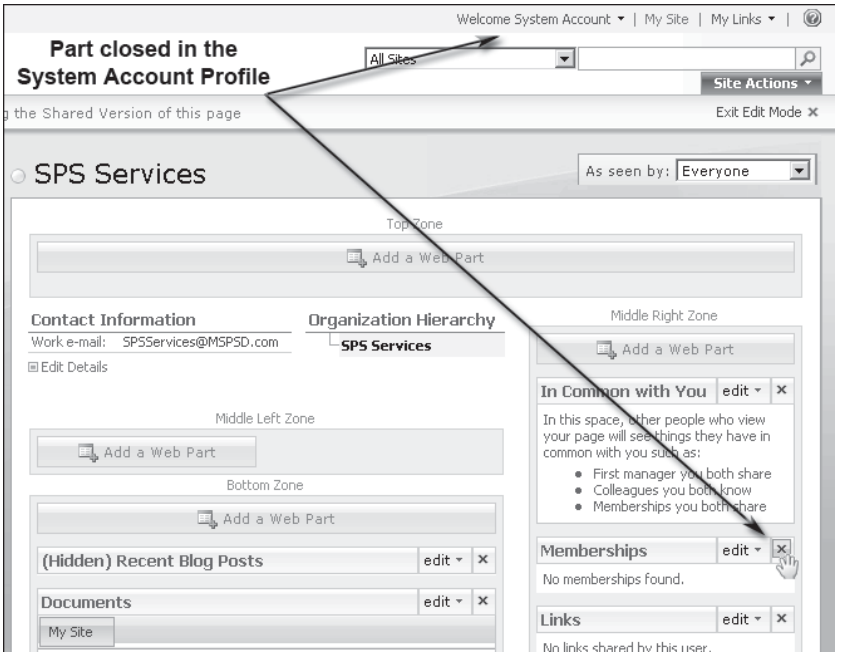
  - User Name that does not resolve conflicts. This uses the user name as the My Site name; conflicts are possible in a medium to large organization.
  - User Name that resolves conflicts using domain_username. This uses the user name as the My Site name and if there is a conflict, it resorts to domain_username.
  - Domain and user name (will not have conflicts) uses the domain_username format for all sites; I recommend this for any large organization.
- **Language Options** This enables/disables users to select the language of their personal site. If enabled, users can choose the language when their My Site is created (note: language packs must be installed for this feature to be available).
- **Multiple Deployments** This enables or disables the My Site functionality to support global deployments; if enabled, users can have a single SSP My Site but are able to browse any site regardless of the SSP attached to it. As mentioned, it is recommended that you implement a profile replication solution before this is enabled (note that Microsoft has extensive documentation available on profiles and user planning).
- **Default Reader Site Group** This sets the default reader group for the My Sites (the visitors to a user's site). The default setting is Authenticated Users which means any valid user on the network that has authenticated (against AD); in a public site, Anonymous Users would be included.

Designing the My Site Profile Page While I never found it documented in SharePoint 2003 and so far still haven't seen it mentioned anywhere for 2007 either, I might be the first to address this topic. There is a built-in feature regarding the My Profile page used in My Sites, which allows the System Account to set what appears (or does not appear) on the user My Profile pages. This, in effect, allows the System Account to set the content that appears on that page, including setting what is mandatory.

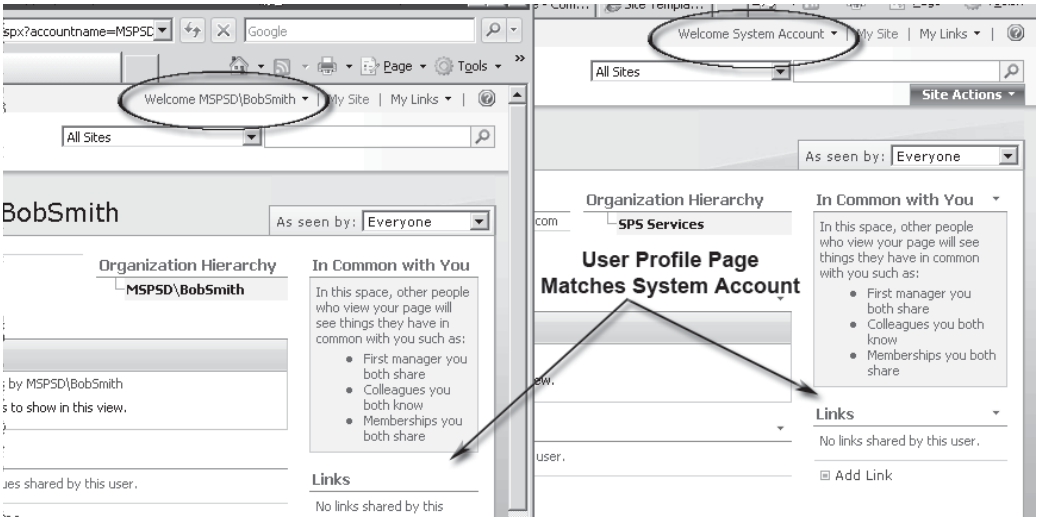
For example, if a user had an existing My Site with the default settings (because of a fresh install), the My Profile page would look something like this:



Now, if for some reason, you decided to not include something, for example, the Memberships web part, the System Account My Profile page can be edited to close the part:



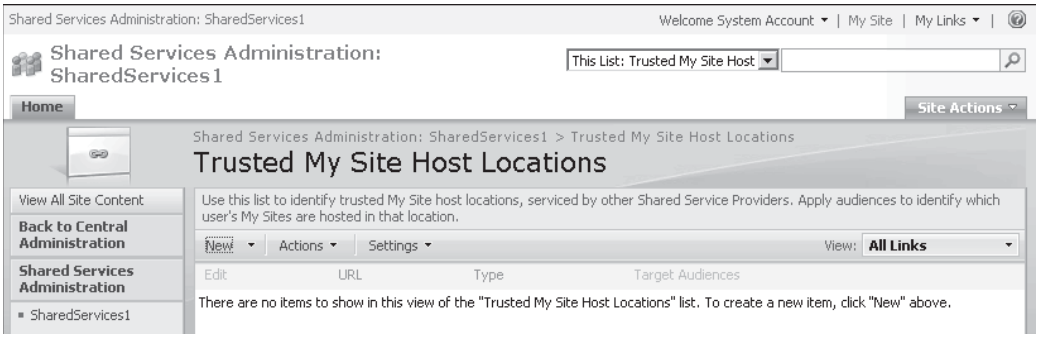
When the user refreshes (all users by the way), the part is no longer shown:



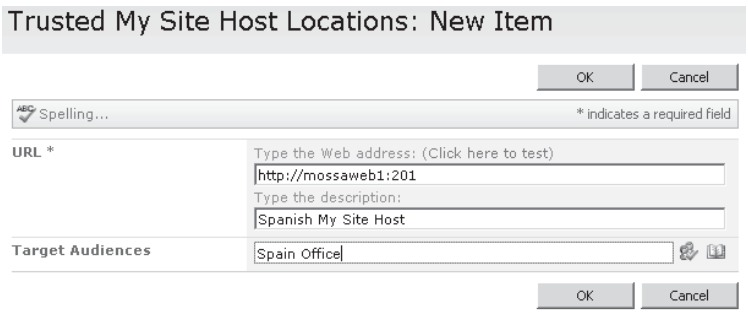
Web part settings such as whether they can be moved, closed, or so on, which are applied to the System Account My Profile page, are applied across all My Sites for the particular SSP.

Set Up Trusted My Site Host Locations

Trusted My Site locations designate My Site Hosts that are trusted by this SSP. These My Site Locations are specified by the URL to the My Site Host, the type of host and (optionally) select the target audiences (audiences are explained below) that belong to it. For reference, the Trust relationship here is much the same as a DNS trust; if a user belongs to a My Site Host that is not trusted, they will not be able to open their my site from the current site. From the SSP Admin site home page, click the Trusted My Site Host locations link to open the list page:



If you receive an error page instead of the previous page stating "unknown error," you should check your network connectivity. This is because when this page loads it attempts to validate the connection URLs (or appears to) and if the network is not available, it simply returns default error page. To add a trusted location, simply click the New link; this will open the new item page:



What is shown is just an example; at this point only the default SSP has been created and there are no audiences designated yet. (Audiences are explained next.)

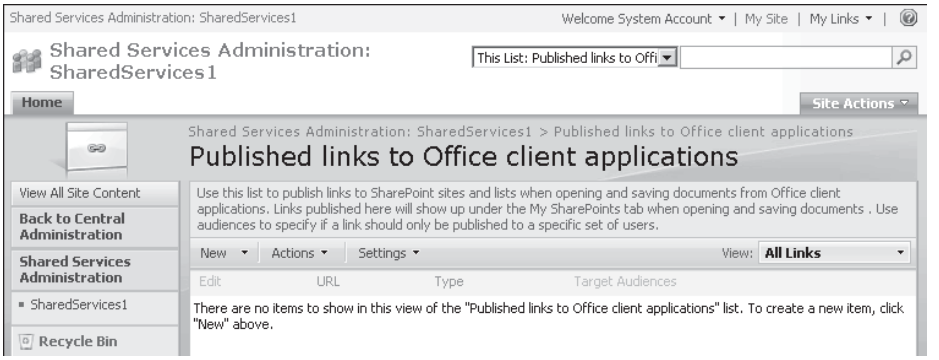
Set Up Published Links to Office Client Applications

Published links to Office provide you with a way of creating general links to various sites and libraries to be shown within the "save as" options in Office. Target destinations are set by type and include the following:

- Team Site
- Portal Site
- Big Document Repository

- Project Workspace
- Document Library
- Data Connection Library
- Slide Library

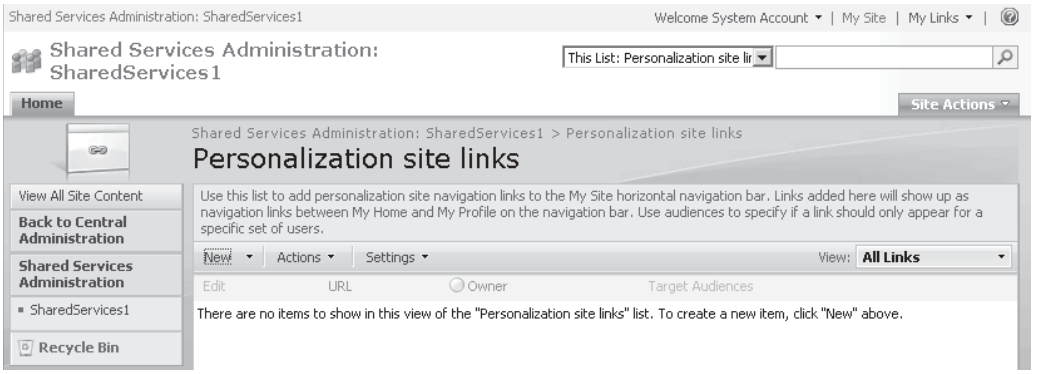
From the SSP Admin site home page, click the Published links to Office client applications link to open the list page:



Note that the links you specify here apply to all users of this SSP in the farm; as such, you need to be careful of security issues. If, for example, you created a document repository to be used for common storage, you would have to set the permissions so that all users can post to it, because otherwise end-users will get either a "permission denied" page or a "request for access" page. In addition, I suggest you not create too many links here—if the user has a large number of options, it could be confusing and users might put things in the wrong place.

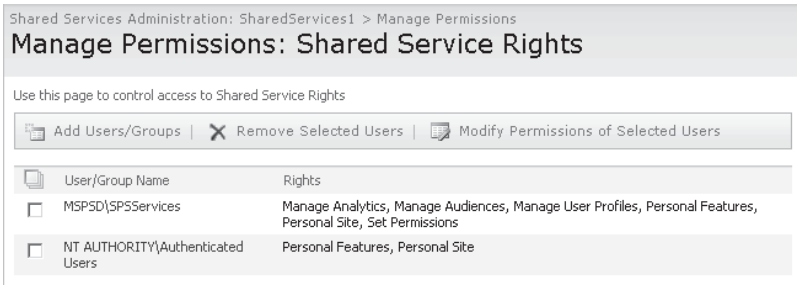
Set Up Personalization Site Links

Personalization site links provide you a way to make standard links available through the user My Site navigation. As links, of course, they can point to anything: a vendor site, a research site, or something else. Links added here are customizable to target audiences (as explained next), which means you can make them contextual to the appropriate users. From the SSP Admin site home page, click the Published links to Office client applications link to open the list page:



Set Up Personalization Services Permissions

Personalization services permissions enable you to designate users that can manage aspects of personalization. From the SSP Admin site home page, click the Published Links to Office Client Applications link to open the list page:

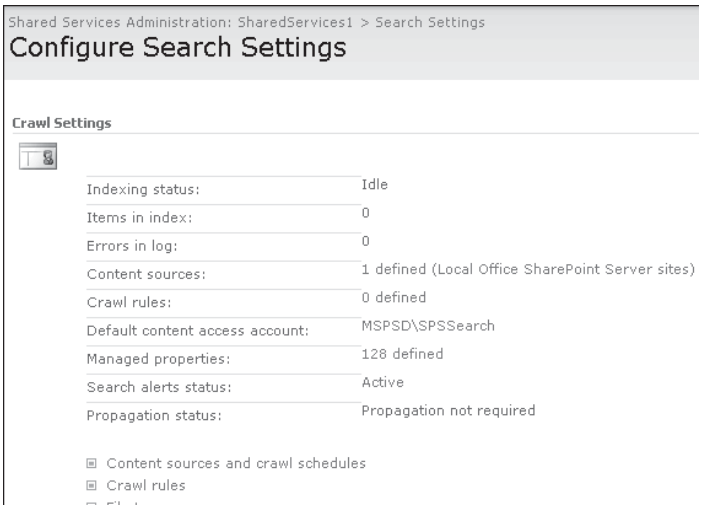


Assignments for individual users and groups work just like adding users to a site (as you either know or will learn later). An individual or a group (like the Authenticated Users group shown previously) can be given all or specific permissions to

- Create a personal site
- Use personal features
- Manage user profiles
- Manage audiences
- Manage permissions
- Manage usage analytics

Set Up Search Settings

The Search settings are used to set indexing schedules, map metadata, create scopes, and so on. From the SSP Admin site home page, click the Search settings link to open the Configure Search Settings page:



As you can see, there is a lot of information here; right away you can see the current status of the search and access all of the search options. The purpose for each of these is as follows:

- Status
  - **Indexing Status** This indicates if the index job is running or idle.
  - **Items in Index** This indicates the count of items.
  - **Errors in Log** This is the number of errors detected in the last indexing job.
  - **Content Sources** This is the number of content sources in this index.
  - **Default Content Access Account** This is the name of the account used to access content in SharePoint.
  - **Managed Properties** This indicates the number of mapped properties and provides a link to Metadata property mappings below.
  - **Search Alerts Status** This indicates whether search alerts are enabled or disabled.
  - **Propagation Status** This indicates whether this index is propagated (in a multi-application server farm) or not required.
- Settings
  - **Content Sources and Crawl Schedules** This adds or removes content sources (sites or file shares) and sets frequency of crawling.
  - **Crawl Rules** This sets the rules for crawling on individual.
  - **URLS** This includes excluding or including items. You can also specify a different authentication (other than the Crawl account) for special circumstances and provides an SSO for remote sources (subscribed sites, and so on).
  - **File Types** This allows you to set proper names for files based on the filename extension. It also allows users to know what a file type actually is, even if it doesn't have the necessary client application installed.
  - **Crawl Logs** This provides a list of crawled content logs that can be viewed, which is useful for determining issues when expected content is not indexed.
  - **Default Content Access Account** This allows you to set the account used for crawling.
  - **Metadata Property Mappings** This provides access to the list of all managed properties that are included in the crawling operations. The standard Microsoft document properties and SharePoint List fields are specified, and you can delete the properties you don't want and add additional ones if needed (for example, adding an attribute found in a CAD drawing file).
  - **Server Name Mappings** This allows mapping of a server URL to a friendlier name, which will be displayed with search results.
  - **Search-Based Alerts** This enables/disables search based alerts.
  - **Search Result Removal** This provides an exclusion list of URLs that should not be included in the search results (such as protected sites). This also provides the ability to immediately remove any existing indexed items on demand.

    - **Reset All Crawled Content** This clears the content index and rebuilds. It also provides the option to disable alerts during the index rebuild.
- Scopes
    - **Scopes** This is the number of scopes defined for this index.
    - **Update Status** This is the current state of indexing.
    - **Update Schedule** This is the schedule set for indexing scopes.
    - **Next Schedule Update** This is the next scheduled run of the job.
    - **Scopes Needing Update** This is the number awaiting update.
    - **View Scopes** This allows you to edit or define scopes (also known as locations that are to be crawled).
    - **Start Update Now** This forces the scope indexing job to run.

NOTE *For reference, a search scope is intended to narrow to a "scope of content" such as IT Documents or Sales Proposals—it is simply a way to narrow the amount of content a user has to search to find what they want.*

- Authoritative Pages
    - **Specify Authoritative Pages** These set URL ranking for indexed URLs. Authoritative pages enable you to adjust the ranking of items found in the index; you can designate a primary, secondary, and tertiary level sites for the ranking and can designate sites to be automatically lower (in ranking) by default.

NOTE *The SharePoint Search account is used for crawling purposes, which means the account must have access to any content being crawled, including file systems, remote sites, and so on. If this account does not have read access, items will not be indexed. Note that while this account may have full access, search results are still filtered by SharePoint Security (users can't see what they can't access).*

Handling Search Errors There are a few errors you might encounter when trying to open search settings. The most common is when an Index server has not been assigned to Search in the initial setup. To correct the problem, use the link shown on the error page to assign an index server or you may have to return to Services on Server to assign the indexing service. Another common error occurs when the index service is not available. This may happen when the system service is either stopped or restarting or if the index server cannot be contacted. You can correct this by verifying the service is running and that you have network connectivity.

Access Search Usage Reports

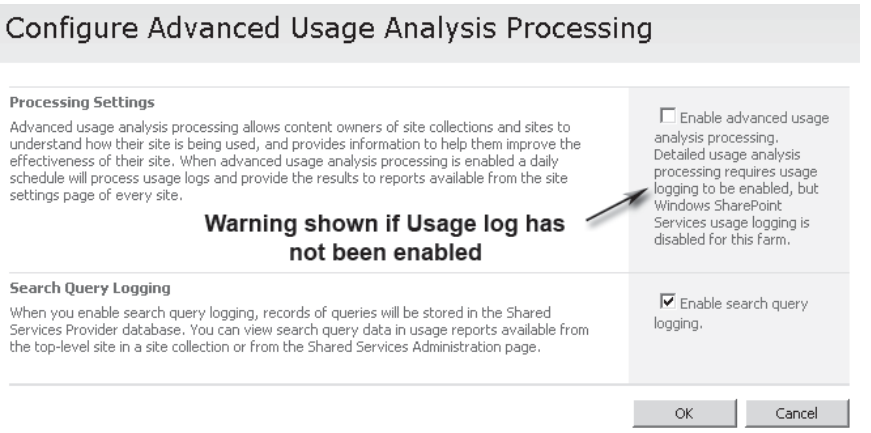
This process calls Analysis Services to list out reports that are available from the search usage analysis. When no data has been collected yet, there is a bug in this feature that returns a fatal .NET error, which will show up in the system event log:

```
.NET Runtime version 2.0.50727.42 - Fatal Execution Engine Error (7A05E2B3) (80131506)
```

Caused by a page called SpUsageSspSearchQueries.aspx, which uses some of the built-in analysis objects, this error has a few bizarre affects. First it may return a Login Prompt or it may return Service Unavailable, but in either case, it shuts down the Shared Service Provider site and cannot be accessed. This is resolved by using the Start | Run command line to enter IISReset, and then click OK to reset IIS.

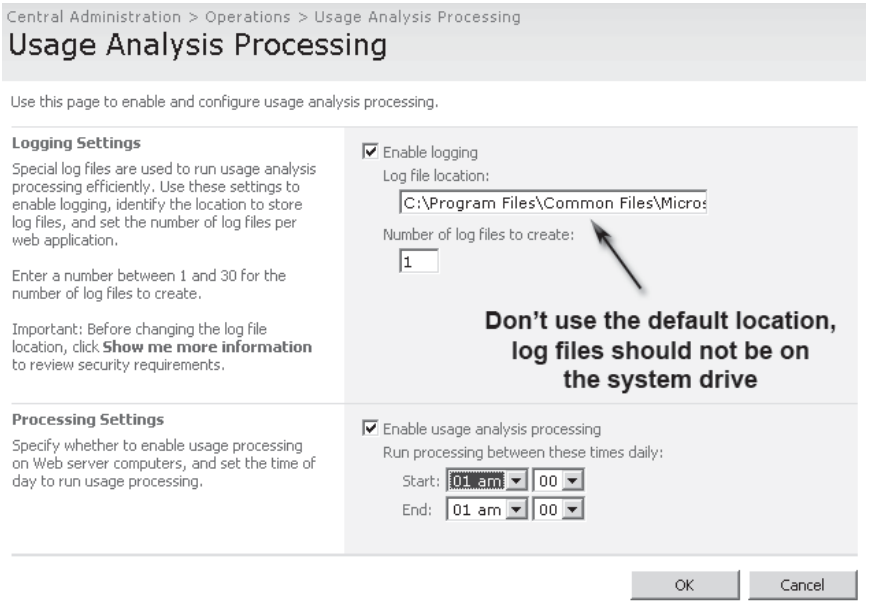
Set Up Usage Reporting

Usage analysis provides you with statistics for how your system is being used. This includes Processing Settings, which log the activities in sites and Search Query Logging, which logs the various search queries used. Both of these can be used to fine tune content and search. From the SSP Admin site home page, click the Usage reporting link to open the list page:



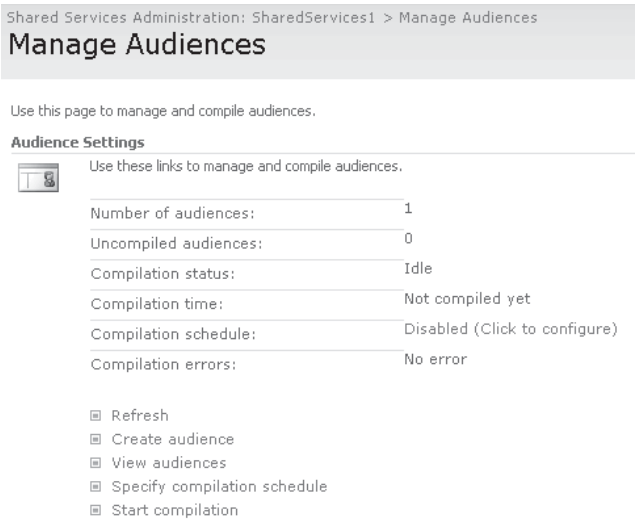
Notice that there is a warning message that the usage analysis logging is not enabled. This is typical when the system is first installed. You can refer to the next section to learn how to turn this on. To start logging for Process or Search, check the check box and then click OK to save the settings.

Turn On Usage Analysis Logging To turn on the usage logging required to collect statistics, Usage Analysis Logging must be turned on for the farm. To do this, open SharePoint 3.0 Central Administration and then click the Operations link. Then, under the Logging and Reporting group, click the Usage analysis processing link. By default, logging is turned off; to enable it, click the check box as shown:



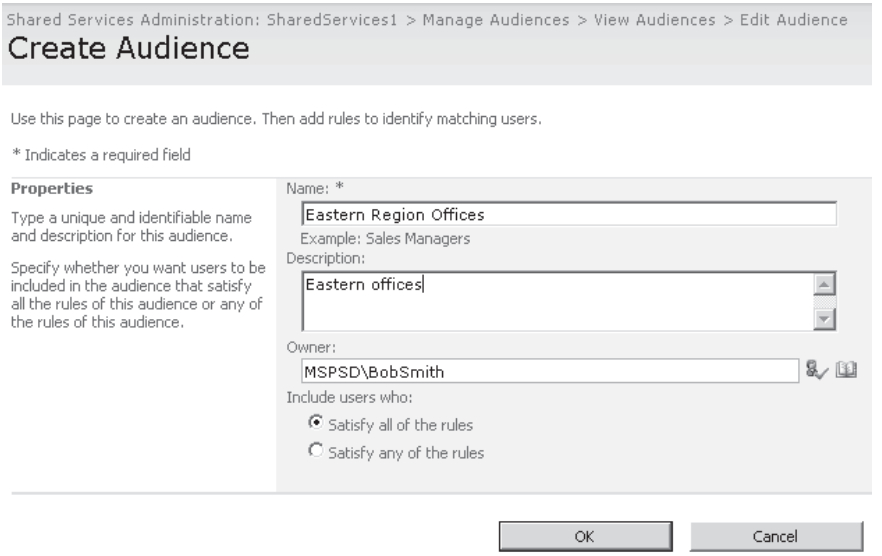
Set Up Audiences

Audiences in SharePoint are one of the best tools for targeting information to users. From the links you can provide in the My Sites to the content targeting within sites, audiences make for a compelling reason to return to the site; this is because content is pushed to them rather than having to search for it. Setting up audiences is a very simple process and can be based on any profile attribute as well as any that are set by Active Directory. There is even conditional processing like equals, contains, and so on. To manage audiences, click the Audiences link from the SSP Admin site home page, to open the Audiences page:

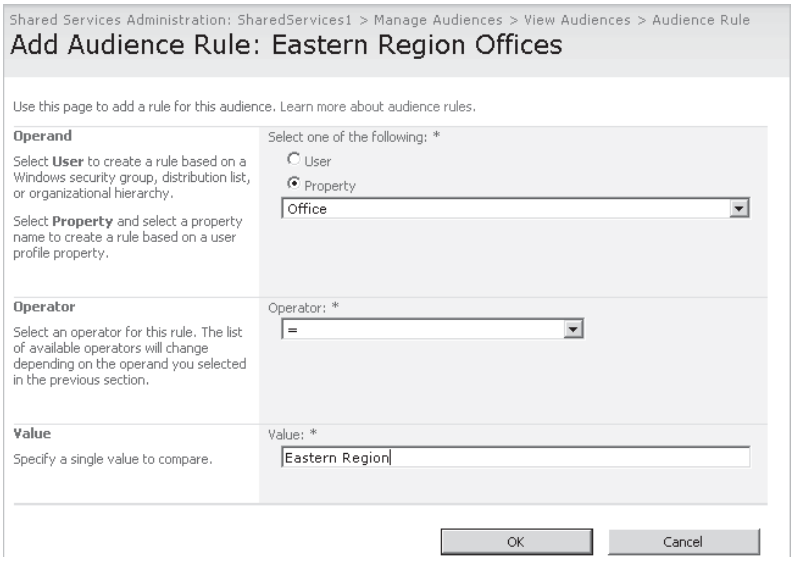


This lists the number of audiences and the compilation status (audiences, like search, must be run or compiled to compare user information against the criteria set by the audience). There is always one audience by default; everyone means all SharePoint users regardless of role. On this page, you can set the schedule for running complies, view existing audiences, and of course, create one. Creating a new audience is a two-step process:

1. Click on the Create Audience link to open the Create Audience Properties page:



2. On this page, you define the audience itself, name, description, and so on. Here, I've entered a new name and set the criteria that all rules (conditions) must be met for a user to be considered part of this audience. Click OK to open the Add Audience Rule page.

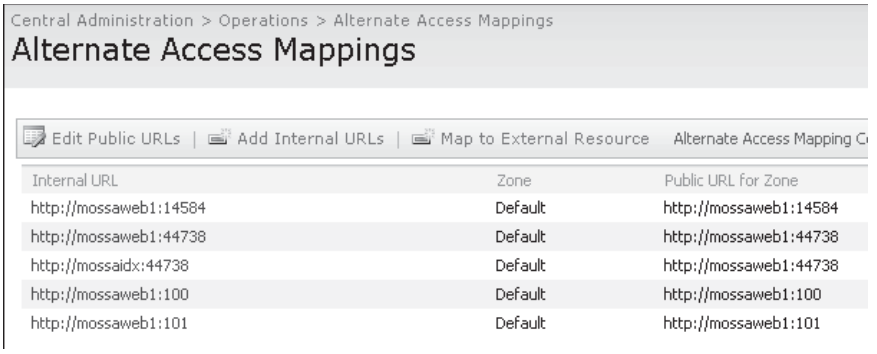


In this example, I selected Office as the property (part of the user profile) and set an equals condition for a specific name; you can choose whatever you like. When ready, click OK to create the audience. You can create as many additional rules as necessary to define who gets picked as a member of the audience. When you return to the Audience Properties page after you create, you should click the Compile audience link to build it. To complete the setup, you should also define a schedule so that the audience is re-compiled automatically.

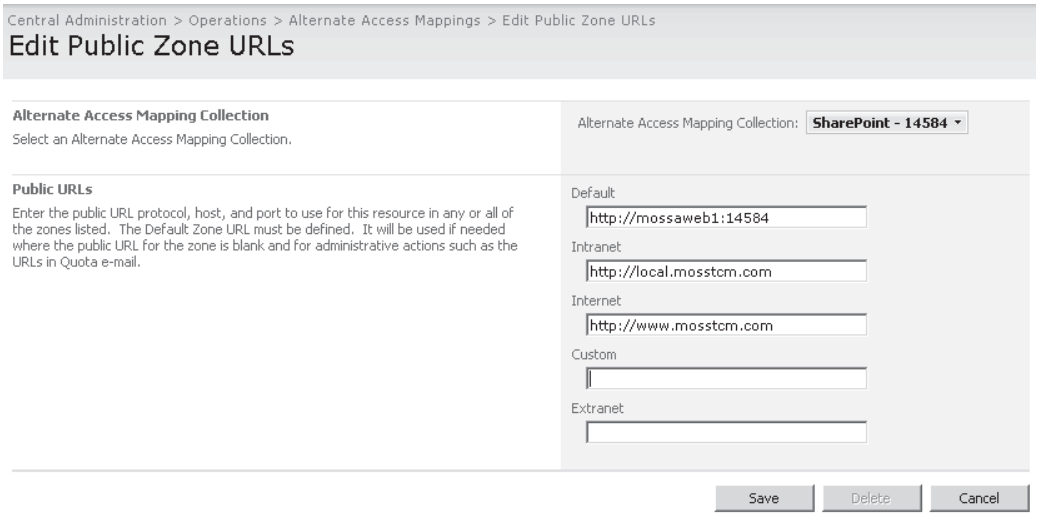
Create Alternate Access Mapping

Alternate Access Mapping (AAM) is a feature in SharePoint that allows you to designate the various URLs to access the site based on the zone (intranet, internet, custom, and so on). This enables you to designate the correct URL to be handled without the trouble of defining host headers in IIS or port redirects in firewalls. As you add applications, the default entry for the application is created automatically based on the system name and the port number. Obviously, this doesn't fly for an intranet or internet site, so AAM handles this for you.

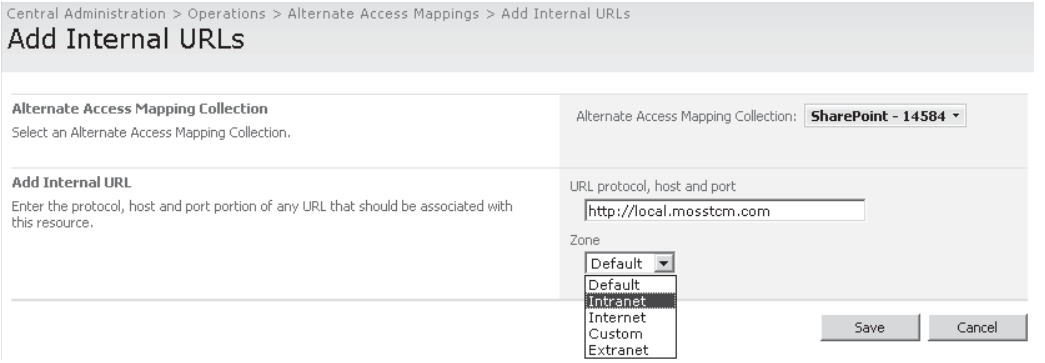
To manage Alternate Access Mappings, open the SharePoint 3.0 Central Administration site and from the home page, click the Operations link. On the Operations page under the Global Configuration group, click the Alternate Access Mappings link, which will open the page to show all current assignments:



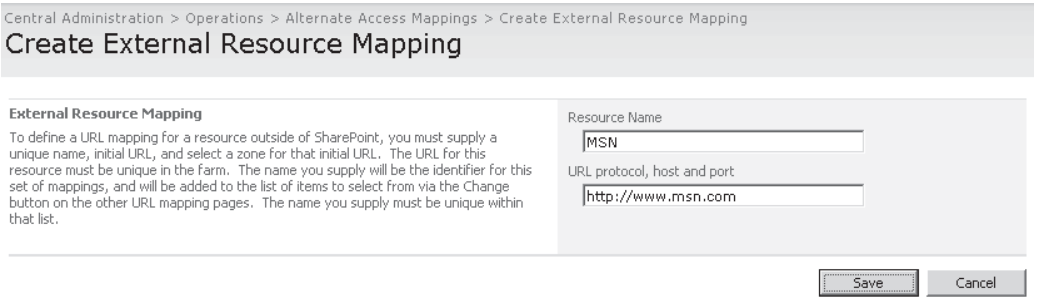
From here, you can select an Alternate Access Mapping Collection that allows you to see all assignments or limit it to a single application, Edit Public URLs, Add Internal URLs and Map to External Resource. Editing the Public URL for an application enables you to set all of the zones at once as shown on the next page.



Adding internal URLs is nearly the same, though you select which Internet Explorer Security Zone is desired:



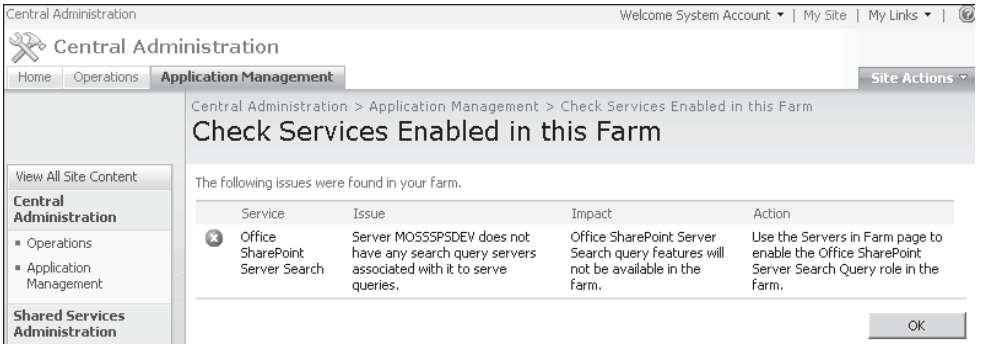
Mapping to an external resource enables you to map in resources that are not within the confines of SharePoint:



As you can see, managing the mappings is fairly straightforward. However, you should be very careful to align your mappings along with your plans for how the site will authenticate. Odd things can happen when a URL gets embedded somewhere; for example, if someone accidentally hard codes a link to an item in SharePoint using an internet name that was not available inside, local users on the intranet would have access issues.

Check Services Enabled in this Farm

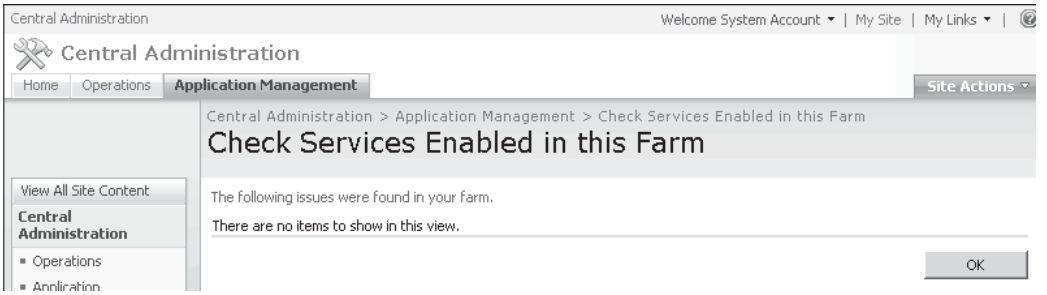
This (almost last) step is to verify that all services necessary in the farm are running and operational—any messages shown will also indicate how to correct the problem. To check the services, open SharePoint 3.0 Central Administration and click the Applications Management link. When the page opens, click the Check services enabled in this farm link under the Office SharePoint Server Shared Services group. SharePoint will check all services and produce a report page:



In this example, you can see that there is an issue because no Search Query Servers were found and that the report page gives an explanation for how to correct it. To correct the issue shown, you would return to the Operations page as instructed, view the server running the Office Search Service, and then check the properties. In this example, you'll see that indeed, the query service was not assigned:



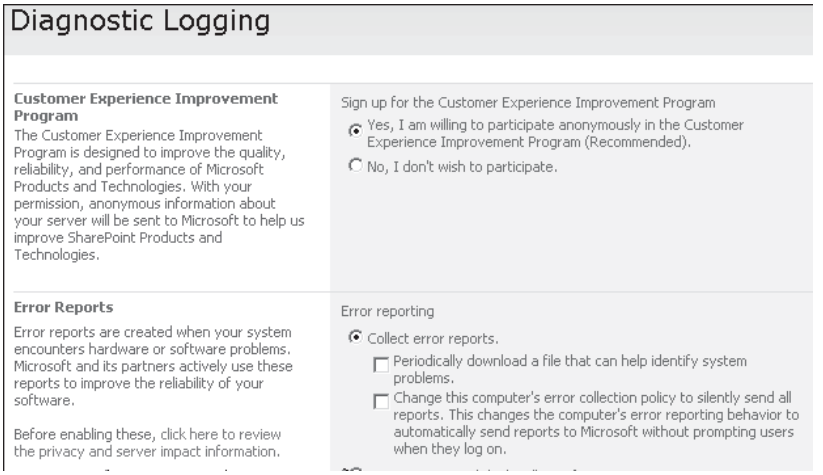
Checking the box, setting the password for the account, and then clicking OK will start the service. When you return to the Check services page, you should see the following:



This indicates that all of the appropriate settings and services for the farm have been configured and that you are ready for standard operation.

Diagnostic Logging Settings (Optional)

Diagnostic logging settings let you know how much logging SharePoint does when errors occur and whether you want these reports to be sent to the Microsoft Customer Experience Improvement Program. You can access theses settings via SharePoint 3.0 Central Administration | Operations. Then, under the Logging and Reporting group, click the Diagnostic logging link to display the options page:



The main groups are

- **Customer Experience Improvement Program** This allows you to opt in or out of this program.
- **Error Reporting** This enables or disables the collection of Error Reports that occur, and decides whether to send them to Microsoft or not.
- **Event Throttling** This enables you to set a level of reporting, which then enables you to increase or reduce the number collected.

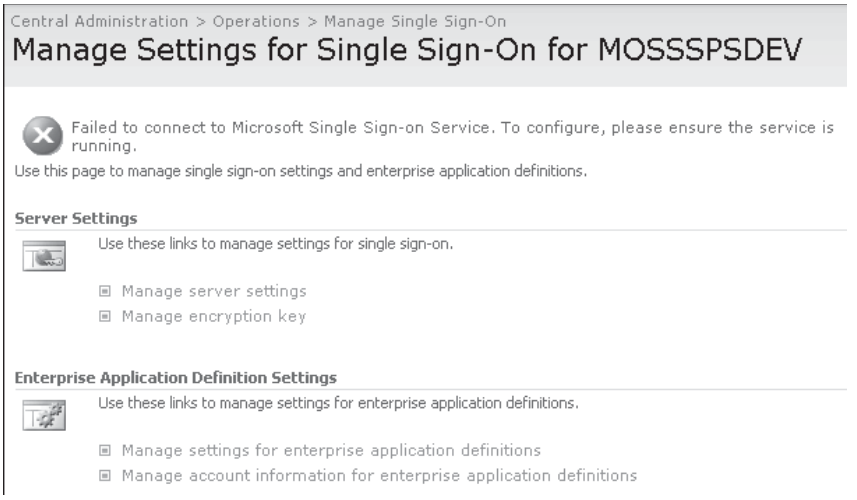
- **Trace Log** This provides a trace log that can be used to diagnose serious errors that occur due to chain reaction problems.

Remember, any logging can cause performance issues and suck up a lot of disk space; be sure to never use the system drive when storing logs and don't turn it on if you don't need it!

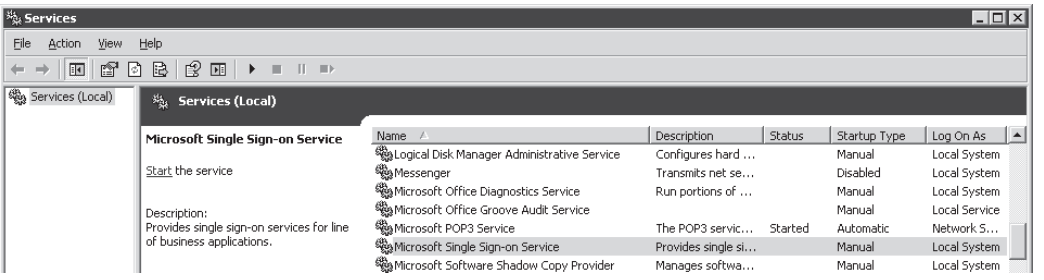
Enable SSO in the Farm (Optional)

The Single Sign-On service provides SharePoint with the ability to provide authentication to external services and systems on a group or individual basis. The SSO service handles authenticating a user against a backend system so the user is not prompted to login. It can also be used for managing authentication information for use in web services, and so on. Remember that this is an optional step; if you are not ready to set up the service at this time, you can defer it until it is needed (some installs may never require it).

To access the Single Sign-On setup, click the SharePoint 3.0 Central Administration | Operations, and then under the Security group, click the Manage Settings for Single Sign-On link to open the Manage Settings page:



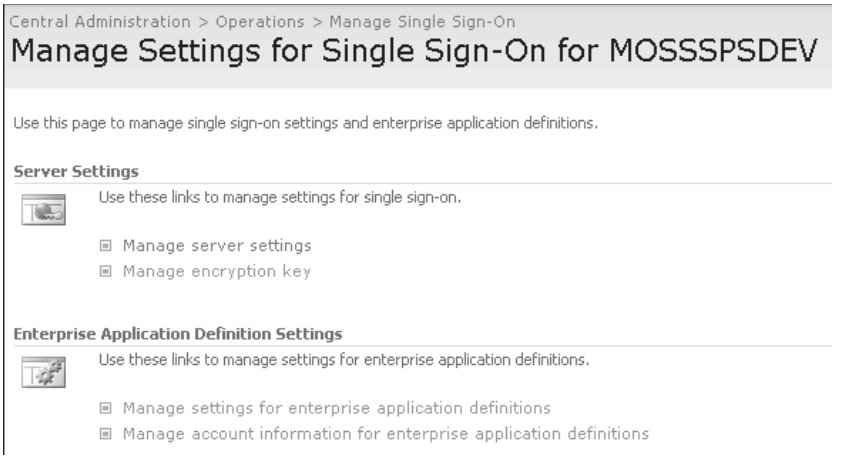
The first time you open this, you can expect to see the error message shown. This is because the service is set to manual and does not start by default. To correct this issue, you should logon to the server running the service, and then use Start | All Programs | Administrative Tools | Services to open the services window. Locate Microsoft Single Sign-on Service in the list and you will find that it is set to manual and has not been started:



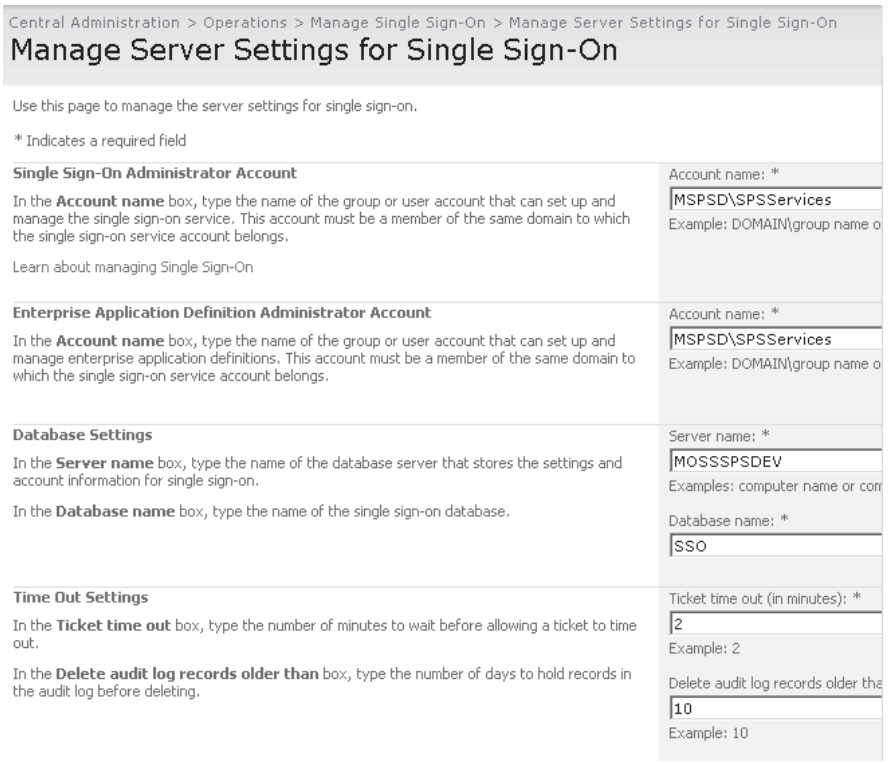
Double-click on the name to open the properties. Then, you must do the following:

1. C1ick the Log On tab. This, by default, is set to the Local System account which is incorrect. You must change this to the SharePoint Services account (SPServices); otherwise, you will receive You do not have rights to perform this operation when you try to create a new server.
2. Click the General tab, change the startup from Manual to Automatic, and then click the Start button to start the service. Click OK to exit the properties.
3. Close the Services window.

When you return to the Manage Settings for Single Sign-On page and refresh, you should see the normal display:



In the first-time installation, there has not been a server installed yet so the only option you have available is to click the Create Server link. This opens the Manage Server settings page:



On this page, you can define all of the settings for the SSO Service as follows:

- **Server Settings**
  - **Single Sign-On Administrator Account** This should be set to the name of the account that can set up and manage the service.
  - **Enterprise Application Definition Administrator Account** This should be set to the name of the (security) group or individual user account that can set up and manage enterprise application definitions (backend systems, web services).
- **Database Settings** This sets the name of the SQL Server and the name to use for the SSO Database.
- **Time Out Settings**
  - **Ticket Time Out** This is the number of minutes to wait before a ticket will time out.
  - **Delete Audit Log Records Other Than** This is the number of days to hold log records.

Once you have made your settings, click the OK button to create the service. It will run for a period of time and then return to the Manage Server page. If an error is detected, you will receive a message stating, "You do not have rights to perform this operation." This indicates that the Microsoft Single Sign-on Services in system services is not running under the correct identity or the account does not have the correct permissions (the Logon as a service right is usually missing). To correct this, check the permissions granted to the account, and then go back to system services, set the identity account, restart the service, and retry creating the SSO service.

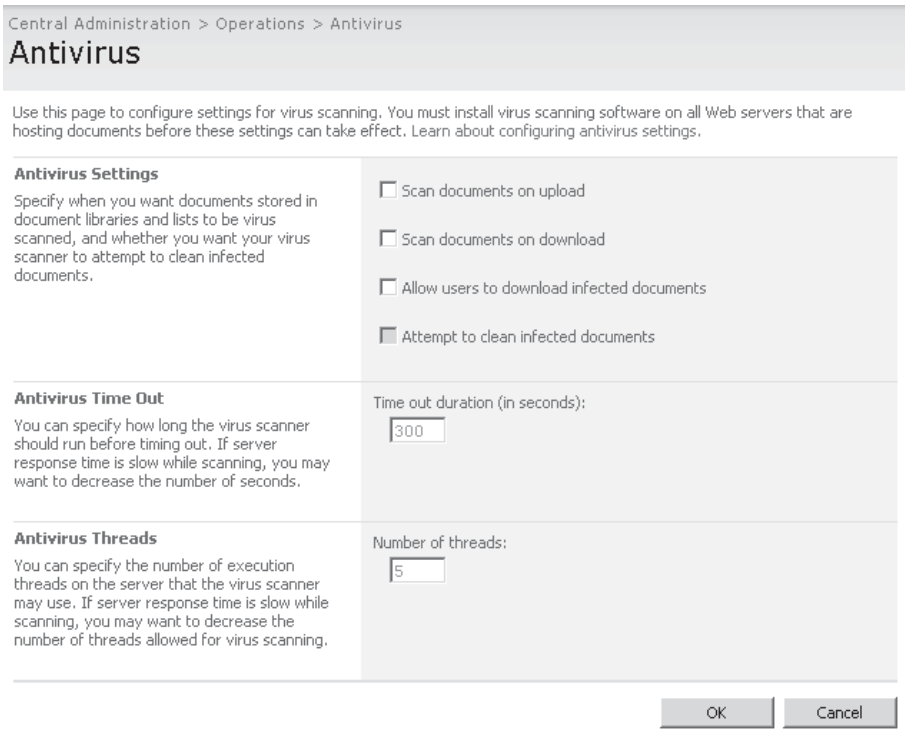
Once you have successfully created the SSO service, you will be returned to the Manage Service page and you will now see that all options are available (Create now changed to Manage). The options available are as follows:

- **Manage Server Settings** This brings you back to the Management page where you can set accounts and database information.
- **Manage Encryption Key** This allows you to create, back up, and restore an encryption key for the server.
- **Manage Settings for Enterprise Application Definitions** This opens up a management page that allows you to define an application for an SSO, which includes:
  - **The Application Contact** This is information about the application and whom to contact for information.
  - The Account type, made up of:
    - **Group** One account for all users.
    - **Individual** One user, one account.
    - **Group using restricted account** This is the same as Group but adds additional security.
- **Authentication Type** Windows or non-Windows.
- **Login Account Information** This allows you to specify up to five fields of information that can be passed to the receiving application (for example, Username and Password for a Forms site).
- **Manage Account Information for an Enterprise Application Definition** This enables you to modify the accounts associated with an Enterprise Application Definition.

Add Antivirus Protection (Optional)

The antivirus settings in SharePoint allow you to automatically scan uploaded documents and content. The settings enable you to control what is scanned and when. While several settings are available, it is important to note that SharePoint does not provide any antivirus utilities itself; a third-party tool is required and it must be installed on all front-end servers.

To access the settings in SharePoint 3.0 Central Administration, click the Operations link under the Security group, and then click Antivirus. This will open the settings page:



As you can see, the options are pretty straightforward:

- **Scan Documents on Upload** If enabled, all documents uploaded through the User Interface or via Microsoft Office are scanned.
- **Scan Documents on Download** If enabled, all documents downloaded from the User Interface are scanned.
- **Allow Users to Download Infected Documents** This allows users to download documents even though they are infected.
- **Attempt to Clean Infected Documents** This cleans documents (if possible) on an upload or download (only enabled if one of the Scans is enabled).

And the final two options:

- **Antivirus Time Out** This is the number of sections to wait on virus scanning.
- **Antivirus Threads** This is the number of threads that can execute at one time (default is 5).

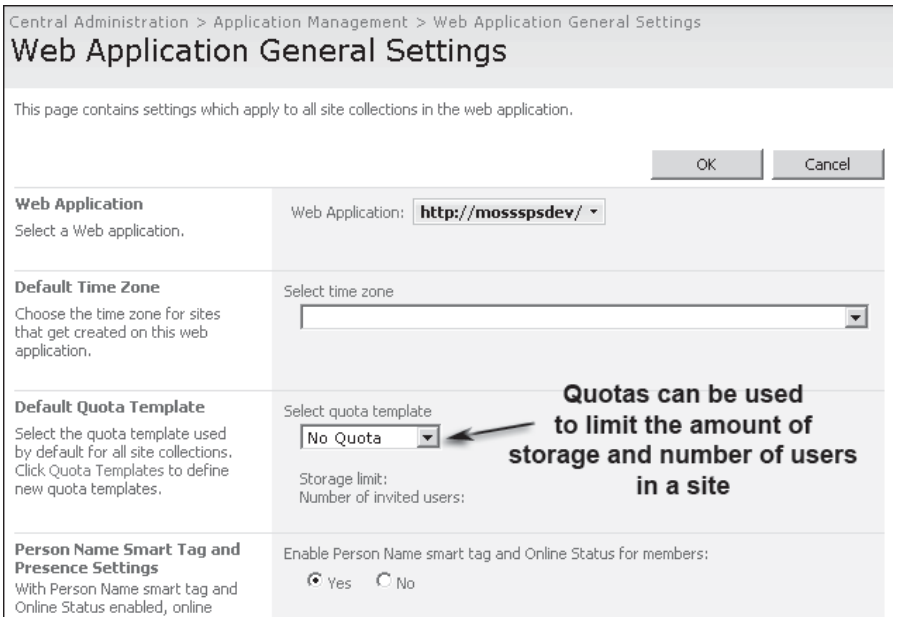
Note that connecting antivirus software to your system is out of the scope of this book; it is best to consult with your software vendor on how to best hook it up.

Post-Configuration Steps

After you have completed the entire configuration covered previously, you are due some congratulations! At this point, you should have a complete system or farm up and running and the first site ready to go. However, as with anything, there are just a few things to set before calling it a day. Note that these settings are not mandatory and don't appear in the administrative task list.

Update Application Settings

The applications you created will still have the default settings and should be updated; this includes setting the time zone, quotas, and so on. Using SharePoint 3.0 Central Administration, click the Application Management link. When the Application Management page opens, under the SharePoint Web Application Management group, click the Web application general settings link:



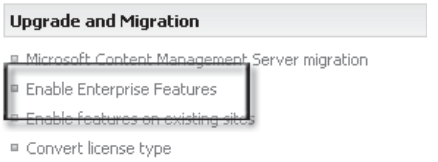
The Web Application provides a drop-down list that allows you to switch between applications; settings for each include:

- **Default Time Zone** This is the time zone for the application; it's helpful to ensure correct timestamps on content.
- **Default Quota Template** This selects a quota (maximum) template for this application.
- **Person Name Smart Tag and Presence Settings** If enabled, User Presence is active in SharePoint. When the mouse is moved over a username, the user profile and or IM presence is shown and it will indicate if the user is currently logged in.
- **Maximum Upload Size** This sets the maximum file size that can be uploaded.

- **Alerts** This enables or disables alerts on the site (if disabled, Alert options are not shown in the User Interface).
- **RSS Settings** This enables or disables Really Simple Syndication (RSS) options for this Application.
- **Web Page Security Validation** This enables or disables page time out (how long a user can be inactive before having to log back in).
- **Send User Name and Password in E-mail** This is used when users are created through SharePoint; if disabled, users cannot access the site unless they already know the password.
- **Backward-Compatible Event Handlers** This enables or disables use of 2003 style event handlers (installed per library and not using the Feature architecture); by default, this is disabled.
- **Change Log** This specifies how long changes logged on site actions are stored, while change logs enable you to compare versions.
- **Recycle Bin** This sets the options for the recycle bin (on or off, how long to hold, and so on).

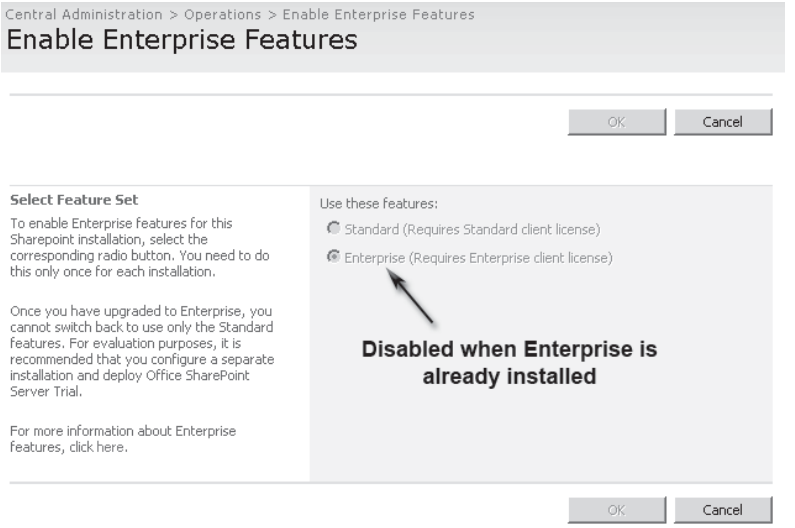
REMEMBER *These settings apply to the application, therefore, all sites are created within it.*

Enable Enterprise Features (MOSS Only)



Depending on the version you installed, you may need to enable enterprise features (typical when you install the standard edition and then upgrade). This does not apply to the Enterprise Edition. Using SharePoint 3.0 Central Administration, click the Operations link, found on the Operations page under the Upgrade and Migration group. Then, click Enable Enterprise Features.

When the page is displayed, if the Enterprise Features are not installed, you will be able to select them. This can only be done once, and once they are installed (or if the Enterprise Edition was installed from the get go), the options are grayed out:



Be aware that you cannot revert back to the Standard Edition once this has been selected. While this has no impact on the SharePoint installation, it does affect licensing and use, specifically in the types of Client Access Licenses (CALs) users must have.

Check the System Event Logs

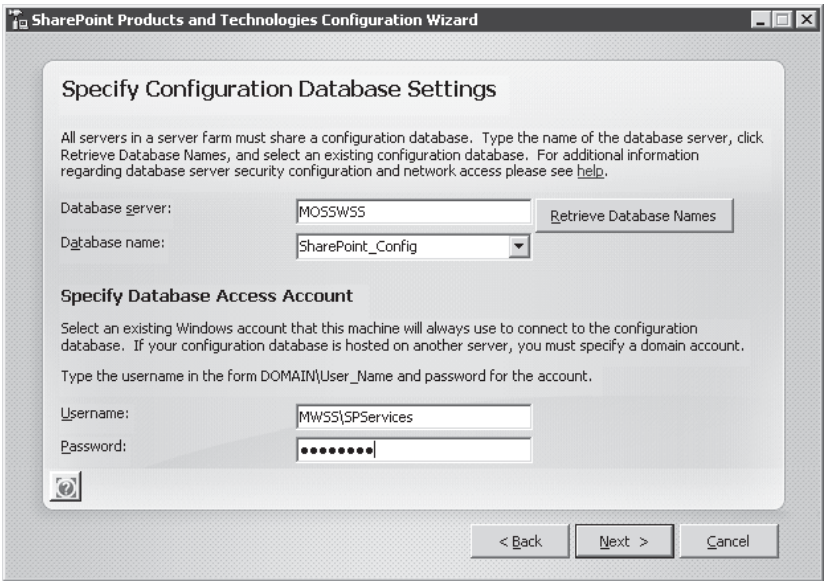
While everything may seem to be running along fine on the surface, there can still be gremlins out there waiting to bite you when you least expect it. After you have completed the configuration, review the Event Logs on every system involved in the farm and if possible, the Active Directory domain server as well. Any errors that appear in the Application or System logs (login failures, for example) should be checked out and corrected before you turn the system over to users.

Joining an Existing Server Farm

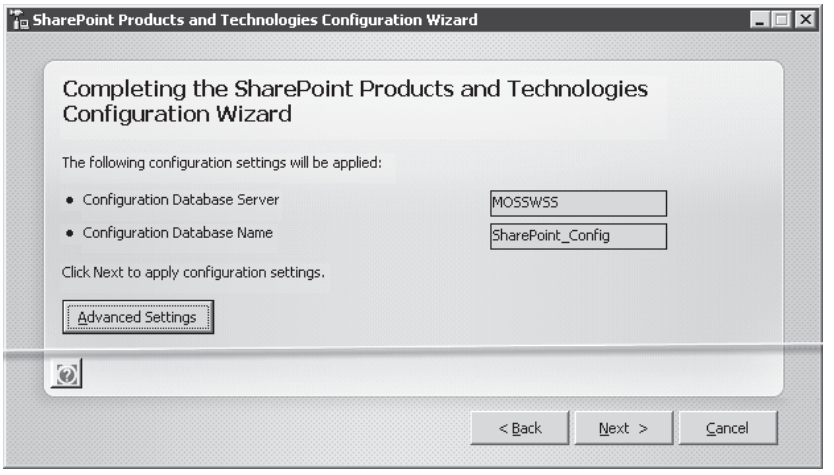
For the Advanced installation, adding additional servers is done by joining an existing configuration database. From that point on, the server is considered part of the server farm and can then be assigned services. In the typical sequence of events, before joining a farm you will have done the following:

- Installed SharePoint on the first server and created a configuration database
- Brought up the SharePoint 3.0 Central Administration Site
- Assigned services to the first server
- Installed SharePoint on the Server joining the farm

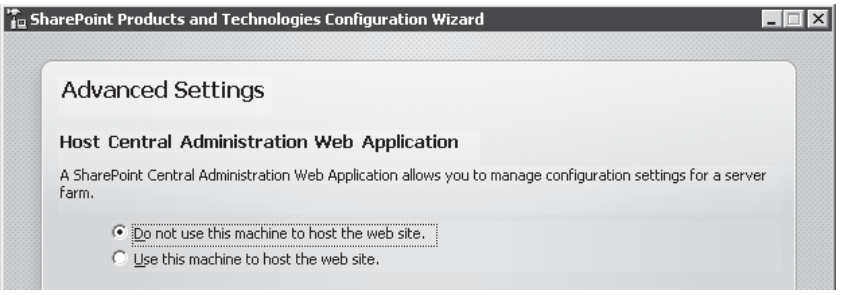
Once installation has completed on a server (joining), the SharePoint Configuration is brought up. The starting prompt will ask you if you wish to Create a new Server Farm or Join an Existing Server Farm; when you select Join an Existing, the configuration database selection page is displayed:



On this page, you specify the Server name that holds the SQL Database (this one happens to be MOSSWSS), and then click the Retrieve Database Names button. After they load, you use the drop-down list to select the Configuration database, and then enter the connection account information as shown. By default, this should be the SharePoint Services account (SPServices). When you click Next, SharePoint's install will complete, and when it's finished, you'll see the completion page:

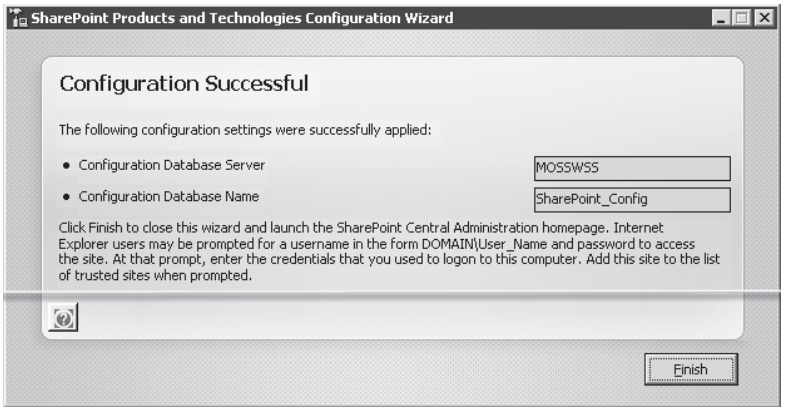


If you click on the Advanced button, you can select whether you want this particular server to host the SharePoint 3.0 Central Administration site or not:

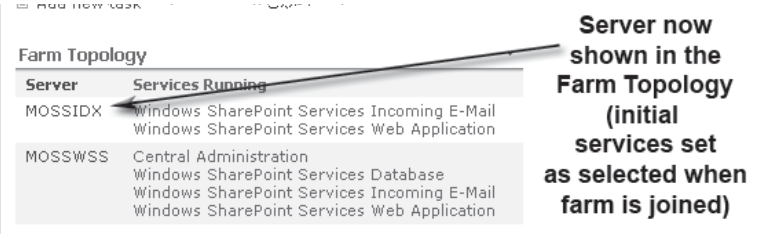


NOTE *When a server is not hosting SharePoint 3.0 Central Administration, the link to it on the local server (in Administrative Tools) will not work because the local SharePoint 3.0 Central Administration site will be stopped. To access SharePoint 3.0 Central Administration from these servers, you should use the correct URL to the Hosting server. Also, if you do start SharePoint 3.0 Central Administration on the local server, you can only see the server itself and no other servers in the farm.*

After the installation is complete, you will get a confirmation that the configuration completed successfully:



Once this has completed, you can now manage the server and its services in SharePoint 3.0 Central Administration. Open the site from the current server (if you chose that option) or use the URL to the hosting server. Click the Operations link, and then under Topology and Servers group, click the Servers in farm link to display the server list:



You can now assign services to the server as desired.

NOTE *Since the server information displayed is coming from the SharePoint Data store, all member servers are shown even if the physical hardware is down.*

Removing a Server from the Farm

Servers have the ability to join or leave a farm at any time. This makes it easy for you to enable a server to play backup roles if a part of a farm is disabled temporarily. It also means that a stand-by can be ready to go at any time. That said, however, SharePoint will warn you if you attempt to disconnect from a server farm with a popup that states, "It is recommended that you uninstall SharePoint Products and Technologies from the server, rather than removing the server from this page." As this recommends (and I agree), rather than leave a farm and join another, it is better to uninstall SharePoint completely and join up during the installation process. While this may sound like more work, it actually isn't; the SharePoint uninstall is very clean and only takes a few minutes (see the uninstall section coming up).

Starting Windows Load Balancing Services

If you followed the instructions for setting up a load-balanced server farm, load balancing should have been stopped during the installation and configuration process. Once this process has been completed, you should start Windows Load Balancing. This can be done from the command line or using the Network Load Balancing Manager utility (NLBMgr). To start from the command line:

1. Log in to the first web server.
2. Open a command window (Run | cmd | OK).
3. Enter in the command:

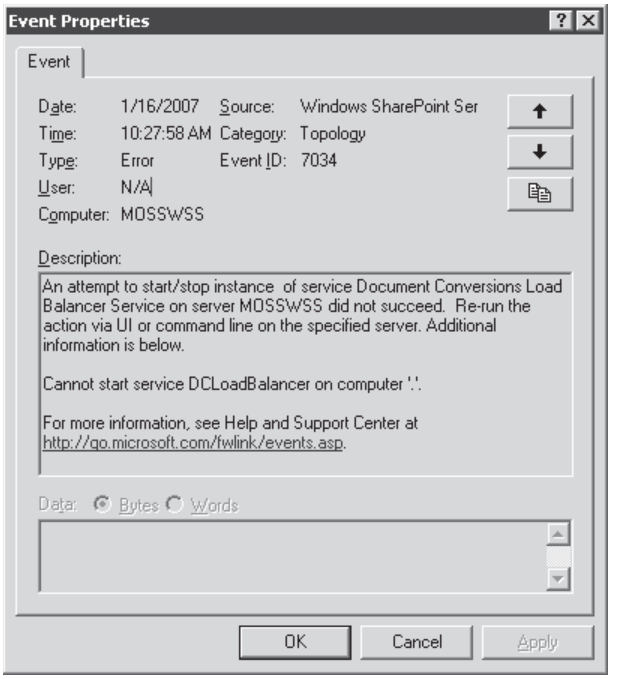
```
wlbs start <server farm/DNS name or IP Address>
```

To use the Network Load Balancing Manager utility:

4. Select Start | Run, enter in NLBMgr, and then click OK.
5. When the NLB Manager window appears, connect to the cluster you want to start.
6. Under the cluster container, click the host that you want to stop.
7. On the Host menu, point to Control Host, and then click Start.

Repairing or Uninstalling SharePoint

There are some instances in which a SharePoint install will fail. For example, there is a known issue of receiving a "Page Not Found" message when attempting to access administration pages. One known error occurs with the MOSS Advanced installation when attempting to start the Document Conversions Load Balancer Service. A brief working page is shown, and then a page not found error is displayed. You can verify this by checking for the error in the System Application Event Log:



You have two options when an error like this occurs: you can attempt to run the SharePoint Repair process or simply uninstall and then reinstall SharePoint. While I'm unsure as to the exact cause, it almost always appears to be privilege-related, is usually on the file system (the actual ACLs), and is, unfortunately, random; I get an average of 1 in 10 with this problem.

Repairing

Before you try to do a repair, you should exhaust the possible common issues such as checking out your settings for key accounts like the Install and Services accounts, looking at the access they've been granted to SQL, and most important, the access they have on the actual file system. In SQL, both of the accounts, Install and Service, should have System Admin privileges. In the file system, they should have read and execute access to the SharePoint Hive and the Inetpub directory trees, as well as any other file locations created for SharePoint (for example, if you installed in a folder other than Inetpub). If you make any changes or the privileges appear to check out, try a reboot, and if the problems still exists on the restart, attempt to use the Repair option.

The Repair feature basically checks all of the files and *most* of the database settings and will usually recover most SharePoint related issues. That said, be aware that it is not very good at correcting system file access errors, file ACL problems, and similar. However, it is a good last resort before a complete uninstall. To run the Repair operation, do the following:

1. Select Start | Control Panel | Add/Remove Programs.
2. When the Add/Remove Programs dialog window appears, scroll down to find the product you installed. For MOSS, the title is Microsoft Office SharePoint Server 2007; for WSS, the title is Windows SharePoint Services 3.0. Click the appropriate title to select it, and then click the Change/Remove button.
3. When the SharePoint install program starts up, you will have the option to select either Repair or Remove:



4. Click the Repair radio button and then click the Continue button. This will start the repair process with a progress bar.
5. When this has completed, you may or may not have to reboot your system, but at minimum, be sure to run an IISReset.
6. Repeat the process that first produced the error (such as trying to start the Document Conversions Load Balancer service as mentioned previously). If the error reoccurs, you must do a full uninstall.

Uninstalling

Uninstalling is rather uneventful and, thankfully, quite good at cleaning up the system. In addition to removing all traces of SharePoint, it nicely cleans up SQL Server and IIS as well as removing any databases created and any sites/application pools created (except for SQL, see last step). To uninstall SharePoint, do the following:

1. Select Start | Control Panel | Add/Remove Programs.

2. When the Add/Remove Programs dialog window appears, scroll down to find the product you installed. For MOSS, the title is Microsoft Office SharePoint Server 2007; for WSS, the title is Windows SharePoint Services 3.0. Click the appropriate title to select it, and then click the Change/Remove button.
3. When the SharePoint install program starts up, you will have the option to select either Repair or Remove; click the Remove radio button, and then click the Continue button. A dialog box will then be displayed showing the progress of the uninstall.
4. When the uninstall has completed, you will have to reboot your system.
5. If this was the only server installed thus far, you must also clear out the Configuration databases.
   a. On the restart, log into the server running SQL Server.
   b. Select Start | All Programs | Microsoft SQL Server 2005 | SQL Server Management Studio, expand the databases to view them, and then right-click on the following database names:
      - The SharePoint Configuration Database (default SharePoint_Config)
      - The SharePoint 3.0 Central Administration Content Database (default name is SharePoint_AdminContent_<GUID>)
      - The WSS Search Database (default name is WSS_Search_<server name>)
6. Select delete.

NOTE *Your database names may be different; refer to your installation notes.*

As I mentioned, the SharePoint uninstall does a good job of removing everything. This is helpful if you uninstalled to correct a problem, because after you've removed the databases you can immediately reinstall SharePoint. Of course, make sure you have rebooted the system and be sure to follow the checklists for installation so you don't have problems the second time around!

CAUTION *If you uninstall SharePoint from a Basic installation, the default website will remain stopped, and must be restarted manually if you are removing SharePoint permanently. Be aware that in the stopped state, some services (like SQL Reports) may experience errors. This is usually corrected by starting the Default website at port 80 and then rebooting the system.*

Summary

This chapter covered the major steps needed to complete the installation of SharePoint using either WSS or MOSS. Although many of the items covered are typical when setting up once for an installation, you will often return to these settings as site usage grows and additional features are added. In the next chapter covering SharePoint Central Administration, you'll begin to work with the foundation that you've created here.

This page intentionally left blank

CHAPTER 7

Exploring SharePoint Central Administration

SharePoint provides several levels of administration for managing SharePoint at the instance, shared service, site, and sub site levels. At the instance level, SharePoint 3.0 Central Administration is the core control site for a particular SharePoint instance, and controls the primary operations of the platform (much the same way Internet Information Services Manager is used to manage IIS). Base operations such as server management, application management, site management, and core services are controlled through this site. In MOSS, Central Administration (CA) includes the Shared Services Administration as a sub site used to manage User Profiles, Office SharePoint Search, and other shared/enterprise features and services.

There is only one Central Administration site created per instance when SharePoint is installed. In a Single Server environment, this simply controls the site on the server. In a farm, the CA site can run on one or more front-end servers, and can be enabled or disabled. This is because CA is still accessible on a server where the service is disabled, but in that case, it will only show the local server's services. In MOSS, there is only one Shared Services Administration site as well (though you can have many Shared Services Providers).

At the site level, administration is based on where you are in a given site collection. At the root level (or top site), Site Administration applies to the entire collection of sites below it (for those that inherit the parent settings). Administration in sub sites can manage only that site and those created below it. You can see how this all fits together in Figure 7-1.

Note that many of the items covered here may not mean much to you at this point. If you are not completely familiar with applications, sites, Shared Services, and so on, don't worry; I will identify what each of these items are for and when you will need them. As you work through other aspects of SharePoint, you will be constantly revisiting Central Administration and eventually all of the options mentioned here. If this is your first time viewing this, I suggest you glance through everything first, just to familiarize yourself with the options and utilities available.

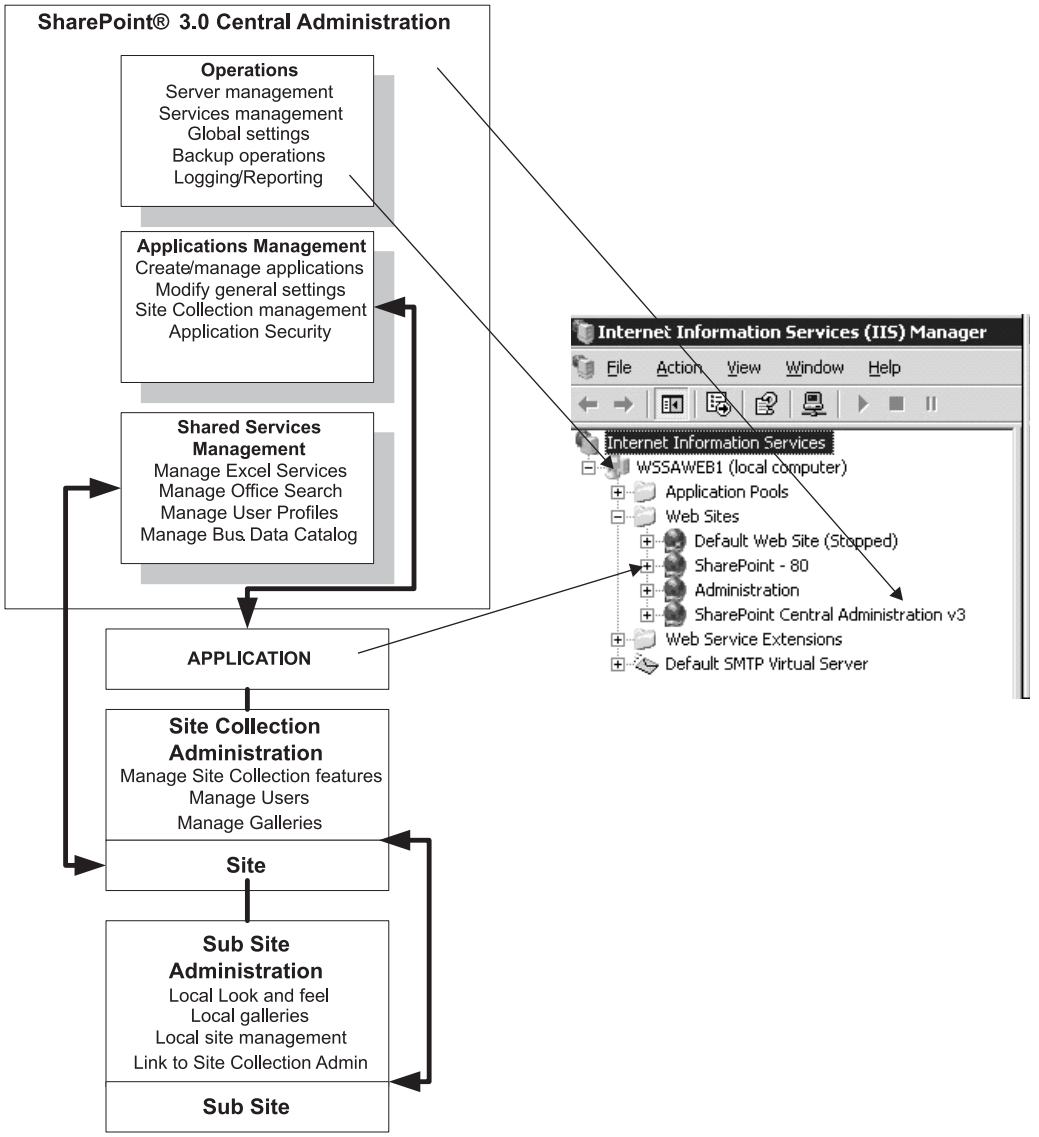


FIGURE 7-1 Overview of SharePoint Administration

Using Central Administration

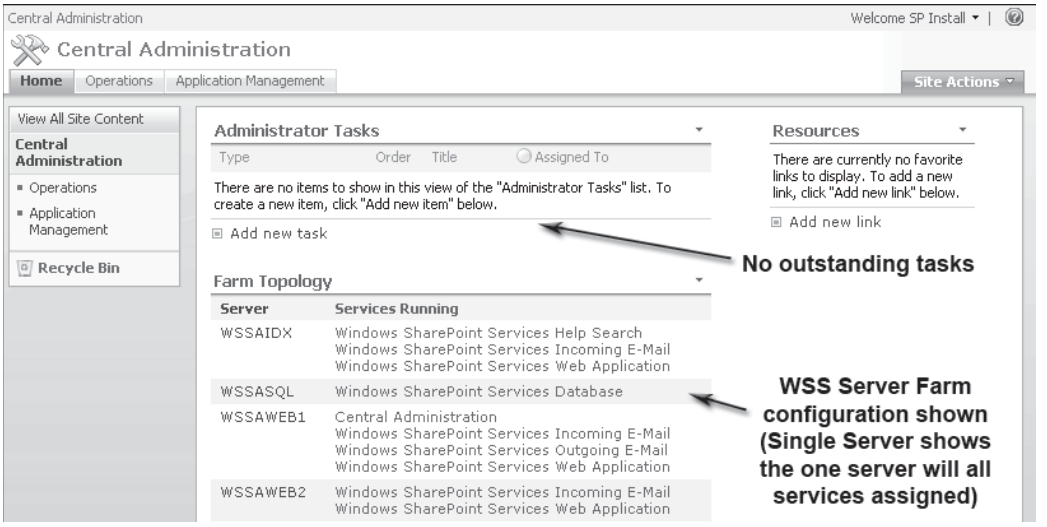
The SharePoint Central Administration (CA) site is installed regardless of whether the installation is Basic or Advanced. The only difference between them is what is done automatically and the additional features that are installed. If the Basic install method is used

for either WSS or MOSS, the installation process handles creating everything for the first site, including a default Shared Services Provider for MOSS. Some minor settings (e-mail settings, logging, and so on) have to be added, but otherwise, it is ready to go. As you saw in the configuration, this is not the case when using the Advanced installation method for a server farm. The CA site is created, but no SSP, applications, or sites are. MOSS requires an SSP to set up and WSS requires you to use the STSAdm Command-Line tool to create the first site (and any other top level sites after that).

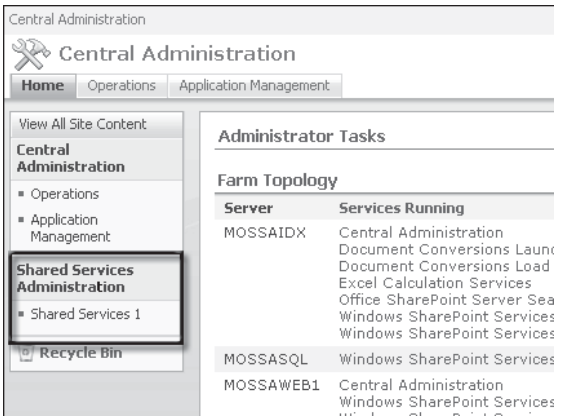
The CA site is set up to handle managing the server(s) in a farm, assigning services to servers, creating and managing applications, and in MOSS, managing the Shared Services. The Operations side works with servers and assigned services and roles. The Application Management side allows you to create and manage applications and the sites beneath them. In MOSS, Shared Services provides control over farm-wide services including search, Excel Calculation Services (ECS), and the Business Data Catalog (BDC).

Central Administration Home

The CA site home page provides you access to the other areas of administration and the Administrator Task List. This page is identical for both WSS and MOSS (except for an additional link to Shared Services Administration in MOSS). There are several ways to open the CA site. For one, select Start | All Programs | Administrative Tools | SharePoint 3.0 Central Administration on the server. When the home page appears, it will look something like this:



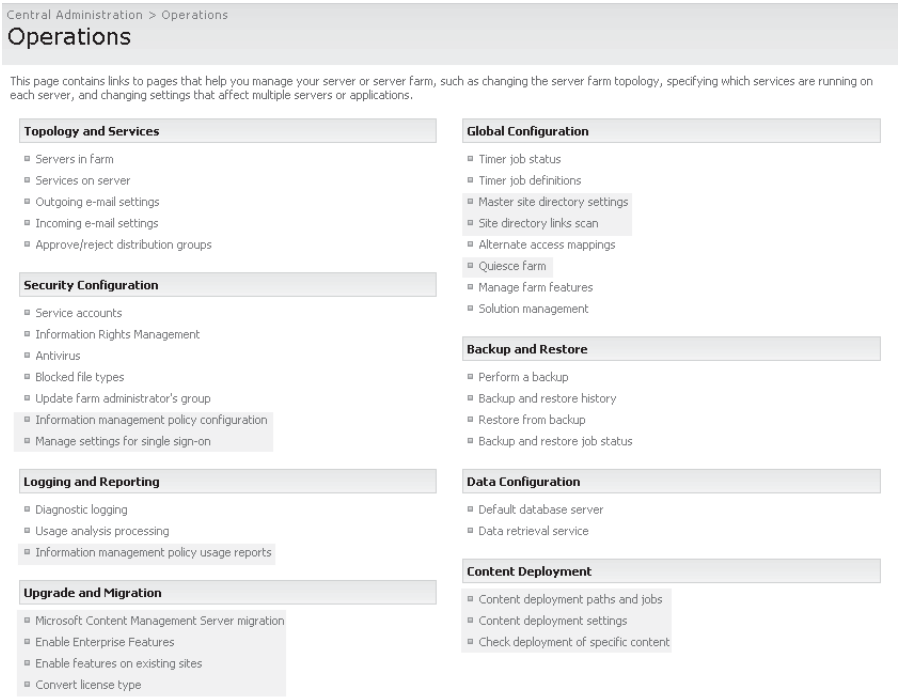
In MOSS, the home page looks exactly the same except for the left-hand navigation.



Of special note is the Administration Task List, which is primarily used during configuration, but is also used periodically when something in the farm needs addressing and you need an easy place to create your own tasks. In addition, there are two web parts shown by default: the Farm Topology, which lists both the current server(s) and services they are running, and the Resources link list, a standard list for adding administrative-related links. This page can be edited and additional web parts added as needed—the same as any other web part page in SharePoint.

Central Administration Operations

The operations side of CA allows you to manage the global aspects of the platform, including server and topology management, security, and logging. To see this, open the CA site via Start | All Programs | Administrative Tools | SharePoint 3.0 Central Administration. From the home page, click the Operations tab or link to display the Operations page. This looks the same in WSS or MOSS; the only differences are a few additional options and sections that apply only to MOSS as highlighted:



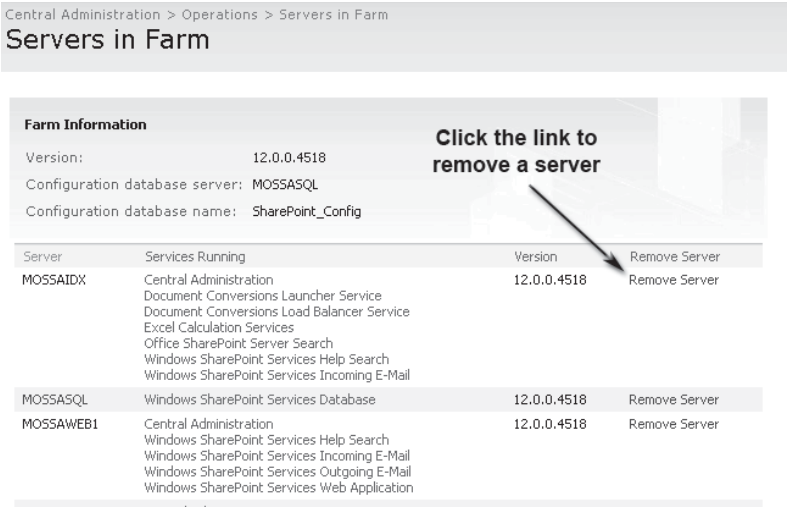
Now I'll go through each option, and I'll explain when and why (if applicable) it is used. All of the options that follow are accessed from the Operations page mentioned previously.

Topology and Services

This section applies to the actual server and assigned services (the topology) of the SharePoint instance.

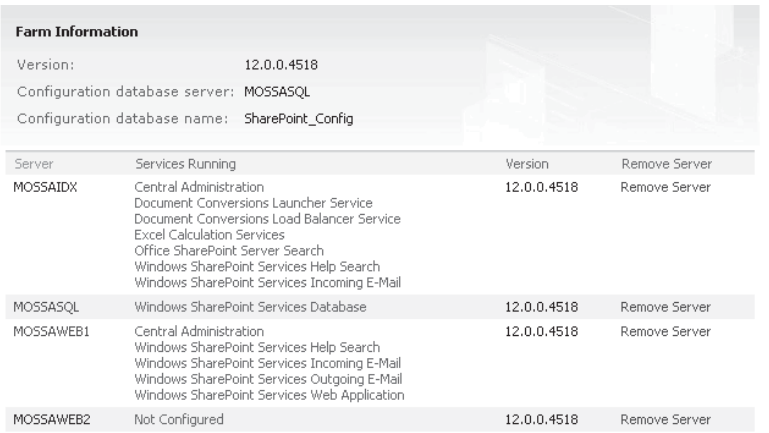
Servers in Farm

- **When you need it** You'll use this option to view or remove servers from the farm.



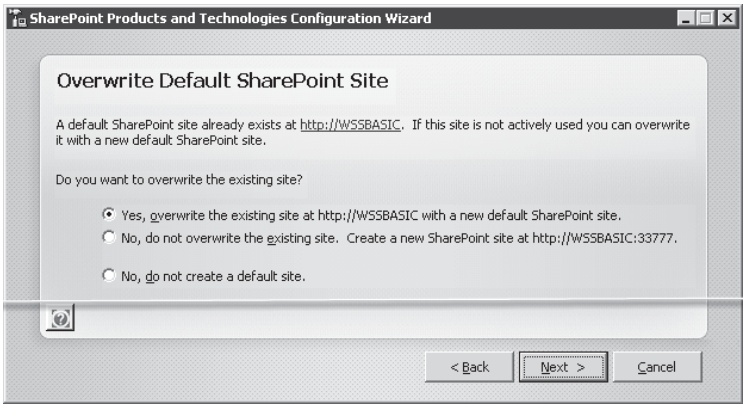
NOTE *The information displayed here comes from the database; if a server is physically down, it will still appear here.*

This option is provided simply for viewing existing and removed servers (adding a server is when a server joins an existing farm during installation—it cannot be added manually). Before you remove a server or uninstall (see below), you should stop all services that server is running. Return to the Operations page, click the Services on Server link, select the correct server in the drop-down list selection, and stop all services being run by that server. On return, you'll see that all services have been stopped and the server is listed as not configured:

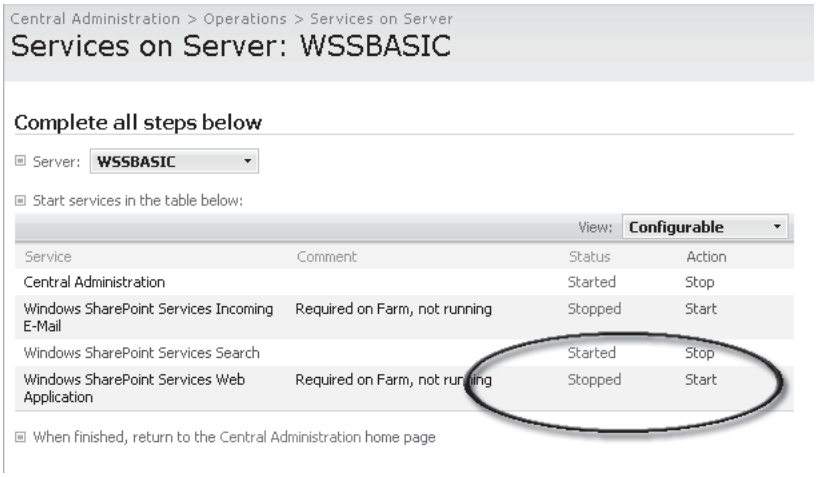


If you are off-lining a server for maintenance or replacement, you can remove a server via the Remove Server link. If you do this, you'll receive a warning message indicating that it is better to uninstall SharePoint from the server directly rather than simply remove it. In most cases, if you are indeed removing the server for good (from the farm anyway), follow this advice. Note that I have had limited success with using Remove; on some systems, attempting to remove the server simply returns a DCLauncher unexpected error message. At that point, the only recourse is to uninstall SharePoint from the server.

Note that while this option is shown in the Basic installation type, you cannot add servers to create a server farm; in fact, if you attempt to attach to the configuration database from a Basic installation, the connection is blocked (you'll receive an error that means that remote connections to the SQL Express database are not enabled). Also, if you remove the server in an installation, your installation will be corrupted and all content and site data will be deleted. The default website will still be provisioned, even though the actual site has been deleted. A server in this state can be reconfigured using the SharePoint Products and Technologies Configuration Wizard, but only to restore the Basic configuration. You do have an option to choose a site other than the default; when you start up the Configuration Wizard, it will detect the state of the server and display the option as shown:



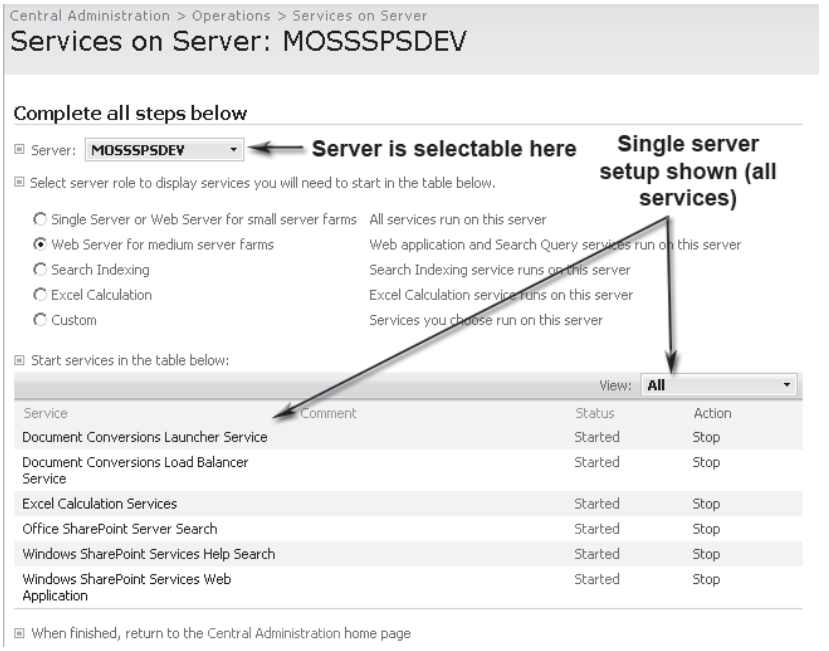
After the reinstall has completed, the Web Service will be disabled. From the CA site, click the Operations link, and from the Operations page, select Services on Server:



Click the Start link to start the web service.

Services on Server

- **When you need it** You'll use this option to view, start, stop, and change the services that are running on servers.



Assigning services to specific servers is done by selecting one from the Web Application drop-down list. Select Change Server, and then click the server name from the list. When one is selected, the current services running on that specific server are shown as well as the current state (usually Started, Starting, or Stopped). Required services that are stopped (such as Windows SharePoint Services Search) show a message that the service is Required on Farm.

The complete details on the individual setup of services are covered in Configuration, but I'll briefly cover what each does. Settable services include the following (for each of these, click the name of the option to set the parameters and use the Start link to start the service):

- **Document Conversions Launcher Service (MOSS Only)** This enables the Content Management Document Launcher Service on the selected server. Any front-end server can run this service but at least one is required, and a Load Balancer must be available to start this service. Option: You can select Load Balancer and the Port number that the service will run under (default is 8082).
- **Document Conversions Load Balancer Service (MOSS Only)** This enables the Content Management Document Load Balancer Service on the selected server. Any front-end server can run this service, but at least one is required; this must be running before a Launcher service can be started. Options: Security Scheme (HTTP the default or HTTPS) can be used with the service running under default Port number 8093.
- **Excel Calculation Services (MOSS ONLY)** This starts the Excel Calculation Service, enabling it to accept requests for documents. The settings for this service are part of Shared Services administration. There are no options available.

- **Office SharePoint Server Search (MOSS Only)** This specifies how Office SharePoint Server search will operate. Options include the Account name to use as the Service account, the SharePoint Content Account, the Database name to hold the search database, the name to use for the Search database and database authentication method (and account), and the Indexing schedule (how often the search will crawl and index).

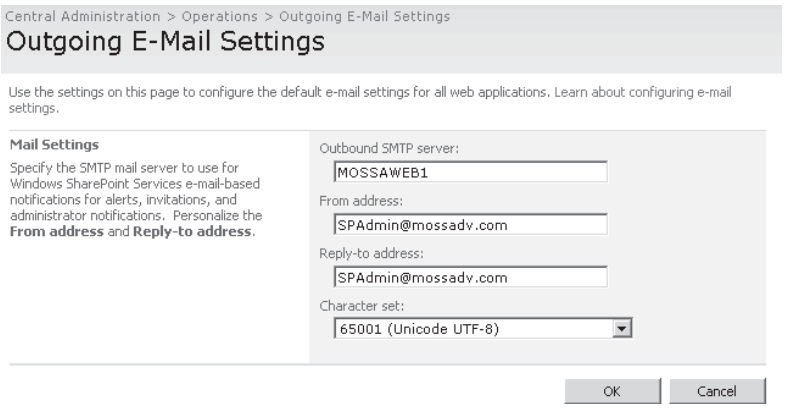
CAUTION *If you define this service, a database is created; if you later redefine the service, you must choose a different database or delete the original one created—otherwise, the creation attempt (and starting the service) will fail.*

- **Windows SharePoint Services Search (WSS Only)** This is the primary search service for WSS to index and search content in local sites (these are stored in the database). The only option for this service is to start it.
- **Windows SharePoint Services Help Search (MOSS Only)** This is a helper service in MOSS for indexing and searching content in local sites and doesn't propagate indexes back and forth (these are stored in the database). The only options for this service are to start or stop it.
- **Windows SharePoint Services Incoming E-Mail** This service handles incoming e-mail sent to sites and lists.
- **Windows SharePoint Services Web Application** This service runs the web services to display sites and is required to be running on all front-end servers in the Web Server role. The only options for this service are to start or stop it.

Services found on servers can be turned on or off as desired. However, be aware that if you've installed MOSS and installed the Web Role only (versus Complete), not all services will be available—only those that pertain to web front-ends. If you have to change this for some reason, you will have to uninstall and reinstall SharePoint on the server you want to add services to.

Outgoing E-mail Settings

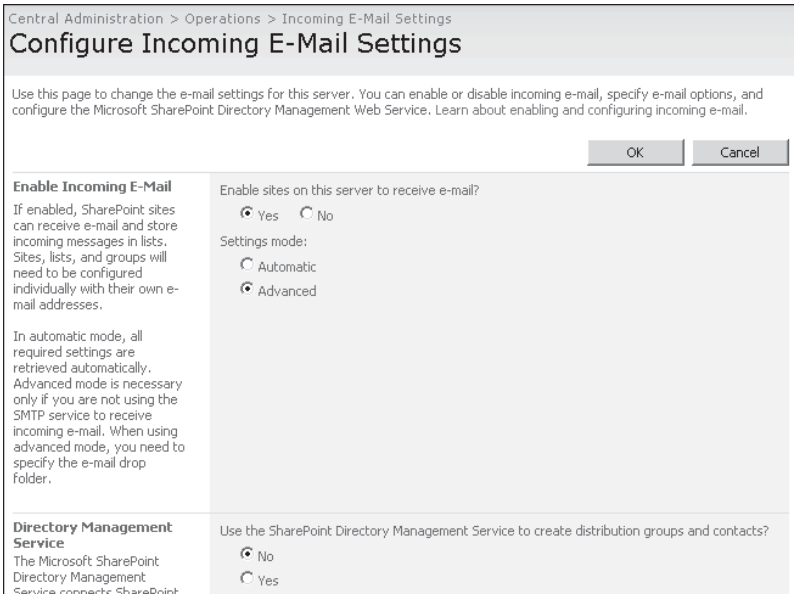
- **When you need it** You'll use this option for a one-time setting for outgoing mail (invitations, alerts) via SMTP.



This allows you to set the name of the outbound SMTP server and the from and reply-to e-mail addresses to be used by SharePoint when it sends e-mail, as well as the character set to use for messages (UTF-8 is the default).

Incoming E-mail Settings

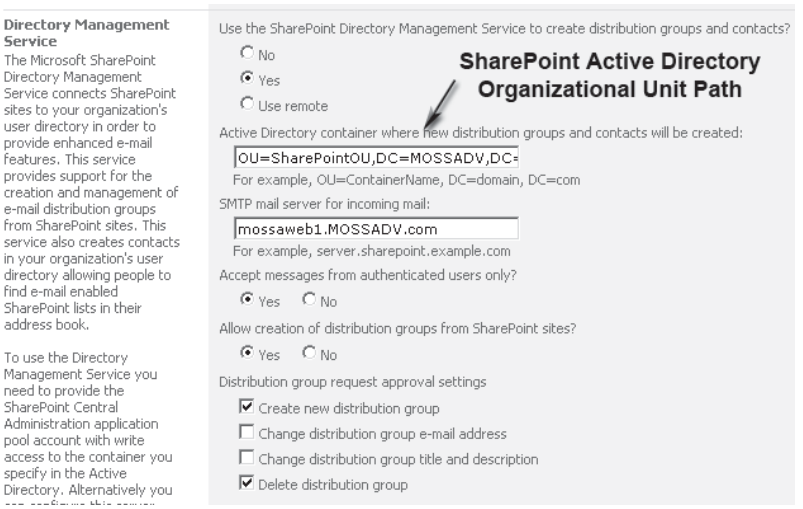
- **When you need it** You'll use this option for a one-time setting for incoming mail to SharePoint Sites and Lists.



This enables configuration of the incoming e-mail message settings for the server or farm. Options include: You can enable sites to receive e-mail and decide whether to automatically get mail folder settings from the server (Automatic setting) or to specify them yourself (Advanced setting). You can also decide whether to use or not use SharePoint's Directory Management Service to create distribution groups and contacts (your choices are Yes, No, or Use a Remote Service). You can also choose the incoming e-mail address (for display in web pages), as well as the list of safe e-mail servers (those it will accept mail from). You can also choose whether to use the SMTP Automatic setting or the e-mail drop folder if employing the SMTP Advanced setting.

The directory service settings are important in terms of how and where you will store distribution groups and users within SharePoint, with your choices being either none or via

a local or remote directory service. The No option has no choices, whereas the Yes option has several:



Selecting Yes means you must specify the following:

- **The Active Directory Container** This is the (LDAP) Query path to the SharePoint AD Organization Unit. If used, this is specified when SharePoint is installed and is used as the container for new users and for holding distribution groups and contacts when they are created by SharePoint (note that an OU can be used to centralize SharePoint users even if the auto-account creation has not been enabled).
- **SMTP Server for Incoming Mail** The SMTP server or exchange server to accept new mail.
- **Accept Messages from Authenticated Users Only** Yes means that only messages from users that are authenticated (against the local AD) will be accepted. No means anyone, without exception.
- **Allow Creation of Distribution Groups in SharePoint Sites** Yes means groups can be created from within SharePoint Sites (by end users).
- **Distribution Group Request Approval Settings** Sets the permissions allowed on group requests, including create, change, or delete.

Selecting the Use Remote option is almost the same as the previous example; however, instead of specifying the path to the local AD, you use the path to the SharePoint Directory Web Service:

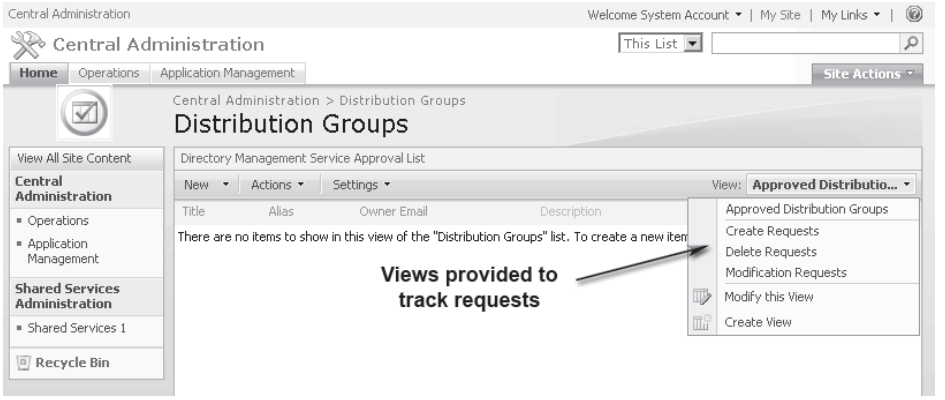


Options here include:

- **Accept Messages from Authenticated Users Only** Yes means that only messages from users who are authenticated (against the remote AD) will be accepted; No means anyone, without exception.
- **Allow Creation of Distribution Groups in SharePoint Sites** Yes means groups can be created from within SharePoint Sites (by end users).

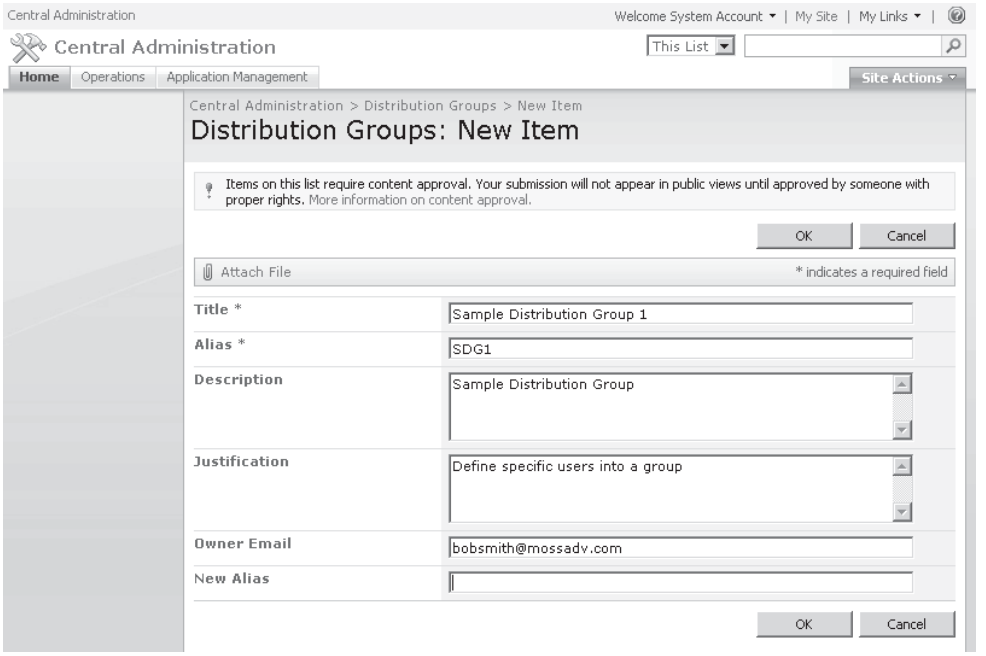
Approve/Reject Distribution Groups

- **When you need it** You'll use this option to add, modify, delete, or approve/reject requests to create Distribution Groups in SharePoint and SharePoint Sites.

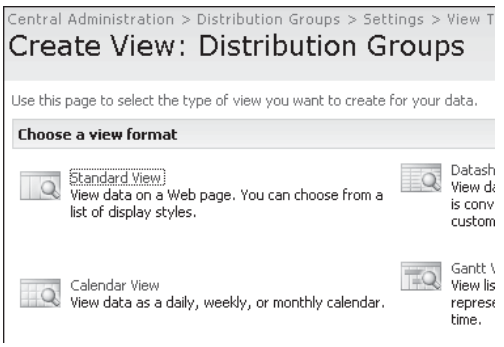


This option provides access for adding, creating, or deleting approved distribution groups (mailing lists) that are available to SharePoint users. This also provides (for Farm Administrators only) approval/reject capability for requests to create or modify groups

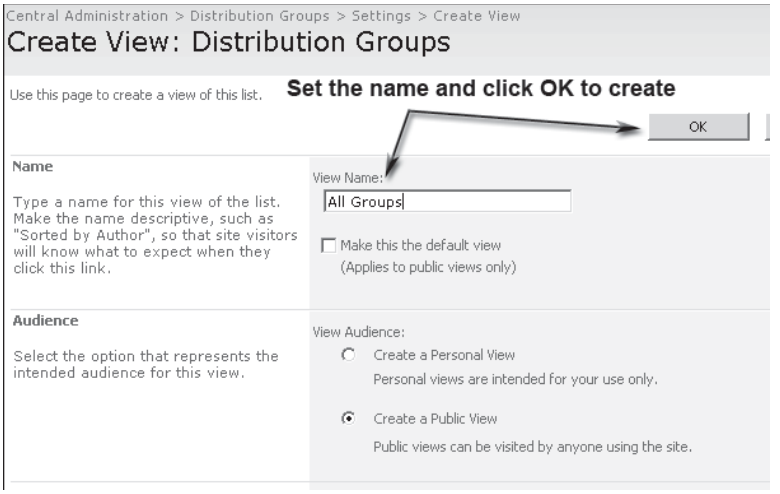
(if enabled). To create a new distribution group, click the New link to open the New Distribution Group page:



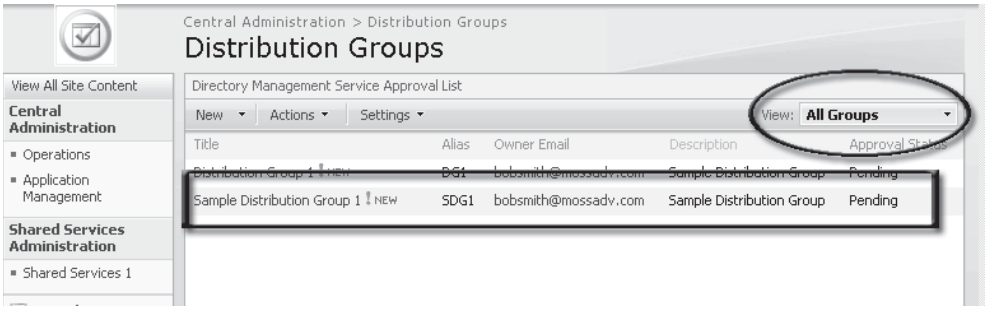
Once you have created a new group, you will be returned to the list. You will notice, however, that the group does not appear in the list. This is because the state of the item is set to pending, and there are no views provided for this. To view pending items, a new view to list all items must be created. From the drop-down list menu on the List View page shown previously, click the Create View link and then select a Standard View:



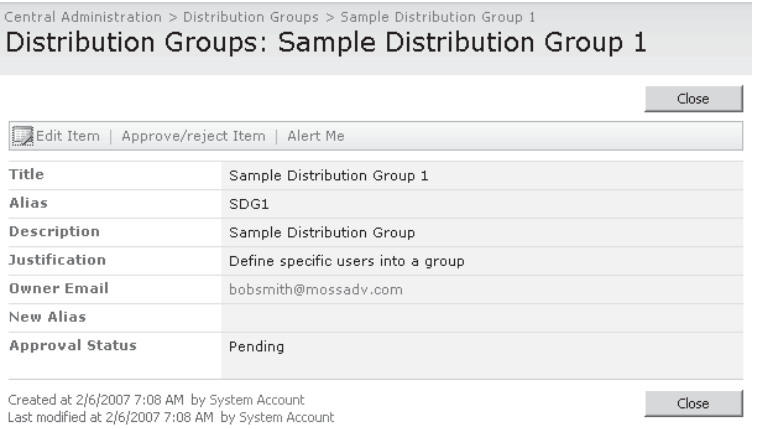
Name the view All Groups as shown:



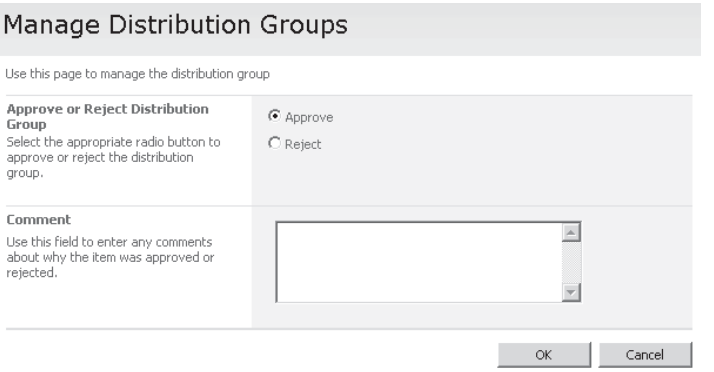
Click the OK button to create the view (leave all other options as is). When you return to the view, make sure the view is selected, and you should see the group you created:



You can now access this item to approve or reject it; mouse over the title and from the drop-down list, select Approve/reject Item:



From the view, click Approve/Reject item:



If you reject the item, it will remain in the list as rejected. If you approve it, it will be listed in the Approved Distribution Groups view and will be available to users. The ability to create and approve/reject groups here, as well as make requests or create groups within sites, is based on the settings in the Incoming E-mail settings. If the path to the directory service is incorrect or the account accessing AD does not have update access, you may receive blank Unknown Error pages when you try to delete or approve/reject a group. If this option is disabled, SharePoint users cannot create or request groups.

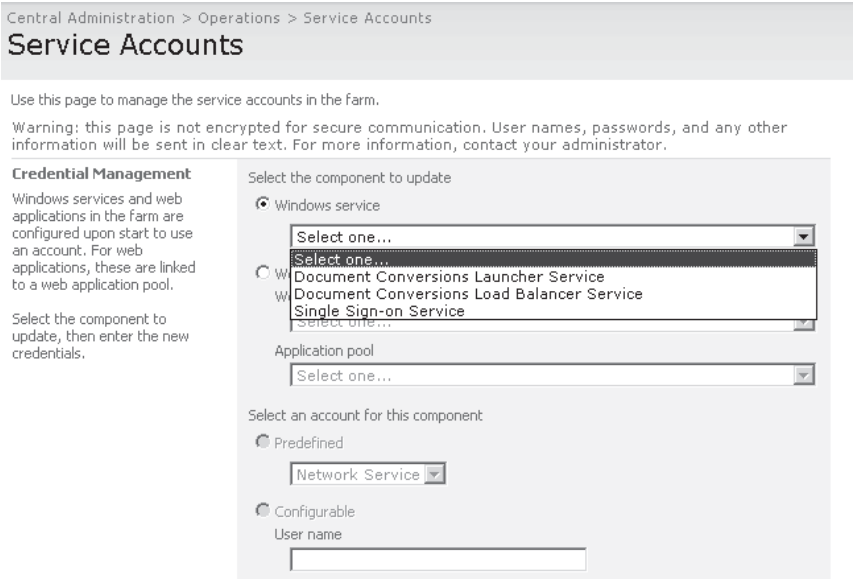
Security Configuration

This section applies accounts used for running SharePoint and other security related items.

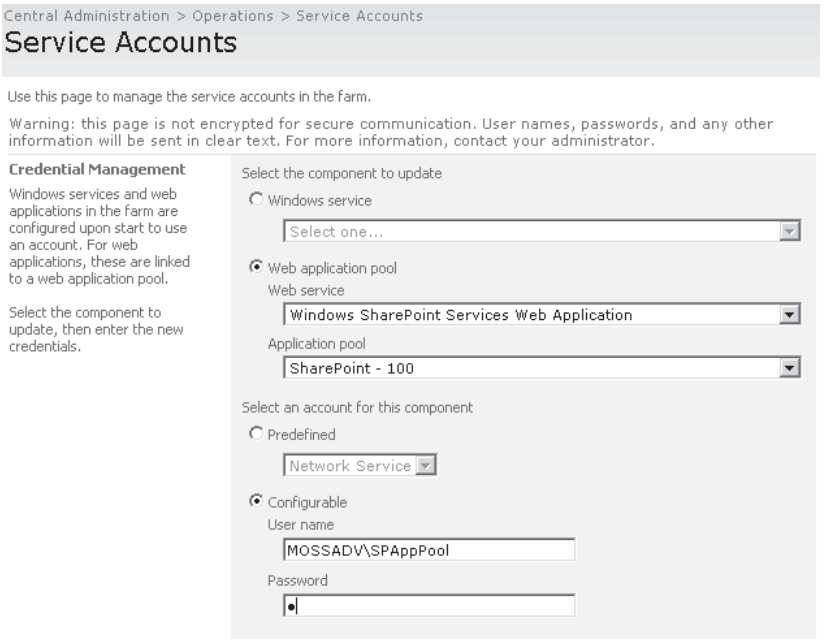
Service Accounts

- **When you need it** You'll use this option to set the Service Account options during configuration or to update the SharePoint Services and Content accounts.

This option enables you to set the accounts used by services and application pools within SharePoint. This is usually done during the configuration, although you will need to use this anytime you wish to change the password (or the account) while running a service or an application pool (in IIS). You can set any one of the SharePoint Services:



Or you can set an application pool accounts for web applications:



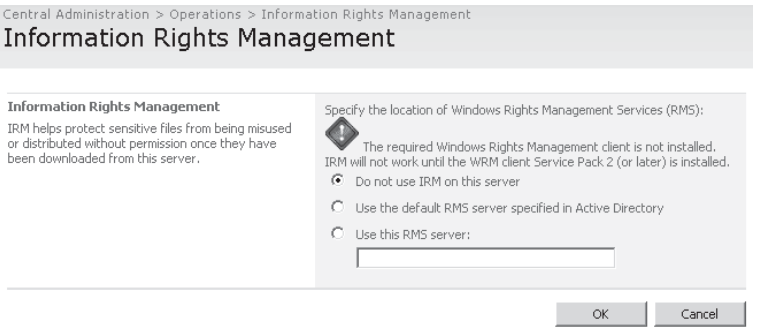
If you do make a change, you must run an IISReset (from the desktop Start | Run…, enter iisreset, and then click OK). If you changed a service, from the Operations page, click the Services on Server link, make sure you are viewing the correct server, and then verify the service is running. If you changed an application pool, open the IIS Manager and verify that the application pool changed is running. If there is an account privilege or password problem, the application pool icon will indicate this with a red warning and show Unspecified Error as the status.

Information Rights Management

- **When you need it** You can use Information Rights Management to enable/disable Rights Management for the server/server farm.

This option is used to enable or disable Information Rights Management functionality within SharePoint. Part of the overall Enterprise Content Management (ECM) functionality of SharePoint, IRM, or the Rights Management System (RMS) enables you to restrict access to documents and presentations based on type of document, extension, and so on. RMS is not solely part of SharePoint; it is implemented across several products, including Microsoft

Office. While RMS is very good at limiting access to sensitive materials, it does have some limitations in what it offers against malicious programs:

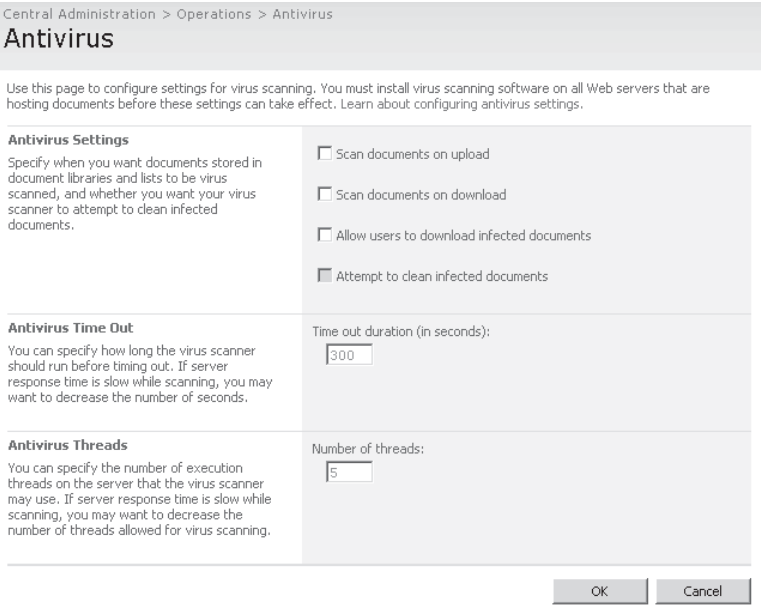


If RMS is enabled in the SharePoint farm, site users can set up rights management on individual documents and form libraries allowing them to pick and choose what types of files are restricted (this is covered under the Library settings in the next chapter). Be aware that this only enables RMS to work with SharePoint; an RMS server must be available in the Active Directory domain. General information on RMS can be found at the link indicated above to download the client, and for more information on how Office uses RMS, see http://office.microsoft.com/en-us/help/HA101029181033.aspx.

Using this feature requires that Microsoft's Windows Rights Management Client be installed (and it must be installed on all front-end servers and client machines). If the client is found but RMS is not configured, you will receive a warning message (as shown). If the client is not installed (including Service Pack 2), you must download it from the RMS information site: http://www.microsoft.com/windowsserver2003/technologies/rightsmgmt/default.mspx.

Anitvirus

- **When you need it** You can use this setting for antivirus scanning options (requires third-party software).

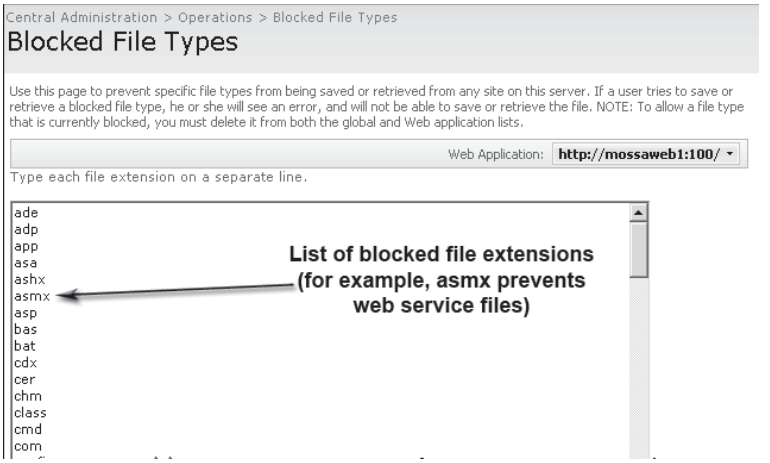


This option enables the setting of the antivirus scanning options for documents. When an antivirus program is installed (on all front-end servers), it can be automatically invoked by SharePoint when documents are uploaded and downloaded. Infected files can also be locked to prevent download. If the software package permits, uploaded files can be cleaned automatically. The final two settings are for the time-out duration (in seconds) to allow for a scanning operation and the number of threads (active instances) that can be invoked at one time (the default is 5).

Using this option requires third-party software and is not a part of SharePoint. You should consult directly with your vendor to ensure compatibility.

Blocked File Types

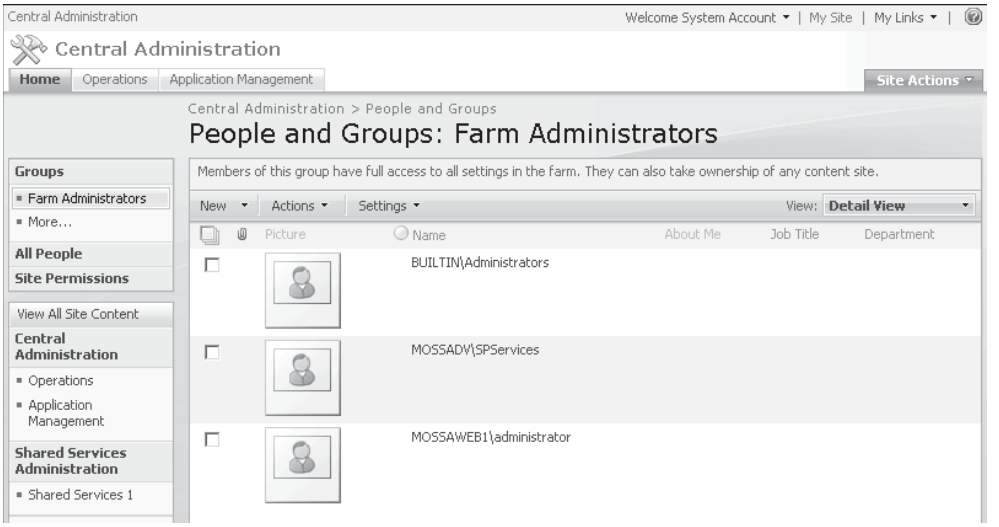
- **When you need it** You can use this option to add or remove file types blocked from being uploaded.



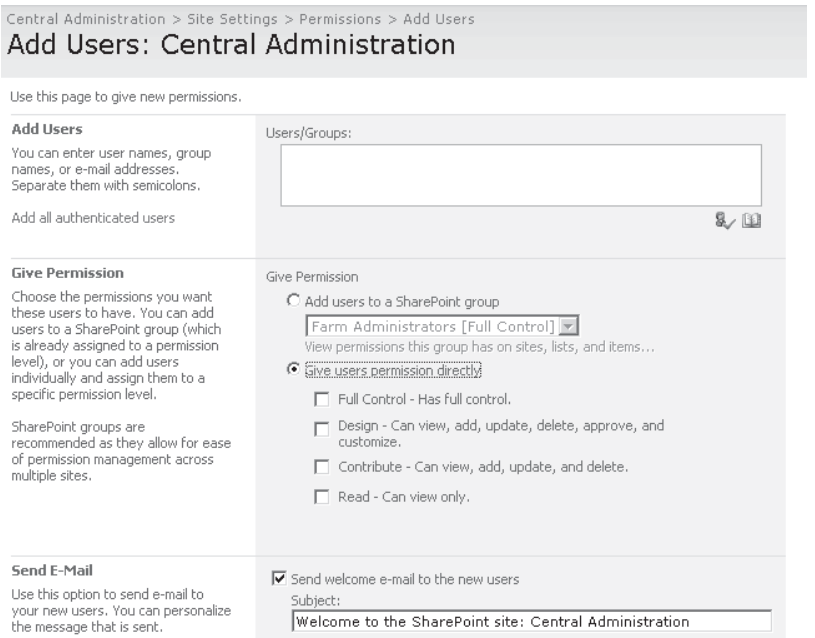
This option allows you to add or remove file types (file extensions) from being uploaded into SharePoint. As shown, Web Services files (with an extension of .asmx) are blocked to prevent execution within SharePoint. When a user attempts to upload a blocked file type, SharePoint responds with a message that the type of file is not permitted. Note that some files typically used in development are blocked (like asmx)—if you are using SharePoint to hold development type files (asmx, exe), you may need to remove some types listed.

Update Farm Administrator's Group

- **When you need it** This option is set during configuration and can be used any time you want to add a user or a new group to the Farm Administrators (Central Administration Users).



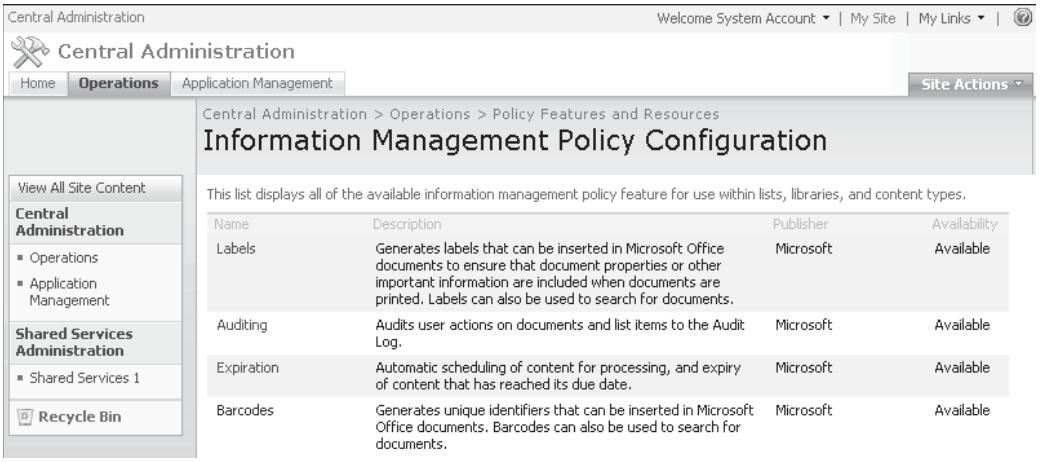
The purpose of the Farm Administrators group is to designate domain users and groups that have access to Central Administration and can manage servers and services based on the rights assigned. For example, when adding a new user (by clicking the New link and then selecting Add User), you can see the rights that can be assigned:



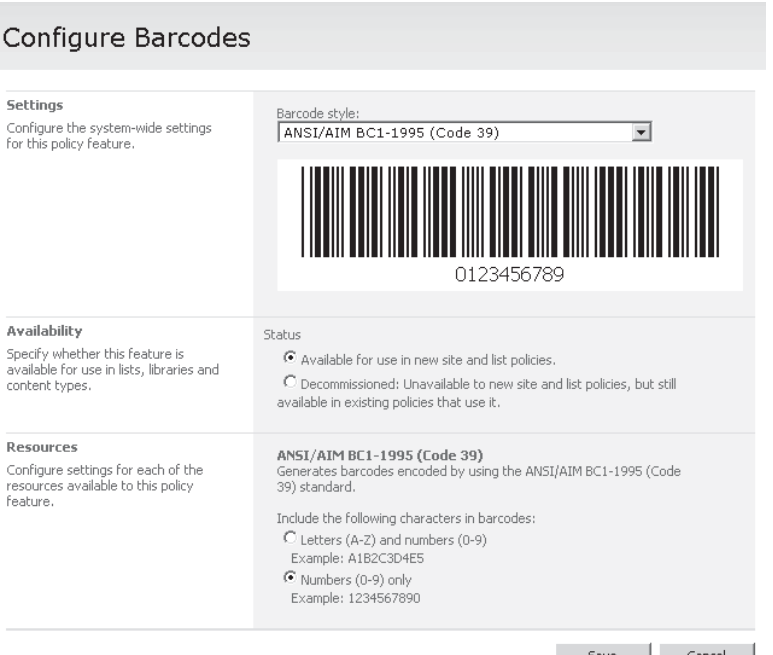
NOTE *These users and groups are only added to the CA site (not sites under applications).*

Information Management Policy Configuration (MOSS Only)

- **When you need it** You can use this option to view, enable, or disable Information Management Policies on content.



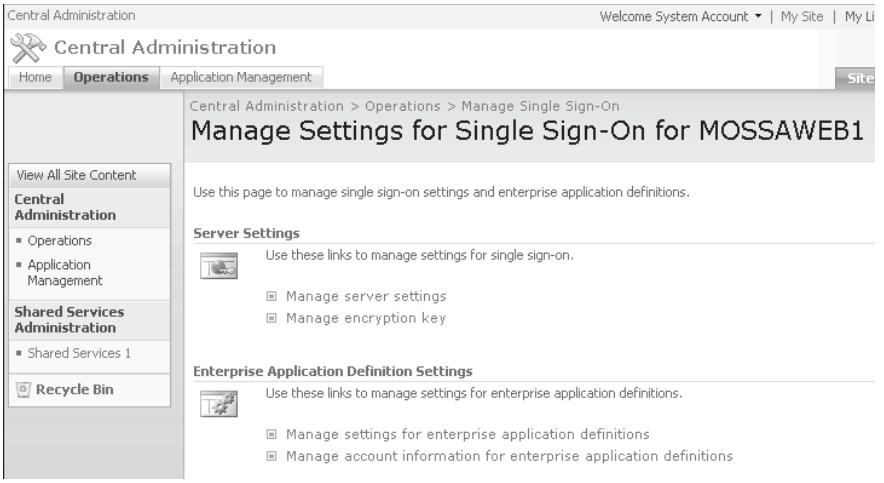
With this option you can view, enable, or disable Information Management Policy features for content management rules applied to documents and lists within SharePoint Sites. A *policy* is a set of rules for content types for basic retention, covering who has access and the ways users are complying with the policy. These policies amount to a series of instructions on how to deal with certain kinds of content and use assemblies to handle transitions and so on (this is covered under Content Management). The most common use for this is records management, but any information that has rules around it can have a policy. A few are provided out of the box, and you can view any of them; for example, the Barcode policy is shown here:



Policies are installed as features and cannot be added or removed from here, simply enabled or "decommissioned."

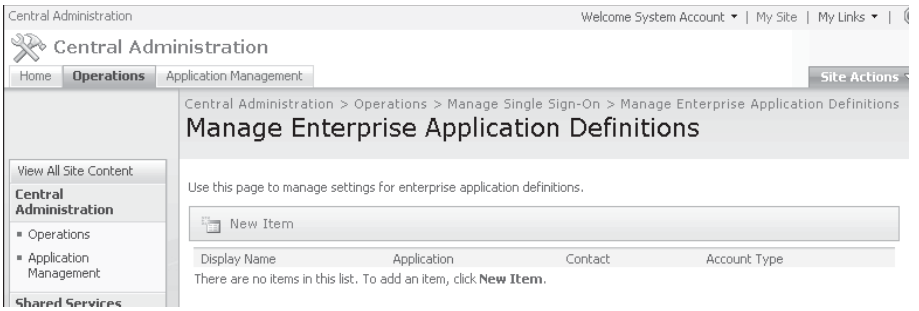
Manage Settings for Single Sign-on (MOSS Only)

- **When you need it** You can use this option to create, update, or remove Single Sign On Application Definitions and to modify SSO service accounts.



This option allows you to manage the SSO service. However, as a one-time setup during configuration, you will mainly use this to add, update, and remove Application Definitions. If this service was not set up during the configuration phase of the installation, you will see the message, "Failed to connect to Microsoft Single Sign-on Service" (usually because the service is stopped). When this occurs, the only link enabled is Manage Server. Rather than repeat the setup here, refer to Chapter 6, perform the setup, and then return to work with application definitions.

To create, modify, or delete an application definition, click the Manage Settings for Enterprise Application Definitions link:

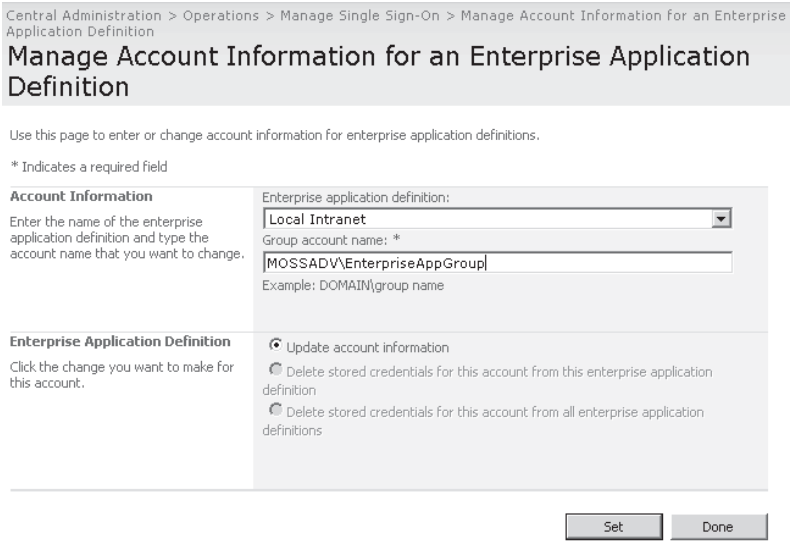


From this list page, you will see any defined applications. To remove one, you can simply mouse over the name and select Delete. To create a new one, click the New link, which will open the Create Application Definition page. On this page, you can supply the application information, type of authentication, and from one to five fields that can be

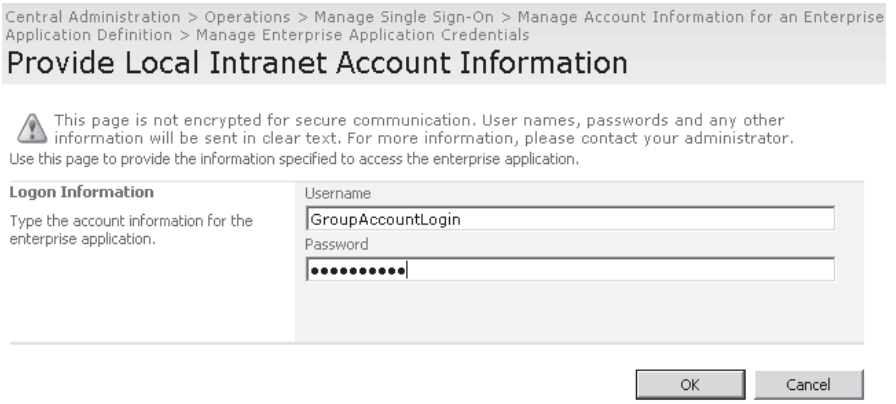
passed for authentication. In forms-based applications, these fields are typically Username and Password, as shown (also note that the Password field can be masked to prevent viewing when entered). The specific information needed here is as follows:

- Application and Contact Information
  - **Display Name** This is a friendly name for users.
  - **Application Name** This is an application name for creating data connections and for use by developers.
  - **Contact E-mail** This is the contact for this application.
- Account Type
  - **Group** One account is used by all users.
  - **Individual** Each user has an individual account.
  - **Group Using Restricted Account** One account is used by all users with restricted access.
- Authentication Type
  - **Windows Authentication** This is either enabled (NTLM) or disabled for all others.
- Login Account Information
  - From one to five field names that will be prompted for and passed to the application to authenticate; these can be masked as needed for passwords.

Once you have created an application definition, you must assign account(s) for it. To manage accounts for existing applications, from the Manage Single Sign-on page, click the Manage Account Information for Enterprise Application Definitions link, which opens up the Manage Account Definition page. From here, you can use the drop-down list to select an application and specify the account information. For the previous example, use a single group account as shown:



To set the credential values, click the Set button, which will open the authentication options where you can specify the values that are passed to and from the SSO to the application (these are the one to five fields defined in the Application Definition itself). In this example, only two fields Username and Password are required, and these are displayed as shown:



Click Done to save the credentials. At this point, the Application Definition is complete. If there is any need to change credentials, you can always come back to modify them if needed.

NOTE *When working with application definitions, you must be logged in as a Enterprise Application Definition account or as a member of the assigned group (set in Manage Server settings).*

Logging and Reporting

This section covers the logging, diagnostics, and log reporting features.

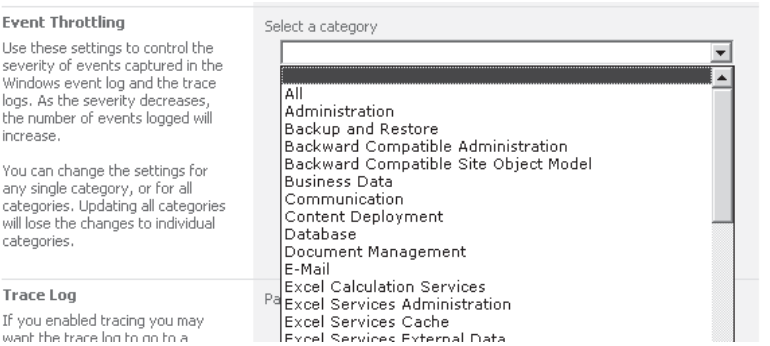
Diagnostic Logging

- **When you need it** You can use diagnostic logging to set levels and items to log for diagnostic testing or troubleshooting.

Diagnostic logging in SharePoint is designed to help detect and hopefully avoid problems in the installation. It's also where you can set some general reporting options for Microsoft to use (anonymously) for bug fixes and future service packs. There are four sections to these settings as follows:

- **Customer Experience Improvement Program** By turning this on, Microsoft may contact you regarding its efforts to improve the customer experience; this includes notifying you when it has a potential fix for issues. This is completely optional but recommended.
- **Error Reports** This section provides you with the option to collect or not collect error reports caused by system faults and unrecoverable software errors (I'm sure you've seen the "Send this error to Microsoft" message before). I recommend you keep this enabled, because the more information about errors we can get to Microsoft the better. Also, be sure to enable this to run in silent mode so as not to disrupt console operations.

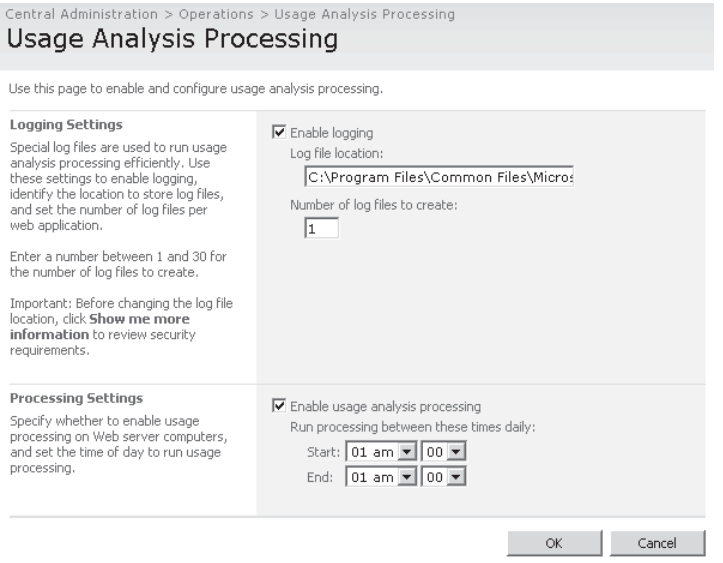
- **Event Throttling and Trace Log** These sections let you specify a specific category you may want to throttle, as well as the location, maximum number, and amount of time you need to maintain trace logs. Event Throttling enables you to pick a specific service to monitor and then set the least critical event to report to the System Event Log and to the Trace Log. As you can see, there are a number of services that can be traced:



- **Trace Log** This allows you to set where the log files will be created. By default, this is the SharePoint Hive Logs directory and should be changed, since this will most likely be on the C drive (as a best practice, log files should never be on a system drive). Other options include the number of log files to retain before overwriting and the number of minutes before the log is cycled. In this case, 96 log files are maintained and refreshed (creating a new one and closing the old) every 30 minutes (48 hours worth). One thing to keep in mind is that logging can bog down your system and shouldn't be used unless needed.

Usage Analysis Processing

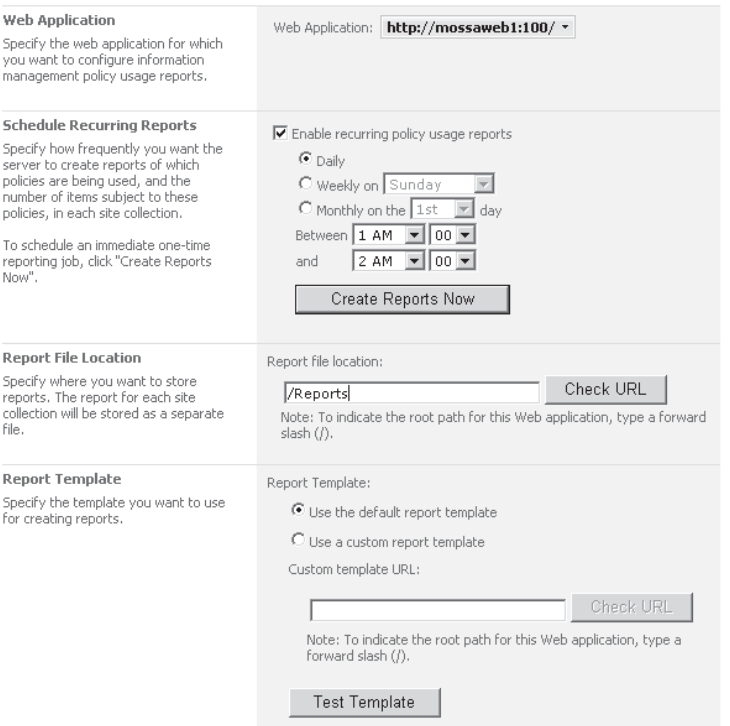
- **When you need it** You can use this option to enable/disable usage analysis logging.



Usage analysis processing is used to increase logging for analysis of site activities, user actions, and so on for reporting. From this page, you specify the location for the logs and the number of logs to create for each application (from 1 to 30). The location defaults to the SharePoint Hive Logs directory, which is usually on the system drive. This should be changed to a non-system drive that has sufficient space to store the logs (remember the number of logs is multiplied by the number of applications). The Processing Settings determine when to run usage processing and allow you to fine tune it to a specific period during the day (you might, for example, want to monitor only peak activity in the afternoons). As I've mentioned before, logging can bog down your system and shouldn't be used unless needed. Always be sure files are on a non-system disk that has plenty of space.

Information Management Policy Usage Reports (MOSS Only)

- **When you need it** You can use this option to set the location and schedule of Policy Usage reports and run reports on demand.



This option sets the schedule and location for the Information Management Policy reports (based on the policies that are defined in Information Management Policies mentioned previously). Reports can be scheduled to run automatically and on demand if needed; the location of the reports is specified as the relative URL to the location (/ = root of the site).

Upgrade and Migration (MOSS Only)

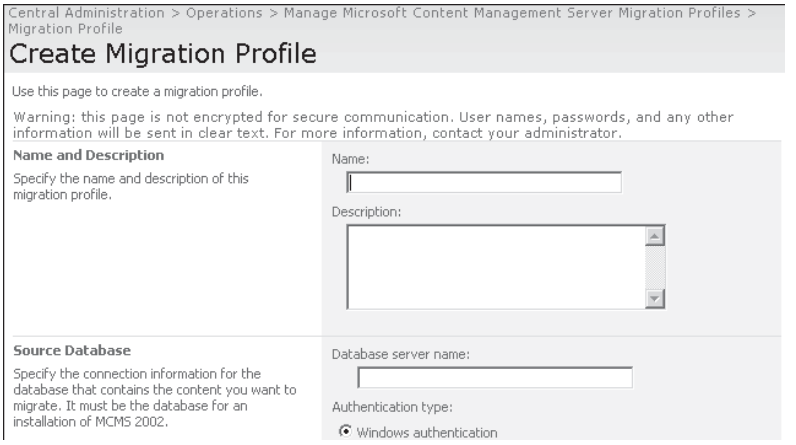
This section applies to MCMS 2002 systems migration to MOSS and license/feature upgrades to SharePoint.

Microsoft Content Management Server Migration

- **When you need it** You can use this option to migrate existing MCMS 2002 content to SharePoint.



This option enables you to move existing MCMS 2002 content directly into a SharePoint Site. Using Migration Profile, you can set up a migration plan and then run it to migrate existing content. To create a migration profile, click the New Migration Profile link:

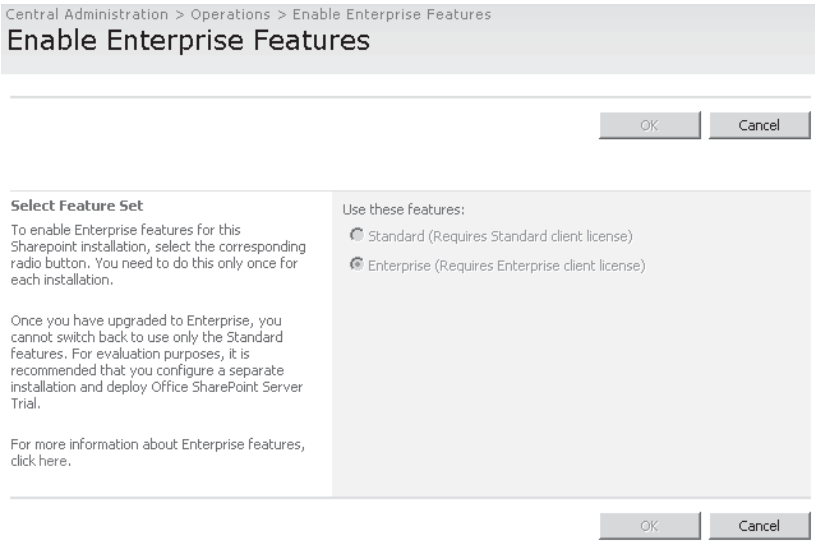


At the start, you will enter a name for the profile and supply a description. In the next section, you must supply the MCMS database information; here, you enter the (MCMS) database server name and authentication type (the account and password is required). Click the Connect button to establish a connection, which then enables the Database Name drop-down list to select the database for the migrating site. Next, you select the destination in a SharePoint Site, where you then use the drop-down list to select the Web Application and then choose a site collection within it (/ = the root). Finally, select the MCMS site that will become the top-level site in SharePoint.

In the next section, you can enable or disable the migration of ACLs (Access Control Lists), which is otherwise known as file security for the content (in short, pulling the same MCMS security into SharePoint). Then, you can select the language for the site (language packs must be installed on all front-end servers), and finally, you can set the notification options to report the migration status. Click the OK button to save the settings, and when you return, you will see the new profile listed. To run it immediately, mouse over the name to expose the menu options.

Enable Enterprise Features

- **When you need it** You can use this feature for a one-time upgrade from the Standard Edition to the Enterprise Edition.



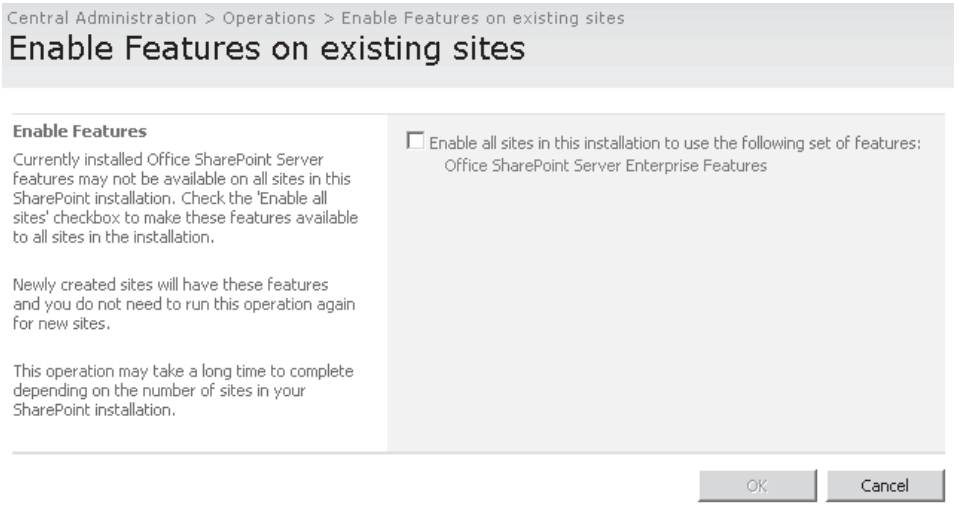
This one-time use option will upgrade the Standard to the Enterprise Edition and enables the following features:

- Business Data Catalog
- Excel Services
- Report Center
- InfoPath Forms Services
- KPI and Filter Web Parts

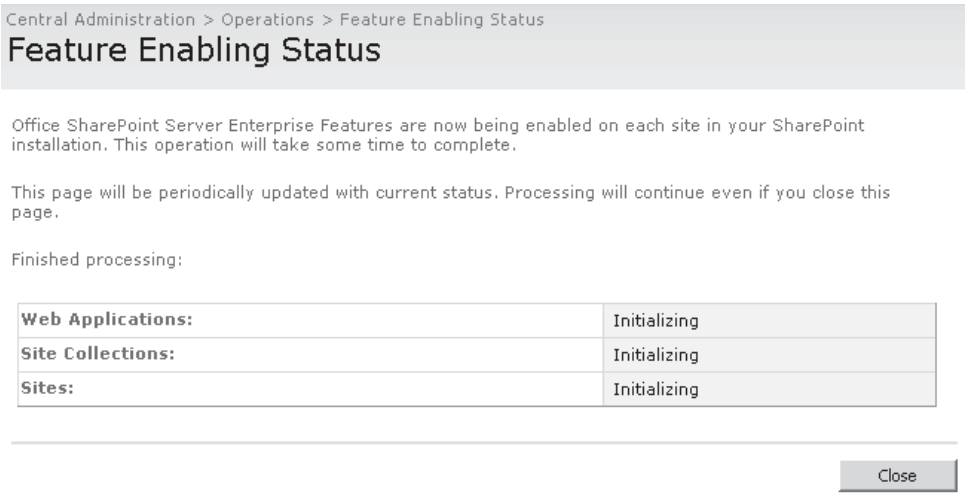
Configuration of the services here are through the Shared Services Provider Administration site hosting the services. These features are installed automatically (and the page disabled as shown), if you install the Enterprise Edition to begin with. Be aware that Enterprise Edition Client Access Licenses are required to use it—not having them won't affect operation but will give you a problem with Microsoft.

Enable Features on Existing Sites

- **When you need it** You can use this to enable enterprise features on existing sites.



This option enables the Office SharePoint Server features on existing applications and sites. This is usually done one time after the initial sites have been set up, but may be needed periodically to enable the features on new sites. To enable the features, click the Enable All Sites… check box and click the OK button. This will display the Feature Enabling Status page:



This page will continue to refresh as it iterates through the sites; once started, processing will continue even if you navigate away from the page. When this has completed, you are presented with a finished page. Click the Close button to exit.

Convert License Type

- **When you need it** You can use this option for a one-time conversion of the install from one license type to another.

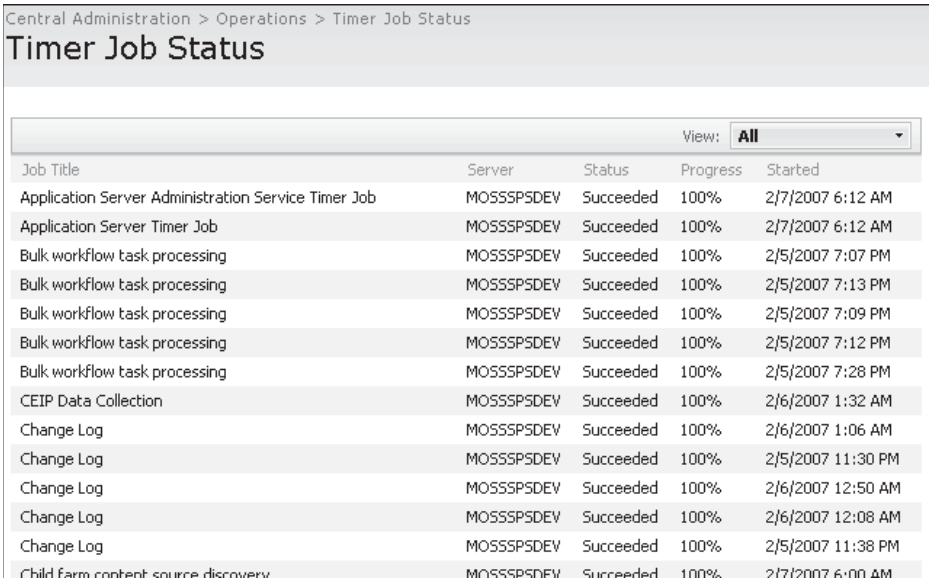
This option enables you to update the installation license from one product version to another, such as when you want to go from a trial to a full license. If you are upgrading, simply enter the product key and click OK to validate the key. When you return to the Operations page, the license will be converted (this can only be done one time).

Global Configuration

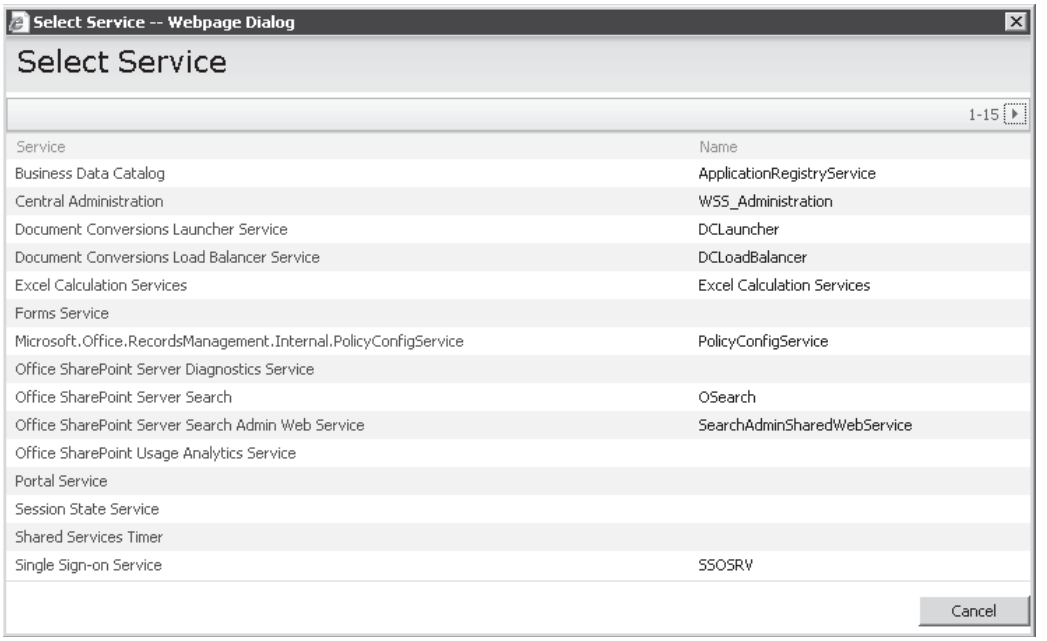
This section applies to global settings of the platform.

Timer Job Status

- **When you need it** Use this option to check the current status of any SharePoint job (including provisioning and search) running under the timer service.

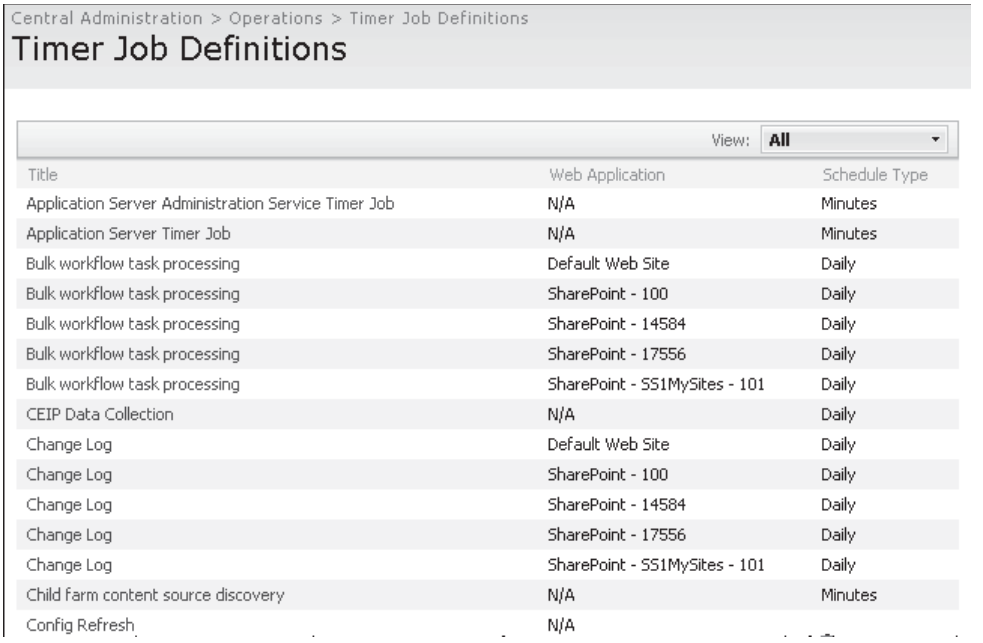


This is a simple status display that allows you to view current jobs that are running under the OWSTIMER service. Several views are provided, including All, Services, or Web Application jobs. For example, if you select to view by Service, you can choose to view jobs for a specific one:



Timer Job Definitions

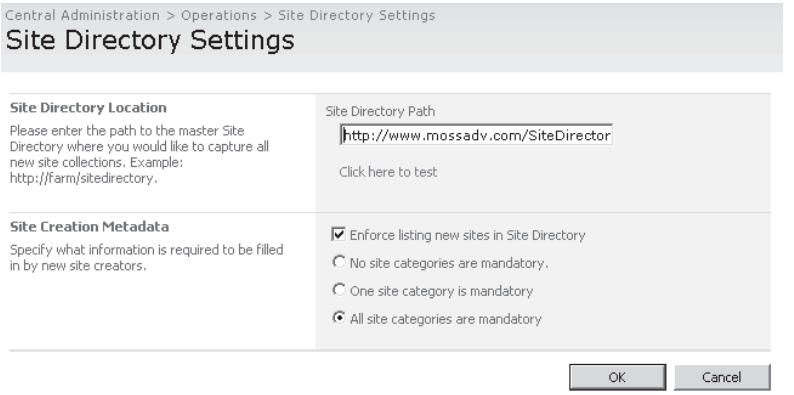
- **When you need it** You can use this option to view or disable/enable a timer job.



This option allows you to view and disable/enable job definitions run by SharePoint and services. Clicking on an individual job enables you to disable it (if enabled) or enable it (if disabled). Like the timer job status, several views are provided (All, Services, or Web Application). Note that jobs are not created here (this only views/controls them); they are created by other services.

Master Site Directory Settings (MOSS Only)

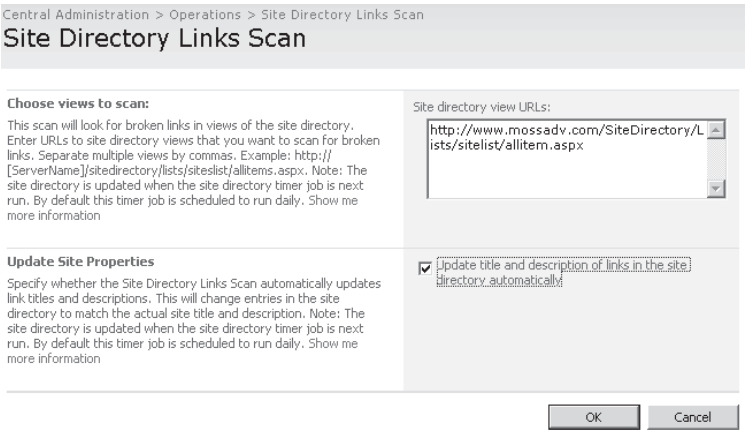
- **When you need it** You can use this option to set the default path for creating new site collections.



This option enables you to set a specific location for site collections instead of allowing users to choose. If set here, you can also set the metadata rules for adding, which can include no rules, must add to Site Directory, and part or all of the metadata set as required. Note that changes made here do not affect existing site collections.

Site Directory Links Scan (MOSS Only)

- **When you need it** You can use this option at any time to validate and cross reference links in a Site Directory.

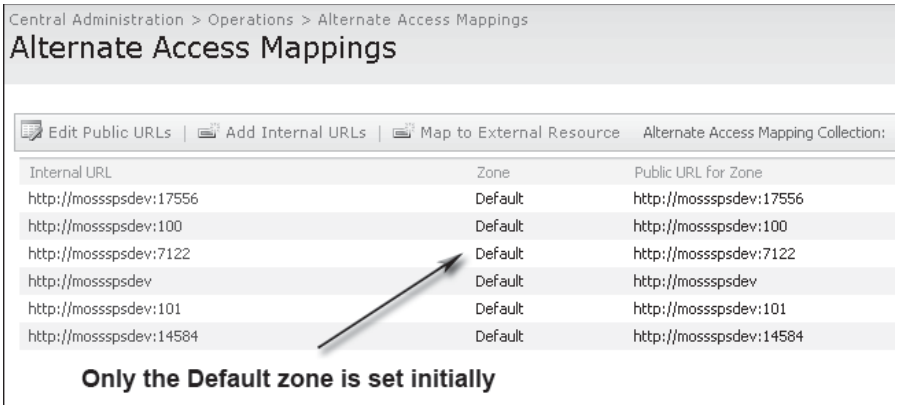


This option handles two different utilities for Site Directory links (site directories are used in the portal site types; it provides a custom list that can list all sites created under the

portal). The first utility will scan any listed site directory location(s) entered and validate all links within them (eliminating bad ones). This is ideal for highly volatile sites where sites are commonly removed, but is also just a good check to ensure that the sites listed are indeed present. The second option enables you to cross-validate the titles and descriptions of sites listed in the directory against the actual site. This is handy when full descriptions are not added to the directory, and if these have changed, it is also a good check to correct potential misspellings in titles.

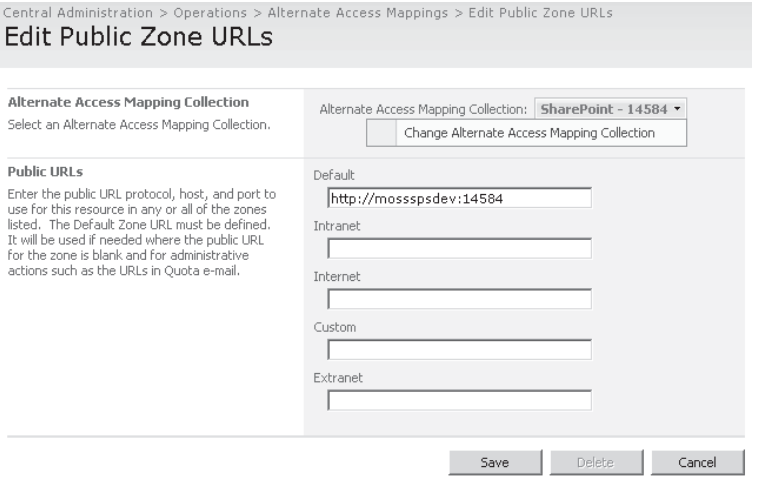
Alternate Access Mappings

- **When you need it** You can use this option to map friendly URLs to Internet Zones.

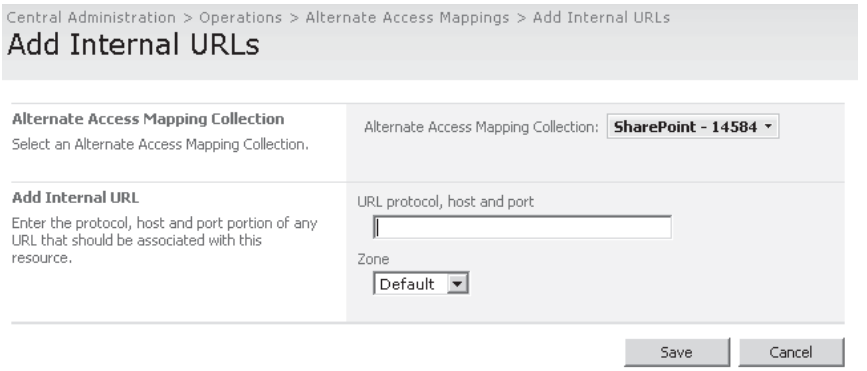


Alternate Access Mappings (AAM) allow you to set alternate friendly URLs for each Internet zone (Intranet, Extranet, and so on) that can be assigned to each application (each application has a collection for each Internet zone) for both public and internal access. Public URLs are what users use to get to a SharePoint site, and are also shown in the browser as bookmarks. Internal URLs are used for URL requests that are sent to the site. You can have any number of internal URLs associated with single public URL in a load-balanced environment (Server Farm). Each application can support five collections for each Internet zone type: default (set for you), Intranet, Extranet, Internet, and Custom. When a request for an internal URL is made from a zone, the lists on the pages returned include the public URL for the zone.

Choices here include setting the public URLs, adding internal URLs, and adding external references (external site links). Public URLs define the friendly URLs used to access the site from the public side:



Internal URLs enable you to create AAMs for internal site links. The name mappings here are available on the internal side:

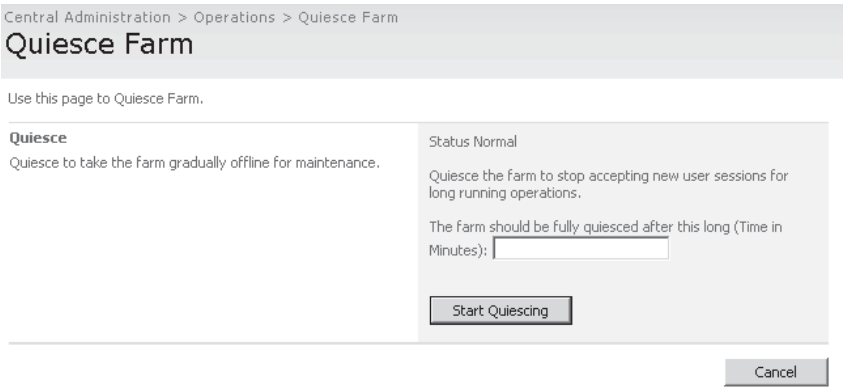


External references enable you to create AAMs for external sites (you only need the name and URL). These mappings are not embedded anywhere and are dynamically used so they can be changed at any time if needed.

NOTE *AAMs do not accommodate for authentication; in public-facing sites, authentication is usually required to implement a front-end identity management server.*

Quiesce Farm (MOSS Only)

- **When you need it** You can use this option when the Farm should be brought down gradually for maintenance.

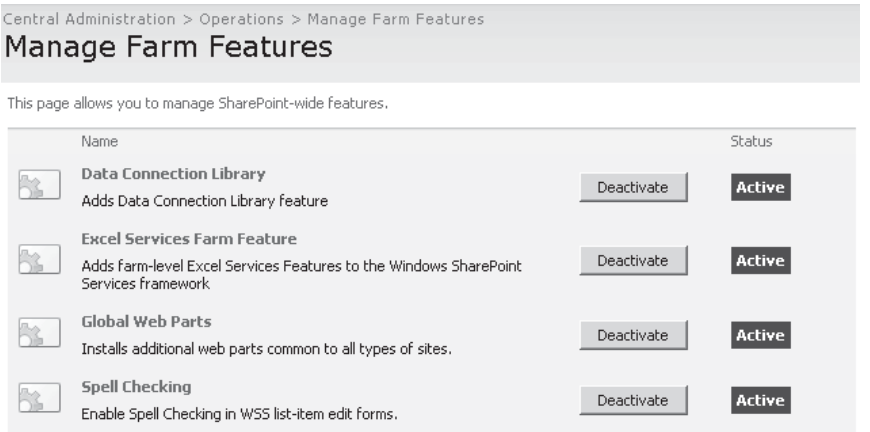


This option enables you to shut down the farm services gradually by denying new connection requests while letting existing sessions end without forcing them to close. From this page, you can select the amount of time to allow for the shutdown (in minutes), and when you click Start Quiescing, new connections will automatically be stopped. When selected, you are shown the current status while the shutdown is in progress (you can stop the process by clicking the Stop Quiescing button). When the process has completed, it will display a new page with the message "The farm is quiesced. There are no open sessions." At

this point, all connections are prevented and nothing is active, so the entire server farm (or any server in it) can be brought down. Once you have completed maintenance or whatever was required, you can return to this page and click Reset Farm. This will open all connections and operations will resume normally.

Manage Farm Features

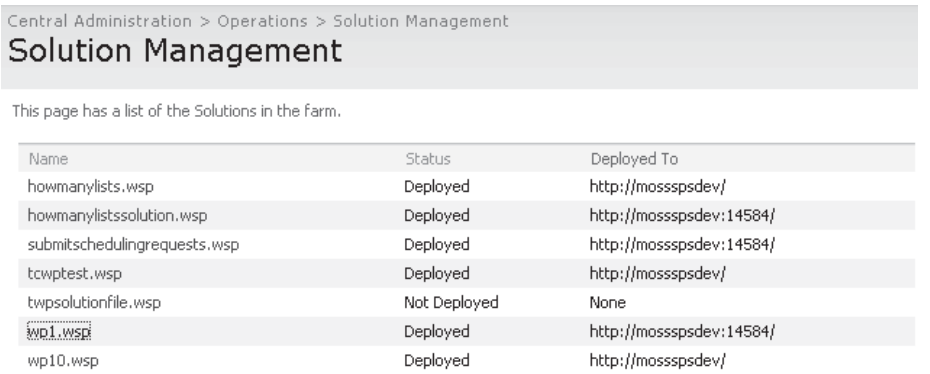
- **When you need it** This option allows you to enable or disable farm-wide features.



This option allows you enable or disable farm-wide features (such as the Excel Service). As you can see, Active indicates that the service is enabled, whereas Deactivate disables it. When the feature is disabled, click Enable to enable the feature.

Solution Management

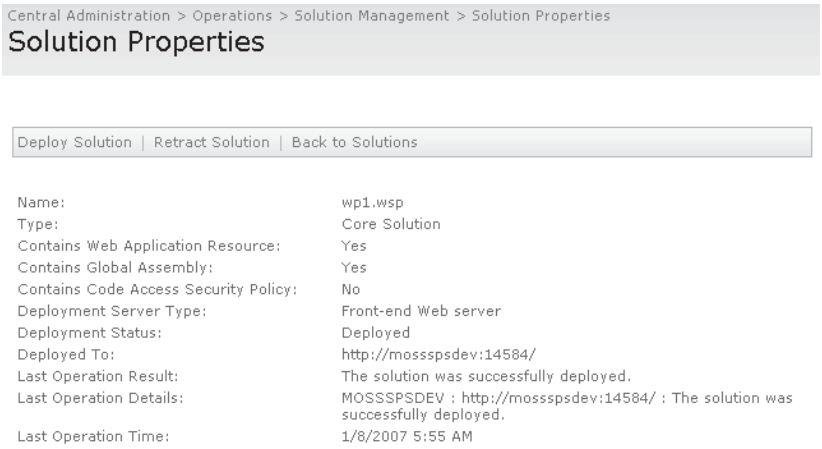
- **When you need it** This option enables the deployment and retraction of solutions from the farm.



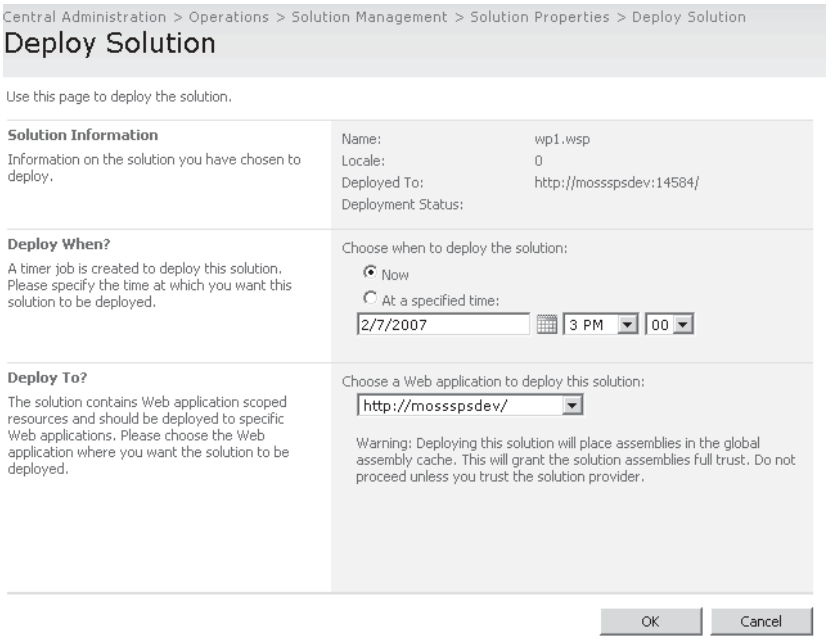
Name	Status	Deployed To
howmanylists.wsp	Deployed	http://mossspsdev/
howmanylistssolution.wsp	Deployed	http://mossspsdev:14584/
submitschedulingrequests.wsp	Deployed	http://mossspsdev:14584/
tcwptest.wsp	Deployed	http://mossspsdev/
twpsolutionfile.wsp	Not Deployed	None
wp1.wsp	Deployed	http://mossspsdev:14584/
wp10.wsp	Deployed	http://mossspsdev/

Solutions are like features but come in the form of definitions and web parts. From the ones displayed in the previous image, you can see a filename extension of .wsp, which

indicates that this list shows only web parts. Clicking any one of the items will show you the state and options available for that solution:



Solutions are not added or removed here—they are installed using the STSAdm command line. This option only permits you to either deploy an installed solution or retract it from the farm (necessary when uninstalling it). Actual activation of a solution is done at the site level. As you can see, your options are either to Deploy Solution (this one is already deployed) or Retract Solution, which means you want to remove any existing deployments. To deploy a solution, you are given options as to when to deploy it and which applications to deploy it to:



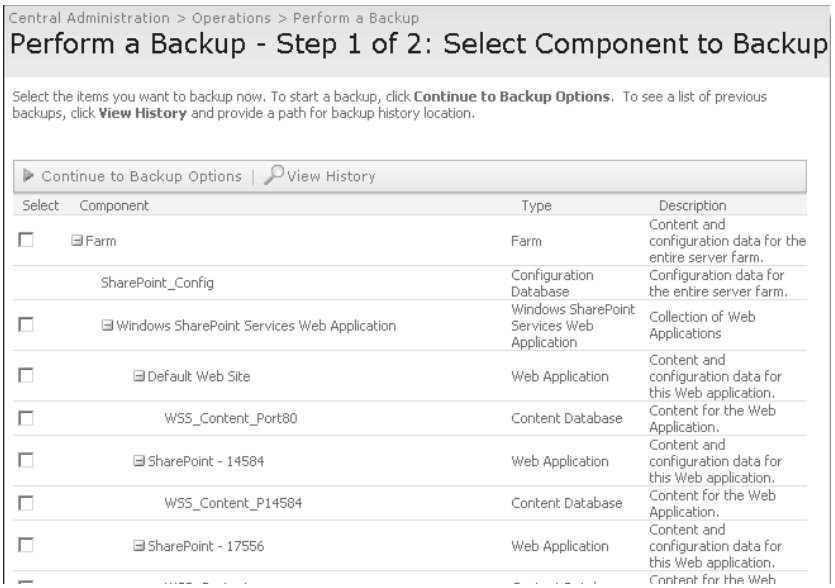
As you can see, you can choose an exact date and time to deploy. This allows you to stage in changes, pre-notify users, and so on. Retracting a solution works exactly the same way.

Backup and Restore

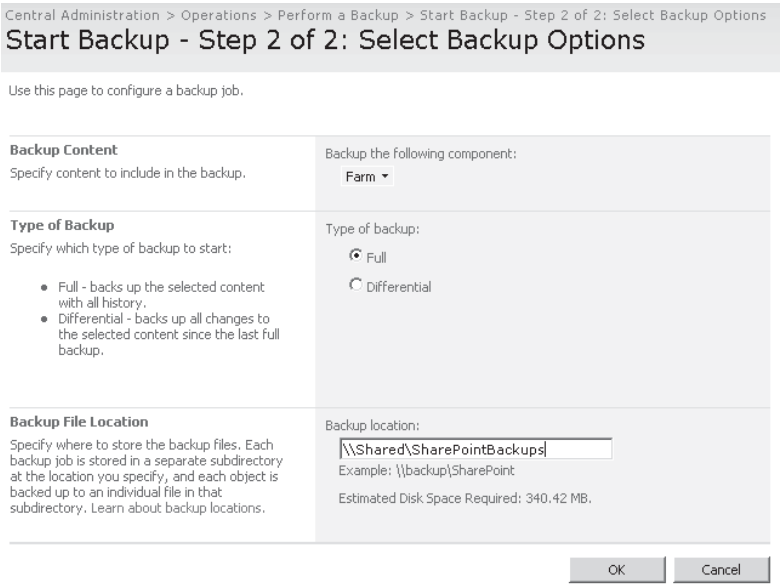
This section covers the utilities SharePoint provides for handling the backup and restoration of sites, much the way SQL Server does with databases. Features include the ability to set up backups appropriate to site usage, perform quick restores, and provide scheduling of differential and full backups to ensure sufficient restore capabilities.

Perform a Backup

- **When you need it** You can use this option to create or schedule a SharePoint backup.

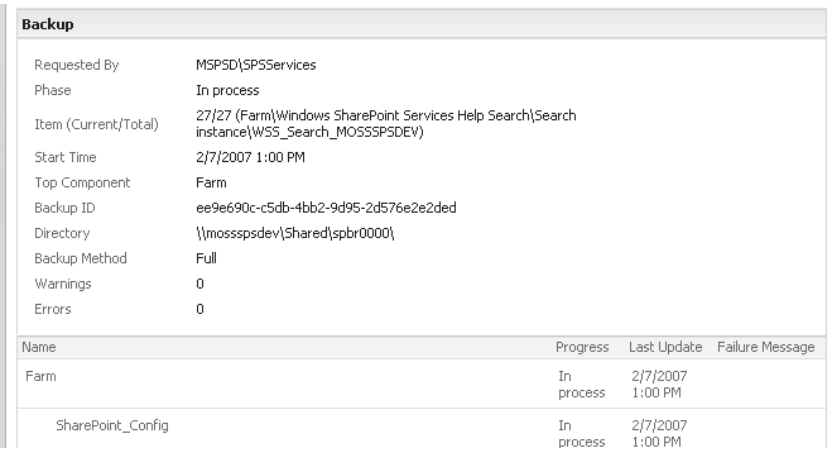


This option allows you to back up any part of SharePoint from a single application to the entire farm. In this first step, you select items to back up by clicking the check box next to each (clicking the farm selects everything). When you click OK, you are presented with the backup type and location options:



Full backup is obviously a complete backup. Differential, on the other hand, will only back up changes made since the last successful backup (remember that in a recovery, you would have to restore to the last full backup and then restore each incremental backup). The location is specified as a files share (this is required), as shown by \\Share, and then the folder within that share (this can also be a web share or an FTP address). The share you specify here is required; you cannot use a local device or path (like c:\). This share should be defined on a network storage device, SAN, or similar disk with capacity to hold the size of the site (you should figure about 20 percent over what is shown as the estimated size). For developers, you can easily create a local shared folder to use.

Once you have started a backup, SharePoint creates a Timer Job and begins the processing. As the job progresses, it will refresh the status to indicate what is being backed up—any errors will be shown as well:



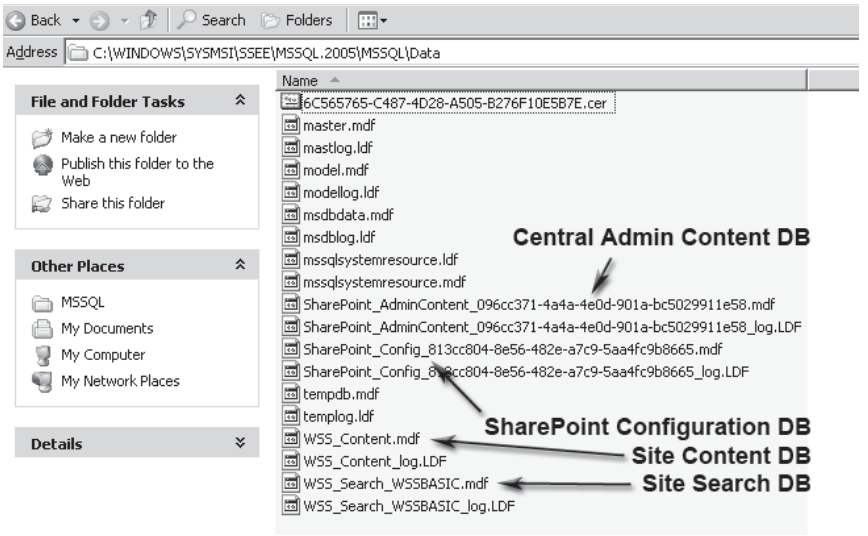
Note that you can navigate away from this page. Since the backup is running as a Timer Job, it will continue to run if you leave the page or close the browser.

SharePoint and SQL Server

Having a backup using SharePoint will cover you for most recovery situations, but I always recommend that you perform a regular SQL Server backup as well. In the event of a failure, particularly a SQL Server failure, having both backups betters the chances of recovering.

For SQL Server 2000 or SQL Server 2005, you should use the database backup utilities provided and set them up on a schedule so that you are always prepared to restore if needed. If you used the Basic installation, the version of SQL installed is the Windows Internal Database, a modified version of SQL Express (limited to 4GB maximum storage). This, unfortunately, does not include any tools, and since it does not have a SQL Agent, it cannot schedule or run jobs for backup (this must be done manually). In addition, the Windows Internal Database instance (under the name of <servername>\MSSQL$MICROSOFT##SSEE) has all network connections closed. This prevents you from using another SQL installation to access it.

There are two ways to back up the databases, and in both cases, you'll need to know the names of them, as well as which ones are auto-generated during the installation; these are MDF and LDF files located in <system drive>\WINDOWS\SYSMSI\SSEE\MSSQL.2005\MSSQL\data. The MDF files will indicate the actual names that were assigned to the databases (the database names include GUIDs that are generated at the time of install):



Backup Method 1 Using the SQLMaint utility, you can easily back up the databases individually or create a batch file to automate it. Run from the command line, the executable is located in <system drive>\WINDOWS\SYSMSI\SSEE\MSSQL.2005\MSSQL\Binn. The basic command to back up a database is:

```
sqlmaint -S <servername>\MICROSOFT##SSEE -D WSS_Content -BkUpDB -BkUpMedia DISK
-BkExt "BAK" -UseDefDir
```

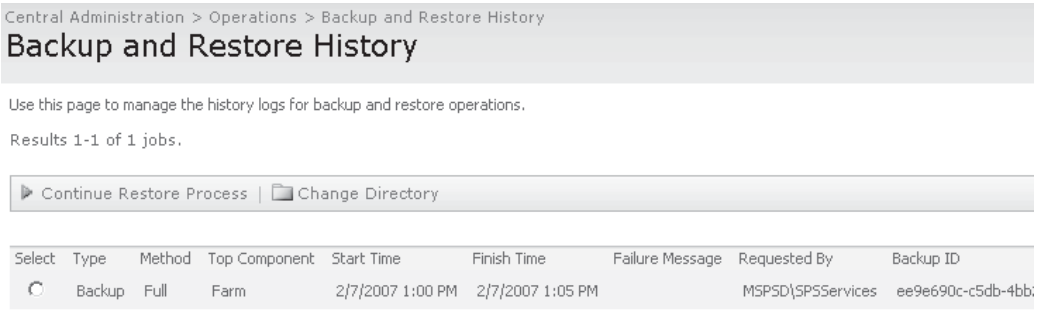
The previous command will back up the WSS_Content database to the Default Backup directory (<system drive>\WINDOWS\SYSMSI\SSEE\MSSQL.2005\MSSQL\Backup\WSS_Content). Note there are several options to this command; entering **sqlmaint** with no options will return the command-line syntax.

Backup Method 2 Shut down all services for SharePoint, SQL Server, and IIS. Once you have them stopped, copy the MDF/LDF files to a backup location and then shutdown and reboot the system.

Also, another alternative available is to download both Microsoft SQL Express and SQL Server Management Studio Express at http://msdn.microsoft.com/vstudio/express/sql/download/ and upgrade the existing installation.

Backup and Restore History

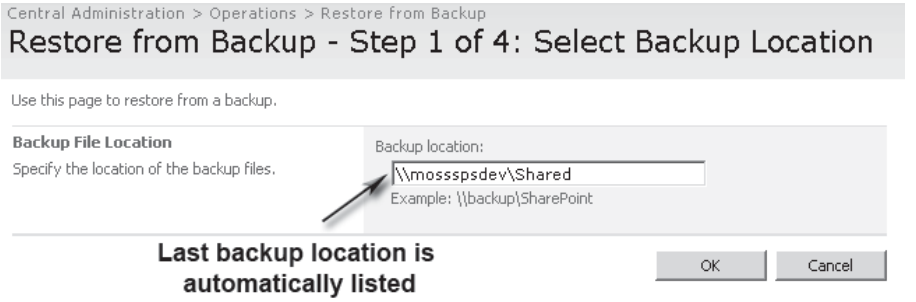
- **When you need it** This option allows you to view the backup histories of sites and site collection backups on the farm.



This option allows you to view the history of backups that have been done; you can also restore backups from this page. If you click a backup instance to select it, you can click the Continue Restore Process link to continue the restore process. If the backup you are looking for is not found, you can also use the Change Directory link to view other backup locations.

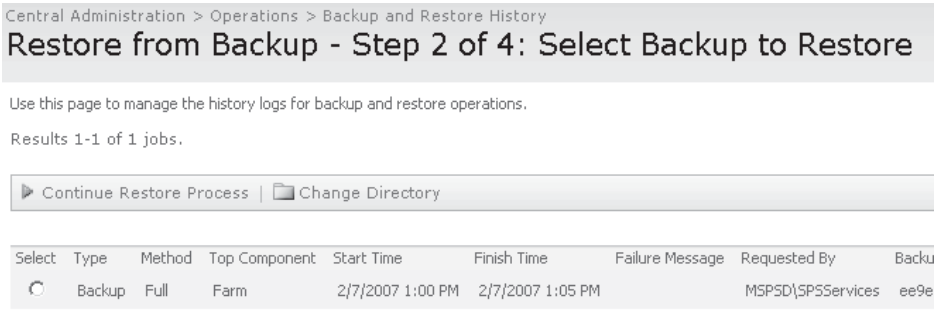
Restore from Backup

- **When you need it** You can use this option to restore a farm or site collection from a SharePoint backup.

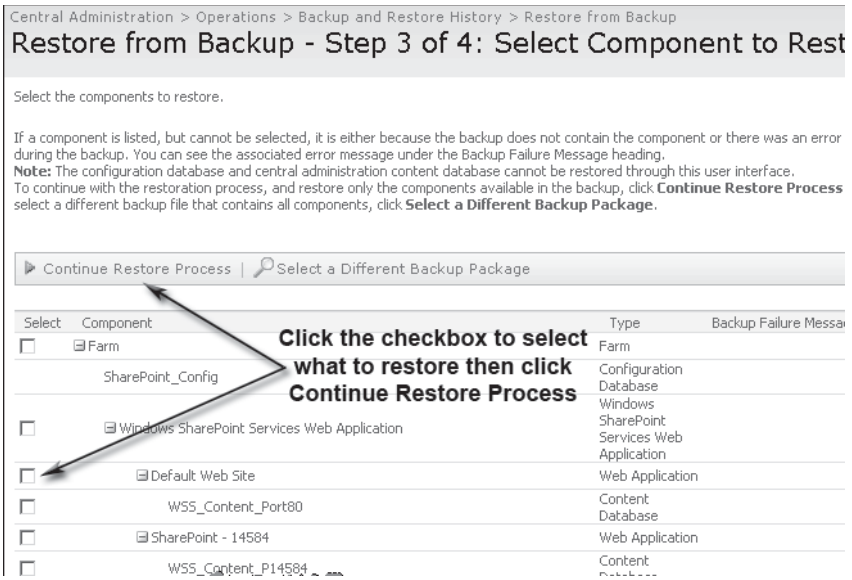


The restore option is a four-step process that allows you to restore a SharePoint Farm or Site to a backup. To restore from a backup:

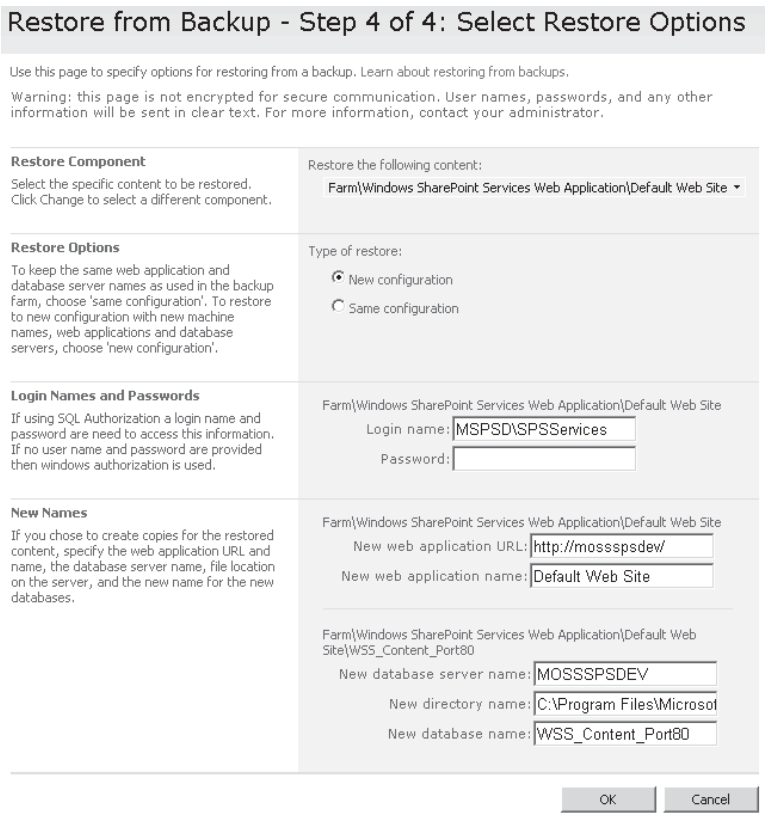
1. Select a location (file share) where backups are located and then click the OK button. This will display the backup history found in that location (this looks the same as the Backup and Restore History view):



2. Click the radio button to select the backup and then click the Continue Restore Process link to continue the restore process. If the backup you are looking for is not found, you can also use the Change Directory link to view other backup locations. When you have selected a backup, the contents of the backup package are displayed:



3. Once you have selected what you wish to restore, click the Continue link to display the destination options of where to restore to:

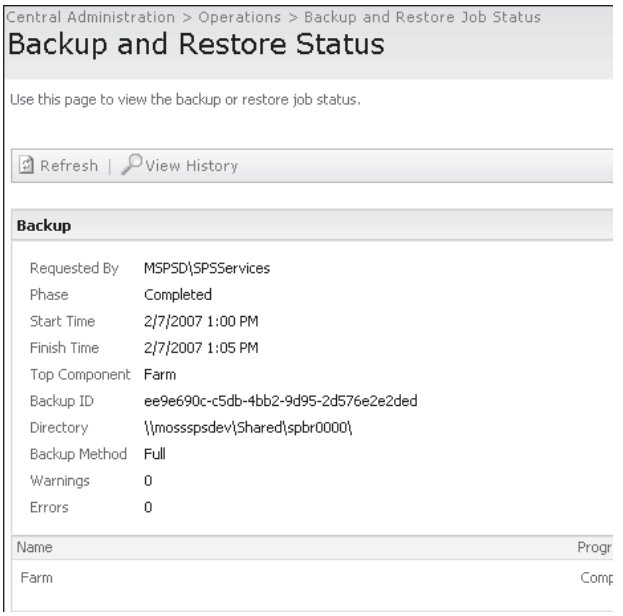


As you can see, you have the option of restoring exactly what was originally saved, or you can change the name and location of where it is restored.

NOTE *Using restore is a method of duplicating a site complete with content and one way to reproduce a portal type site.*

Backup and Restore Job Status

- **When you need it** You can use this option to view the current status of a backup or restore operation.



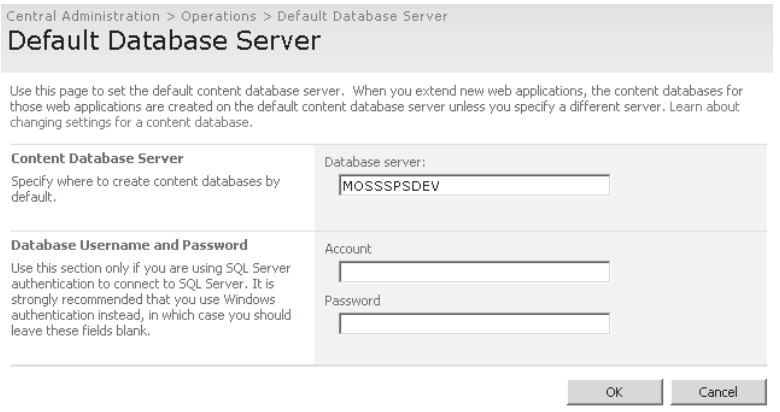
This option allows you to browse the history of backups that have been performed (use the Restore option to restore a backup).

Data Configuration

This section sets the default database server SharePoint will use when creating sites and provides configuration for data access and retrieval services.

Default Database Server

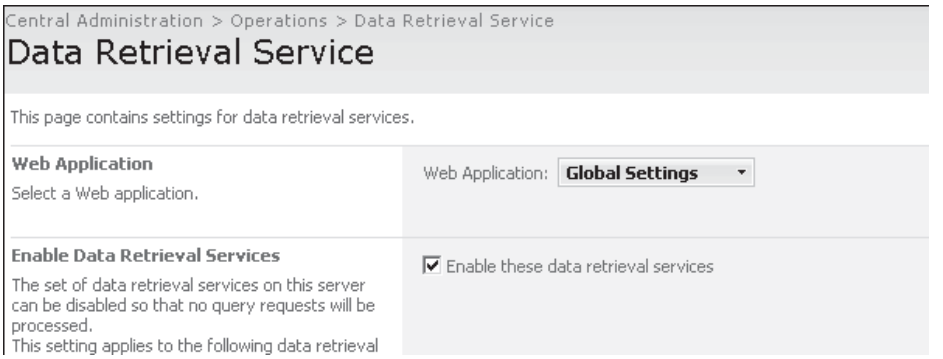
- **When you need it** You can use this option to set the default database server to be used when new sites are created.



This option allows you to set a default database server (and database account, if necessary) to be used when new sites are created (this has no affect on existing sites). This option can also be set using the STSAdm command (see Appendix A).

Data Retrieval Service

- **When you need it** You can use this option to enable or disable database and query access.



This option allows you to set the Data Retrieval Service settings for the farm or an individual application. The choices here are:

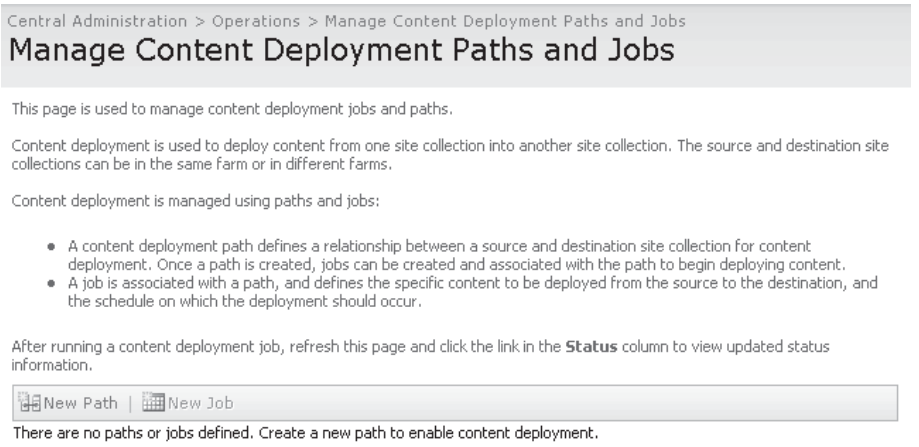
- **Enable Data Retrieval Services** If this is enabled, the farm/application can read data from WSS, OLEDB, SOAP, and XML. Basically, it means that most information in SharePoint can be read by SharePoint. Unless you are in a highly restricted environment, this option should be enabled.
- **Limit Response Size** This sets a limit on the amount of data that can be returned by a SOAP request; the default is 10kb.
- **Update Support** This enables or disables the ability to update data in non-WSS sources and in the case of SQL, determines whether Stored Procedures can be used. The default is disabled, though I suggest you enable this to enable use of stored procedures.
- **Data Source Time-out** This is simply a time-out limit on how long SharePoint will wait for a response from a data source.
- **Enable Data Source Controls** This enables or disables use of internal SharePoint controls that perform query requests (such as the Site Aggregator, for example). The default is to have this enabled. I don't suggest disabling this unless there are some restrictions on data queries in your environment.

Content Deployment (MOSS Only)

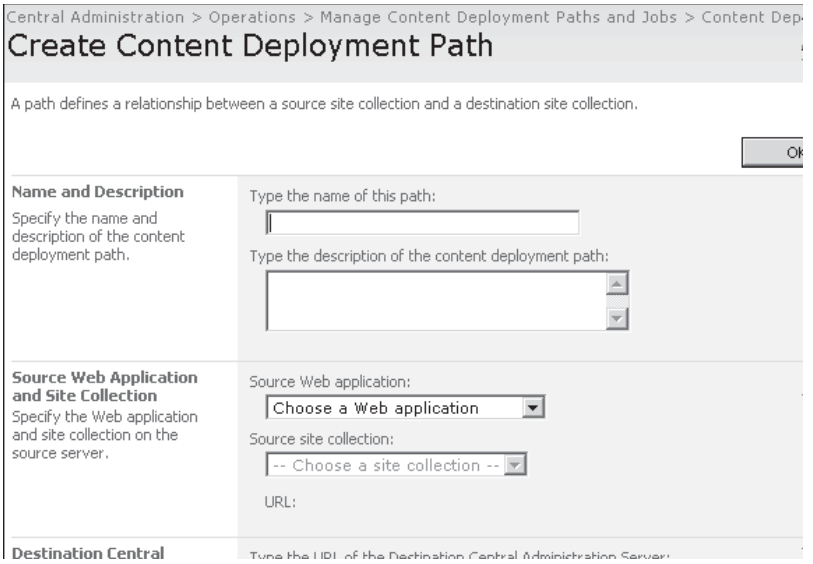
This section provides the settings for preparing publication paths and managing the actual content deployment process between sites.

Content Deployment Paths and Jobs

- **When you need it** You can use this option to add, modify, or delete Content Deployment paths between sites.



To modify or delete a path, you can simply use the mouse over menu on the name. To create a content path, click the Create link:



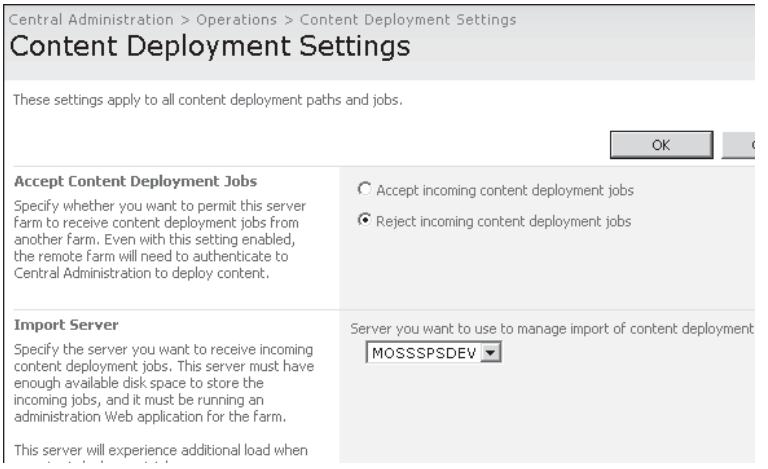
Creating a content path gives you the following options (once created, a job can be associated to it):

- **Name and Description** This is a name and description that you can assign to this deployment path (this should be a reasonable name that will indicate what this path is for.

- **Source Web Application and Site Collection** This allows you to select an application and site as the content source.
- **Destination Central Administration Web Application** This defines the URL to the application/site destination Central Administration Site (it needs this to update).
- **Authentication Information** This allows you to specify the account to use when moving the reading/writing content (note that this account must have access to both source and destination!).
- **Destination Web Application and Site Collection** This allows you to select the destination application and site to receive the content.
- **Usernames** This enables or disables including usernames (those that are associated with the original) with the content. This should be disabled if you are deploying to a site that has a different user community.
- **Security Information** This allows you to select the permissions that are deployed with the content. The options are All, Roles, or None.

Content Deployment Settings

- **When you need it** You can use this option to enable/disable as well as change storage locations for Content Deployment jobs.



This option allows you to enable or disable content deployment to this farm. Content Deployment jobs come from many sources, and these are invoked either programmatically or from the command-line tool, STSAdm. Further options include:

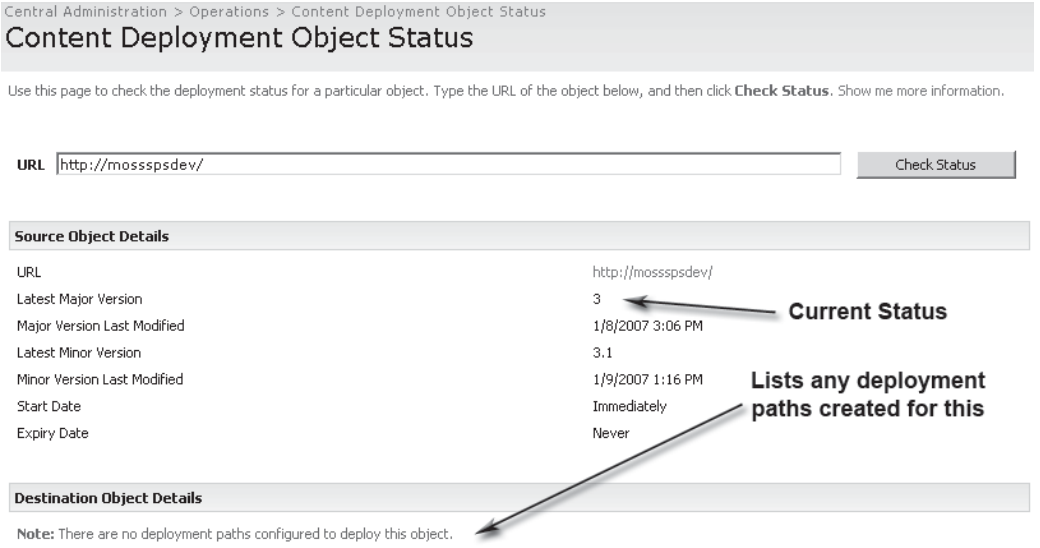
- **Accept or Reject requests** This turns inbound content deployment on or off. Note that Reject is the default.
- **Import Server** This specifies the server to use for incoming Content Deployment jobs. Be aware that the server will take a performance hit when content jobs are running and must have enough disk space to accommodate a copy of inbound content.
- **Export Server** This specifies the server to use for outgoing content deployment.

- **Connection Security** This enables or disables and requires encryption for the content deployment connection.
- **Temporary Files** This allows you to specify the file location to use when processing inbound or outbound content. This, by default, will be the system Windows Temporary directory. I highly advise you change this to be a non-system drive that has plenty of space.
- **Reporting** This sets the number of reports you want to save on a Content Deployment job.

Note that when you click OK to save your settings (or Cancel), you are directed to the Central Administration Site Administration page. (I suspect this is a misdirected link!)

Check Deployment of Specific Content

- **When you need it** You can use this option to check the status of any site item to verify last deployment.



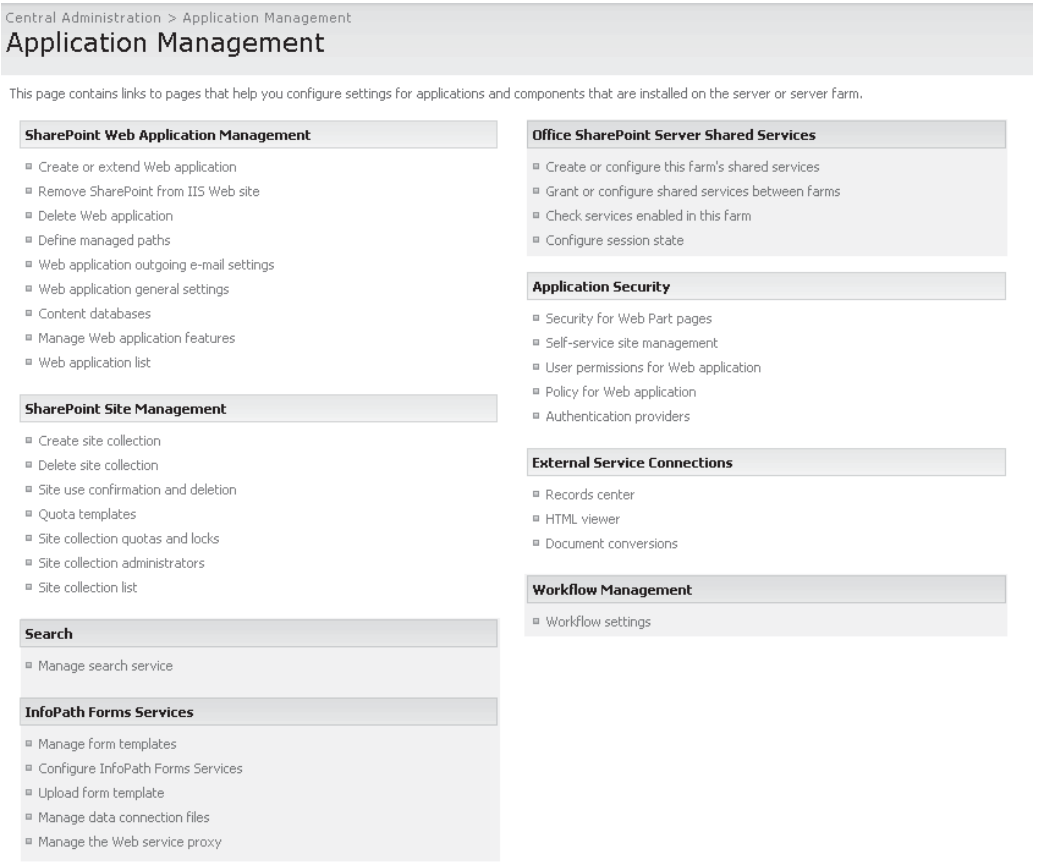
This option allows you to check the status of any site or content to verify when it was last deployed and if any deployment paths have been created for it.

Application Management

The Applications Management side of Central Administration allows you to manage applications and sites within SharePoint installation. From here, you can create or remove applications and sites, modify the search service, access shared services settings (MOSS), set application-level security settings (including authentication providers), manage external service connections, and manage workflow settings for applications.

To open Application Management, open the Central Administration via Start | All Programs | Administrative Tools | SharePoint 3.0 Central Administration. From the CA home page, click the Application Management tab or link to display the Application

Management page. These instructions are the same for both Windows SharePoint Services 3.0 and Microsoft Office SharePoint Server, though MOSS has additional sections as highlighted:



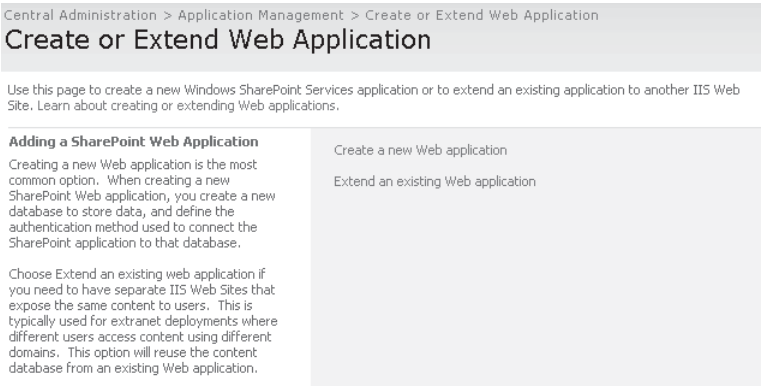
Next, I'll cover each Application Management option, including why and when (if applicable) it is used; all of these options are opened from the Applications Management page shown previously.

SharePoint Web Application Management

This section provides all of the management tools needed for handling applications and top-level Site Collections.

Create or Extend Web Application

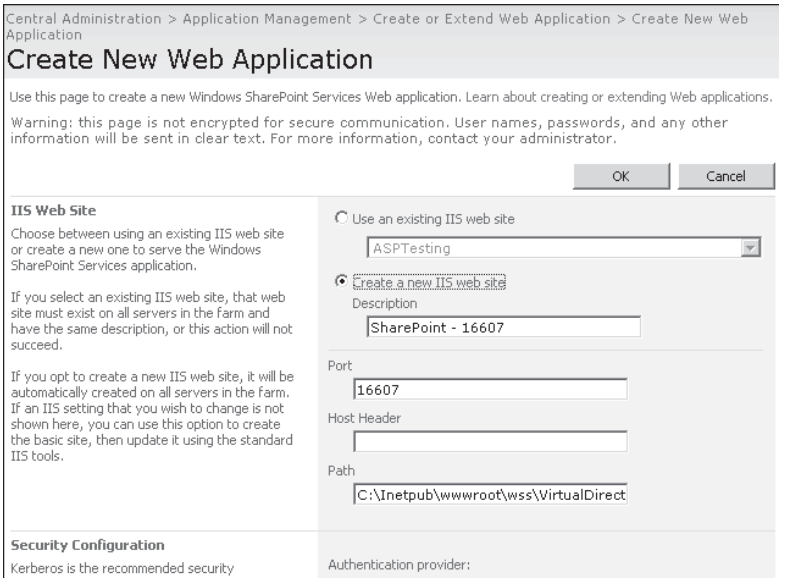
- **When you need it** You can use this option to create a new application or extend an existing one.



This option allows you to create a new application or SharePoint-enable an existing IIS application (virtual directory). Creating the first web application is covered in Chapter 6. If you used the Basic installation method, an application and top-level Site Collection has already been created for you. If you've followed the configuration steps, this has already been done as well. This option is for additional applications created.

To create a new application:

1. Click the Create a New Web Application link. This will now open the Create Application page:



2. When the Create page opens, you can specify the following:

   - **IIS Web Site** This defines the name and the port number to be assigned to the application. SharePoint will pick a random (unused) port number or port 80 if it is available (true when creating the first site). You do have the option of selecting an existing IIS Virtual Directory if one has been created and does not already

have SharePoint on it. You can also specify the Host Header (a different name to use in the URL), and you can specify the physical file path to be used to hold the application (by default, this is under <system drive>\inetpub\wwwroot\wss\virtualdirectories path; since virtual directories contain very little content, you can accept the default).

- **Security Configuration** This allows you to define the following:
  - **Authentication Provider** This is set to either Negotiate (Kerberos) or NTLM (Windows Integrated). The preferred method is NTLM, because if you choose Kerberos, you will have to make some adjustments to the servers to accommodate this. Specifically, you'll need to create network system names for servers to avoid the "double-hop" problem of losing authentication between servers. Review the Microsoft Documentation on how to use Kerberos to be sure you do this correctly.
  - **Allow Anonymous** This enables/disables anonymous access to the application sites.
  - **Use Secured Sockets Layer (SSL)** This turns SSL on or off on application sites.
  - **Load-Balanced URL** This specifies the network, load-balanced URL and Internet zone.
  - **Application Pool** You can use an existing application pool or have one created specifically for this application (recommended), and you can also designate the account to run the application pool. This should be the designated SharePoint Application Pool account (SPAppPool), and that account must have Logon as a service right.
- **Reset Internet Information Services** This either restarts IIS automatically (an IISReset) after the application is created or manually. Note that if you do select Automatic, you are returned to an Error page when the application is created. This is because the IISReset is running when the page tries to open. Simply use the Go Back to Site link to return to Central Administration.
- **Database Name and Authentication** This allows you to specify the database server, the name of the content database that will be used for this application, and the way to authenticate (using Windows or a SQL account). Note that the name of the database is set to WSS_Content or WSS_Content_<Application GUID>. I suggest you change this to a more easily identifiable name, for example, WSS_Content_Site1234.
- **Search Server** This allows you to specify a search server. If only one search server is available, there will be no options enabled here.

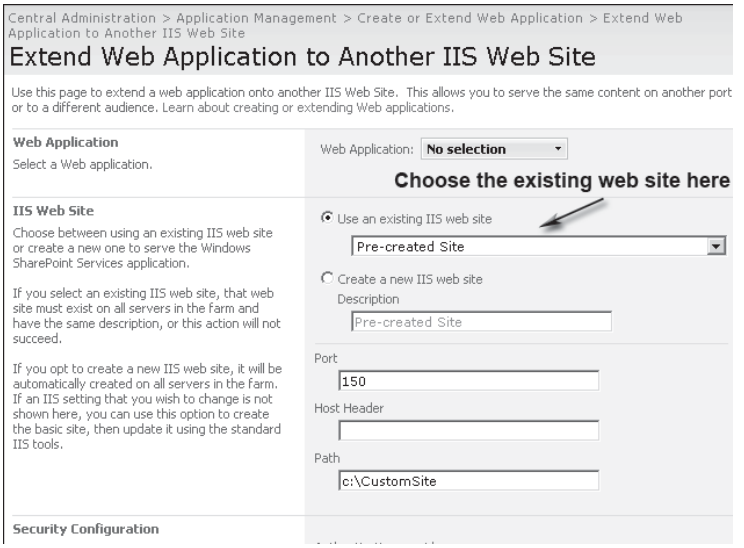
NOTE *Remember that application settings apply to all sites created under it.*

3. Click the OK button to create the application; the ticker page will be shown while it works. What is happening behind the scenes is this:
   - A new (physical) folder will be created to hold the application.
   - A new application pool will be created using the name and account specified.
   - The new folder will be provisioned as a web within IIS (and assigned the application pool created previously).

- The Virtual Directory will have a filter added as a SharePoint application and a Web.config file will be added to the application directory.
- If set, an IISReset will be performed once created.

To extend an existing application for SharePoint requires that an application (website) already be set up in IIS and that you already have a SharePoint Application you want to attach it to. To create a new website in IIS and then to extend it, do the following:

1. From the desktop, select Start | All Programs | Administrative Tools | Internet Information Services (IIS) Manager.
2. Click + next to the server name to expand the tree if necessary.
3. Right-click on the Web Sites folder and select New | Web Site.
4. When the wizard opens, click Next.
5. Enter a description for the site, and then click Next.
6. Select the IP Address and the Port number to use. You can also specify a Host Header here; click Next to continue.
7. Specify the physical folder to use to hold the website. You can use the Browse button to navigate to a location and create one if needed, or you can simply create a folder using Windows Explorer and then enter that here. You can also set the Anonymous Access settings for this site; if you intend to provision it for SharePoint, this setting will be overridden. Click Next to continue.
8. On the next page, you specify the access. By default, only Read is selected, but for a SharePoint site, you should enable Run Scripts and Execute as well. Click Next and then Finish to create the website.
9. Return to SharePoint 3.0 Central Administration | Application Management | Create or Extend Web Application. From the Create or Extend page, click Extend an Existing Application; this will open the Extend Application page (you'll notice it looks pretty much the same as the Create page):

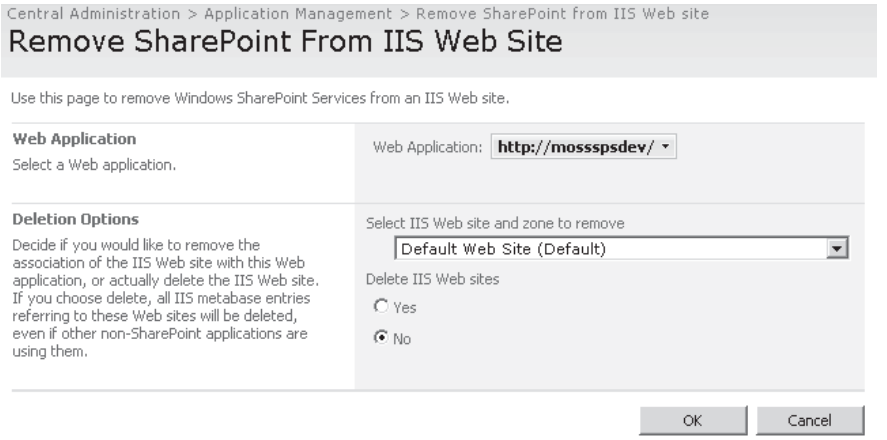


NOTE *You must run an IISReset before the application will appear in the drop-down list.*

10. Select the web application to use in the drop-down list (when first opened, this is set to No Selection).
11. Select the radio button to Use an Existing IIS Web Site, and then from the drop-down list, select the web you just created (in this example, I've used port 150. When the website is selected, it will fill in this information as shown).
12. You can then specify the Security Settings and Load Balanced URL. The other settings for the application pool are already set by the application you selected.

Remove SharePoint from IIS Web Site

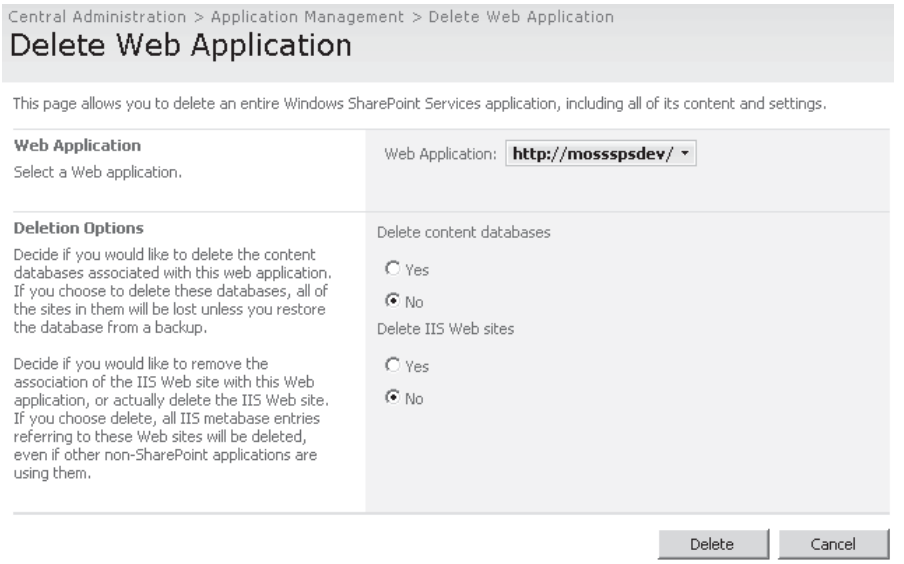
- **When you need it** You can use this option to remove SharePoint Extensions from an IIS website.



This option allows you to remove SharePoint from an IIS website and optionally remove any websites underneath it. While you won't use this option much, it basically enables you to de-provision a website so that it can be used for other purposes. For example, if you had already provisioned the default website (port 80) and need to restore it for other uses, you can de-provision it here without removing the site itself.

Delete Web Application

- **When you need it** You can use this option to delete an existing SharePoint application.



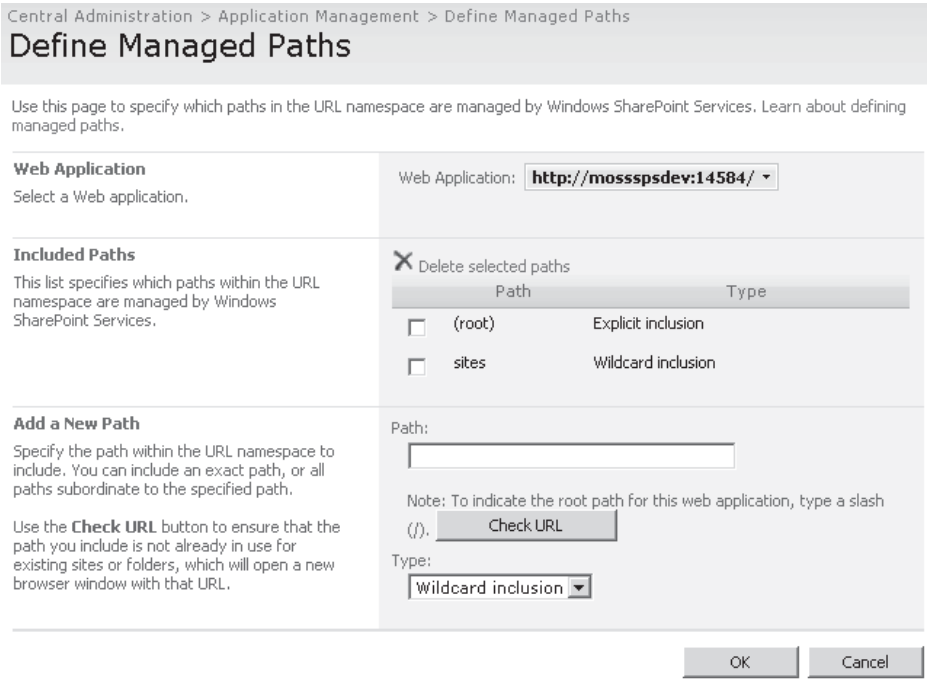
This option enables you to completely remove a SharePoint application from IIS. You also have two deletion options here:

- **Delete Content Databases** Selecting Yes will permanently delete all content databases associated with this application. Selecting No will still de-provision the site, but the content databases will remain intact (these can associated to other sites).
- **Delete IIS Websites** Selecting Yes will delete any websites that are within the application; No will leave them intact but they will not be SharePoint-enabled.

CAUTION *Make sure that you have the correct application selected before you confirm deletion!*

Define Managed Paths

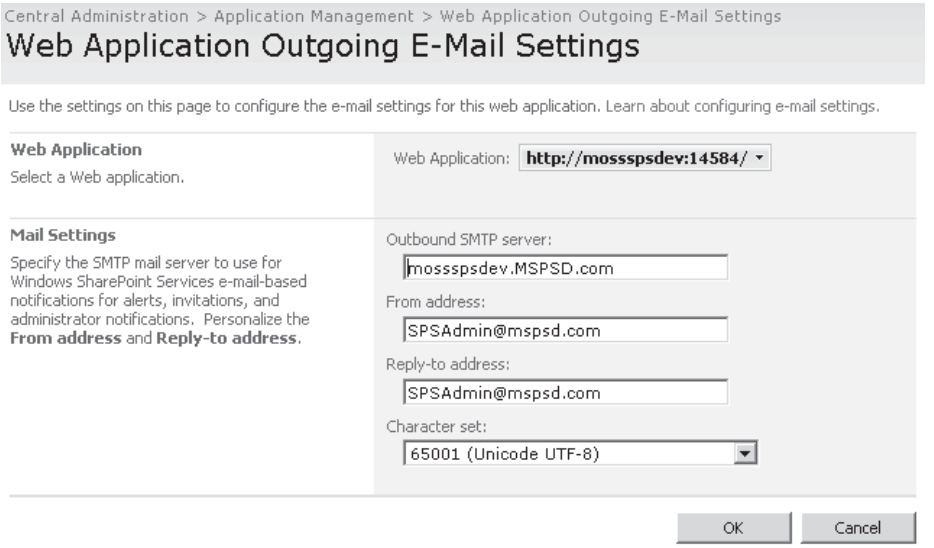
- **When you need it** You can include or exclude folders within the site from SharePoint control.



This option allows you to either include or exclude folders from being controlled by SharePoint. The primary purpose for this is to allow other applications to run under SharePoint and have requests for them ignored. As an example, if you install SQL Reports on the same application as SharePoint, two directories are created: Reports and ReportServer. If not excluded, any request to a report would be picked up by SharePoint instead of SQL Reports and the request would be denied. The same situation is true for running a .NET web application under SharePoint.

Web Application Outgoing E-mail Settings

- **When you need it** You can use this option to set or modify the SMTP server settings for the application.

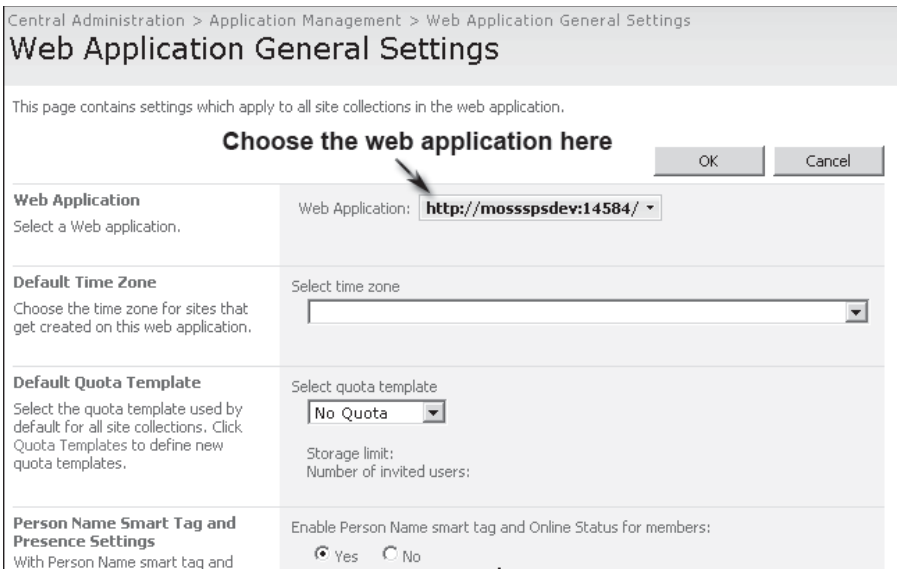


This option (usually set only once per application) allows you to set or modify the outbound e-mail server settings for a specific application. This is the mail server that will send any outbound e-mail from the application (alerts and so on). From here you specify:

- **Outbound SMTP Server** This is the server the application will use to send out e-mails from SharePoint. This is either a designated SMTP server or the front-end web server SMTP service (this must be installed with IIS).
- **From Address** This is the from address that will appear in e-mails.
- **Reply-to Address** This is the reply to address that will appear in e-mails.
- **Character Set** This sets the character set for e-mails (UTF-8 is the default, meaning allow full HTML and special characters).

Web Application General Settings

- **When you need it** This option is used to set or modify the general application settings (time zone and so on).



This option provides access to general settings for an application (set on an application-by-application basis). From here, you can set all of the settings that apply to the application as a whole (and by extension, every site underneath it).

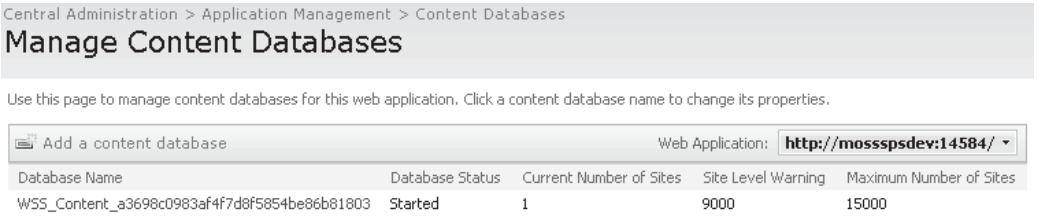
- **Web Application** This provides a drop-down list that allows you to switch between applications.
- **Default Time Zone** This is the default time zone for the application and is helpful to ensure correct timestamps on content.
- **Default Quota Template** This selects a quota (maximum) template for this application.
- **Person Name Smart Tag and Presence Settings** If enabled, User Presence is active in SharePoint. When the mouse is moved over a username, the user profile and or IM presence is shown and it will indicate whether the user is currently logged in.
- **Maximum Upload Size** This sets the maximum file size that can be uploaded.
- **Alerts** This enables or disables alerts on the site (if disabled, Alert options are not shown in the User Interface).
- **RSS Settings** This enables or disables Really Simple Syndication options for this Application.
- **Web Page Security Validation** This enables or disables page time-out (or how long a user can be inactive before having to log back in).

- **Send User Name and Password in E-mail** This is used when users are created by SharePoint; if disabled, users cannot access the site unless they already know the password.
- **Backward-Compatible Event Handlers** This enables or disables use of 2003-style event handlers (installed "per library"). This is disabled by default.
- **Change Log** This specifies how long changes logged on site actions are stored; change logs enable you to compare versions.
- **Recycle Bin** This sets the options for the recycle bin: on or off, how long to hold, and so on.

REMEMBER *These settings apply to the specific application (and its sites) and are not global.*

Content Databases

- **When you need it** You can use this to add or remove content databases from an application.

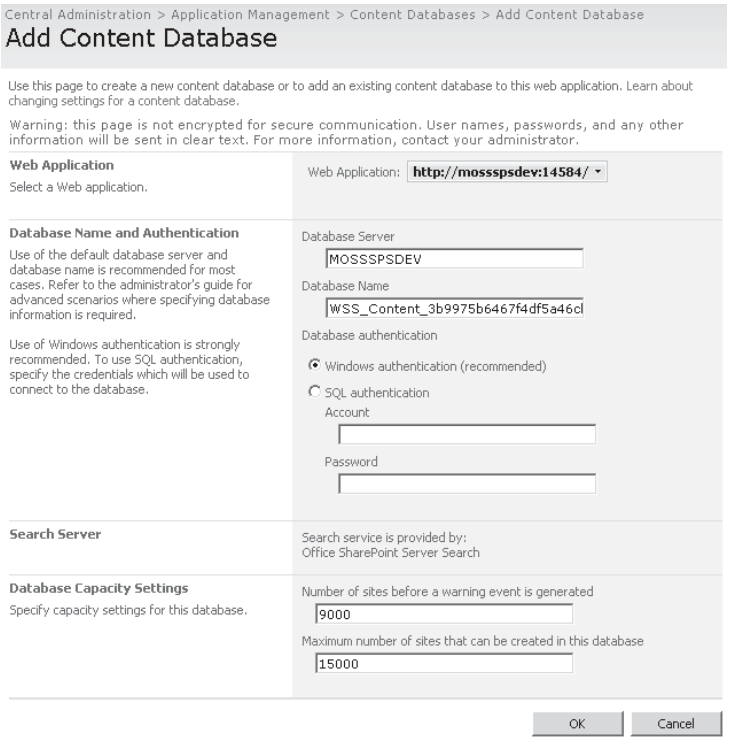


This option provides a list of all content databases in use by the selected application. As you can see next, there are a series of things that are included for each database:

- **Database Name** This is the name assigned to the Application database when it was created or to a custom database name as created by hand.
- **Database Status** This indicates whether the database is available or not.
- **Current Number of Sites** This indicates the current number of sites deployed in the database.
- **Site Level Warning** This is a set number, after which a warning is sent to the administrator that a set site limit is/will be exceeded. The default is 9000 independent sites; most of you will never come close to this count, but it is an acceptable level from a backup/restore perspective and a good point to consider when adding additional content databases.
- **Maximum Number of Sites** This is the total number of sites that can be created in this content database. The default for this is 15,000 sites—again, something most will never hit. However, this limit, if hit, will prevent additional sites from being created.

Creating a new content database allows you to expand the capacity of an existing application. For example, if you were to exceed 9000 sites in one database, you might opt

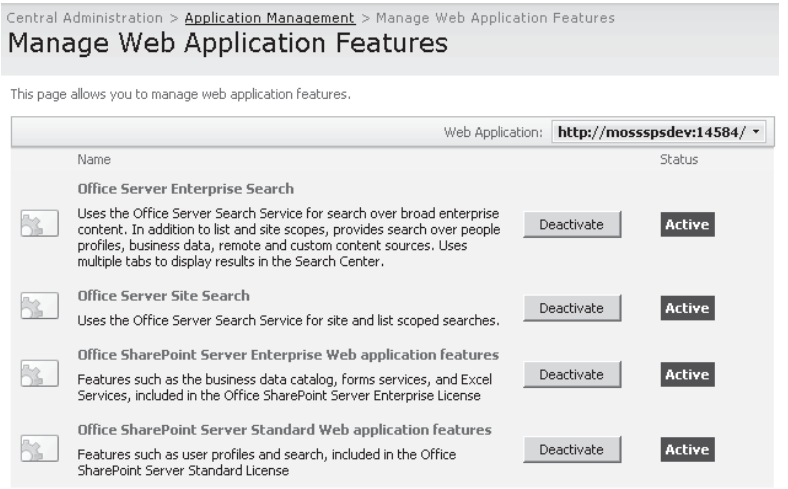
to add another one, giving you another 9000 sites to work with. The database create process is handled by SharePoint automatically. To create one, click the Add a Content Database link to open the Add Content Database page:



NOTE *In addition to the previous settings, you can also select a search server if more than one is installed. Otherwise, it simply displays the service it will use.*

Manage Web Application Features

- **When you need it** You can use this option to activate or deactivate application-wide features.

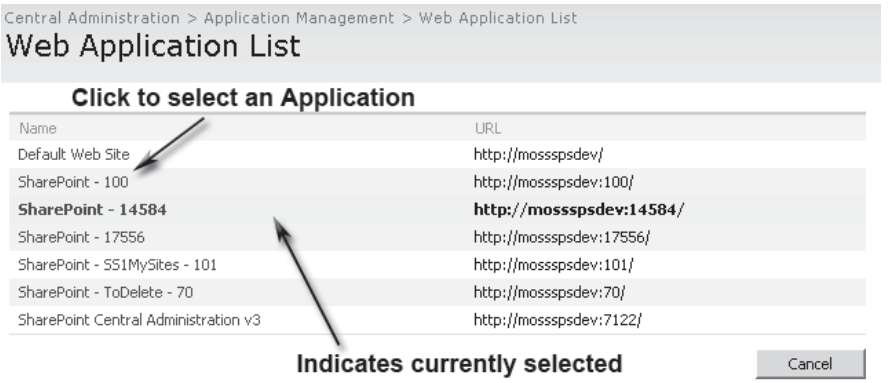


This option allows you activate or deactivate application-level features already installed. If disabled, the specified feature is not available in sites in that application, whereas if enabled, all sites in the application can use the feature.

NOTE *Disabling at the Site level overrides the settings here.*

Web Application List

- **When you need it** You can use this option to select or switch the current application.



While you can usually select the application you are working with on individual pages, you can also use this option to set the current application so that you do not have to specify

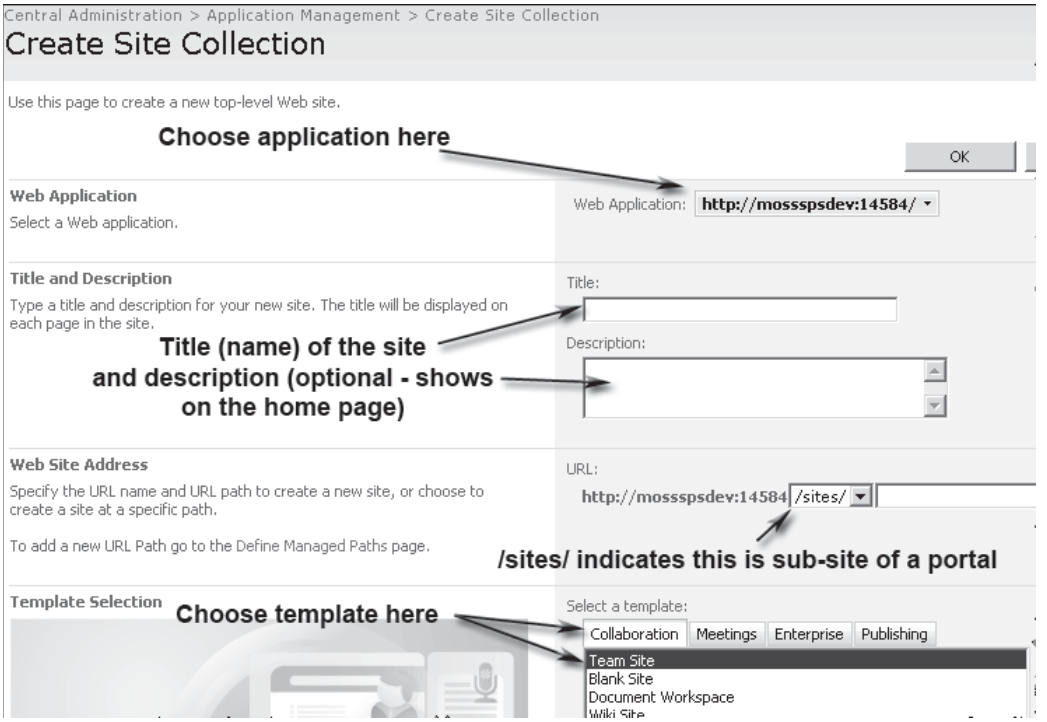
it in each page. Simply click an application name to make it current. Clicking Cancel will leave the current application selected.

SharePoint Site Management

This section covers all of the options for Site Collection Management for the selected application.

Create Site Collection

- **When you need it** You can use this option to create a new site collection in the current application.



This option enables you to create a new site collection (top-level site) within the current application. From here, you can choose the following:

- **Web Application** This allows you to choose the web application you will be creating the site under.
- **Title** This is the title (name) of the site as it will appear on the site pages.
- **Description** This is the description (optional) of the site and what it is for/used by. This information appears on the home page of the site.
- **Web Site Address** This is either a root (/) when it's the first site in the application or a member of a portal (indicated by /sites/) and the URL to use for the site. Remember to keep site URLs short and do not use special characters. For example, a "Process Excellence" site might use a URL of simply **pe**. Be aware the URL name must be unique in the portal. You can also select another URL path (this requires

that you add that path to SharePoint Managed Paths). See the previous information or click the link shown.

- **Template Selections** This provides the various "out of the box" and installed site templates available (Collaboration/Meetings are WSS site types and Enterprise are MOSS site types). Publishing sites are MOSS Portal site types. Refer to the SharePoint Site Definitions chapter for site types and for what each is used for.
- **Primary Site Collection Administrator** This is the login for the user that will be the primary administrator/contact for the site.
- **Secondary Site Collection Administrator** This is the login for the user that will be the secondary (backup) administrator/contact for the site.
- **Quota Template** This sets a quota for this site (see "Site Collection Quotas and Locks" covered later in this chapter).

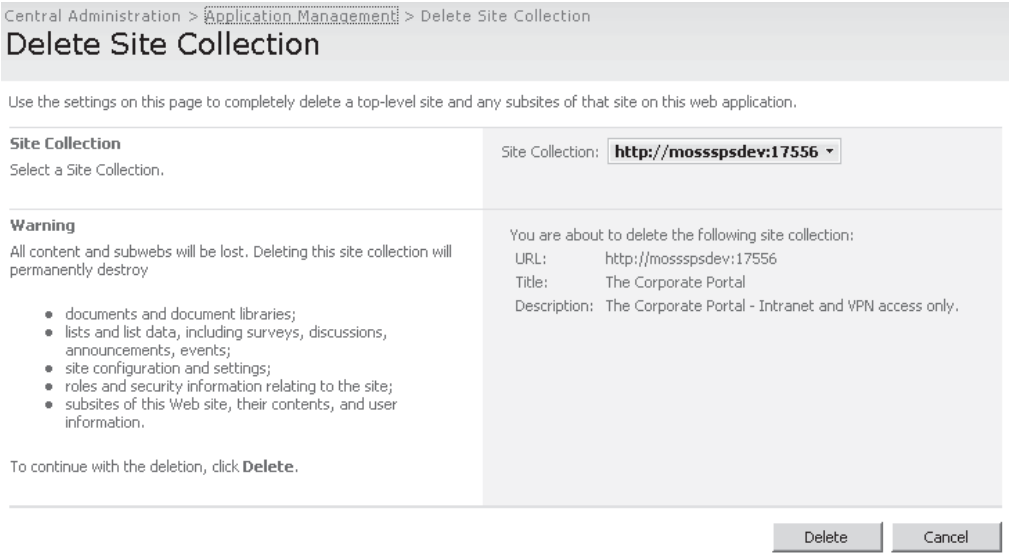
Once you have made your selections, click the OK button to create the website. A few things will be validated (like the URL and the user names), and assuming everything is OK, go ahead and create the site. In the background, SharePoint will

- Obtain the site definition (the combination of ASPX pages, ONET.xml file, and definitions in the database).
- Provision the new site under the application.
- Extend the site for SharePoint.
- Save the site definition back to the database.

Once the site has been created, you can navigate to it. Be aware that only the Primary and Secondary Site Collection Administrators you specified are the defined users for that site.

Delete Site Collection

- **When you need it** You can use this option to select or switch the current application.

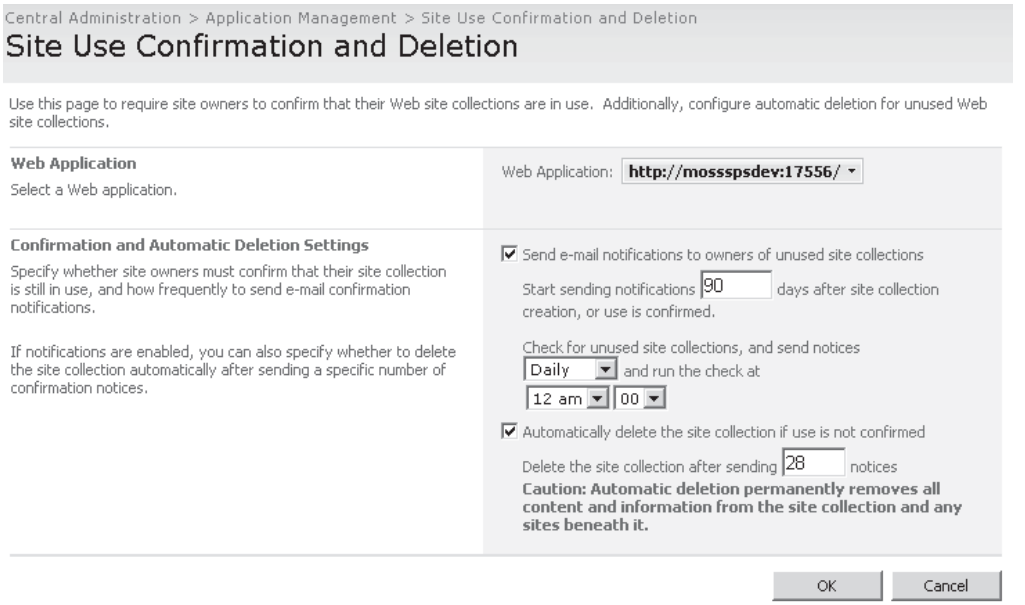


This option allows you to delete an entire site collection from an application. When it first opens, you must select which site collection to delete. To do this, click on the site collection drop-down list and then select Change Site Collection.

CAUTION *Deleting a site collection is permanent; while it will remain in the database for a short time, there is no Recycle bin for sites (yet), so you must rely on backups.*

Site Use Confirmation and Deletion

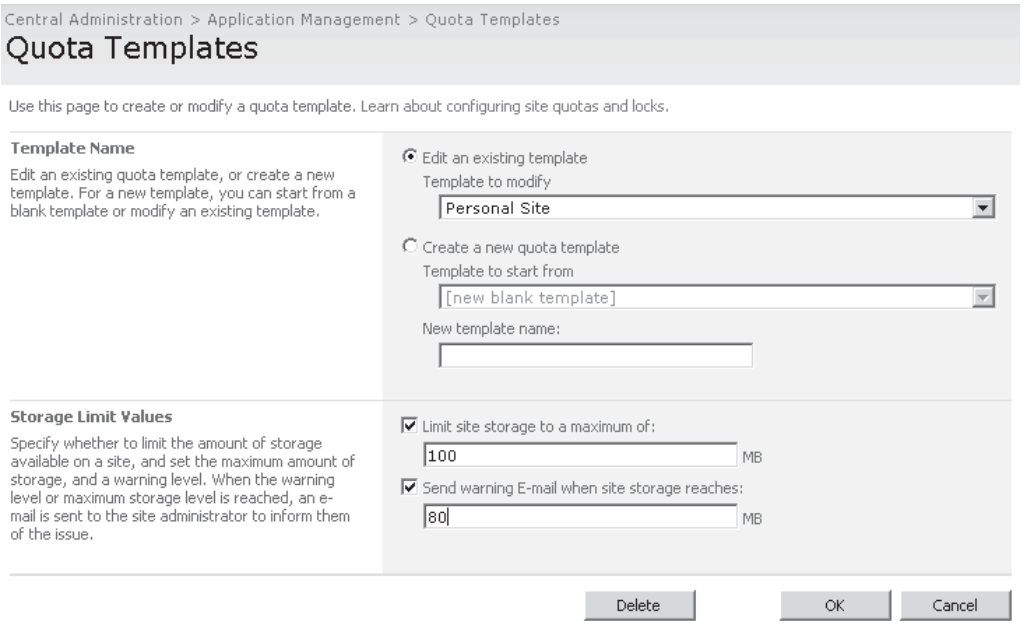
- **When you need it** You can use this option to enable or disable unused site notifications and automatic deletion.



This option either enables or disables notifications and automatic deletions of sites that go unused for the time you specify. While this is disabled by default, I recommend you at least set the notifications, as dead sites create useless overhead and can end up confusing users. I do not suggest using the automatic delete unless you are very vigilant about retaining backups. This is because often an unused site may indicate to a viewer that the site owner is no longer active.

Quota Templates

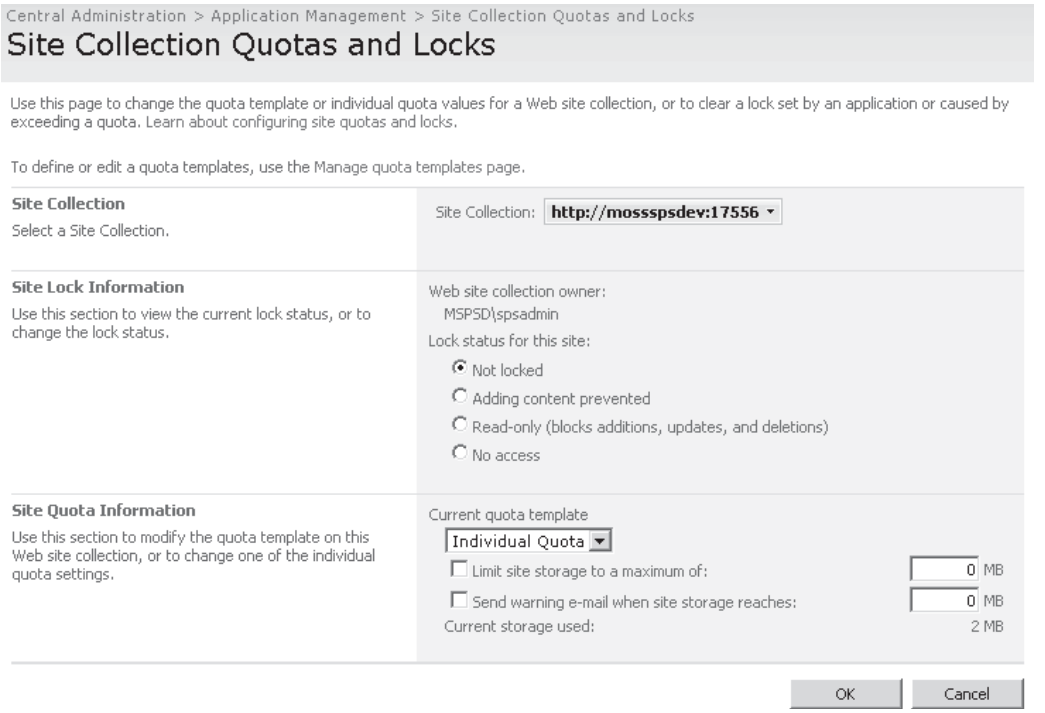
- **When you need it** You can use this option to create, modify, or delete Quota templates.



Quota templates are general "maximum" settings for a particular entity in SharePoint (a site, a My Site, and so on), which provide you with a way to control how much storage they can take up (the application owner is notified by e-mail if getting near the maximum). In MOSS, a default template is created for Personal Sites, in order to limit individual users to 2GB (though this can be changed). Quotas are an excellent way to set a level of service (SLA) for sites for restoration and backup purposes. For example, it is much faster to restore a site that maintains less than 2GB versus one that is 20GB.

Site Collection Quotas and Locks

- **When you need it** You can use this option to create, modify, or delete Site Quotas and Storage Limits on sites.



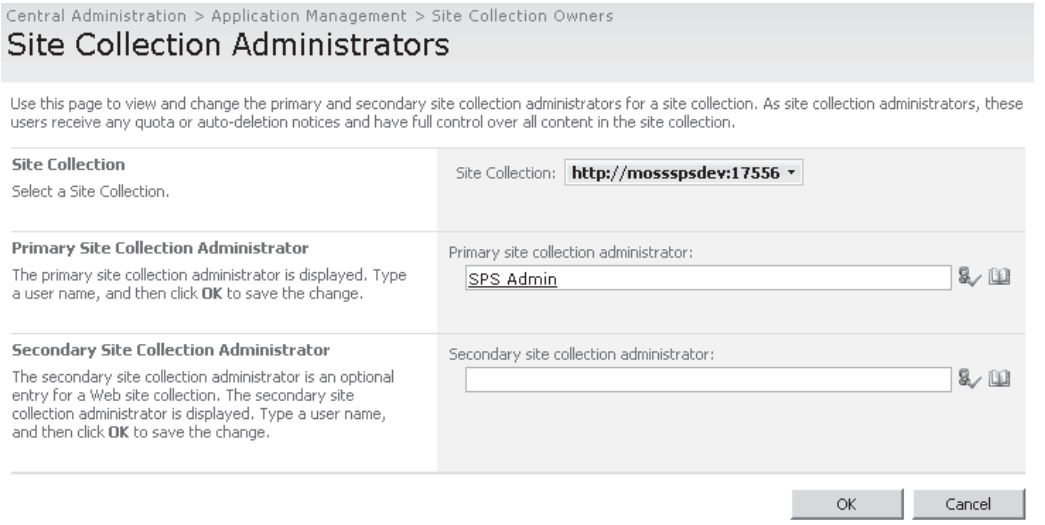
This option allows you manually (one-off) set or apply the Quota templates to a specific Site Collection. The options here include:

- **Site Collection** This is a drop-down list that allows you to select a specific Site Collection.
- **Site Lock Information** Not Locked means normal access. Adding Content Prevented means that the site can be viewed, but no items can be added. Read-only makes the site accessible, but nothing can be changed. No Access locks out all user access except for the Administrator.
- **Site Quota Information** This allows you to select a quota template or set an arbitrary quota for the site.

Site Collection Administrators

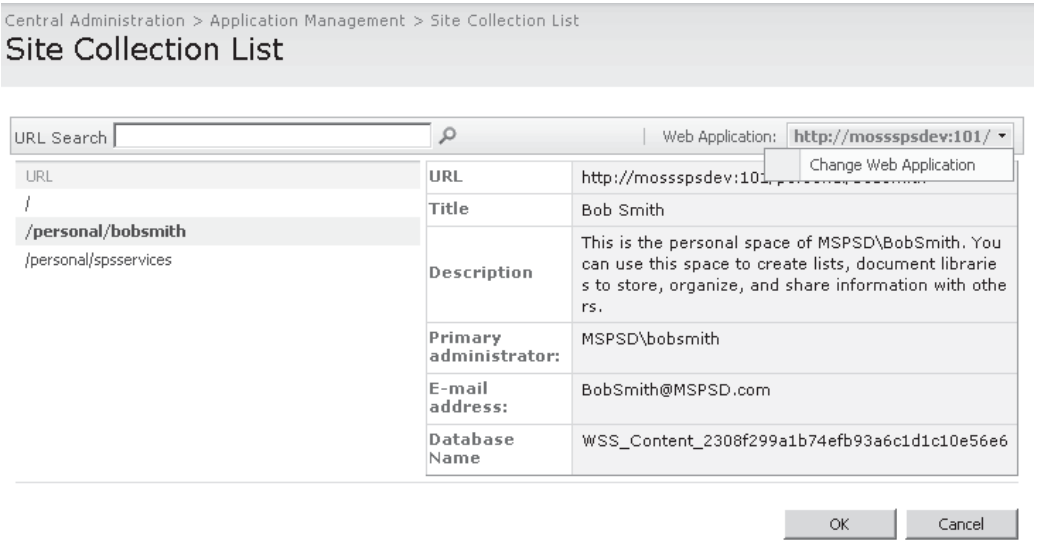
- **When you need it** You can use this option to add or remove Site Collection Administrators from site collections.

This option allows you to change the Primary and Secondary Site Collection Administrators on any Site Collection for the current application selected:



Site Collection List

- **When you need it** You can use this option to view details about Site Collection(s) within an application.



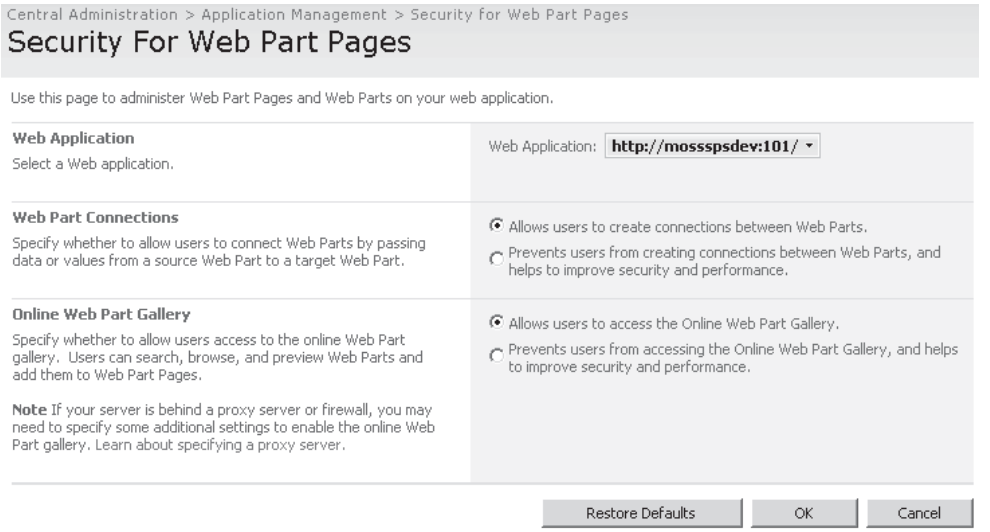
This option allows you to see all of the websites that are defined for a selected application. If you click on any one of the site links (in the left-hand side of the list), you'll see that the information about the site including the title, description, and so on, are shown.

Application Security

This section handles the overall security for a selected application.

Security for Web Part Pages

- **When you need it** You can use this option to enable or disable Web Part Connections and access to the Web Part Gallery for an application.

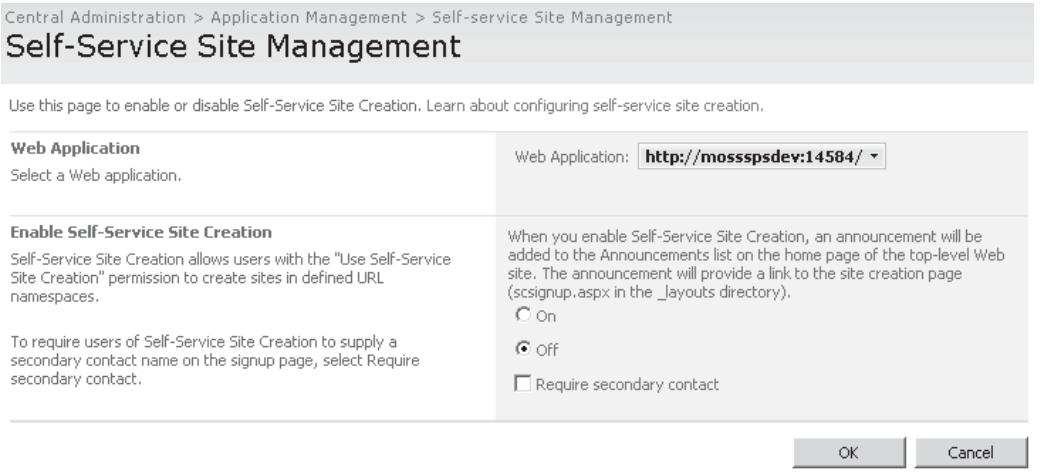


This option enables the security settings for web part pages for all sites within an application (Microsoft recommends these be disabled but that is impractical in Collaboration/Collaboration Portal sites). The options are listed next:

- **Web Application** This uses a drop-down list to change applications.
- **Web Part connections** This enables or disables the user's ability to connect a web part to another web part (commonly done with filtering web parts).
- **Online Web Part Library** This enables or disables the access that end-users have to the Web Part Online Gallery. If disabled, users cannot see web parts to add (this does not affect lists and other types of content).

Self-Service Site Management

- **When you need it** You can use this option to enable or disable user self-service site creation.

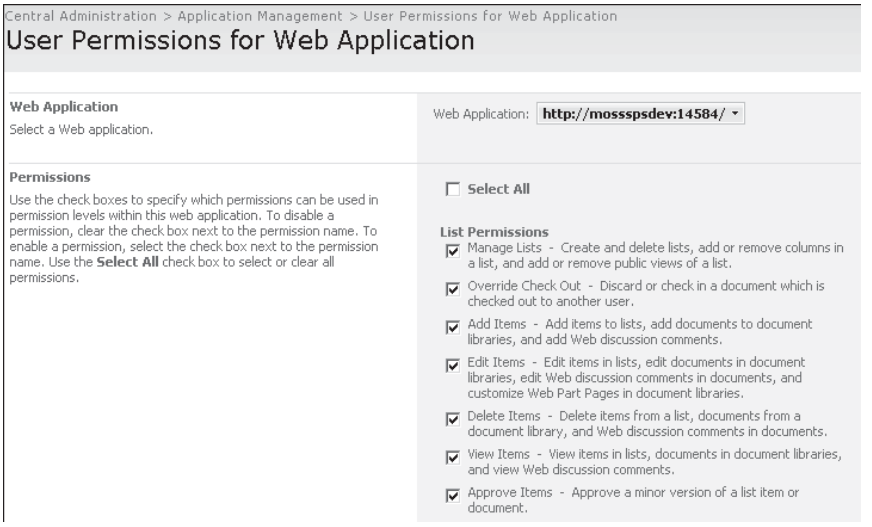


This option enables or disables the user's ability to create their own sites (under designated URLs). If enabled, it requires a secondary contact, meaning that at least two Administrators, both Primary and Secondary, are required before a site can be created. Enabling this option will create a new Announcement (on the top-level site home page list if one exists), which will then provide users with a link to create new sites. Other than a secondary contact restriction, there is no approval required for site creation.

CAUTION *If enabled, you are essentially making site creation a free-for-all; if the users are disciplined enough to handle this, then by all means turn it on. In a corporate intranet or public Internet, however, I highly recommend you enforce a governance model and disable this.*

User Permissions for Web Application

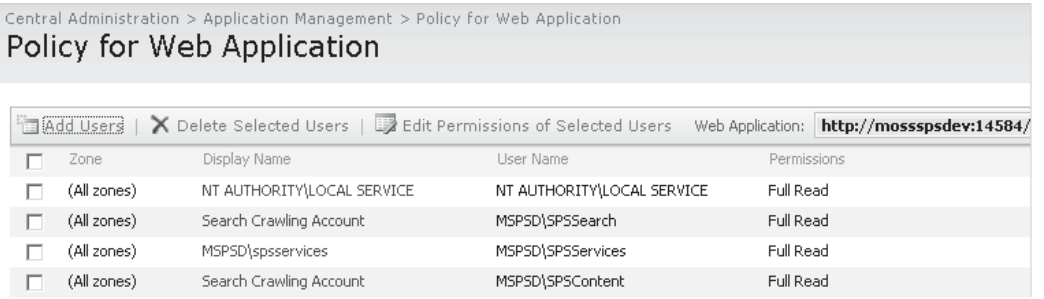
- **When you need it** You can use this option to set or view user permissions within an application.



This extensive list of permissions enables you to enable (checked) or disable (unchecked) the various SharePoint activities users are permitted within the sites of the application. The granularity is set up to enable you to finely tune access to your governance model. By default, all users are enabled—if you make any changes, simply click the OK button to save them. Remember these settings apply to all users in all sites under the application selected.

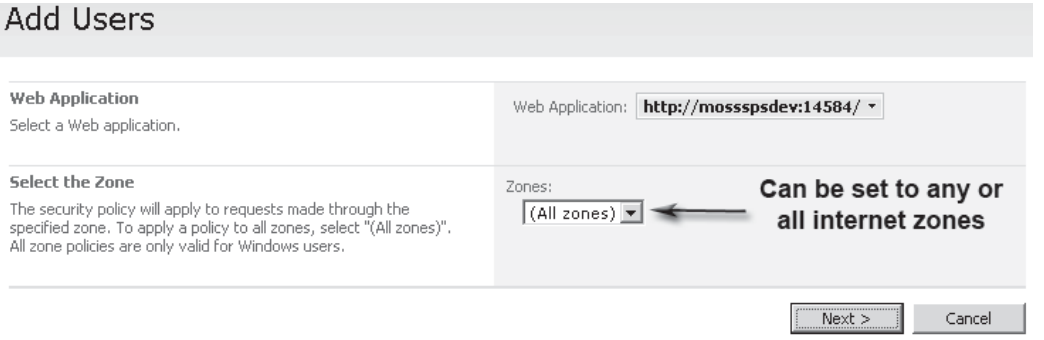
Policy for Web Application

- **When you need it** You can use this option to set the user policy for web applications.

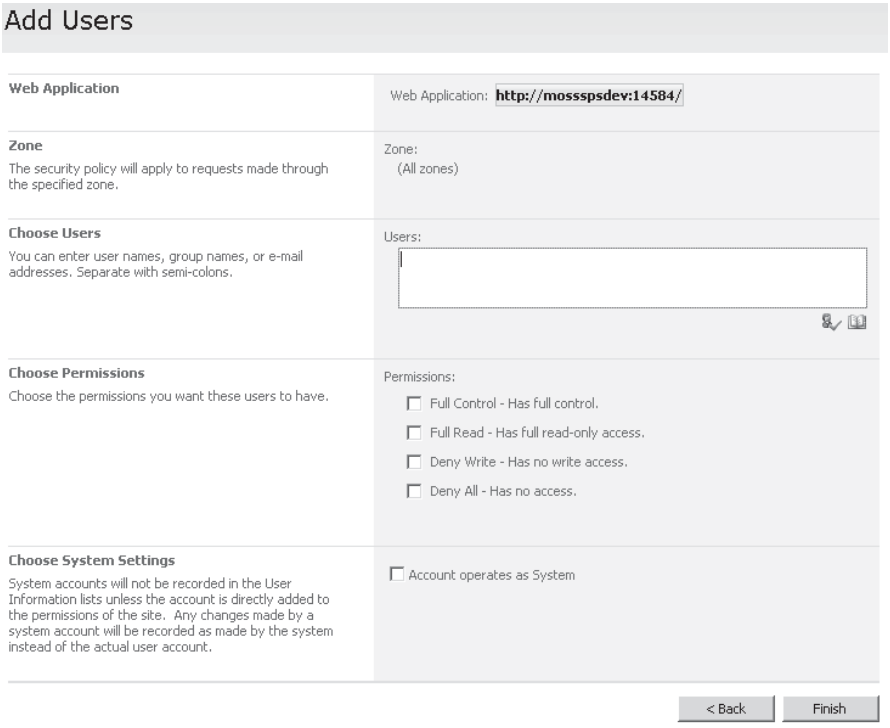


This option allows you to specify individual users and their rights within an application (at the application level, not the site level). This only affects the application you have selected. As shown, this usually includes the global accounts like the SharePoint Search account, which needs read access across the entire application in order to index content. Granularity is provided to the Internet zone level—either all or a selected zone (for example, Extranet).

When adding users, you will go through two steps. First, when you click Add Users from the policy list page, you are asked to select a zone:



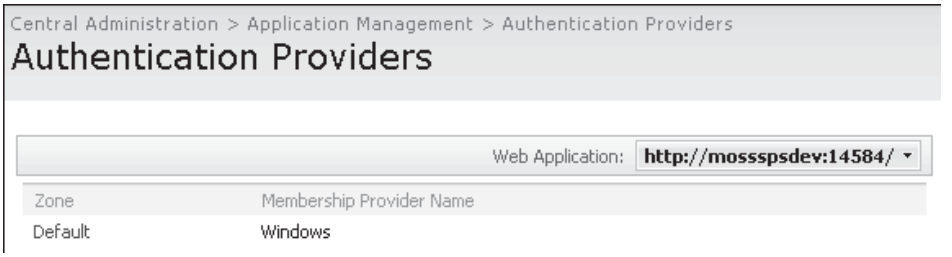
After you have selected the zone choice, clicking Next brings up the page where you can add the user and set the rights to be granted (the permission settings are pretty obvious):



Click the Finish button to add the user(s).

Authentication Providers

- **When you need it** You can use this option to view the list of authentication providers available for this application.



This option provides a list of the authentication providers currently installed for the application in question. This information is provided by the web.config file in the individual site (and Central Administration file as well), and is based on the authentication applied. This page allows you to modify settings for existing providers. In the example, this indicates

that only the Default zone has been defined and uses Windows Authentication. To edit the settings, click on the authentication provider you want to edit (in this case, Default) to open up the editing page. On this page, you can choose:

- **Zone** This is either set to default or to a selection you've made when you have multiple zones defined.
- **Authentication Type** There are two authentication types.
  - **Windows** Windows Integrated Authentication (NTLM)
  - **Forms** Forms-based authentication (requires separate authentication data source)
- **Web Single Sign On** This login information provided via SSO. This must be an Application Definition within SSO; check this on the Operations page by clicking Manage Settings for Single Sign-on under the Security Configuration group.
- **Anonymous Access** This enables or disables access by anonymous (non-authenticated) users.

If you choose Windows as the Authentication Type, the options are:

- **IIS Authentication Settings** These settings include:
  - **Integrated Windows Authentication** This is either NTLM or Negotiate (Kerberos). If Kerberos is selected, other server settings are required.
  - **Basic Authentication** These are user credentials (name/password) that are passed between systems in clear text (non-encrypted), meaning it can be captured by monitoring network traffic. This is the least secure login method, but it can be useful in public sites that enable login for content targeting.

If you choose either Forms or Windows Single Sign-on as the type, the options are:

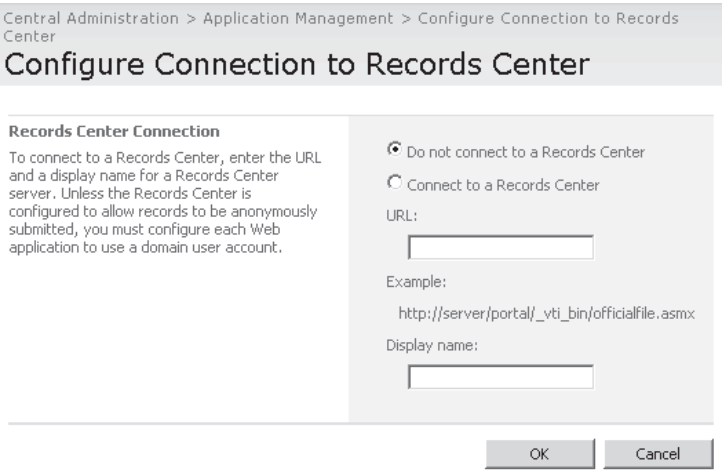
- **Membership Provider Name** This must be defined in the web.config files of both the general site and the Central Administration site.
- **Role Manager Name** This is the name of the Role Manager as defined in the web.config file(s), though this is optional.
- **Client Integration** This enables or disables launching client applications to view content (like a document); typically, you would disable this in a public-facing site.

External Service Connections

This section covers connections to SharePoint and SharePoint's connection to a records center site.

Records Center

- **When you need it** You can use this option to disconnect or connect an application to an existing Records Center Site.



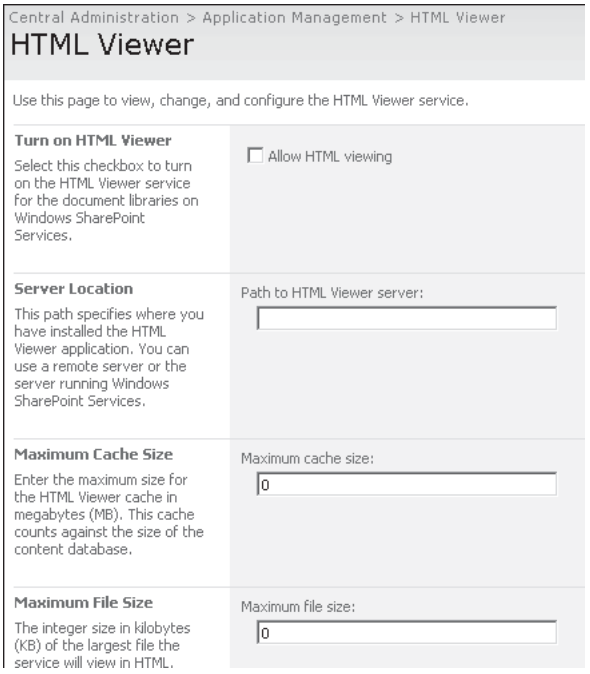
This option allows you to disconnect or connect the current application to an existing Records Center Site (either stand-alone or as a part of a site collection). The site is selected by entering the full URL to the site and then specifying a display name (the display name is what users see in the UI). If records cannot be anonymously submitted to the Records Center site, the account used to run the application must be a domain account (needed to cross between applications and sites). Note that an application can only be connected to one Records Center.

HTML Viewer

- **When you need it** You can use this option to enable or disable the HTML Viewer service.

This option allows you to enable or disable HTML viewing and specify the service for SharePoint to access it. HTML viewing is used to view documents and presentations when the client system does not have the necessary application (like Word). Options here include the following:

- **Turn on HTML Viewer** This enables or disables the service (from running in SharePoint).
- **Server Location** This specifies the full URL path to the HTML Viewer service (this is set when you install it).



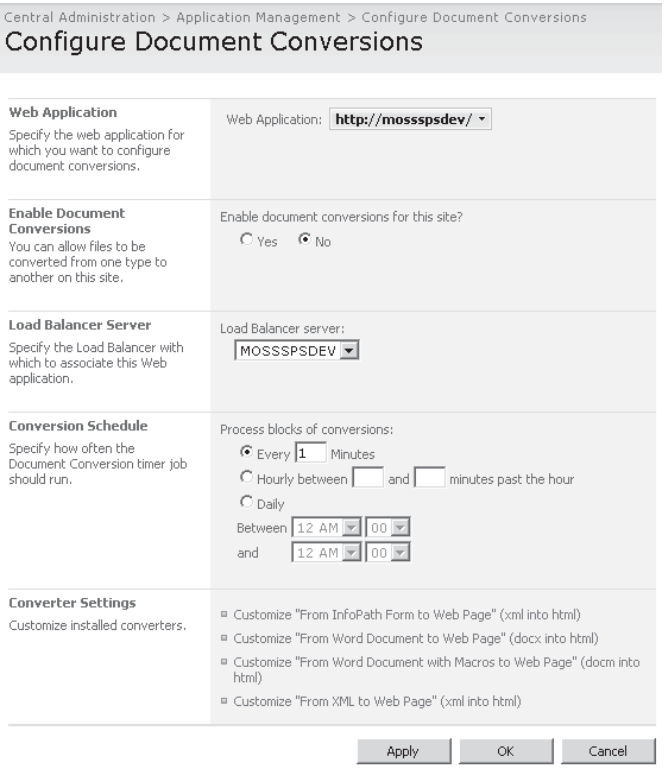
- **Maximum Cache Size** This sets the maximum size in megabytes to be used for caching files; 0 means no maximum.
- **Maximum File Size** This sets the maximum size in megabytes for files that can be opened with the viewer; 0 means no maximum.
- **Time-out Length** This is the amount of time in seconds to wait for a file to be loaded by the Viewer.

The HTML Viewer is a separate service and is not installed by default with SharePoint. Instead, you must download this and install the service separately, which you can do from http://www.microsoft.com/downloads/details.aspx?FamilyID=C62E0232-9BF6-48FC-829E-5C34D5C8B15F&displaylang=en.

After downloading, install the viewer on the server. By default, the HTML Viewer uses port 8093 (or whatever you've previously set). This port must be available on the Windows SharePoint Services server as well as on the Office HTML Viewer server (and opened in the firewall). You can verify that the service is up and reachable by simply entering the server URL (with port 8093). The return will be basically gibberish, but it proves the service is up and running; if it is not running, you will receive a message stating that "Internet Explorer cannot display the web page."

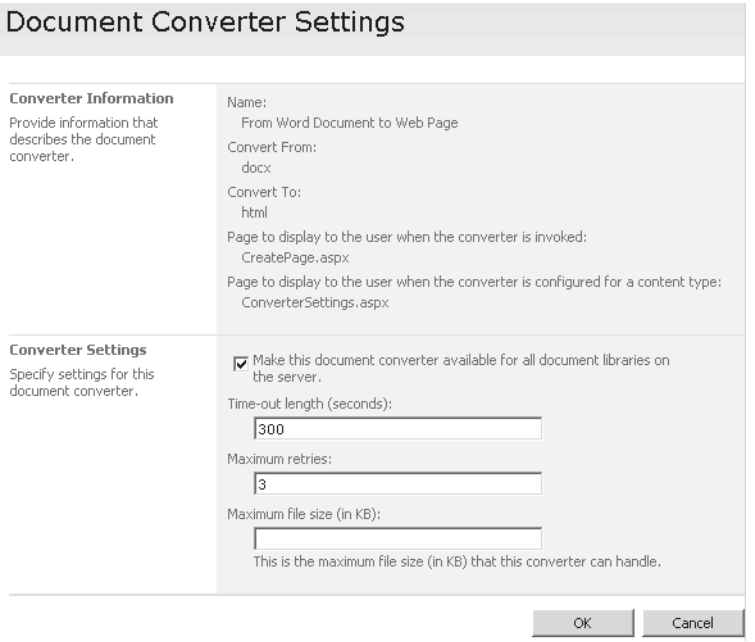
Document Conversions

- **When you need it** You can use this option to enable/disable and customize document conversions.



NOTE *This option always appears but only applies to MOSS (not WSS).*

This option enables or disables document conversions (used in content deployment and conversions within libraries). If it's enabled, you set the Load Balancing server and the Conversion Schedule (when content conversions will be done). Below this are the Converter settings for installed converters (converters are not installed here; you can only modify existing ones). The defaults are shown—for example, InfoPath Form to Web Page (xml to html) publishes an InfoPath form from its native xml into a flat html page. Depending on the converter, you can modify some of the options they run under. For example, click Customize From Word Document to Web Page (docx to html) to open the Document Converter Settings page:



From here, you can enable or disable the option that determines whether this converter can be used on all libraries or not (in other words, available for everyone or used selectively) and set the limits for conversion files:

- **Time-out Length** This is the time in seconds to allow for a conversion to complete.
- **Maximum Retries** This is the maximum number of attempts it will make to do the conversion.
- **Maximum File Size** This is the blank (no limit) or a number in KB for the maximum size file it will attempt to convert.

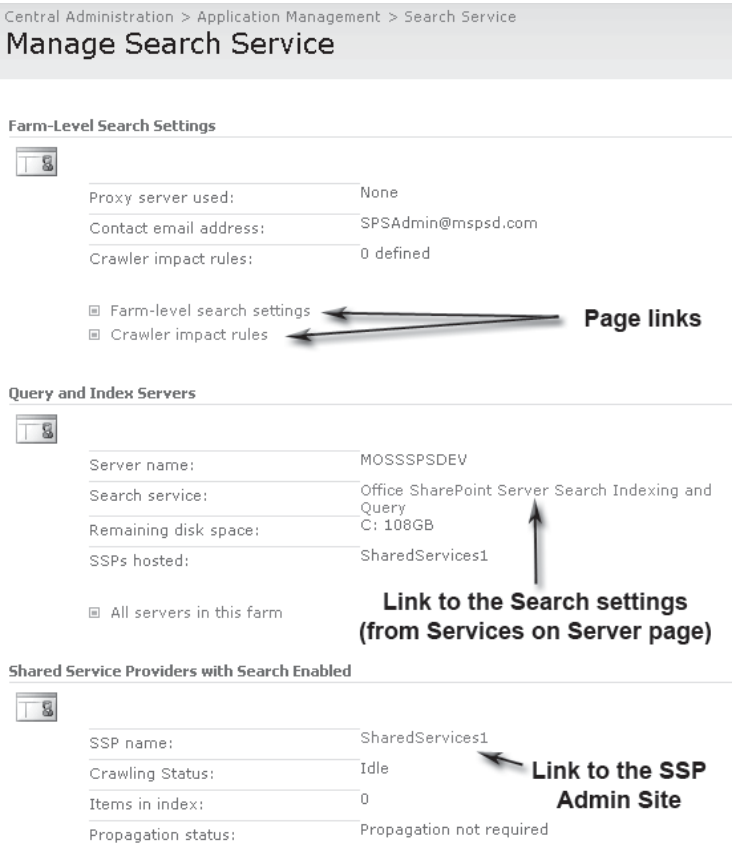
The converters shown in the illustration are simply the default ones installed with MOSS; your installation may have other/additional converters, so keep in mind that your list might be different.

Office Search (MOSS Only)

This section covers the settings for the Office Search Service (see more on Search in Chapter 18).

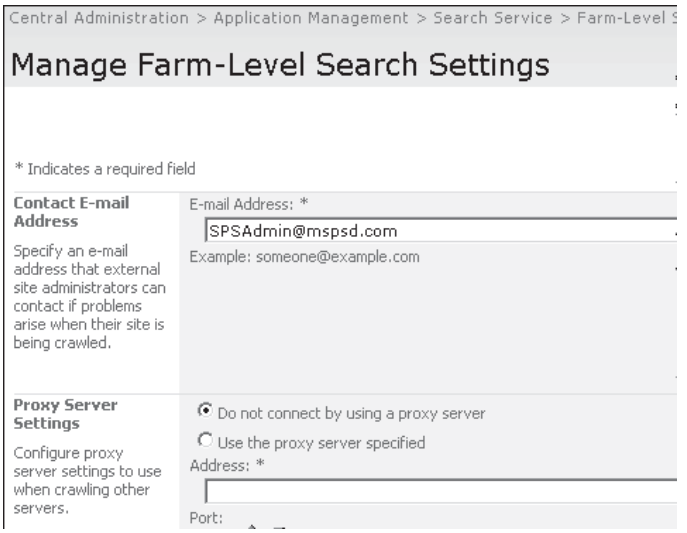
Manage Search Service

- **When you need it** You can use this option to modify the account running the search. You can also modify search schedules and scopes.



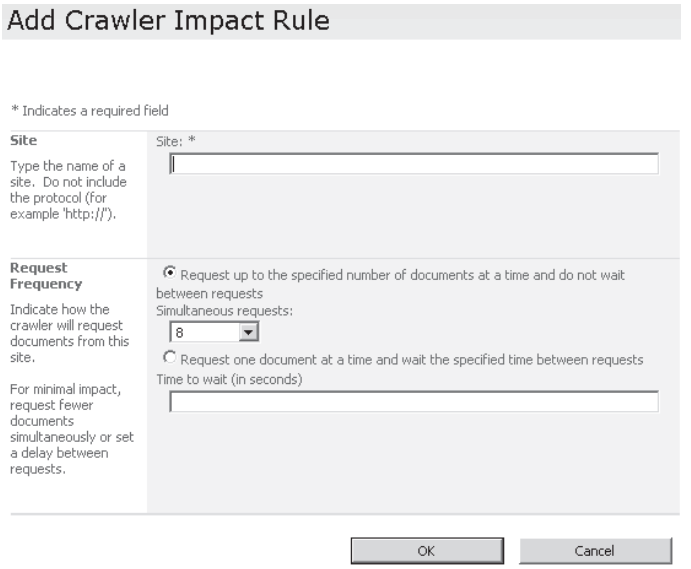
This option opens a snapshot page for displaying all statistics about the current farm server's status, including some important points like how much space is left on the index drive. This also provides links to a few settings:

- **Farm-level Search Settings** This link brings you to the settings page:



- **Contact E-mail Address** This is an address to send a message to if errors are detected while crawling content.
- **Proxy Server Settings** This is either disabled or enabled. If enabled, you must specify the address and port of the proxy server. You must also enable or disable bypassing local intranet addresses, and if necessary, provide a list of addresses to bypass.
- **Time-out Setting** This is the maximum time in seconds for a connection to be established and how long to wait for an acknowledgment.
- **SSL Certificate Warning Configuration** This enables or disables warnings when SSL certificate names don't match.

- **Crawler Impact Rules** These rules allow you to tweak the Crawler settings on individual URLs. When this opens, it is simply a list, and no rules are defined by default. Clicking Add Crawler Impact Rule opens the Rule Definition page:
  - **Site** This is the URL of the site you want to set a rule for.



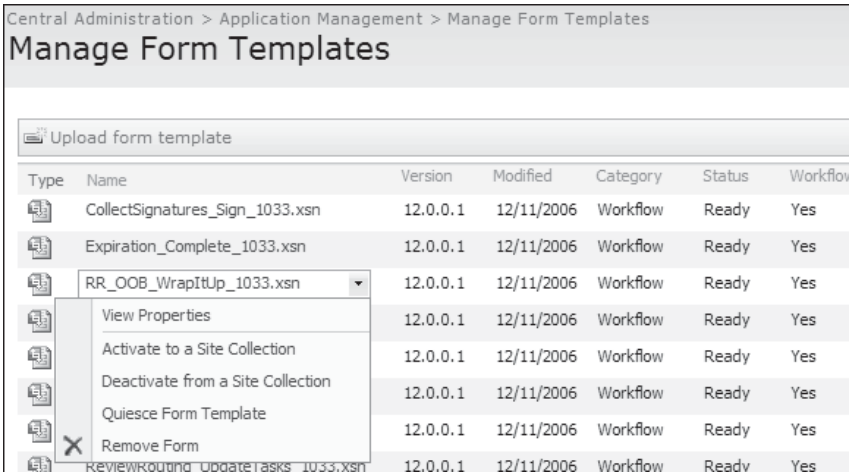
  - **Request Frequency** This allows you to increase or decrease the number of simultaneous requests (the higher, the more of a performance hit) or to set the crawl to request only one document at a time and set the number of seconds to wait for each (this provides the lowest possible impact on operations, but search indexing will be considerably slower).
- **Office SharePoint Server Search Indexing and Query** This brings you to the Search Service Settings (the Services on Server page available from Operations).
- **Shared Services Name link** This opens the Shared Service Provider Administration site.

InfoPath Forms Services (MOSS Only)

This section provides access to the settings and management of the InfoPath Forms Services within MOSS. This includes managing form templates as well as data access and proxy settings.

Manage Form Templates

- **When you need it** You can use this option to deactivate and remove InfoPath forms for site collections.



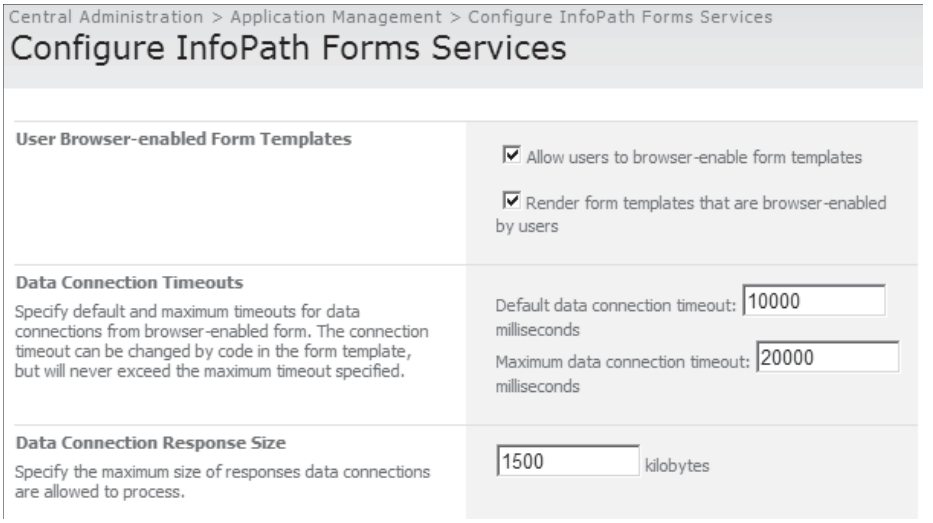
This option provides a list of all installed Form Templates in the farm. From the list, each item provides a drop-down list menu that allows you to do the following:

- **View Properties** This includes the name and category of the form, the identification information (form title, publish date, and so on), and security information (trust level, digitally signed, and so on). You can modify the category name from the view.
- **Activate to a Site Collection** This enables you to activate a Form Template in a SharePoint Site Collection (and all sites within it). This only applies to user forms—pre-installed forms that are part of the MOSS workflow cannot be activated or deactivated.

- **Deactivate from a Site Collection** This enables you to deactivate the Form Template from a SharePoint Site Collection (and all sites within it). This only applies to user forms, because pre-installed forms that are part of the MOSS workflow cannot be activated or deactivated.
- **Quiesce Form Template** This works just like the farm quiesce, wherein you can set an amount of time in minutes to make the template unavailable for new requests. Active workflows and forms using the Form Template are not affected. Once started by entering a time and clicking the Start Quiescing button, the Quiesce Form Template page will refresh to show you the status. Shutdown will continue even if you navigate away from this page. When the process has completed, this page will indicate it has finished (you can remove the form template at this point).
- **Remove Form** This removes the form from the farm.

Configure InfoPath Forms Services

- **When you need it** You can use this option to configure or modify Forms Services settings.



This option allows you to configure the InfoPath Forms Services to enable or disable user Form Templates and set data connection properties. This particular option is covered under Configuration so this is a bit of a repeat. Technically these options should already be set. From this page, the options are:

- **User Browser-enabled Form Templates** These templates allow you to enable or disable users to browser-enable Form Templates and render Form Templates that are browser-enabled (these are usually both on or both off).
- **Data Connection Time-outs** These set the default and maximum time the service will wait for a connection to a data source (10000 milliseconds = 10 seconds).
- **Data Connection Response Size** This sets the maximum amount of data that can be retrieved (the default is 15kb).

The User Form Settings enable or disable using forms with browsers. The Data Source Settings allow you to control how much and how long to wait for data responses; the more time given, the longer the system will wait on requests and the greater the impact on performance. The maximum setting will prevent IFS from waiting too long. The next sections all deal with data source access as shown:

- **HTTP Data Connections** This enables or disables SSL for HTTP connections to data sources (the default is enabled; for development this will be disabled).
- **Embedded SQL Authentication** This enables or disables database connections with embedded SQL connection strings that include the user name and password (the default is disabled; this should only be enabled in intranets and other low security environments).
- **Authentication to Data Sources** This enables or disables whether users can use authentication embedded in data connection files (the default is disabled; enable this if data connection files are secured).
- **Cross-Domain Access** Enables or disables cross-domain data connections, meaning a form can connect to a data source outside of the local domain (the default is disabled).

The HTTP Data Connections setting controls whether access to Internet data sources require SSL or not (in production, they always will). The remaining options are security based; the Embedded SQL selection determines whether forms can connect to data sources using a basic text user ID and password embedded in the connection string. Authentication enables you to embed connections into data connection files; while not visible to the end user, they are still basic text. Lastly, Cross-Domain Access determines whether a form can reach outside of the current domain. In general, the last three options here are always disabled in a production setting (unless there is a specific need, such as multiple domains in the same organization, and so on).

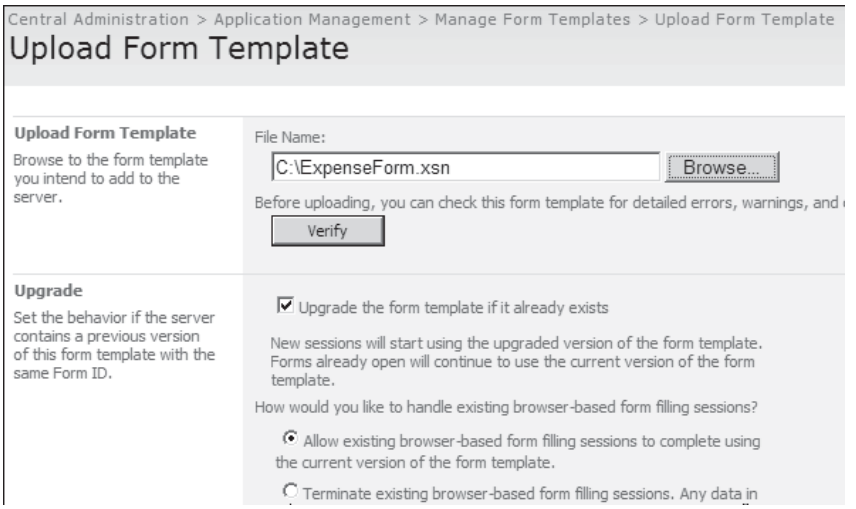
The last few sections are actually the throttle settings:

- **Thresholds** These set the number of postbacks and actions permitted per postback allowed on a form (this prevents excessive postbacks to the server, typical of a denial of service attack).
- **Form Session State** These control how long sessions remain active, the maximum in memory storage for the form, and the location of where to store the form Session State (the default is the Session State Service).

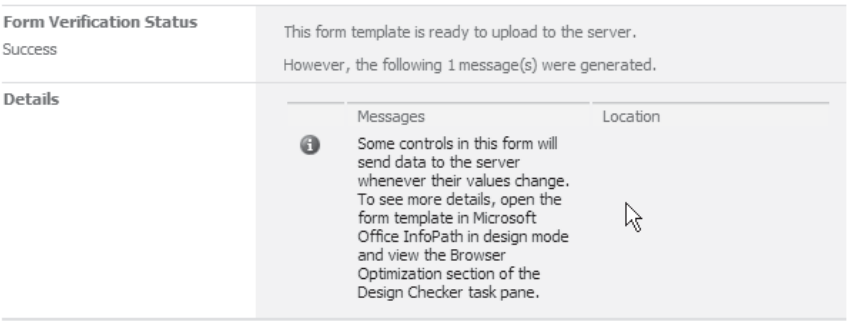
The thresholds are used to limit how much activity is permitted for a single form; the default numbers shown are more than adequate and should not be altered unless there is a definite need to change them. The Form Session State basically keeps the state of a particular form while it is being used, and you have the option of storing it in the server process (best for low bandwidth users and public facing sites) or in the database (less vulnerable to a server failure but rendering takes extra time).

Upload Form Template

- **When you need it** You can use this option to upload a new or revised InfoPath Form Template.



This option enables uploading InfoPath Form Templates to a farm. An uploaded form can be accessed (and activated) using the Manage Form Templates option. When preparing to upload, there is an option to verify/validate the form before uploading. This will alert you of any dependencies or issues you need to know about. To validate a form, use the Browse… button to find the file you want to upload. Then click the Verify button, which will return a status report on the form:



Note that when you return to the page after validating, you have to use Browse… again to select the file. To upload a new Form Template, select the file and then set the Upgrade Options:

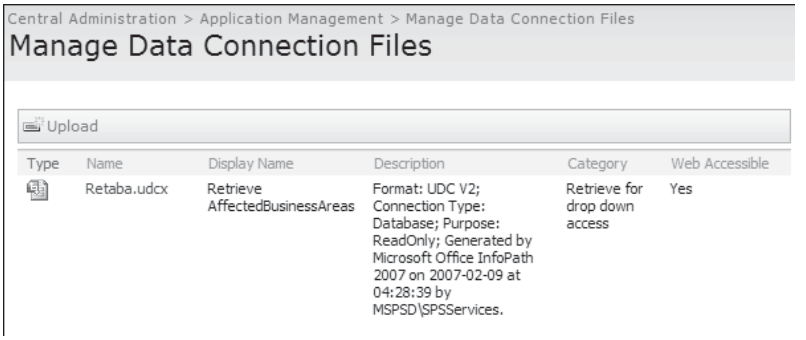
- **Upgrade the Form Template If It Already Exists** If checked, this will upgrade (overlay) an existing form if there is one to be found. If enabled, you must choose:
  - **Allow Existing to Complete with the Old Form** This allows all current sessions to complete. All new requests get the new form.
  - **Terminate Sessions** This immediately terminates any existing sessions including wiping any data that has not been saved. All new requests get the new form.

After uploading a form, it will appear in the Manage Form Templates list. When initially loading, you will see a status message that says "Installing." Then, when complete, the

status will change to "Ready." Note that you may have to view the properties to set the category for the form.

Manage Data Connection Files

- **When you need it** You can use this option to add, modify, or delete data connection files.

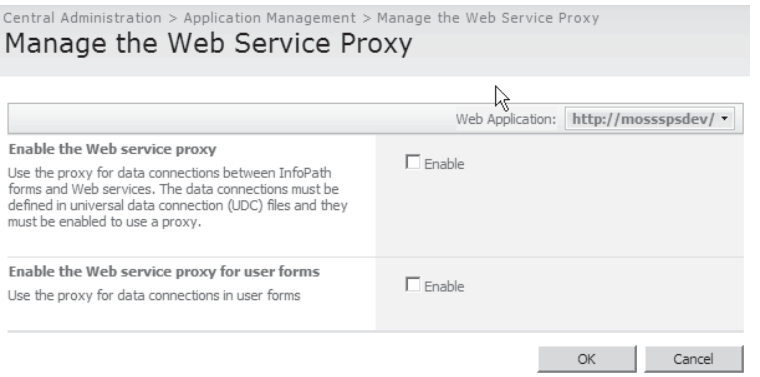


This option allows you to add, modify, and delete data connection files that are used by InfoPath forms. These allow connections to data sources to be encapsulated and reused for multiple forms. Either farm-wide or in a site collection, data connection files allow the actual connection and execution commands to be extracted away from the form developer, eliminating the need to know the actual connection information (account, password, and so on) since they can reuse the existing definition.

Data connection files are either Office Data Connection files (.odc) or InfoPath-created Universal Data Connection files (.udcx or .xml). A data connection file defines a specific kind of data query or update against xml, web services, SQL server, and any SharePoint list. Data connections common across the farm are stored here, but they can also be local to a site collection (using local Data Connection Libraries).

Manage the Web Service Proxy

- **When you need it** You can use this option to set or modify the Forms Service Proxy.



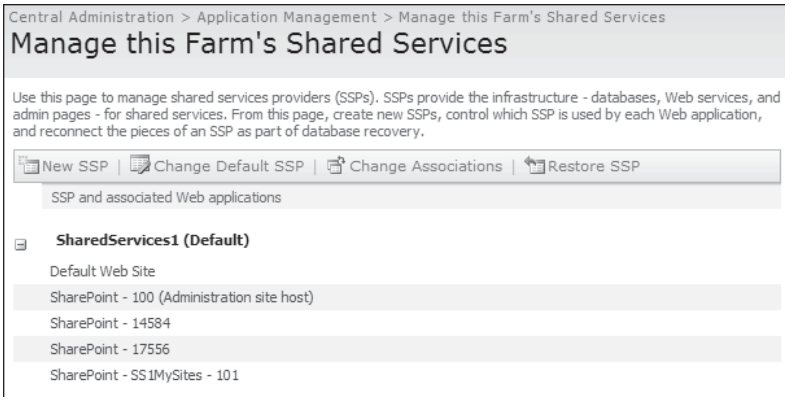
This option allows you to enable or disable using a proxy for the Forms Service data connections and user forms. If enabled, the proxy defined is used when any request for a connection is made.

Office SharePoint Server Shared Services (MOSS Only)

Shared Services are a required component of MOSS and must be installed and configured before the first sites can be created. As such, the complete configuration of SSPs is covered in detail in Chapter 6. Rather than duplicate much of that detail here, I will be fairly brief; for more in-depth setup and configuration settings, go to Chapter 6.

Create or Configure This Farm's Shared Services

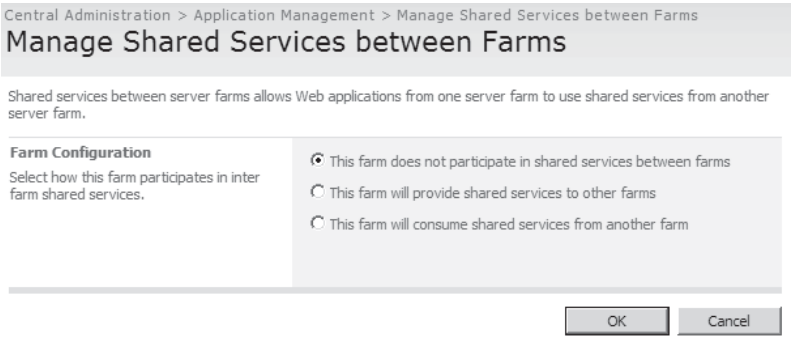
- **When you need it** You can use this option to change settings for Shared Services and add/change SSPs and associations.



This option opens the Manage This Farms Services page. From here, you can create a new Shared Service Provider (SSP), change the default SSP, change application/SSP associations, and restore an SSP (from a backup). Note that this is the same page you are brought to if you use the Manage Shared Services link on the Central Administration Home Page.

Grant or Configure Shared Services Between Farms

- **When you need it** You can use this option to disable sharing, or alternatively, share or consume Shared Services with another farm.



This option enables setting the SSP sharing options for this server farm. Built into MOSS, you can share Shared Services between different server farms allowing you to use a common SSP across any number of installations. Individual server farms can, of course, provide stand-alone services for use by the applications in the farm, but it can also provide services to or consume services from another. While the individual server farms may perform completely different functions, the common services (like My Sites) are shared. The options here are:

- Farm Configuration, set to one of the following types:
  - A farm that does not participate in shared services between farms. This is the default and standard setting for a stand-alone farm.
  - A farm that will provide shared services to another farm. This means that the current farm will be sharing services. When selected, you must choose the SSP you wish to share, and you can assign Process Accounts (domain\Username) to be used for the share (adding them here grants or revoke those accounts rights).
  - A farm will consume shared services from another farm. This means that the current farm will be a consumer of another farm's shared services. If selected, you must specify the database server, name, and authentication method (Windows Integrated or SQL Account) for the parent farm's configuration database (this gives the current farm access to the SSP configuration stored in the parent database). You will also have the option to either choose an existing SSP on the parent or create a new one (to be used here). Note that the options for the Provider/Consumer choices are shown when the radio button is clicked to enable this option.

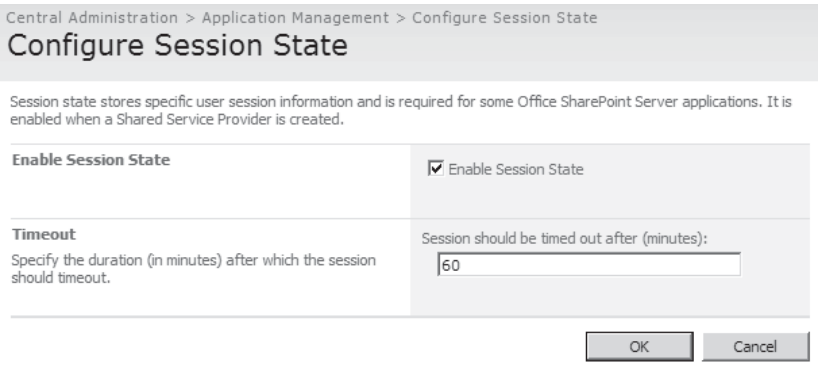
Check Services Enabled in this Farm

- **When you need it** You can use this option to check the current SharePoint services operation status.

This option reviews the SharePoint installation and verifies the state of all required services; if any required services are not running or some other issues exist, you'll be notified here as well as given guidance on how to correct the issue. When all is well in the farm, you'll see the message "There are no items to show in this view."

Configure Session State

- **When you need it** You can use this option to enable or disable session state and session length.



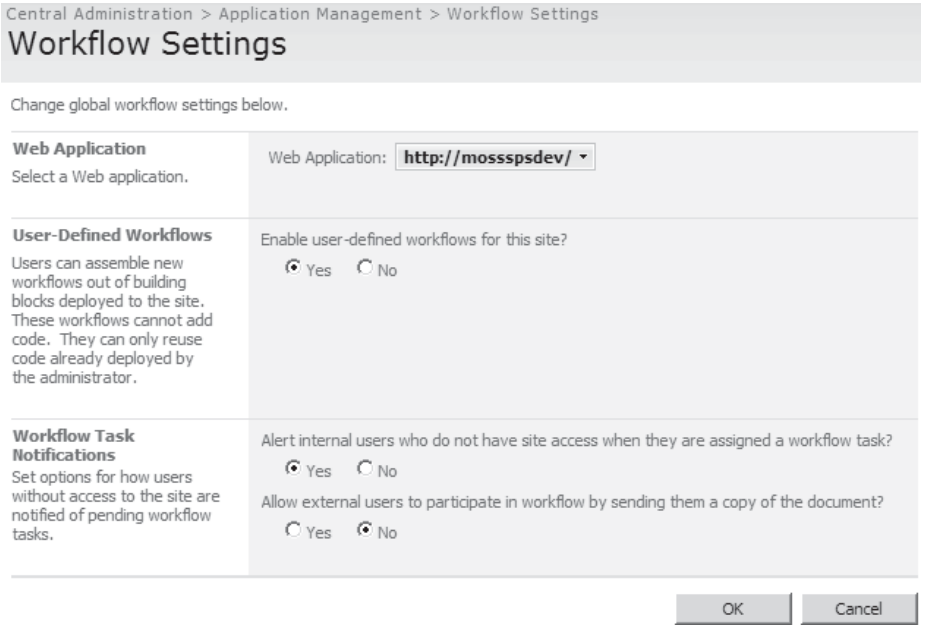
This option will enable or disable the user session state and how long a session should be maintained. The session state is based on when a user logs in and how long their session can remain inactive before requiring the user to log in again. The default setting is enabled with a session length of 60 minutes. In closed and otherwise secured environments, you can disable this option to eliminate users from ever having to log in more than once.

Workflow Management (MOSS Only)

This section covers workflow settings and enabling and disabling user-defined (designer-created) workflows.

Workflow Settings

- **When you need it** You can use this option to enable/disable user-defined workflows and set notification options.



This option enables you to enable/disable user-defined workflows (those built with SharePoint Designer) and set notification options for the selected application. The options here are:

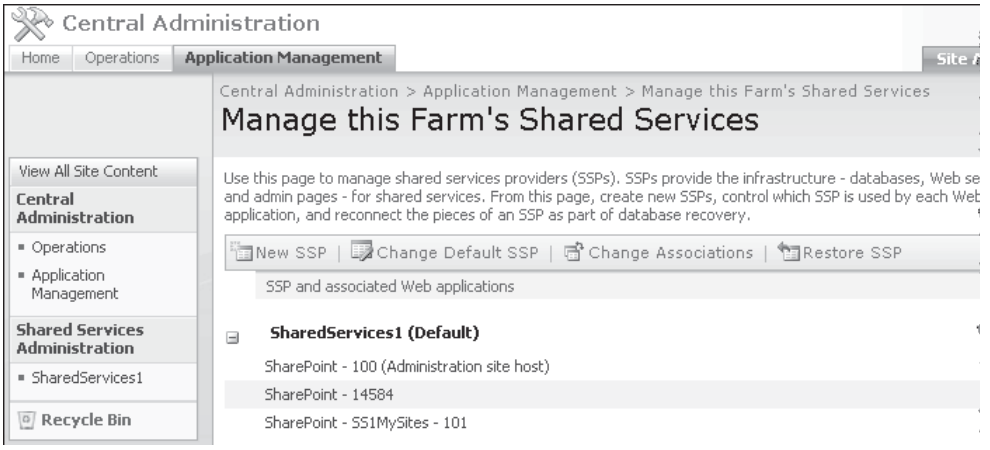
- **User Defined Workflows** When this is enabled, it means designer built workflows can be published and used in a site. When disabled, it prevents workflows from being published to the site (any that are there already will not be disabled). Be aware that the settings here do not affect the out-of-the-box workflows (Approval and so on) that are provided by default in lists and libraries (see Chapter 16 for more information).
- **Workflow Task Notifications** This is pretty self-explanatory. The Alert Internal Users option, if enabled, notifies a user when they've been assigned a workflow task

in a site that they do not have access to (thus they can contact the site owner for access), and the Allow External Users option, if enabled, will send a copy of the document along with notifications in a workflow (this is under the assumption that they have no access to the site).

Shared Service Provider Administration

Shared Services are a required component of MOSS, and Shared Services Administration provides you with the ability to create and modify Shared Service Provider settings for the server farm. Note that this is not the same as the Shared Service Provider Administration site; the SSP Admin site is used only to manage the SSP and its services. This site, on the other hand, provides management above that, allowing you to manage SSPs and the applications that are associated with it. Shared Services Administration is covered here; the Business Data Catalog is covered in Web Chapter 1 (available online).

From this administration page, you can create and work with Shared Service Providers as well as set or change sites associated with them. The purpose of allowing multiple SSPs is quite simple; you can provide a variety of shared services and My Sites across multiple applications or between farms. Some uses include keeping things such as My Sites organized by division instead of corporate wide; another might be to provide services to an extranet that has a completely different user base (and not providing My Sites). Accessing Shared Services Provider Administration is done through SharePoint 3.0 Central Administration. From the home page, click the Shared Services Administration link from the left navigation to open the Administration page:



As you can see, any existing SSPs are listed, as well as all of the applications associated with them. In this example, you can see the default setup: the first SSP and My Site applications and the applications for the Central Administration site and the first site created (at port 14584). Also, the one Shared Service Provider shown is indicated as the default, which means that any new application will be created under that SSP, though, of course, these can be changed. There are four primary operations provided here, as shown next.

New SSP

Creating a new Shared Services Provider can be done at any time once the farm has been set up initially, and at least one should be created before creating sites (though you can optionally not create an SSP, and the services will not be available). Creating an SSP requires that you

create at least one but suggests two new applications: one to host Shared Services and the other to host the My Sites Host. You can use a single application, but this means that the SSP and My Sites cannot be backed up separately.

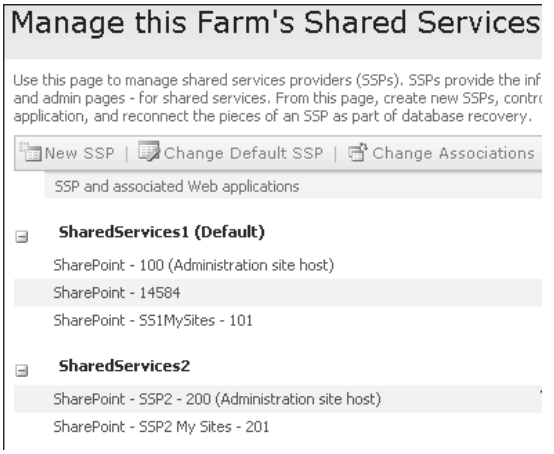
To create a new SSP:

1. Using SharePoint 3.0 Central Administration, navigate to the Applications Management page and create one or two applications. Set the port numbers so that you know they are related (I use 100 and 101 for the first, 200 and 201 for the second, and so on). What you use doesn't matter as long as you know where to find it. When you return to Shared Services Administration, you'll see that the two applications are assigned to the default SSP automatically (this will change when you assign them below).
2. From the Manage This Farm's Shared Services page, click the New SSP link to open the Create Shared Services Provider page, and then add the information for each section:
    - **SSP Name** This is where you specify the Name of the Shared Service Provider (this is arbitrary, and you can call it anything, however, you should name it so that it can be identified as an SSP, like SSP2 or SSP_EngDiv).
    - **Web Application** This is where you select the application you created for this SSP from the drop-down list.
    - My Site Location
        - **Web Application** This is where you select the application you created for this SSP's My Site Host from the drop-down list.
        - **My Site Location URL** This is the relative URL to be used for My Sites. By default (and typically), this is the root of the application (/), but you can choose to extend this, for example: /MySites/. This dictates the URL the users will need to get to their personal sites, for example, http://server:101/personal/spsservices/default.aspx or http://server:101/MySites/personal/spsservices/default.aspx.
        - **SSP Service Credentials** This specifies a user name and password for the account that will be used by the SSP to run jobs and communicate with other servers. This is typically the SharePoint Services Account (SPService).
    - SSP Database
        - **Database Server** This is the database server name that will support the SSP.
        - **Database Name** This is the name of the database for this SSP. I suggest you name this to indicate what it is, for example, SSP2_DB or SSP_EngDiv_DB.
        - **Database Authentication** This specifies a user name and password for the account that will be used by the SSP to access SQL Server. This is either Windows Integrated (the default) or a SQL Login Account.
    - Index Server
        - **Index Server** This allows you to choose which Index Server this SSP will use. This is set if only one or a drop-down list selection of all are available.
        - **Path for Index File Location** This is where the SSP index files will be stored. By default, this is set to C:\Program Files\Microsoft Office Servers\12.0\

Data\Office Server\Applications. I highly suggest that you never use a system drive (like C:) to hold index files (or log files) and that the storage be placed on a disk with plenty of space to handle the index files. (There's no exact estimate because it all depends on how much content exists and indexing settings).

- **SSL for Web Services** This selects Secure Sockets Layer (HTTPS) access for this SSP's Web Services (this requires a certificate that must be installed in IIS beforehand).

3. Click OK to create the SSP. When the operation completes, you should see the Success completion page, which lists all the services that have been added.
4. When you return to Shared Services Administration, you will see the new SSP with the two applications you used associated correctly to the SSP:



Change Default SSP

This option allows you to set the default SSP to be used when new applications are created. When you initially set up MOSS, your first SSP will be designated as the default. When you create another SSP, you can switch between them. Whichever is default is the one that new applications will be automatically associated to. Regardless of when created, application associations can be changed at any time. FYI, the default SSP cannot be deleted (you have to have at least one). To change the default:

1. Using SharePoint3.0 Central Administration, click the Shared Services Administration link to open the Manage This Farm's Shared Services page. Then, click the Change Default SSP link:



2. Simply use the drop-down list to select the SSP as the default (the current one is indicated). Then click OK. You will receive a warning message regarding changing the default:
   - **Existing Web Applications Retain Their Associations** This means that the applications don't automatically switch to the new default; they stay where they are unless you change the association.
   - **New Applications Created Will Automatically Associate to the New Default** This means that any new applications created from now on automatically get assigned.
3. Click OK to change the default, and when you return to Shared Services Administration, you will see that your change has been made:



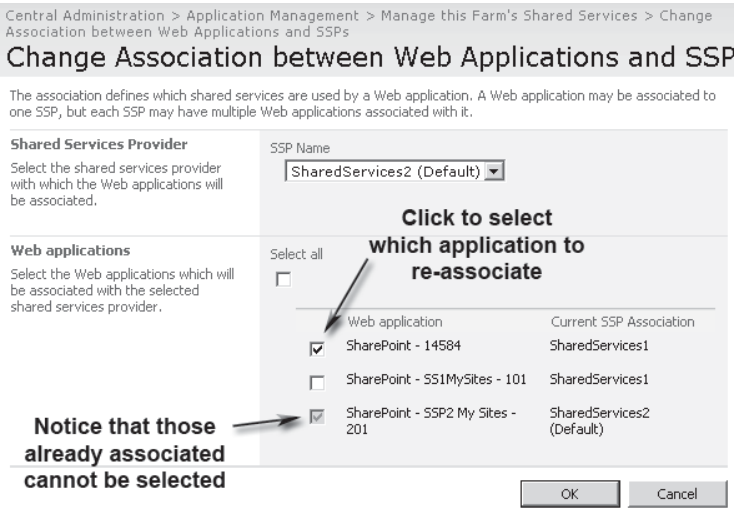
Change Associations

This option allows you to change which SSP an application hosting SharePoint Sites is associated with. You use this when you're changing SSPs, modifying your information topology, maxed out on capacity in an SSP, or recovering from a problem.

NOTE *I do suggest that once an application is created and associated, it's left alone. I only create new SSPs as topology dictates and typically make sure the default association is set correctly before I create applications. However, there are times when this must be done, as I mentioned previously.*

To change SSP/application associations:

1. Using SharePoint 3.0 Central Administration, click the Shared Services Administration link to open the Manage This Farm's Shared Services page. Then, click the Change Associations link:



2. Click the check box next to the application(s) to move and then simply click the OK button. Just like when changing default SSPs, you'll get a warning about changing associations:

   - Services may take time before they are available; for example, search must crawl the new content before it will appear.
   - Any web application–specific data in the old SSP (the BDC, for example)—remains there but can't be accessed (unless association is switched back).
   - All service-level configurations will remain in the old SSP, meaning your settings stay there and do not carry to the new SSP. You will have to use the new SSP administration to verify/make settings.
   - You must review the process accounts list on the SSP Properties page to remove accounts that no longer require access (to the old SSP).

3. Click the OK button to change the association. On return to the Manage This Farm's Shared Services page, you'll see that the application has moved:



Restore SSP

This option allows you to restore an SSP database from a SharePoint backup. This is the second step in a restore process when you are recovering an SSP; once the database has been restored, you use this option to "attach" the SSP to the database (in effect, associating a new SSP definition and application(s) with a restored SSP database).

1. Using the CA site, click the Shared Services Administration link to open the Manage This Farm's Shared Services page. Then click the Restore SSP link:



2. The settings here are identical to those when creating an SSP (see previous example); except, instead of creating the databases, you select the existing ones. Be sure to verify the correct accounts are used and that these have access to the restored databases!
3. Click OK to restore the SSP.

Summary

In this chapter, we've covered all of the aspects of Central Administration and the parts of SharePoint it manages, including applications, top-level sites, and for MOSS, managing Shared Services Providers. You've also seen that Central Administration handles the global aspects of the install, including setting server roles and managing services. In the next chapter, we'll dive into the next layer in Site Administration.

CHAPTER 8

SharePoint Site Administration

There are three levels of Site Administration: the Site Collection (or root site), which controls the primary settings for the site and all sites beneath it; sub site administration, which, depending on the settings above it, has local options; and My Site Administration, which is used by individual users to control their own personal sites.

In the Portal Site model (Collaboration and Publishing), there is a top site (the portal/parent), and all sites below it are considered sub sites. In the Team Site model (basically all others), there is a top site then all sites below it are considered sub sites. When the two are combined, the models are still the same; the Team Sites still control the sub sites below them and are loosely attached to the Portal Sites via the Site Directory. You can, in fact, join and leave a portal (that option is explained next). Joining a portal makes the site searchable and content accessible by relative URLs.

The hierarchy of administration dictates the services that are available for the sites below it. For example, a feature must be installed in the Site Collection for sub sites to use it. A feature must be enabled in a sub site for users to access it.

Remember that all SharePoint sites, including Central Administration, are constructed the same way; for example, CA is simply a top-level site collection that has a specific Site Definition that provides you with the features you need there. The same goes for Shared Service Providers (SSPs); the SSP administration site is simply a top-level site. In the case of My Sites, there is a top-level site collection (the My Site Host) and the My Sites themselves are top-level sites, and again, a Site Definition provides the personalization features.

While much of Site Administration has to do with site organization, the focus here is not on the supplemental things done in the User Interface (such as moving pages, for example), but on the actual Site Administration, which includes user management, feature management, and gallery management. Note that I will touch on all of the options but will skip details on items (like Content Types) that are covered elsewhere in the book.

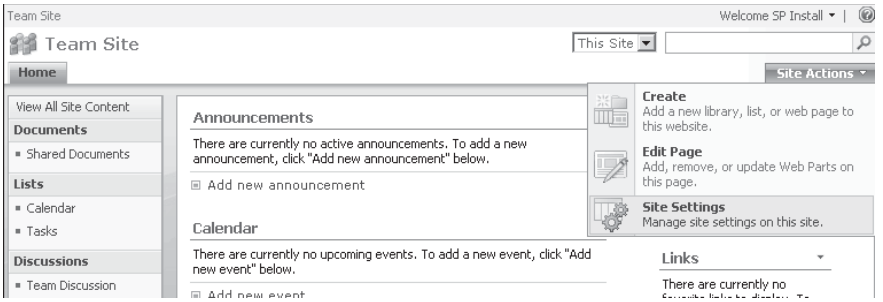
Site Collection Site Administration Group

Access to the Site Collection Site Administration is achieved from either the home page of the site or in the case of a Portal Site, any one of the top-level site collections. This is available

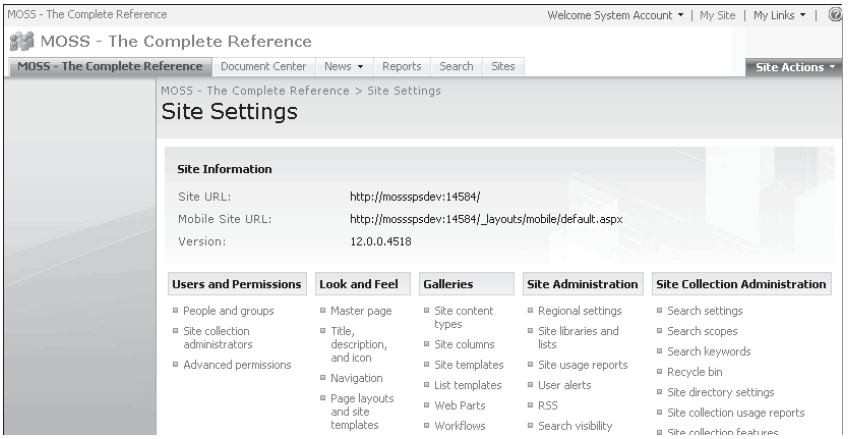
under the Site Settings drop-down menu as shown. Note that in this case it happens to be a Collaboration Site type:



In WSS sites, you access it from the home (default) page:



As I've mentioned, I'm not focusing on the content and navigation because this is covered in Chapter 9 on the User Interface. At this point, you're only interested in the Aite Administration settings. When the Site Administration page for a Site Collection opens, it looks something like this:



This is the top level of Site Administration. As you can see, the previous example is from a MOSS Collaboration (Portal) Site type. In the top level, in WSS and Personal (My Sites) sites, you see most of the same options, with the exception of the enterprise-level options (Search and so on).

Sub sites represent the second or third level in administration, and have most of the same options, but these settings are site-specific for themselves and for any site created below them. The options available are the same for any sub site, whether or not it inherits the security of its parent site; security (users) can be different, but the site features are still controlled by the parent. Sub sites have no options for management of features and so on—you will notice that the Site Collection Administration group has a link only to the parent site administration.

In the case of a Portal Site (Collaboration or Publishing), sites created from the portal root have the same options as the WSS sub site shown previously. Each of the options available is explained next, but note that some of these options apply only to MOSS (mainly the Enterprise Search and ECM features) and one only to WSS; each of these are indicated as necessary.

Users and Permissions

This section covers the management and settings for users, user groups, and permissions throughout the site. This includes creating new users and groups and the details of assigning the activities individual users and groups are permitted on the site.

People and Groups

- **When you need it** You can use this option to add and remove users and assign them to groups or add and remove groups.

People and groups set up your user governance and security within the site. You add people that can access the site and assign them to a group that grants the user specific rights. This can get a bit confusing when considering Security Groups, Users, and Site Groups, so let's make sure this is clear:

- **Users** These are a login account in Active Directory.
- **Security Groups** This is an AD security group of which users are members.
- **Site Groups** These are defined within SharePoint as groups to which users are assigned, granting them rights within the site/portal (big surprise: Site Groups are SharePoint lists of users). I prefer to call these *Rights* Groups.

When you add a user to SharePoint, they become an individual user and are listed that way in the people lists shown below. When you add an AD Security Group to SharePoint, all members in the group become members in SharePoint, though they are not individually listed. In both cases, you assign them to a SharePoint Group, which gives them certain rights in the site.

Site Group/User Rights The complete list of available rights is as shown in the following table.

Permissions	Description
List Permissions	
Manage Lists	Create and delete lists, add or remove columns in a list, and add or remove public views of a list.
Override Check Out	Discard or check in a document that is checked out to another user.
Add Items	Add items to lists, add documents to document libraries, and add web discussion comments.
Edit Items	Edit items in lists, edit documents in document libraries, edit web discussion comments in documents, and customize Web Part Pages in document libraries.
Delete Items	Delete items from a list, documents from a document library, and web discussion comments in documents.
View Items	View items in lists, documents in document libraries, and view web discussion comments.
Approve Items	Approve a minor version of a list item or document.
Open Items	View the source of documents with server-side file handlers.
View Versions	View past versions of a list item or document.
Delete Versions	Delete past versions of a list item or document.
Create Alerts	Create e-mail alerts.
View Application Pages	View forms, views, and application pages. Enumerate lists.
Site Permissions	
Manage Permissions	Create and change permission levels on the website and assign permissions to users and groups.
View Usage Data	View reports on website usage.
Create Sub Sites	Create sub sites such as Team Sites, Meeting Workspace Sites, and Document Workspace Sites.
Manage Web Site	Grants the ability to perform all administration tasks for the website as well as manage content.
Add and Customize Pages	Add, change, or delete HTML pages or web part pages, and edit the website using a Windows SharePoint Services–compatible editor.
Apply Themes and Borders	Apply a theme or borders to the entire website.
Apply Style Sheets	Apply a style sheet (.CSS file) to the website.
Create Groups	Create a group of users that can be used anywhere within the site collection.
Browse Directories	Enumerate files and folders in a website using SharePoint Designer and Web DAV interfaces.

Permissions	Description
Use Self-Service Site Creation	Create a website using Self-Service Site Creation. (Note that this is not the My Site—enabling this enables user to create sub sites.)
View Pages	View pages in a website.
Enumerate Permissions	Enumerate permissions on the website, list, folder, document, or list item.
Browse User Information	View information about users of the website.
Manage Alerts	Manage alerts for all users of the website.
Use Remote Interfaces	Use SOAP, Web DAV, or SharePoint Designer interfaces to access the website.
Use Client Integration Features	Use features that launch client applications. Without this permission, users will have to work on documents locally and upload their changes.
Open	Allow users to open a website, list, or folder in order to access items inside that container.
Edit Personal User Information	Allow a user to change his or her own user information, such as by adding a picture.
Personal Permissions	
Manage Personal Views	Create, change, and delete personal views of lists.
Add/Remove Personal Web Parts	Add or remove personal web parts on a Web Part Page. (Note that this applies to parts that are designated as personal.)
Update Personal Web Parts	Update web parts to display personalized information.

Site Groups By default, there are three basic groups: Visitors, Members, and Administrators, which are created for you when you create either a site or a portal (MOSS) site:

- **Visitors** These have read access for the site, but cannot make any changes in the site.
- **Members** These have basic contributor access (they can add and update items).
- **Administrators** These have full rights to manipulate the site (modify the structure and navigation, add/remove users, and so on).

NOTE *Personal sites do not have groups defined by default, but instead have the same functionality as any top-level site.*

The only difference between these groups is the rights that have been assigned. In the MOSS environment, there are several additional groups, which are all set up to be tweaks of the rights for specific purposes as follows:

- **Approvers** Members of this group can edit and approve pages, list items, and documents.

- **Designers** Members of this group can edit lists, document libraries, and work on pages in the site. Designers can create Master Pages and page layouts in the Master Page Gallery and can change the behavior and appearance of each site in the Site Collection by using Master Pages and CSS files.
- **Hierarchy Managers** Members of this group can create sites, lists, list items, and documents.
- **Quick Deploy Users** Members of this group can schedule Quick Deploy jobs.
- **Restricted Readers** Members of this group can view pages and documents, but cannot view historical versions or review user rights information.
- **Style Resource Readers** Members of this group are given read permission to the Master Page Gallery and restricted read permission to the Style Library. By default, all authenticated users are members of this group. To further secure this site, you can remove all authenticated users from this group or add users to this group.
- **Viewers** Members of this group can view pages, list items, and documents. If the document has a server rendering available, they can only view the document using the server rendering.

Default Permission Level Groups Default permission groups are simply some pre-defined groupings of rights you would typically use, and they offer a quick way to assign a specific kind of role. These groups represent the equivalent of what you might assign individually and are just grouped for convenience (don't confuse these with Site Groups):

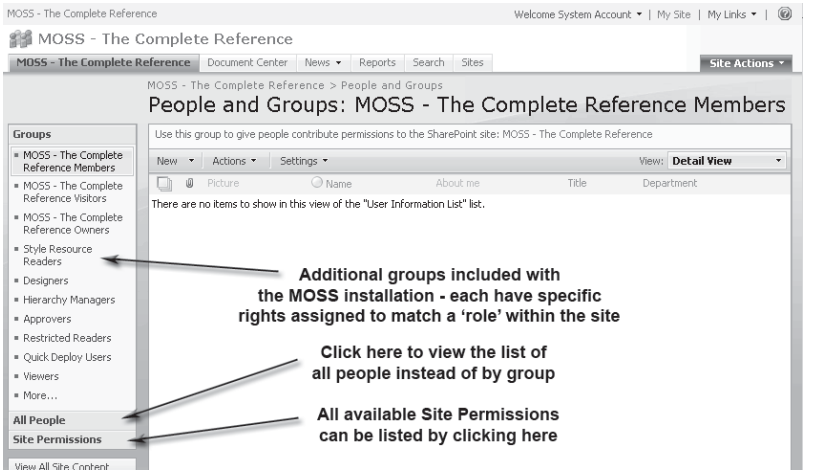
Permission Group	MOSS/WSS	Description
Full Control	MOSS/WSS	Has full control (also known as Administrative level permissions).
Design	MOSS/WSS	Can view, add update, delete, approve, and customize (also known as Member level permissions).
Manage Hierarchy	MOSS	Can create sites and edit pages, list items, and documents (also known as Site Designer level permissions).
Approve	MOSS	Can edit and approve pages, list items, and documents.
Contribute	MOSS/WSS	Can view, add, update, and delete.
Read	MOSS/WSS	Can view only.
Restricted Read	MOSS	Can view pages and documents, but cannot view historical versions or review users rights information.
View Only	MOSS	Member of this group can view pages, list items, and documents. If the document has a server-side file handler available, they can only view the document using it (for example, using HTML viewer instead of Word).
Limited Access	MOSS/WSS	Can view specific lists, document libraries, list items, folders, or documents when given permissions.

NOTE *Personal sites provide the same permission level groups as WSS shown previously.*

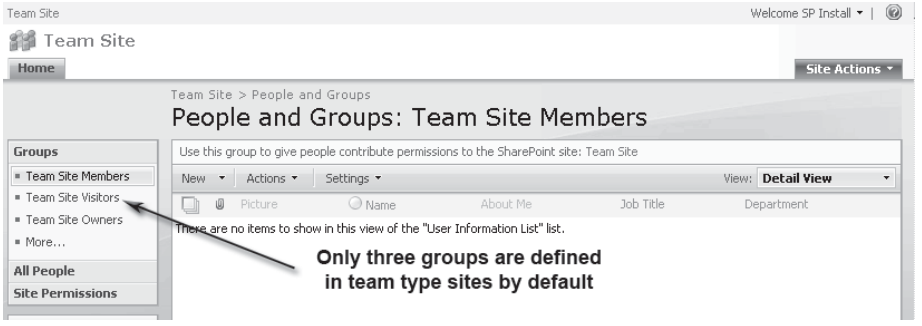
View Existing Users and Groups

To view existing groups, membership, users and site permissions, do the following:

1. Open the Site Administration from the home page
2. Under the Users and Permissions group, click People and Groups. This will open the By Group list:



As you can see in the previous illustration, there are several pre-defined groups for MOSS. You can also see that different views are made available; by simply clicking on each group to view members, you have the view of all users regardless of group. From here, you also have a link to Site Permissions, which provides a list of rights that are enabled in the site (permissions can be turned off site wide). The WSS sites (and MOSS sub sites) are nearly identical, but you'll see that the number of groups is just the standard three:



Note that even though there is a More… link, there are only three for WSS (the same page is used for both WSS and MOSS). This link simply brings you to the list of Groups. In addition to the New link for creating new users and groups (in SharePoint), provided on both

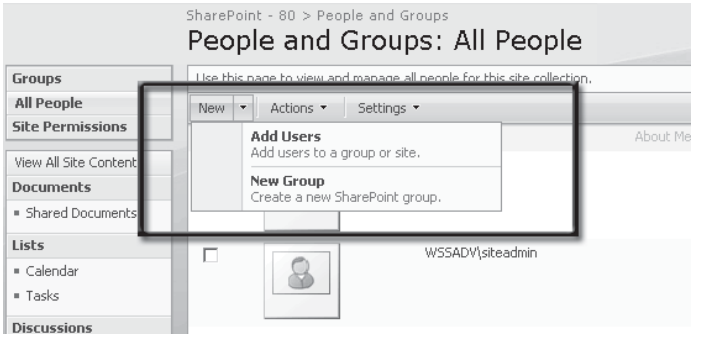
the Groups and People views, there is also an Action menu that provides some additional capabilities:

- **E-mail Users** This allows you to select any number of users and send a group e-mail.
- **Call/Message Selected Users** This allows you to call or message selected users online or during a web meeting.
- On the Group view action menu:
  - **Remove Users from Group** This removes selected users from a group (it does not delete the user account).
- On the All People view action menu:
  - **Delete Users from Site Collection** This deletes the user from the site collection. (Note that this does not affect the My Sites in MOSS, but simply removes them from the current site.)

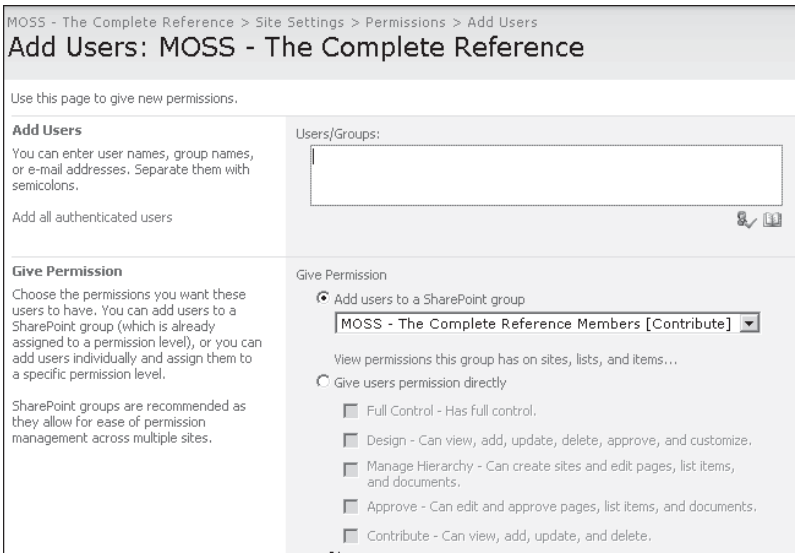
Add/Modify User/Security Group to Site or Site Group

When adding a new user (AD Login) or security group (AD Security group), you can simply select the group they will be added to and add them there, or you can assign groups when viewing a user that has already been added. To add a user or security group or add memberships to an existing account, do the following:

1. Open the Site Administration from the home page.
2. Under the Users and Permissions group, click People and Groups. This will open the By Group list (you can also use the All People view).
3. Click the New link, and from the drop-down menu, select New User:



4. This opens the Add User page. From here, you can enter any number of accounts and designate which group(s) they will belong to:



5. As you can see, you can select a defined group or specify unique permissions. If the user or security group is already a member of the site, this simply makes them a member of the selected group or gives them the specific permissions you assign. If this is a new user or security group, new account(s) are created and assigned to the specified group or given the specific rights.
6. Click OK to add the account or new membership.

Delete User/Security Group to Site or Group

To add memberships to a specific user or group, you should use the previous add method. The only other options available are removing a user or group from a site group or removing the account(s) from the site collection entirely. This option works the same in both the Group view and the All People view:

1. Open the Site Administration from the home page.
2. Under the Users and Permissions group, click People and Groups. This will open the By Group list.

   To remove users from a group:
3. From the Group view, click the Site Group in which you want to remove the user or group in the left navigation.
4. When the Site Group list of users and groups opens, you can click the check box next to each user or group you wish to remove.

5. From the Actions menu, select Remove From Group.

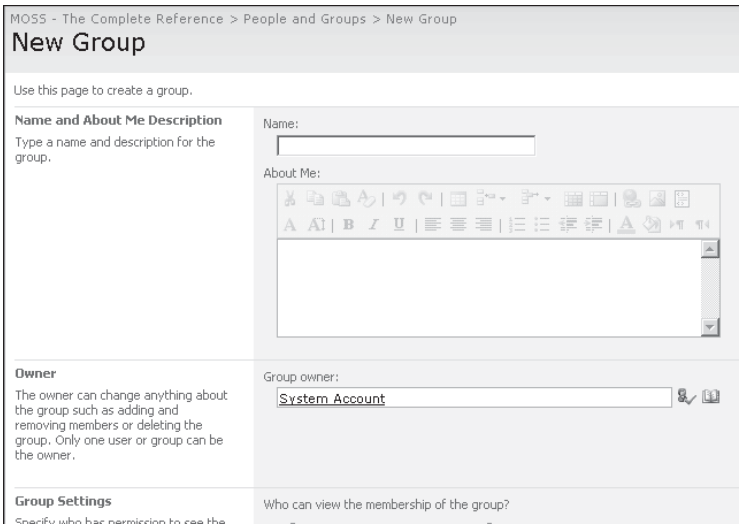
   To remove a user from the site collection:

6. Click the All People view in the left navigation to display the list.
7. When the All People list of users and groups open, you can click the check box next to each one you wish to remove from the site collection.
8. From the Actions menu, select Delete Users from Site Collection.

Add/Modify SharePoint Site Group

Adding a new Site Group (rights group) is done nearly the same way as a user, but permissions can be set as granular as desired. To add or modify user or security groups or add memberships to an existing account, do the following:

1. Open the Site Administration from the home page.
2. Under the Users and Permissions group, click People and Groups. This will open the By Group list (you can also use the All People view).
3. Click the New link, and from the drop-down menu, select New Group.
4. This opens the New Group page. From here, you can enter any number of accounts and designate which group(s) they will belong to:



5. The options to set here are as follows:

   - **Name and About Me Description**
     - **Name** This is the name of the new group.
     - **About Me** This is the rich text field that allows you to describe what the group is for.
   - **Owner** This designates the owner of the group (this defaults to the user creating it).
   - **Group Settings** These provide options:

- **Who Can View the Membership** This can be set to either Group Members or Everyone.
- **Who Can Edit the Membership** This can be set to either Group Owner or Group Members.

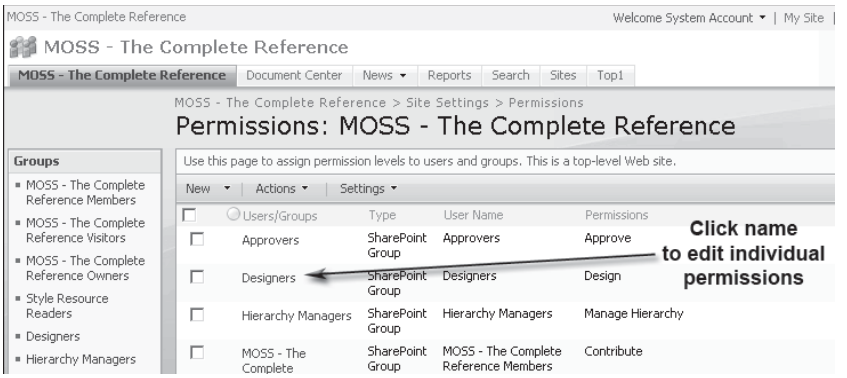
- **Membership Requests** This provides the following options:
  - **Allow Requests to Join/Leave the Group** This enables or disables users from requesting membership to the group.
  - **Auto-accept Requests** This enables or disables whether user requests are automatically accepted.
  - **Send Membership Requests** This designates an e-mail address to send the requests to (presumably, to someone with authority to add them).
- **Give Group Permission to This Site** This allows you to select the group permissions for the site.
- **E-mail Distribution List** This provides the following options:
  - **Create an E-mail Distribution Group** This enables or disables the ability to broadcast an e-mail to the group.
  - **Distribution List E-mail Address** This sets the address for the distribution group if enabled.
  - **Archive E-mail** This enables or disables archiving of e-mail sent to the group; if enabled, e-mails can be stored in a new list (created when the group is) or an existing list (if keeping e-mails centralized).

6. Click Create to create the new group. Once the group is created, you can begin adding users and security groups to it.

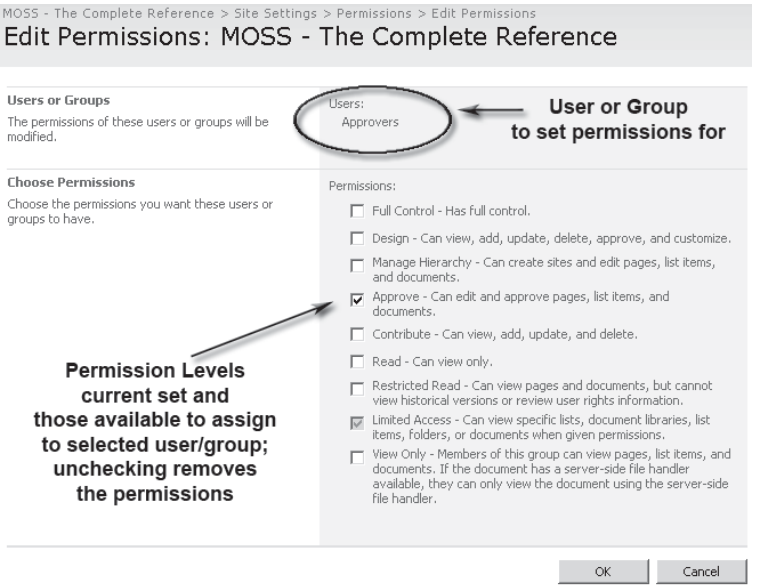
Enable/Disable Site Permissions

Site Permissions allow you to manage individual permissions and provide granular control of individual rights. To open this page:

1. Open the Site Administration from the home page.
2. Under the Users and Permissions group, click People and Groups. This will open the By Group list.
3. On the left-hand navigation menu, click Site Permissions to open the Permissions page:



4. To edit the permissions of a specific user or group, simply click the name to open the Edit Permissions page:



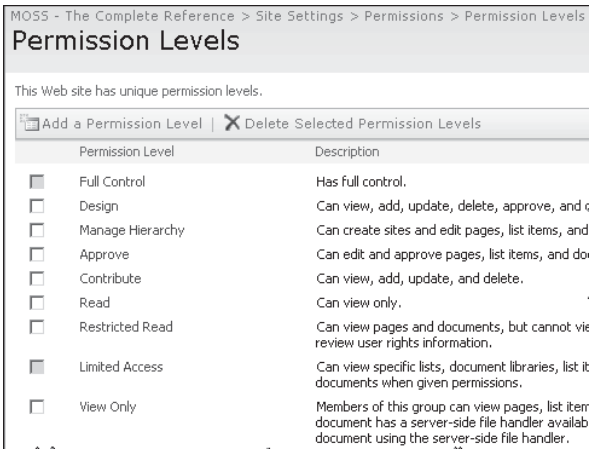
5. Additional options on the Permissions page also include the following:

   - **New** As explained previously, this allows you to create a new user or group.
   - **Remove User Permissions** These remove individual permissions from a user or security group.
   - **Edit User Permissions** These edit all permissions (grant/revoke) for a user or security group.
   - **Site Collection Administrators** These add/remove site collection administrator accounts.
   - **Access Requests** These enable or disable Access Requests to the site (MOSS only).
   - **Permission Levels** These configure granular permissions for this site.

Manage Permissions Level Groups

To manage the Permission Level groups—also known as those non-site groups that are loosely grouped by sets of rights for an individual site—you must do this from the Permissions page. To access the existing groups to view, add updates, or delete, do the following:

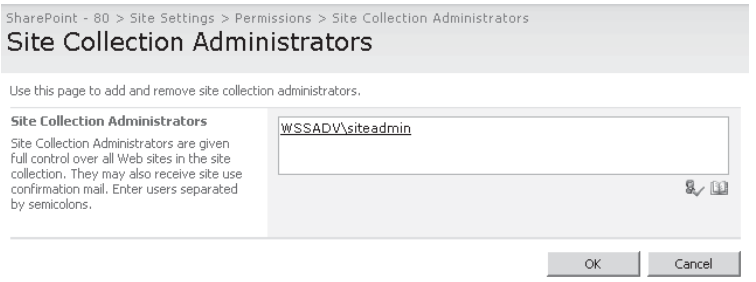
1. Open the Site Administration from the home page.
2. Under the Users and Permissions group, click People and Groups. This will open the By Group list.
3. On the left-hand navigation menu, click Site Permissions to open the Permissions page.
4. Using the Settings link, select Permission Levels. This will open the Permission Levels page:



5. From here, you can add, edit, and delete any permission level.

Site Collection Administrators

- **When you need it** You can use this option to add or remove site administrators.



This option allows you to add or remove site collection administrators. Administrators have full control of the site, pages, and content within it as well as the ability to add and remove users, modify the site template, and so on. When this page opens, any existing (assigned) accounts are shown. Simply add or remove from this list and click OK to save changes.

Advanced Permissions

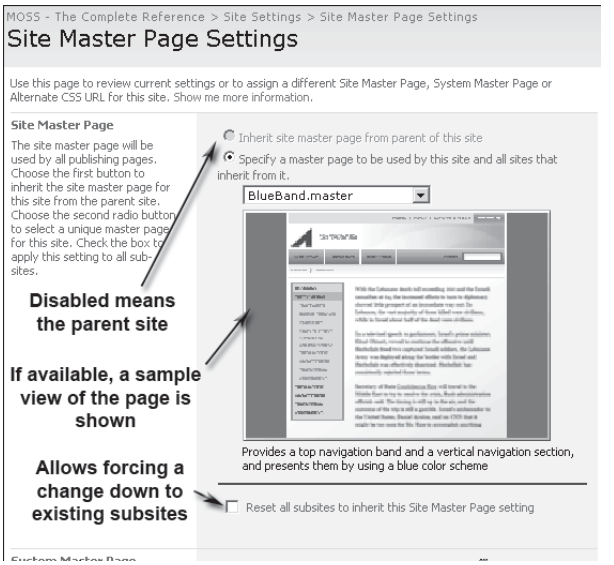
- **When you need it** You can use this option to access site permissions and levels.

This option is a direct link to the Permissions page also accessible from the People and Group page. You saw an example of this usage previously.

Look and Feel Group

This section provides the settings for the way the site looks via which Master Page it uses, the home (welcome) page displayed, and so on. This section also provides access to the page layouts and management of the site navigation displayed.

Master Page (MOSS)



- **When you need it** You can use this option to set Site Master page template, System Master page template, and specify Cascading Style Sheet (CSS) to use for display.

This option allows you to set the Master Page (design) to use for the site. Master Pages are somewhat like a theme in that they dictate both design and structure, but they are really the backbone of SharePoint content management and site structure. They define the layout of the page components and the zones within them. There are two types of pages: typical site pages, which use the Site Master Page as their template, and those for form/view pages, which use the System Master Page. The connection here is simple: for all pages using these masters, you can completely rebrand an entire site in a matter of seconds and even change it on the fly. Custom Master Pages are easily created using SharePoint Designer and Visual Studio.

While on this option, you can choose from the list of installed Master Pages (this list is pulled from the Master Page Gallery, which is covered later under Galleries), and it provides a snapshot view of the page layout (as shown above). From here, you have the option to set the following:

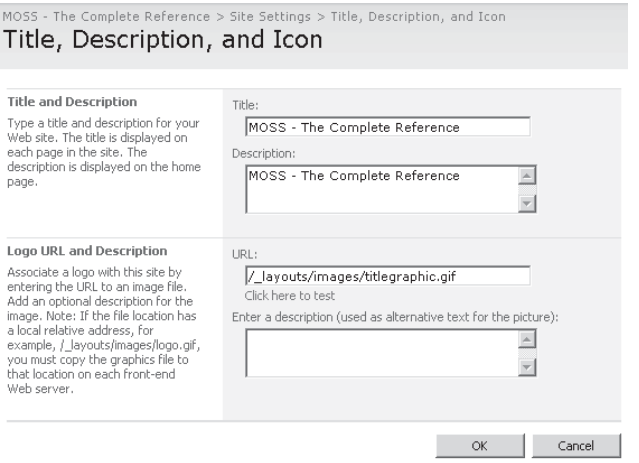
- **Site Master Page** This sets the Master Page layout to be used by the current site and all sites that inherit from it. You can also force any change down to all sub sites by checking the Reset All Sub Sites check box.
- **System Master Page** This sets the Master Page layout for all forms and view pages created in this site. This layout can be inherited from the parent, or you can set a new one (again, this applies to all sub sites). You can also force any change down to all sub-site pages by checking the Reset All Sub Sites check box.
- **Alternate CSS URL** This sets the URL link to the Cascading Style Sheet (CSS) to use with the Site Master and System Master Pages selected. This also has the option of forcing the change down to any sub site pages by clicking the Reset All Sub Sites check box.

NOTE: *When the Inherit <type> Master Page from Parent Options are disabled, you are at the top level (the parent).*

Title, Description, and Icon

- **When you need it** You can use this option to change the title, description, or icon displayed on the site pages.

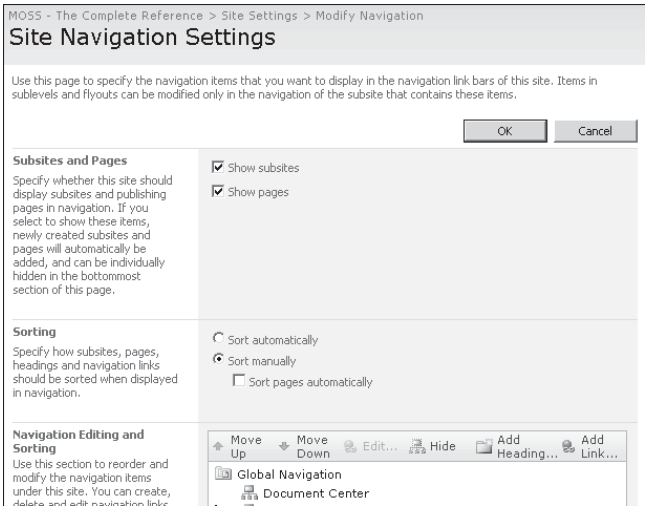
This option allows you to change the title, description, and the icon shown on pages. If you wish to make any changes, you can simply do so and then click OK to save them. One note about specifying the icon file; this must be a relative URL to the image either somewhere on the site or in the file system. The best option is to put the image into a picture library on the site that everyone has access to. Simply upload an image and then use the relative URL to the image in the library. If you use the file system, make sure everyone has read permissions on the file. If you put it in a SharePoint library, make sure the lowest level users (readers in most cases) have access.



Be aware that whenever you reference site content stored in a library, you must always use the Relative URL for an item. Never include http://<server>:<port>, because this causes the user to go outside of the site and then back in—the typical result being prompted to log in repeatedly.

Navigation (MOSS)

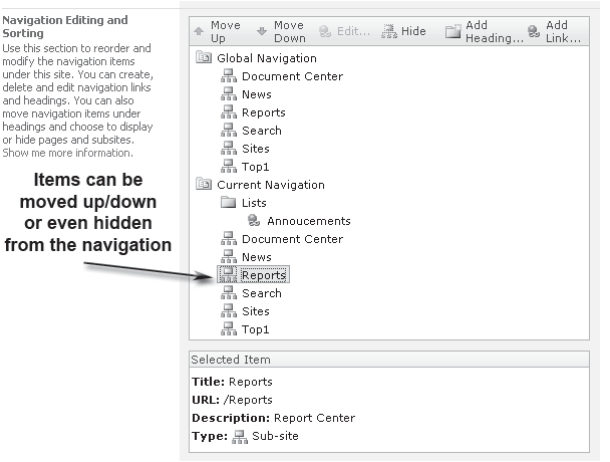
- **When you need it** You can use this option to modify the site navigation.



This option allows you to specify what to include and the order in which sites are displayed within a Portal Site. The navigation options are as follows:

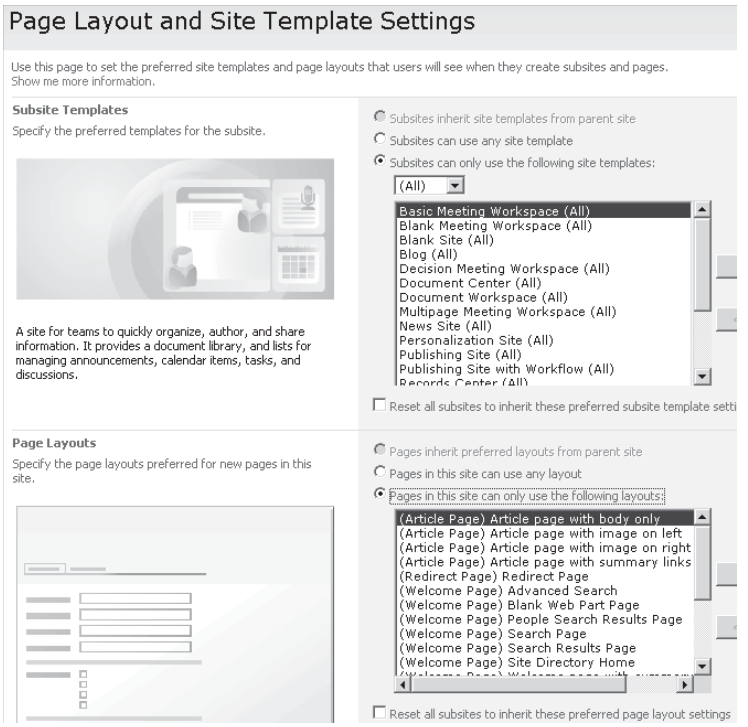
- **Sub Sites and Pages** This includes/excludes sub sites and/or pages in the navigation.

- **Sorting** This sets the sort style for the sites/pages.
- **Navigation Editing and Sorting** This allows you to manipulate how the navigation is organized:



Page Layouts and Site Templates (MOSS)

- **When you need it** You can use this option to choose the preferred site templates and layouts to use in a site.

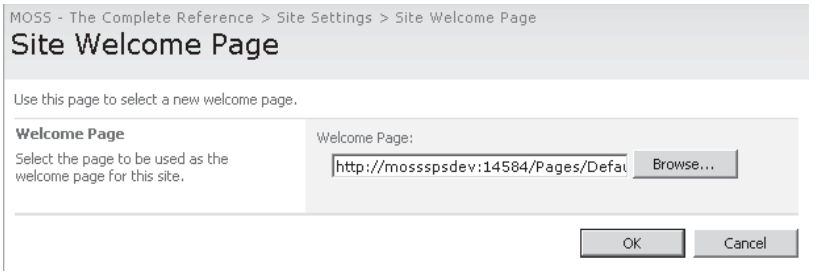


This option sets how sub sites will support site and page layouts. These options can be changed at any time and include:

- Sub sites
  - **Sub Sites Inherit from Parent Site** This means that the sub sites will have the same site type as the parent. If disabled, this is the parent.
  - **Sub Sites Can Use Any Template** This means any available template can be used when creating a sub site.
  - **Sub Sites Can Only Use the Following Templates** This provides a selection list to set a specific set of site templates that can be used (as shown); note that the site templates available depend on your installation.
  - **Reset All Sub Sites** If checked, this forces all sub sites to adopt the same settings (resetting any that are different).
- Pages
  - **Pages Inherit from Parent Site** This means that the pages (in sub sites) must use same page layout as the parent; if disabled, this is the parent.
  - **Pages Can Use Any Layout** This means any available page layout can be used when creating a new page.
  - **Pages Can Only Use the Following Layouts** This provides a selection list to set a specific set of page layouts that can be used (available).
  - **Reset All Pages** If checked, this forces all pages to adopt the same settings.

Welcome Page (MOSS)

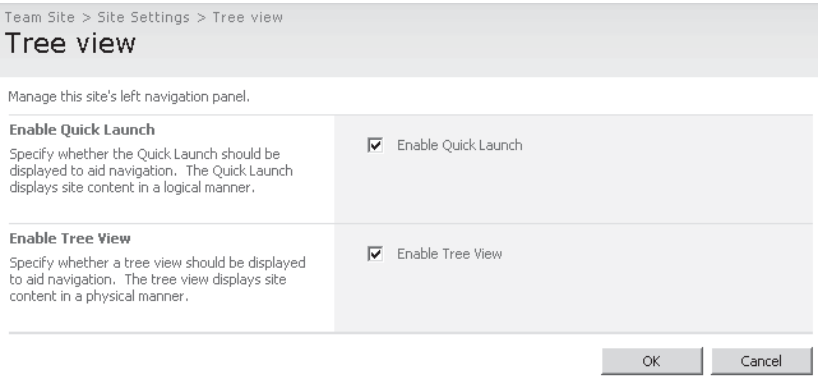
- **When you need it** You can use this option to change the Welcome Page displayed for the site.



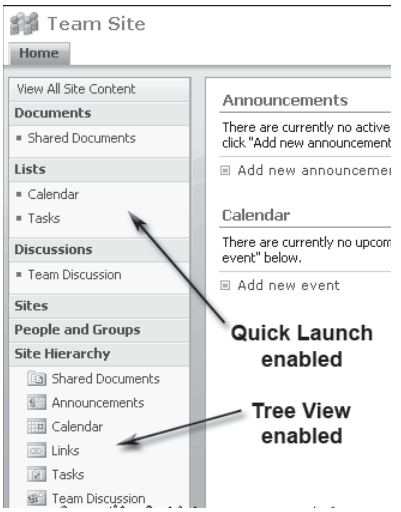
This option sets the page that is displayed when the site is first opened, which is initially set to the default.aspx page in the Pages library for MOSS. This can point to any page/name within the site.

Tree View

- **When you need it** You can use this option to enable/disable Quick Launch and/or Tree View navigation.



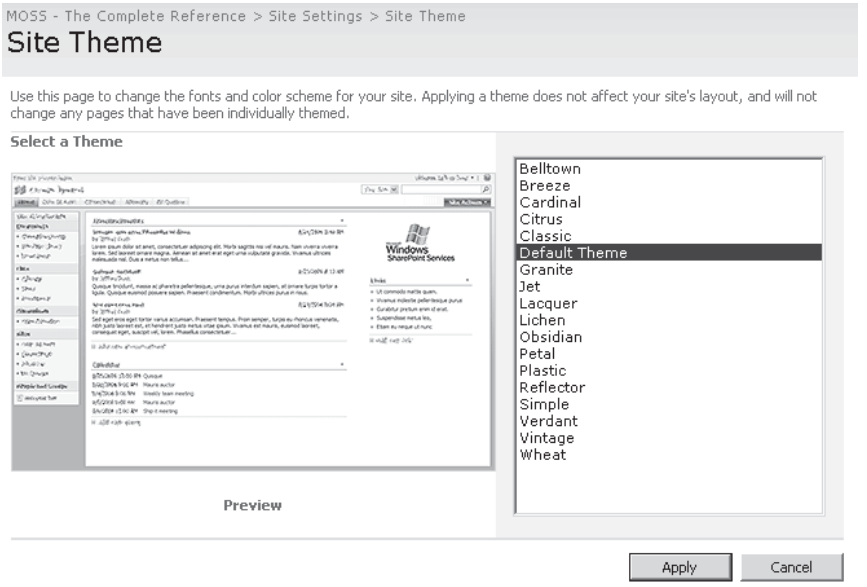
This option enables or disables the Quick Launch and/or Tree View navigation options in a site (unlike the old 2003 Portal version, these do work in MOSS portal sites). If either are enabled, they provide navigation to content and sub sites:



NOTE *Quick Launch is enabled by default. Individual lists and libraries can be included or not included here using the list settings.*

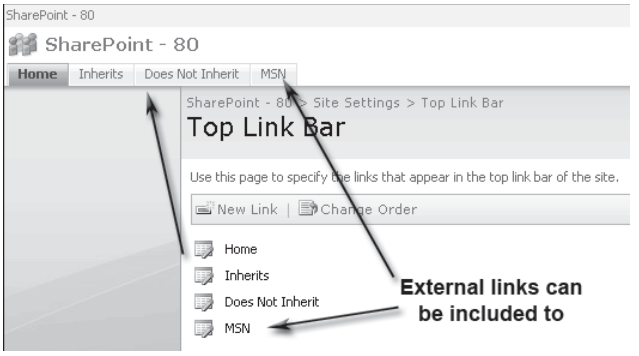
Site Theme

- **When you need it** You can use this option to change the theme (style) of a site.



This option allows you to select from a list of pre-installed themes to stylize your site to suit your taste. Themes can be changed at any time and do not affect the site structure or pages in any way; they simply apply a modified Cascading Style Sheet (CSS) to the site definition of the current site. (Sub sites are not affected either.) Note that changes to the Master Pages and use of an alternate CSS may override these settings.

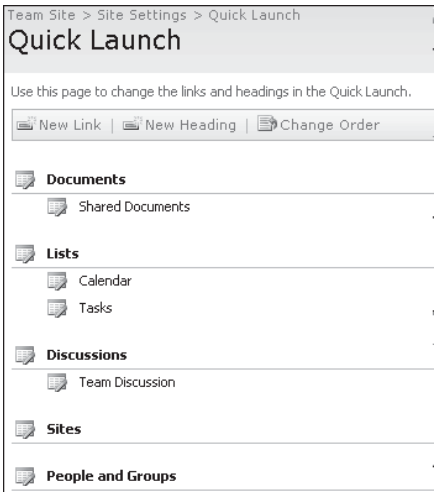
Top Link Bar (WSS)



- **When you need it** You can use this option to modify navigation and links in a WSS site.

This option enables you to modify the order and add or delete link items from the top navigation tab bar shown on the site. Be aware that you can even delete the Home link, effectively removing it from navigation (making only sub sites available). Links are not limited and can point to any valid URL, including third-party sites, SharePoint sites, pages, and event content such as documents and images. To add a new link, click New Link and then enter the URL and name you want to use for the link (new links are always added to the end of the current list). To change the order, click Change Order and then arrange using the drop-down lists provided. To delete a link, click the Edit icon next to the name to expose the Delete button.

Quick Launch (WSS)



- **When you need it** You can use this option to modify the Quick Launch bar properties and display.

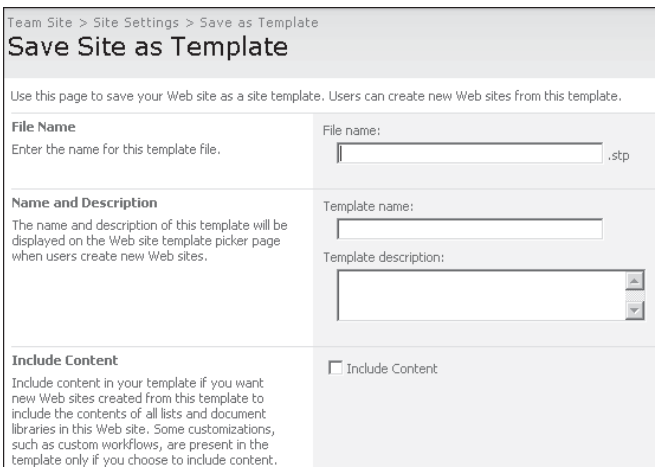
This option allows you to modify the categories that appear in the Quick Launch navigation and the lists/libraries that appear under each. This is basically the equivalent of the Navigation option in MOSS—though as you can see, it's not very sophisticated. Like the Top Link Bar, you can add links to any valid URL, including sites, pages, content, documents, and so on. To create a new link, simply click New Link and then specify a URL and link name.

To create a new heading, click New Heading and then enter the URL and the name. The URL for the default headings found here point to the List view page and passes with it the List or Library type. It's included (automatically) under that heading if the list is set to be included on the Quick Launch bar. The URL for lists is /_layouts/viewlsts.aspx?BaseType=*n* where *n* is the number of the list type. For example, BaseType=1 indicates document libraries. You can manually add your own heading (not pointing to a list type), but it must have a URL to use. If you don't have a particular page to go to here, use / to represent the root of the site. When the user clicks the Heading link on the page, it will simply refresh the home page.

Finally, you can use Change Order to rearrange the order of headings and items within headings. Note, however, that this order can be changed by the list itself. If a list is removed from the Quick Launch and then added back (via the List Settings), it will appear at the end of the list.

Save Site as Template (WSS)

- **When you need it** You can use this option to save a site format, including content as a reusable site template.

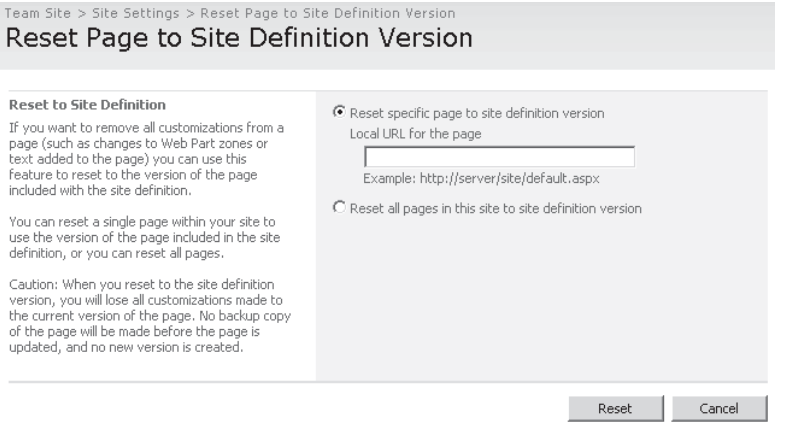


This option allows you to make a new site template from the current site. This includes everything out of the box, including lists, libraries, Quick Launch settings, page layout, theme, and even content—anything that can be packaged into a site template that you can reuse over and over. When user-based site templates are saved to the Site Template Gallery, they can be selected during the site creation process. This enables two very cool features: first, you can set a layout that you like and reuse it for consistency without multiple formats for people to figure out, and second, you can include content (documents, announcements, links, and so on) with it. For example, setting up a project site that has all of the necessary starting documents means not having to upload these each time a new site is created.

From this page, you simply define the template name (this is a file name), the title to use, and a description of what it is used for (the title and description you supply is what you see in the Site Gallery and when selecting the template, so these should be very descriptive). When ready, click OK to save it. If successful, the site template (.stp) file will be saved to the Site Template Gallery (in the Site Collection site), and then you'll be shown a link page that allows you to return to the site or to go to the Site Template Gallery. Unless the gallery is set to require approval on new items, the site template should be immediately available.

Reset to Site Definition

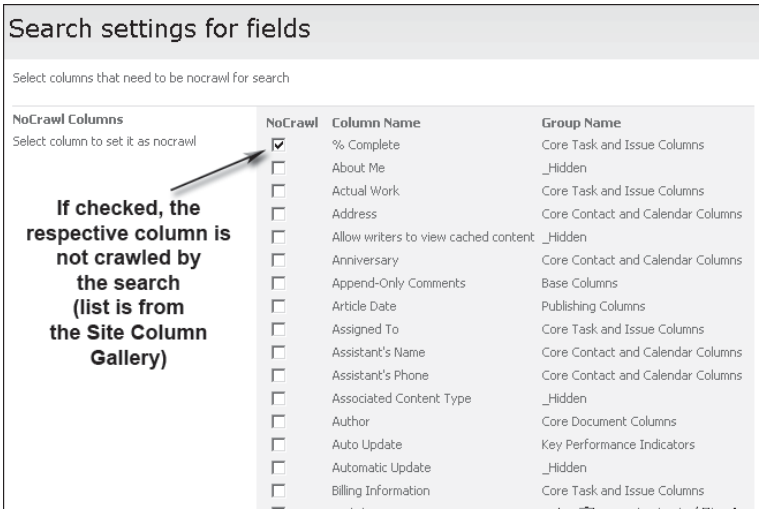
- **When you need it** You can use this option to restore a site or site page to its original definition.



This option enables you to restore an individual site page or all site pages to the original page definition(s) used when the site was created. Any modifications to page layouts, web parts, and text that has been added will be lost and the page(s) reverted to their original state.

Searchable Columns (MOSS)

- **When you need it** You can use this option to modify site columns that are searchable.



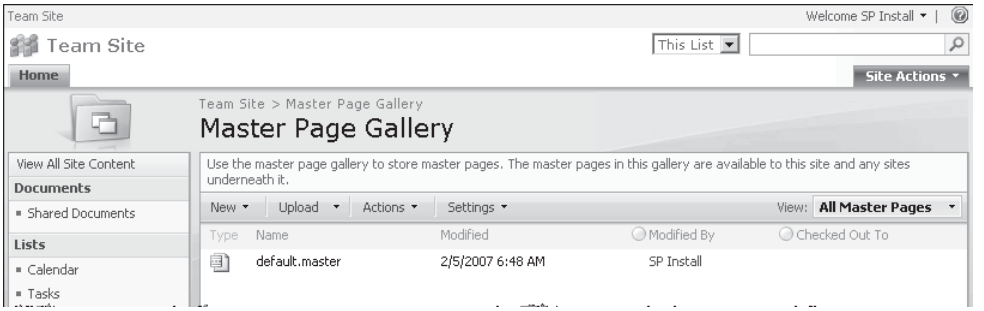
This option enables you to select the site columns that you want to be included (all are included by default) or excluded from the search crawler. This enables you to tweak the search to eliminate custom noise words (in English, words like "the" and "and" are noise in searches, whereas in a business, common acronyms are often of little use in searches). To exclude any column, click the check box next to the name, and then click the OK button. Be aware that this setting applies to the Site Collection search, including the parent site and all sub sites.

Galleries Group

This section provides access to the various galleries used to store Master Pages, page layouts, site columns, and content types used in content management as well as list/library templates and web parts available.

Master Pages (WSS)

- **When you need it** You can use this option to upload or delete site Master Pages.



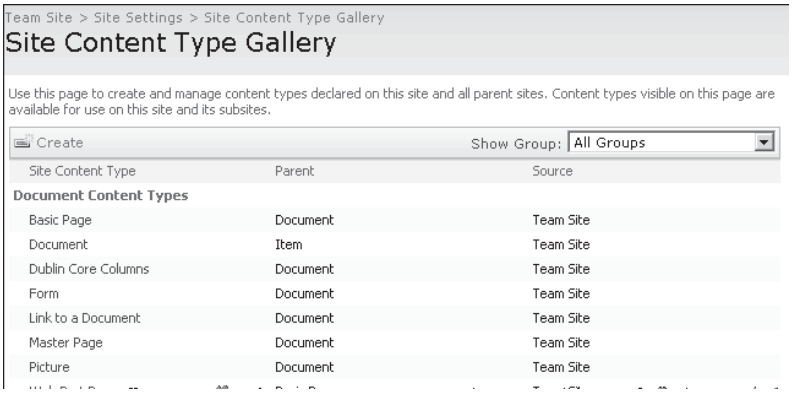
This option allows you to upload or delete Master Pages for the current site. By default, there is only one (default.master—this should not be deleted). When you have custom built Master Pages (built in Visual Studio or Designer), you can upload them here to make them available in the site.

Site Content Types

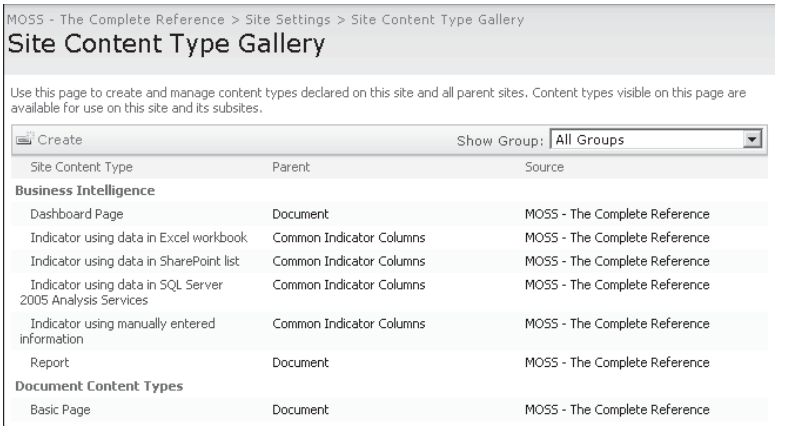
- **When you need it** You can use this option to add, modify, or delete site content types.

This provides access to the site content types within the site, which are covered in complete detail in Chapter 14. However, the short description of content types is that they allow you to define content profiles for specific kinds of list and library items. For example, you might decide that all Workorder documents require that you have a Vendor number. By creating a Workorder content type that has a column defined for Vendor number and Name, you can now easily add this content type to a library without having to add these columns manually. For documents, they can also be considered document profiles, with the complete metadata description about a document that goes along with it.

The content types available depend on what is installed. For example, with WSS, the core elements are defined:



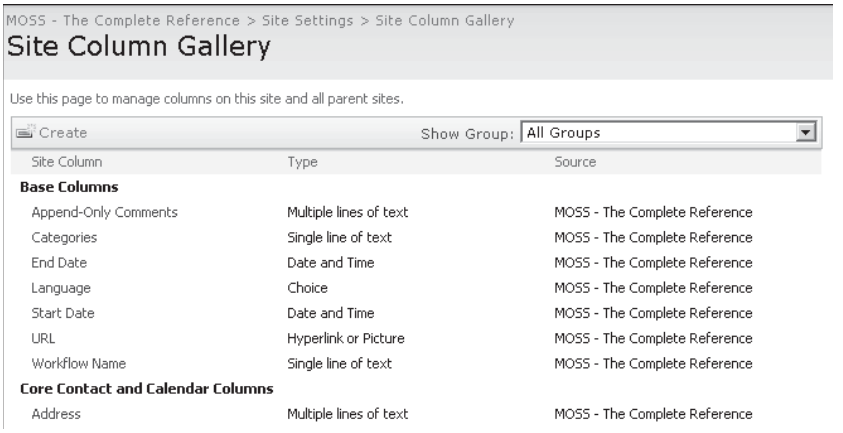
In MOSS, there are several additional ones that are added for the Enterprise-level features, including Business Intelligence as shown (covered in detail in Chapter 14):



Site Columns

- **When you need it** You can use this option to add, modify, or delete site columns.

This provides access to the site columns within the site, which like content types are covered in complete detail in Chapter 14. However, the short description of site columns is that they allow you to define a site-wide column that can be used in lists and libraries anywhere in the site or sub sites. This allows you to create common columns that can be easily modified and, through inheritance, automatically update all of the lists and/or libraries that use them. For example, you could create a Choice Column field called Division and then fill the choices with all of the divisions. Used as a column in a list, the choices are automatically available. If a new division is added, adding the choice to the original site column makes it available in all lists that use it.


MOSS - The Complete Reference > Site Settings > Site Column Gallery

Site Column Gallery

Use this page to manage columns on this site and all parent sites.

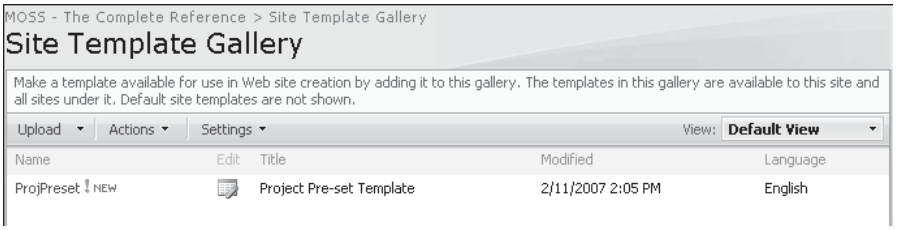
Create | Show Group: All Groups

Site Column	Type	Source
Base Columns		
Append-Only Comments	Multiple lines of text	MOSS - The Complete Reference
Categories	Single line of text	MOSS - The Complete Reference
End Date	Date and Time	MOSS - The Complete Reference
Language	Choice	MOSS - The Complete Reference
Start Date	Date and Time	MOSS - The Complete Reference
URL	Hyperlink or Picture	MOSS - The Complete Reference
Workflow Name	Single line of text	MOSS - The Complete Reference
Core Contact and Calendar Columns		
Address	Multiple lines of text	MOSS - The Complete Reference

Site columns are also used to create Site Content Types; these are all covered in detail in Chapter 14.

Site Templates

- **When you need it** You can use this option to add or remove site templates from the Site Template Gallery.


MOSS - The Complete Reference > Site Template Gallery

Site Template Gallery

Make a template available for use in Web site creation by adding it to this gallery. The templates in this gallery are available to this site and all sites under it. Default site templates are not shown.

Upload | Actions | Settings | View: Default View

Name	Edit	Title	Modified	Language
ProjPreset ! NEW		Project Pre-set Template	2/11/2007 2:05 PM	English

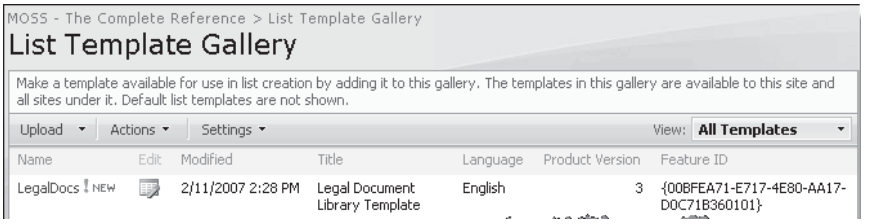
Site templates are site definitions that create a site in a specific layout and include lists, libraries, and/or web parts already defined. All of SharePoint's sites are templates of one form or another and, in fact, use the same single page (the Master Page assigned) as the shell, simply supplying different page layouts, zones, and web parts.

Site templates can be created in a number of ways, for example, using site definitions created in their raw Collaboration Application Markup Language (CAML), which is a mix of XML and HTML coding. Site templates can also be used by SharePoint Designer to create new sites, either based on existing sites or created from scratch. Site templates can also be created by users by simply selecting Save This Site as a Template from the Site Administration page.

Site templates can also include the content (like list items and documents) within them (up to a limit of roughly 2GB), making them ideal for setting up reusable pre-set sites. Custom site templates appear when you create a new site. In addition to the out-of-the-box ones, you will now see a new tab called Custom. These templates are also available in SharePoint Designer.

List Templates

- **When you need it** You can use this option to add or remove list templates from the List Template Gallery.



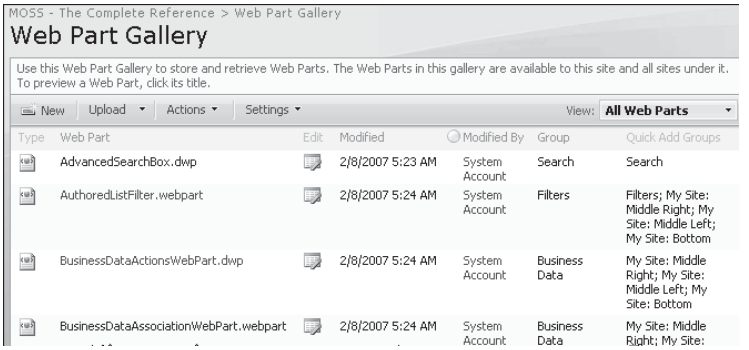
List templates are list definitions that create a list or library with specific columns (and site columns and content types into a template that are available across the sites). Any kind of list within SharePoint, from a Document Library to a Custom list, can be saved as a template, complete with custom ordered columns and custom views. This provides a fast way to reuse common list types that include metadata (columns) and that are common across the enterprise. This facilitates better search, common threads across the information network, and better information management regarding list items and documents.

Like site templates, list templates can be created in a number of ways: in list definitions created using Collaboration Application Markup Language (CAML), by SharePoint Designer, and by users simply selecting this option from the List settings. Also like site templates, list templates can include the content within them (up to a limit of roughly 2GB of storage if attachments or 30,000 items but don't hold me to that). Note that custom list templates appear when you create new content for a site and will appear grouped along with related list and library types.

A good example of where list templates are really useful is with common items like calendars and issue tracking lists, in which additional columns that are specific to the company are useful (such as a drop-down list that includes the building number). Instead of re-creating this list from scratch, you can simply create a template and re-use it.

Web Parts

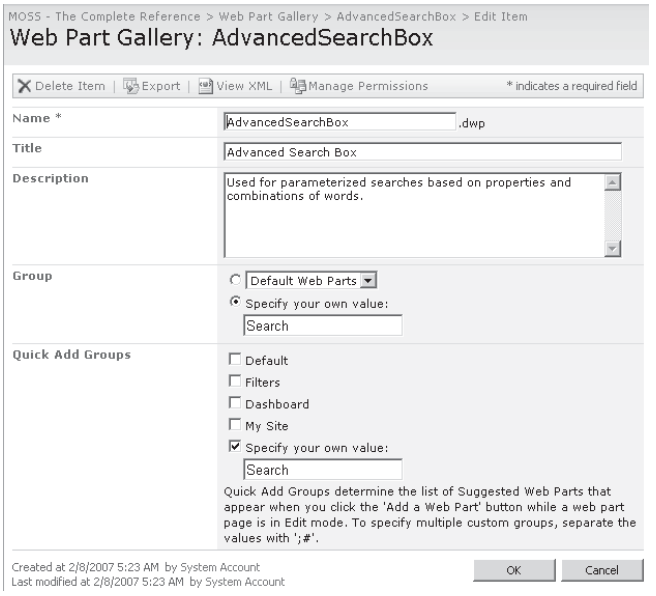
- **When you need it** You can use this option to add or remove web part definitions and preview web parts.



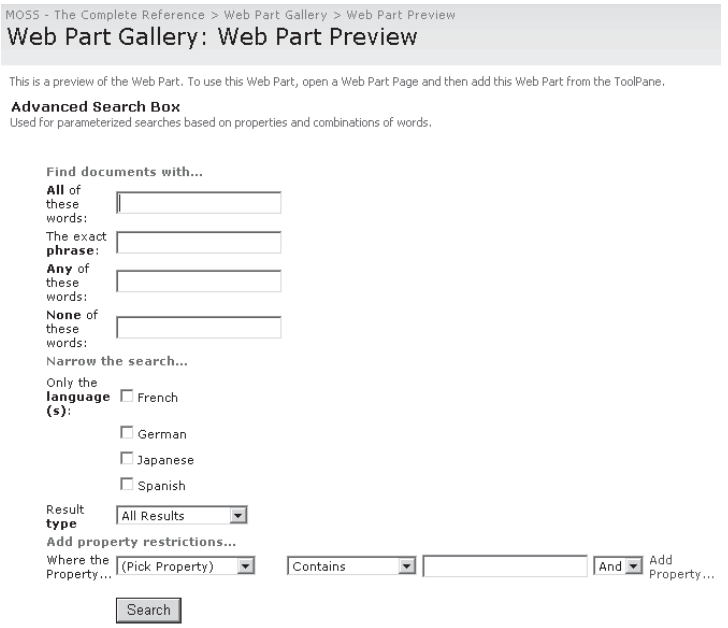
This option provides you with a list of web parts that are currently installed (also known as *solutions*, as you will come to know them soon). These are not the actual web parts but rather the definitions that define the components of the web part solution. This includes references to any web part properties and the actual assembly (DLL) that belongs to it. When you add a web part to a page, the actual reference to the web part in SharePoint is a saved version of this definition. When the page is displayed, SharePoint combines the definition (obtained by a unique GUID for the page and one for the web part on the page) with web part properties (values set for the part) and the assembly to render the web part uniquely on a page. Be aware that this definition is also separate from the solution; while a solution may install a definition, removing (retracting) a solution from a farm or site does not remove this.

Note that you will see both .dwp and .webpart files here; .dwp files are a holdover of web parts from the previous version upgraded to support 2007. These are installed using the STSAdm too. Solutions, the new way of building web parts, use .webpart files as the preferred method. From the list shown, you have the option to edit the web part definition or view the part. To edit the part definition, click on the Edit icon to open the Edit Web Part page.

On this page, you can modify the description, set the group, and quickly add groups under which the part will display (to the end user). To view or test the web part,

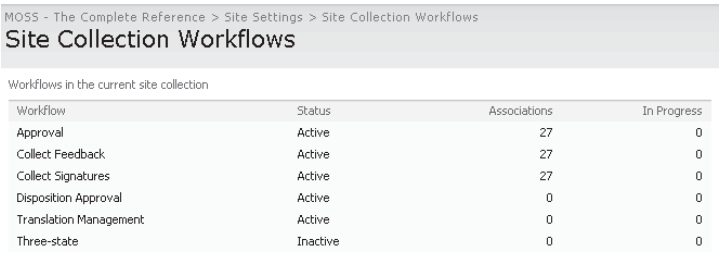


you can often see what the part will look like on the page. (This does not always work, however; many web parts require some sort of pre-setup before they will display.) For example, if you were to click on the Advanced Search web part, you will see what it would look like on a page:



Workflows

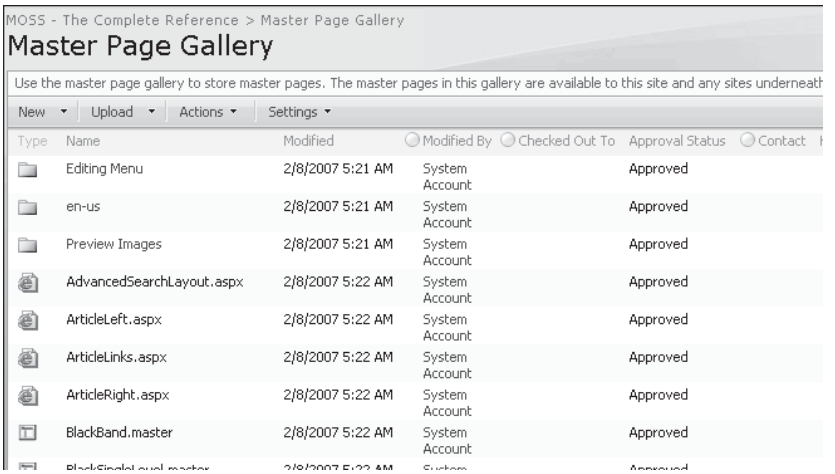
- **When you need it** You can use this option to view the status of current workflows defined in the site.



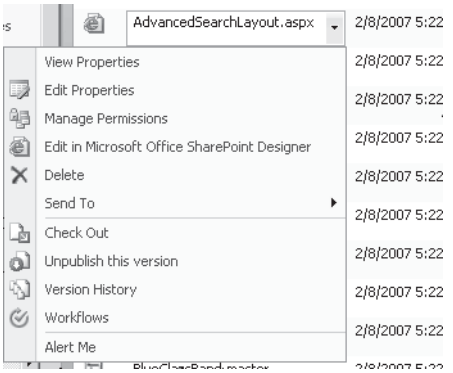
This option provides a status of the current workflows on the site including the status (active or inactive), number of associations (how many lists are referencing them), and the count of each workflow type in progress.

Master Pages and Page Layouts (MOSS)

- **When you need it** You can use this option to add or remove Master Pages and Page Layout Templates.



This option lists all of the Master Pages and Page Layout Templates that are available on the site. From here, custom Master Pages and page layouts created in Visual Studio or SharePoint Designer can be uploaded (or published to the site) and then made available to the site for use either as site or portal wide changes. You have several options for each of these, as shown by the drop-down menu.



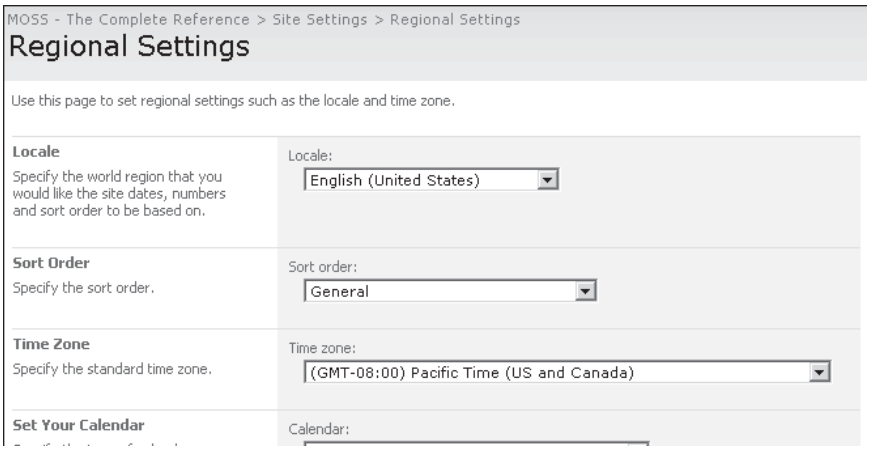
From here, you can have SharePoint Designer edit the page template, check it out, start a workflow, and so on, in the same way you would with any other document. You can also manage the permissions to set who can use the Master Page or Page Layout Template.

Site Administration Group

This section provides the settings that apply to the application general settings, such as setting the region, the ability to enable and report on site usage and statistics, the ability to enable and disable features, and so on.

Regional Settings

- **When you need it** You can use this option to set the regional settings for a Site Collection and sub sites.

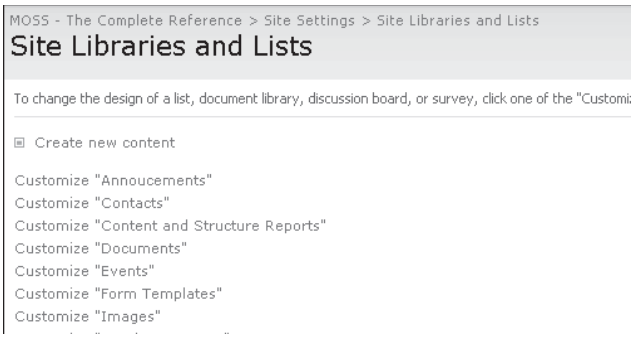


This option sets the regional settings for the Site Collection (and if selected, the sub sites as well), so they include locale, sort order, time zone, and similar information. Note that this does not localize a site (this requires language packs), but sets the proper sort order, date format, and so on, to the region. Options here include (these settings override the default settings in the application):

- **Locale** This is a drop-down selection to select the proper language.
- **Sort Order** This specifies the sort order for items to the proper locale.
- **Time Zone** This sets the time zone (important for the correct date/times on items and documents created and modified).
- **Set Your Calendar** This sets the calendar type appropriate to the locale.
- **Enable An Alternate Calendar** This allows dual calendar features for an alternate calendar type.
- **Define Your Work Week** This sets a normal working schedule for the organization.
- **Time Format** This sets the time format for 12 or 24 (military time) hours.
- **Sub site Settings** This enables or disables sub sites from inheriting the parent settings.

Site Libraries and Lists

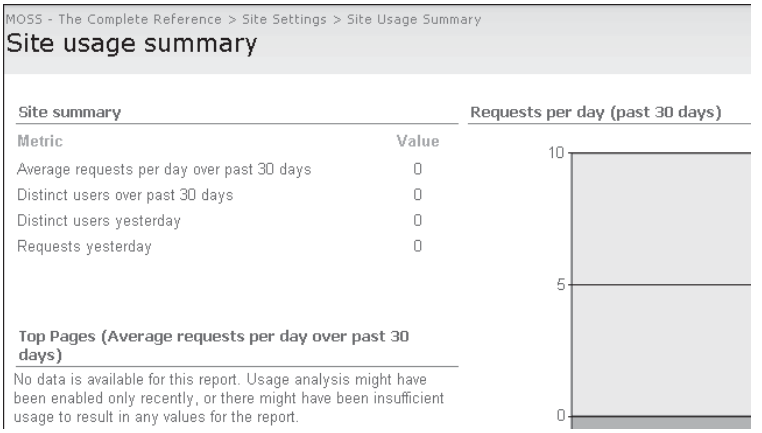
- **When you need it** You can use this option to customize existing site lists and libraries.



This option provides a quick link list for creating new content or customizing existing lists and libraries in the site. The Customize links shown here go to the List settings of the particular list, where you can modify and add and delete columns and views in the list.

Site Usage Reports

- **When you need it** You can use this option to view site usage reports.

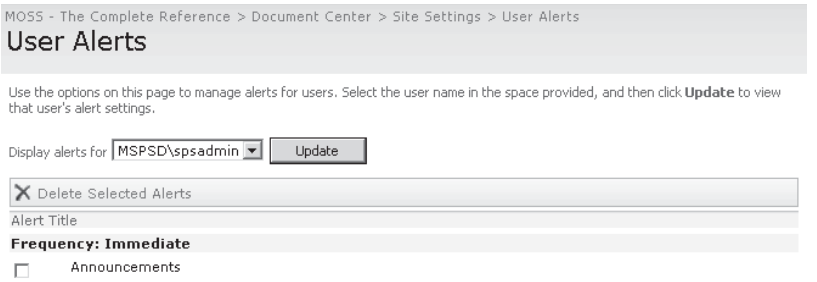


As you can see, this provides several usage analysis reports, which show statistics on site usage, home page hits, and so on. In this case, the report has just started so totals are not available. The report includes the following categories: Site Summary, Top Pages, Top Users, Top Referring Pages, Top Referring Host, and Top Destinations. These are probably the most typical tracking metrics for most public sites (including collaborative ones for determining what is being used). As usage information increases, you can return here to see the statistics, though you should note that statistics don't mean much until you've had more than 30 days worth of logging. As you can see by the partial message shown previously, messages may appear underneath a reporting category that indicate no data is available for this report. This is perfectly normal in most cases and simply indicates that not enough data has been collected to generate the reporting metrics. Simply wait for the next cycle to check the report again. As I mentioned previously, less than 30 days worth of logging is hard to measure. You may need to adjust the Usage Analysis schedule to cover more time in SharePoint 3.0 Central Administration | Operations | Usage Analysis Processing.

There is also the chance that when you open this page, you see the message that states that usage logging and processing must be enabled to view usage reports. This means that one or more services need to be checked and, if necessary, enabled. The first is the usage analysis for the farm (or server). To set/check this, open SharePoint 3.0 Central Administration, and then click the Operations link. Under the Logging and Reporting Group, click the Usage Analysis Processing link. On the Settings page, make sure Enable Logging and Enable Usage Analysis Processing are enabled. Next, go to Shared Services Administration (use the link on the Central Admin home page), and then for each Shared Services Provider open the Shared Services Administration. Next, under Office SharePoint Usage Reporting, click the Usage Reporting link. Make sure that Enable Advanced Usage Analysis Process is checked, as well as Search Query Logging if you are tracking searches.

User Alerts

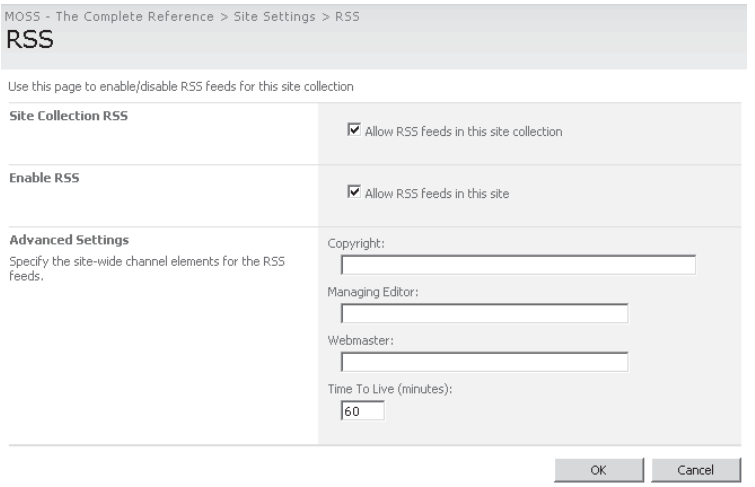
- **When you need it** You can use this option to view and manage user alerts in a site.



This option enables you to view the current alerts that are set for a user (and on what) in a given site. From here, you can only view the general alert type and remove alerts for users. You must be an Administrator to delete user alerts.

RSS

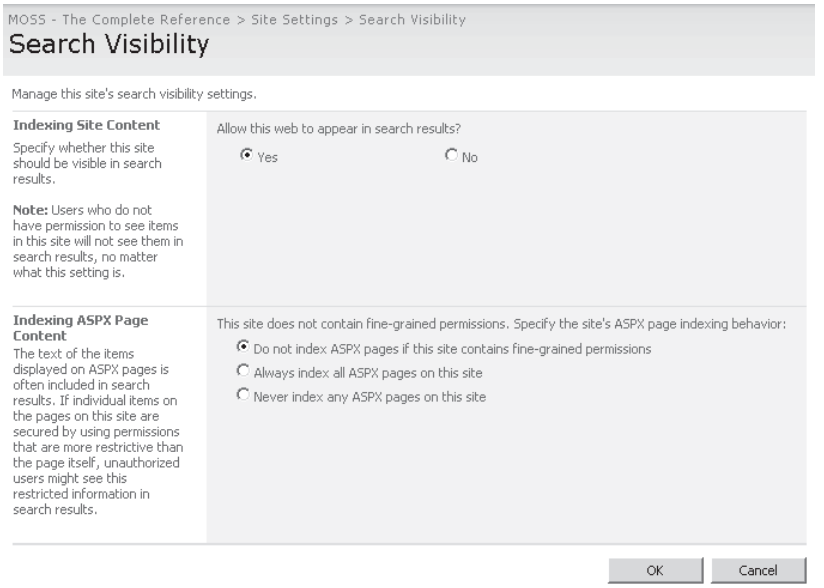
- **When you need it** You can use this option to enable/disable RSS feeds in and out of a site.



This option allows you to enable or disable Really Simple Syndication (RSS) in the Site Collection (also known as RSS feeds out). RSS feeds allow you to set a site-wide copyright notice and contacts for the RSS feeds. RSS feeds are XML files or data feeds that are enabled on SharePoint Lists and are also a common method of producing news-like feeds of information from a variety of sources (visit www.rssfeeds.com). SharePoint lists are based on XML anyway and provide RSS feeds that can be read by other sites in the same way. Thus, instead of simply using your site announcements locally, you can now use your SharePoint list to broadcast them to the web. The copyright information here is optional, but suggested if you intend to publish your RSS feeds to the public.

Search Visibility

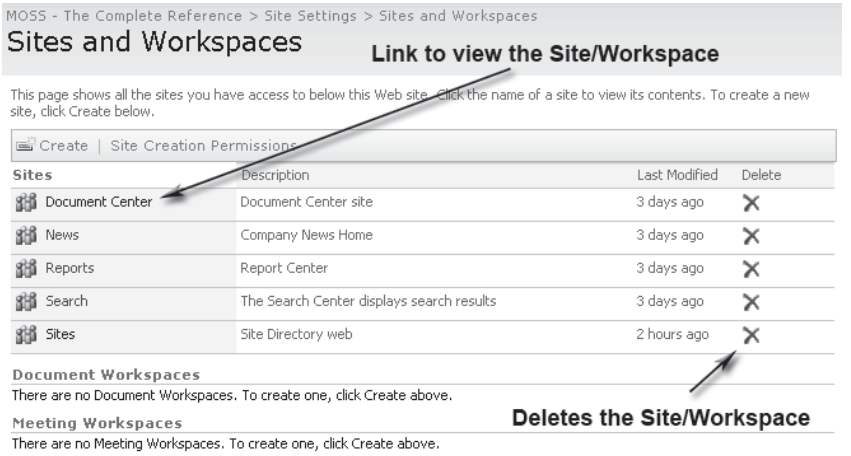
- **When you need it** You can use this option to enable or disable search engine indexing.



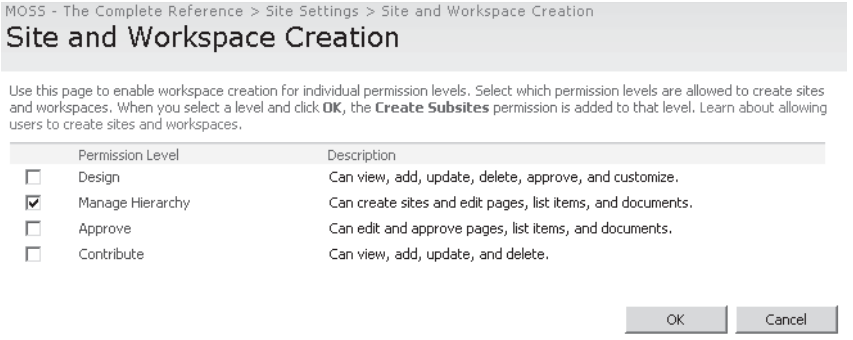
This option enables or disables the search engine from indexing the content in this site and sets up how the ASPX pages will be treated within the site. Enabling and disabling, in effect, is a way of securing the content in this site from being readily found via search. The regular SharePoint security model still applies regardless of these settings; a user must have rights to see an item or document for it to show up for them in search. The settings for ASPX pages should be set to honor not only the page security, but also the web parts on it as indicated (this default is the normal setting). In closed environments such as a publishing site, you might wish to turn ASPX page indexing off, while in a collaborative environment, indexing everything is usually acceptable.

Sites and Workspaces

- **When you need it** You can use this option to view sub sites and workspaces and set site creation rights.

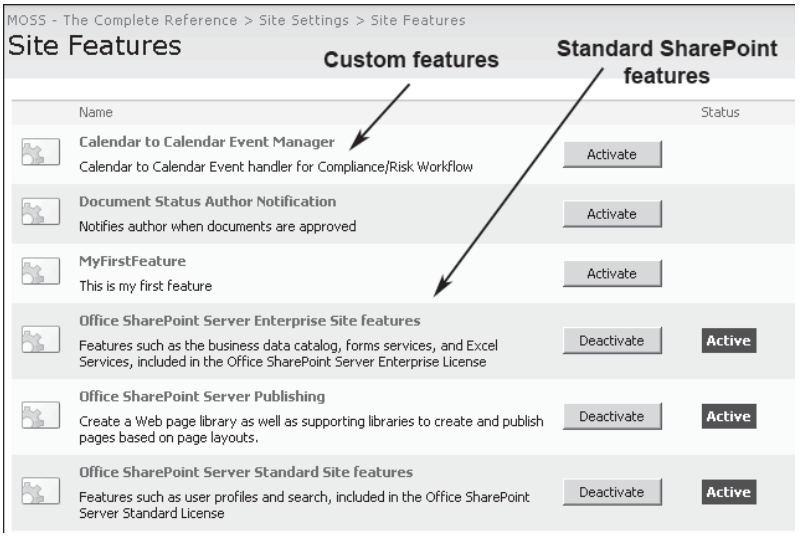


This option enables you to view the current list of sub sites under the current site (in the previous example, this happens to be a Portal Site). From here, you can follow the link to view the site, and you can also delete it. This is the same as going to the site, and then to Site Administration, and then deleting the site—except this doesn't leave you on a dead page. (See the section "Deleting a Site.") This page also provides a link to setting the site creation rights for the site; by clicking the Site Creation Permissions link, you can choose which site group has site creation rights:



Site Features

- **When you need it** You can use this option to activate and deactivate installed features for the site.

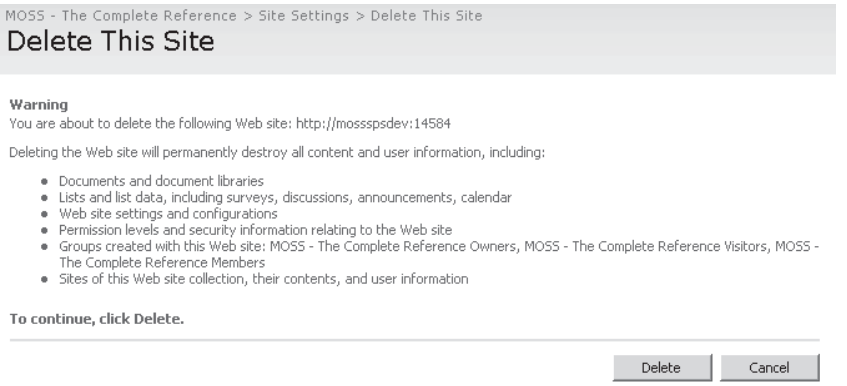


This option allows you to activate and deactivate features within the Site Collection (and subsequently, sub sites beneath it). Features are either those installed with SharePoint (including Enterprise Features) or custom features that include workflow, web parts, and similar. Activation and deactivation is immediate; once active, a feature is available to users, and when deactivated, it cannot be accessed. Note that you cannot install or delete (retract) features here, because that is done using the STSAdm command line.

Delete This Site

- **When you need it** You can use this option to delete the current site.

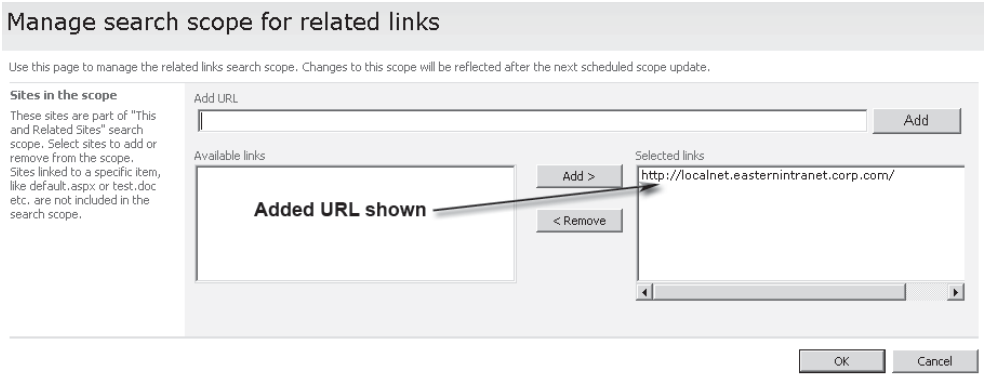
This option, as it implies, allows you to delete the current site you are in. Before the delete actually happens, you are presented with the warning, as shown here:



As it clearly states, all content will be *kaput!* At present, there is no recycle bin for sites, so have a backup before deleting and *be sure* you are at the right level! Once you've confirmed the delete, SharePoint will return you to a simple page that announces "Site Has Been Deleted." Oddly enough, there are no links on this page to return anywhere. To get back to the site, you must reenter the URL into the browser.

Related Links Scope Settings (MOSS)

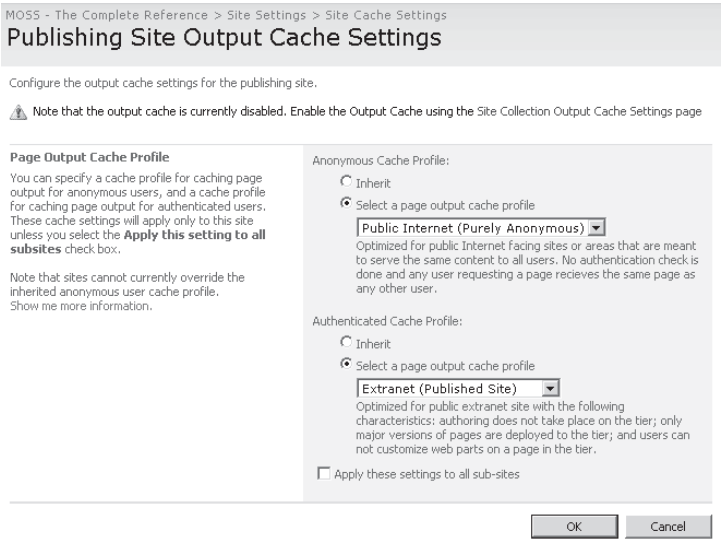
- **When you need it** You can use this option to manage the scope for searching-related links.



This option allows you to add and remove URLs that are part (or not part of) the This and Related Sites search-scope for the site. As noted, sites linked to a specific item (like a page or a document) are not indexed, but general URLs are.

Site Output Cache (MOSS)

- **When you need it** You can use this option to set the policies for page caching in a publishing site.

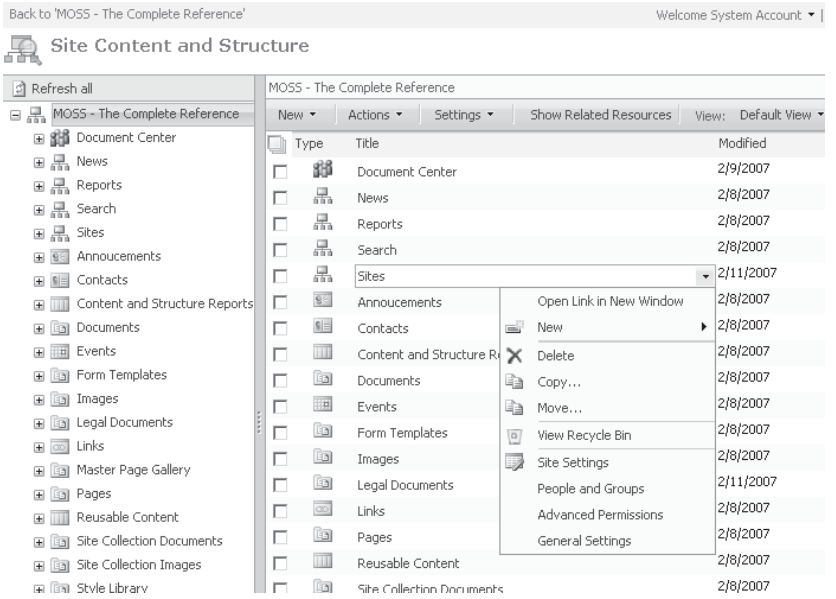


The option allows you to adjust the cache settings on a Publishing Site Output Cache; options here are set for both Anonymous and Authenticated users:

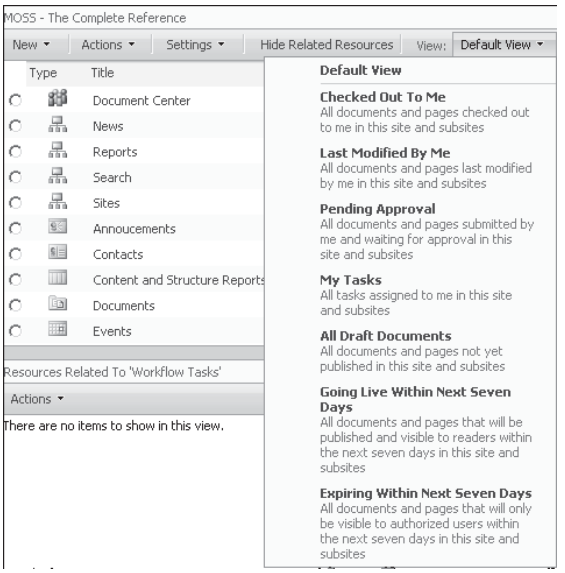
- **Inherit** This takes the parent site settings.
- **Select a Page Output Cache Profile** This provides a drop-down list for selecting the cache policies (limits on how much, how long).
- **Apply These Settings to All SUB sites** This will apply the cache policies to all sub sites.

Content and Structure (MOSS)

- **When you need it** You can use this option to manage (add, remove) content and sites in the current site.



This option allows you to manipulate the sites and content within the current site and provides several options. As shown previously, you can manipulate settings for each area or site. If you check the views, you'll see that you have access to the status of sites and lists:

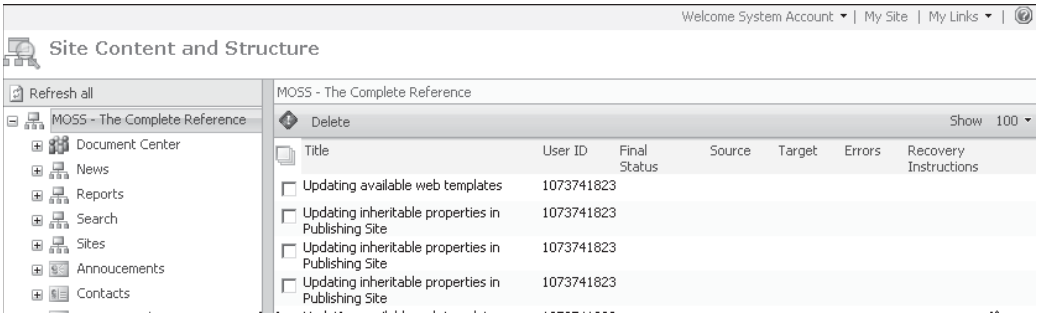


Additional options include:

- **New** This allows you to create a new list, page, or site.
- **Actions** This allows you to copy, move, or delete a site, page, or list.
- **Show/Hide Related Resources** This shows or hides related resources that are required for a specific site or page (for example, an external assembly).

Content and Structure Logs (MOSS)

- **When you need it** You can use this option to view the modification logs to site content and structure.



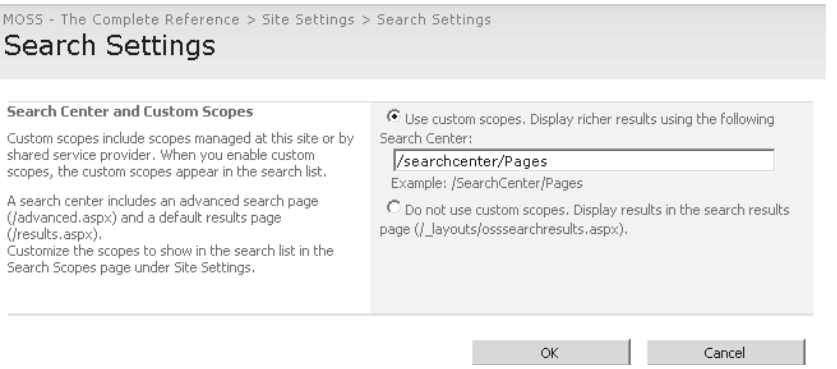
This option provides a view of the logs that track site content and structure changes, and this is an internal log that allows you to determine when changes were made and by whom. This option also gives you a way to find a restore point in the event of a major problem.

Site Collection Administration Group

This section provides access to the settings that apply to a specific site collection within an application. These include search settings, search scopes, site usage, and auditing reports and settings to control site cache operations.

Search Settings (MOSS)

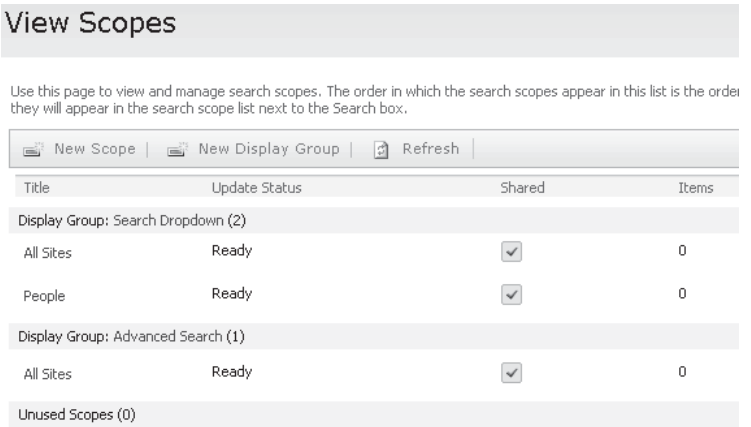
- **When you need it** You can use this option to enable or disable the use of custom search scopes.



This option allows you to enable or disable the use of custom scopes for search. A non-scope search provides only the bare minimum search with the default scope of "All Search Scopes." Typically, custom search scopes are only disabled for public-facing sites where segmentation doesn't make sense.

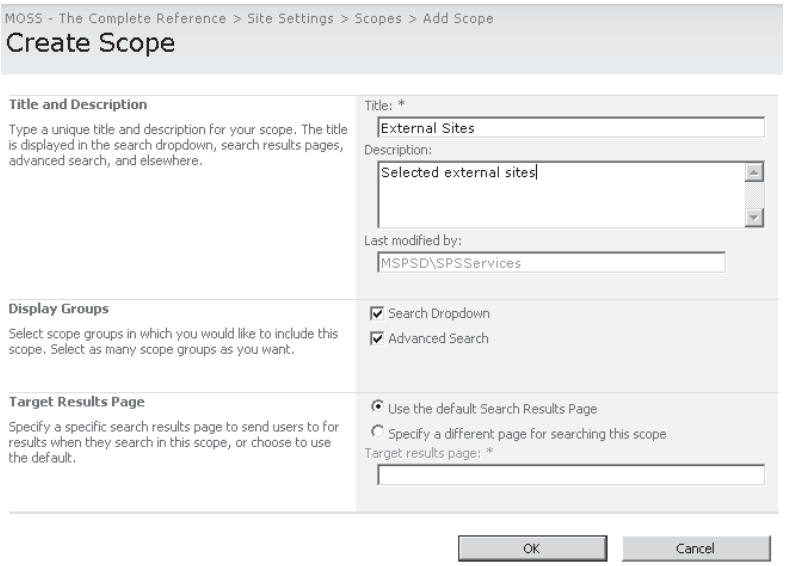
Search Scopes (MOSS)

- **When you need it** You can use this option to add or remove search scopes.

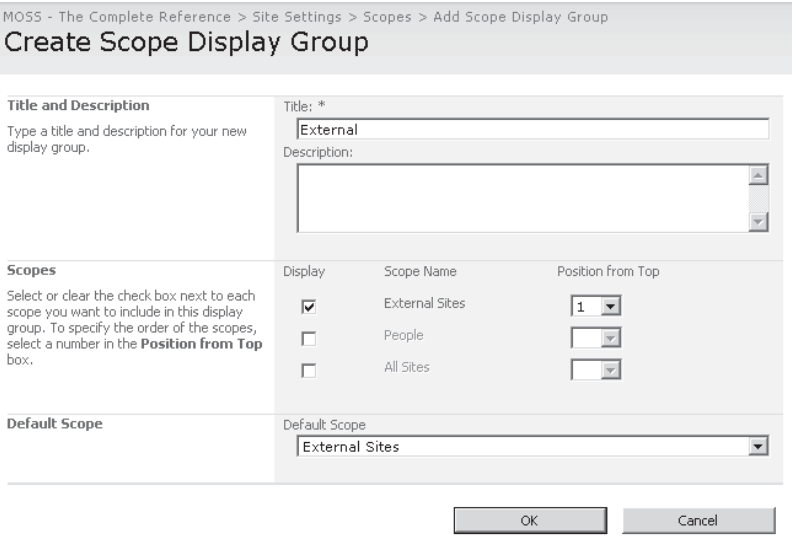


This option lets you view, modify, and add new search scopes to use in searches. Search scopes are a way to set a specific filter on what you are searching for in order to help isolate

it for better results. For example, trying to find a person by searching on all aspects might be fine, but you will get everything related to the person in a general search. If you are just looking for their e-mail address, a scope set up to search only user profiles is much more efficient. The same is true for external uses like vendor sites, technical sites, and so on. To create a new scope, click the New Scope link to open the Create Scope page:



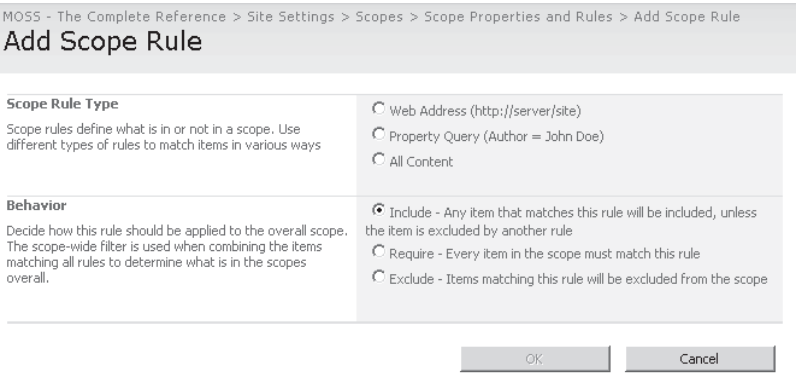
You can set the name, and also, you can direct the search to a custom search page if desired. Once a scope has been created, you can also create a new category for it (optional). From the View Scopes page, click New Display Group:



As you can see in the previous illustration, I've assigned the new scope created to this group (and I've assigned it to the two others as well). When you return to the View Scopes page, you'll see that while the scope has been created (and appears in the new Display Group), there is also an Add Rule link:



This link opens up the Add Rule page, which has several options depending on what is selected:



This gives you the rule selections:

- **All Content** This means all content without restrictions; no other options are enabled.
  - **Web Address** This is either a web folder (URL), SharePoint site, host name, or a domain/sub-domain name.
  - **Property Query** This is a limit to a specific property: one of Author, contentclass, Site, and SiteName, by default.

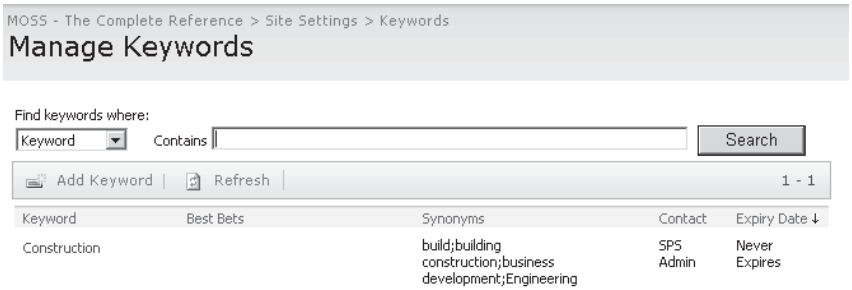
Any property can be used here. To add one, open SharePoint 3.0 Central Administration, and then, on the left navigation bar, click the name of the Shared Service Provider for the application of the current site. In the Search group, click Search Settings, and then on the Configuration page under Crawl Settings, click Metadata Property Mappings. When the list of properties is displayed, mouse over the property you want to include (for example, AboutMe) and select Edit/Map Property**.** When the Property page displays, scroll to the bottom, check Allow This Property to Be Used in Scopes, and click OK.

- **Behavior** Includes the following:
  - **Include** This means that any item that matches the rule is included unless some other rule excludes it.
  - **Require** This means that all items must match the rule to be included.
  - **Exclude** This means that all items that match the rule should be excluded.

Once you have set the rule, click OK to save it. When you return to View Scopes page, you'll see that it is preparing or already compiling. To edit or delete a scope, mouse over the scope name and select either from the drop-down list. Note that editing a scope may require re-indexing.

Search Keywords (MOSS)

- **When you need it** You can use this option to add, edit, delete, and change publishing dates of search keywords.

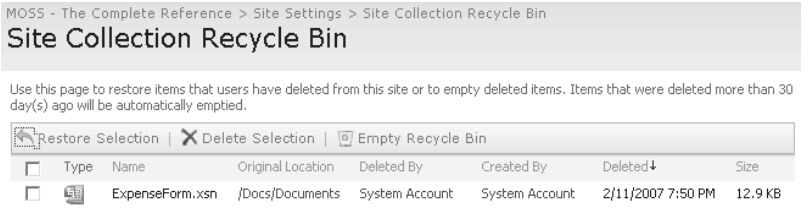


Search keywords provide a way to link keywords with synonyms in order to increase the relevancy of search results returned for users. In short, it's a way to relate something someone might search for to the various words they might use to get there. To add a new keyword, click the New Keyword link. To edit one, mouse over the Keyword Phrase and select Edit. In either case, you have the following options:

- **Keyword Phrase** This is the keyword phrase or word.
- **Synonyms** These are related words users might use to find the content.
- **Best Bests** This is a link that allows you to specify URLs to specific content to be included in results (when this keyword is hit).
- **Keyword definition** This is a simple description of the keyword to appear in the keyword results.
- **Contact** This is a person to contact about this keyword (should it need to change).
- **Publishing** This allows you to set a start, end, and review date for this keyword to be active (blank means don't expire).

Recycle Bin

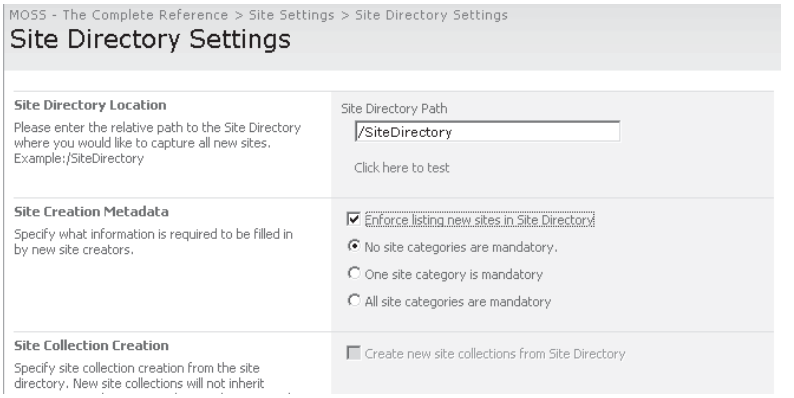
- **When you need it** You can use this option to restore a deleted item or document.



This option allows you to restore content (items/documents) that has been deleted 30 days from the day deleted. Content that is known to not be needed can be purged if necessary and items not recovered are automatically removed permanently.

Site Directory Settings (MOSS)

- **When you need it** You can use this option to specify the settings for the Site Directory.



This option sets the path for any new sub sites that have been added and sets the category rules for being added (must be listed, select one category, or all). Note that if the option is changed, it only affects new sites being created. If the site specified is not a site directory type, changes to this will likely fail.

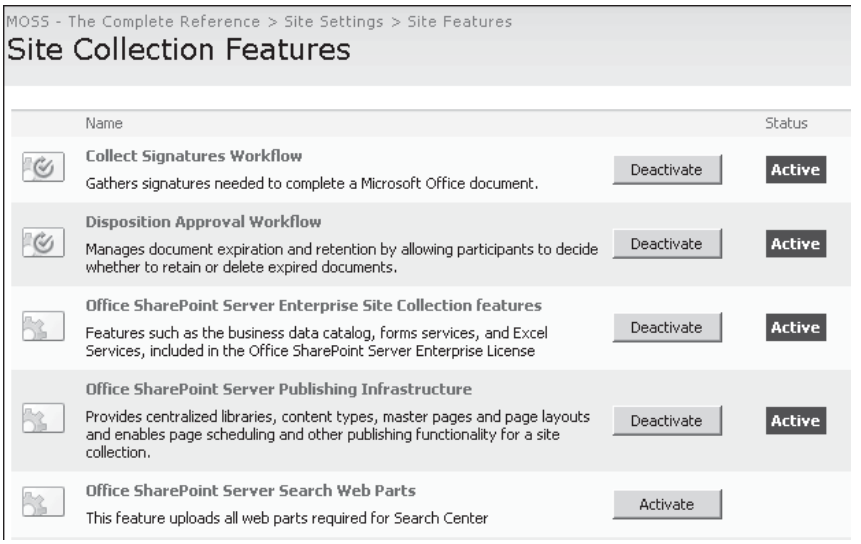
Site Collection Usage Reports (MOSS)

- **When you need it** You can use this option to view usage reports.

This provides usage reports for the Site Collection, which is the same as the site usage reports described previously.

Site Collection Features

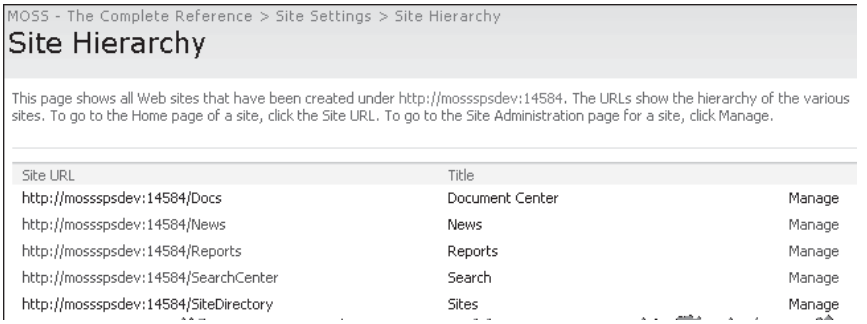
- **When you need it** You can use this option to activate or deactivate site collection features.



This option allows you to activate and deactivate installed features in the Site Collection. Features will include those provided by SharePoint as well as custom features and web parts. Features cannot be added or deleted (retracted) from here; that must be done using the STSAdm command-line tool.

Site Hierarchy

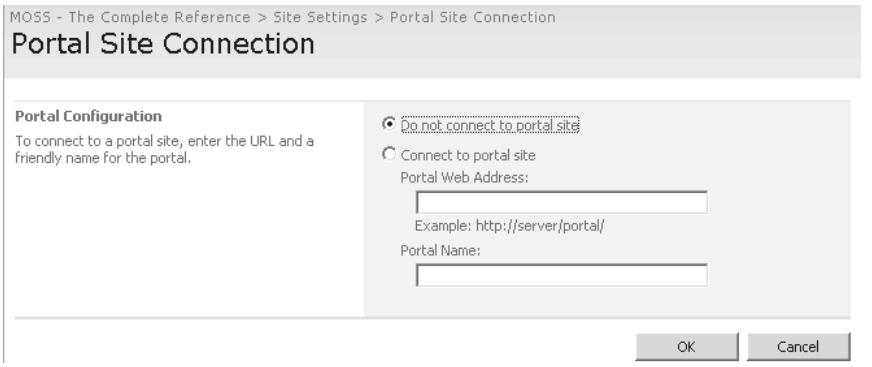
- **When you need it** You can use this option to view and quickly manage sub sites.



This option provides a simple way to navigate to sub sites and to their respective Site Administration pages (via the Manage link).

Portal Site Connection

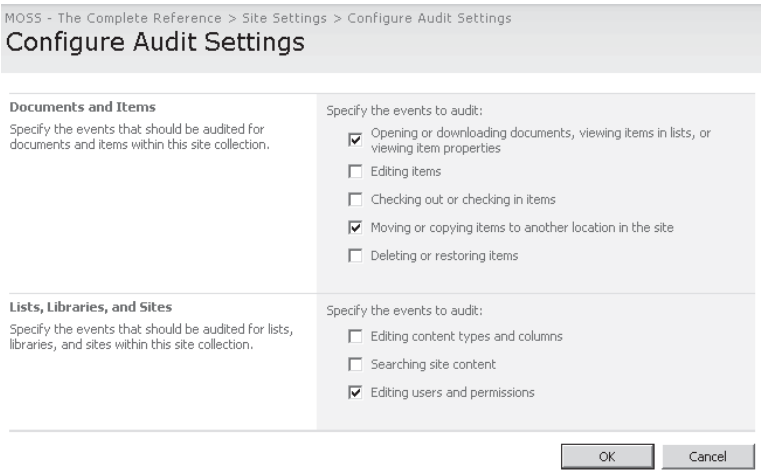
- **When you need it** You can use this option to connect or disconnect from a portal site.



This option allows you to connect or disconnect from a SharePoint Portal (MOSS) site. Connection to a portal provides relative mapping to content and users; this makes the local content available within the search of the portal and gives access to sites using relative URLs. Because actual content between sites is separated, sites can leave a portal at anytime.

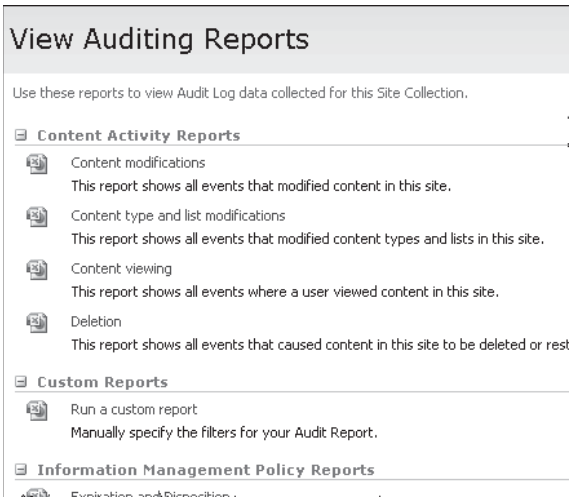
Site Connection Audit Settings (MOSS)

- **When you need it** You can use this option to set SharePoint audit settings.



This option provides the settings for additional audit logging of SharePoint activities. This is often useful when there is a suspected problem or a potential breach of security or authority. All of these options are turned off by default; enabling items here will increase logging substantially. I suggest you use this only when necessary, unless you have a whole lot of memory and disk space.

Audit Log Reports (MOSS)

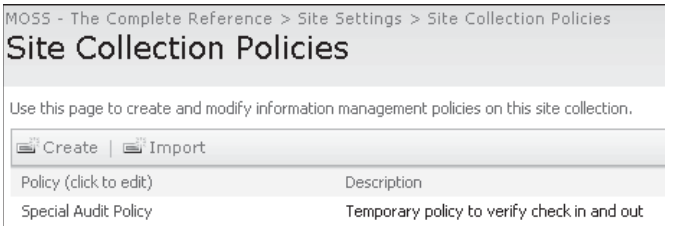


- **When you need it** You can use this option to view auditing reports.

This option allows you to access reports based on the audit logs that are logging the actions specified in Site Connection Audit settings mentioned previously. As you'll notice, there is also a Custom Report option provided, which allows you to generate an ad hoc report based on your selections. Other reports are canned reports that are pre-set. Note that if you are not auditing a particular item needed, reports will return the message "No data for this report."

Site Collection Policies (MOSS)

- **When you need it** You can use this option to add and remove both site audit and label policies and to set expirations.



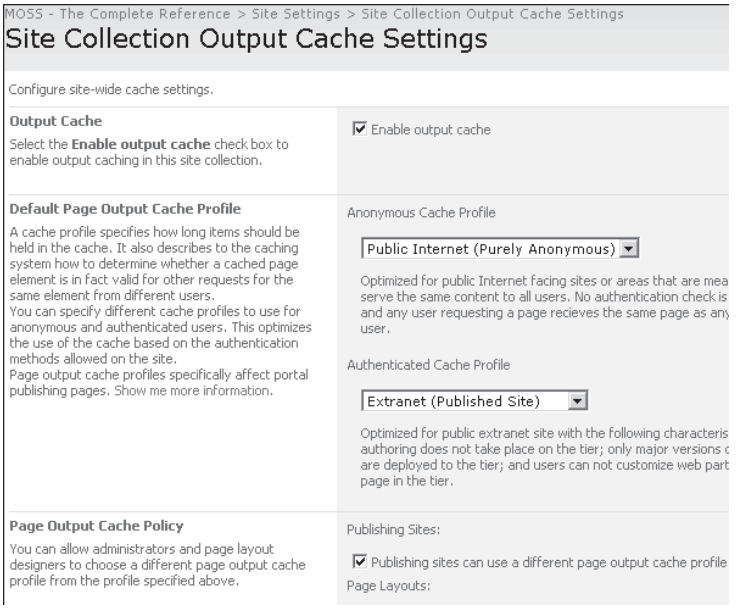
This option allows you to add and remove site collection policies including labels, barcodes, and auditing. When you either create new policies or edit old ones, they have the following options:

- **Name and Administrative Description** This is the name and a description that explains the purpose/use of the policy.
- **Policy Statement** This is a statement displayed to end users when opening items subject to this policy.
- **Labels** These enable or disable labels on documents; labels can automatically prompt the user for input or can be set to automatically use a property or text value (like Confidential). Documents are automatically tagged with them when uploaded.
- **Auditing** This either enables or disables auditing. If it is enabled, you can audit specific events, such as uploading/viewing, editing, checking in/out, moving or copying, and/or deleting or restoring.

- **Expiration** This enables an expiration date on this policy and determines the actions to take when the item expires (this can even initiate a workflow).
- **Barcodes** This enables or disables barcodes and can prompt the user to insert a barcode before saving or printing.

Site Collection Output Cache (MOSS)

- **When you need it** You can use this option to set the Site Collection user and page output cache policies.

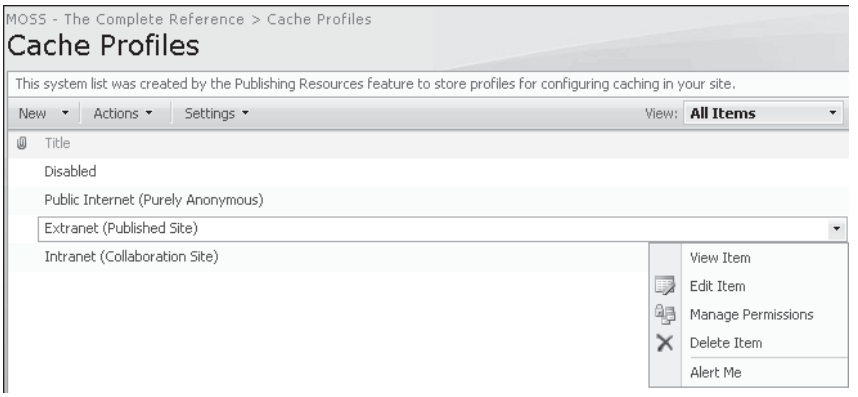


This option enables you to set the output cache settings for the Site Collection and to set profiles to optimize performance. Options here include:

- **Output Cache** This enables or disables the cache (in development this can be off).
- **Default Page Output Cache Profile** This sets the profiles (*type of site use profile* is a better term) to use for both Anonymous and Authenticated users.
- **Page Output Cache Policy** This sets the policy for publishing sites:
  - Enables or disables publishing sites using different page output cache profiles (different from the site).
  - Enables/disables page layouts to use a different page output cache profile (overriding the site).
- **Debug Cache Information** This enables or disables debug cache information.

Site Collection Cache Profiles (MOSS)

- **When you need it** You can use this option to add, modify, or remove site collection cache profiles.

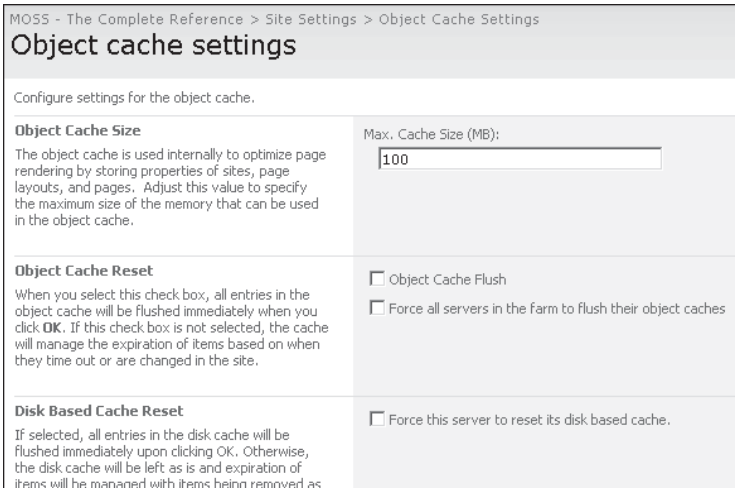


This option allows you to create or edit a profile and modify its permissions. Creating a new profile is done by clicking the New | Profile Output Cache link. To edit, mouse over the name (you can set permissions from this menu as well). Creating or editing a Cache Profile includes the following options:

- **Title** This is the title of the policy.
- **Display Name** This is the descriptive name of the policy.
- **Display Description** This is used to populate the list of available cache profiles.
- **Perform ACL Check** This enables or disables checking of Access Control List (ACL) file security; in all but public environments, this should be disabled to provide better performance.
- **Enabled** This enables or disables caching.
- **Duration** This is the time in seconds needed to keep the cached version available before overwriting.
- **Check for changes** This enables or disables checking content changes on page refresh.
- **Vary by Custom Parameter** This allows you to set a custom parameter used to identify specific customers; refer to .NET documentation on HttpCachePolicy. SetVaryByCustom.
- **Vary by HTTP Header** Refer to .NET documentation on HttpCachePolicy. VaryByHeaders on the browser.
- **Vary by Query String Parameters** This allows variation by the query string; refer to .NET Documentation on HttpCachePolicy.VaryByParams.
- **Vary by User Rights** This enables or disables checking that the user has identical rights on every item on the page.
- **Cacheability** This is set to NoCache, Private, Server, ServerAndNoCache, Public, ServerAndPrivate; refer to .NET documentation on HttpCacheability.
- **Safe for Authenticated Use** This enables or disables authenticated access by administrators and page designers.
- **Allow Writers to View Cached Content** This enables or disables allowing users without edit permissions to have their pages cached.

Site Collection Object Cache (MOSS)

- **When you need it** You can use this option to modify the settings of the Site Collection object cache.

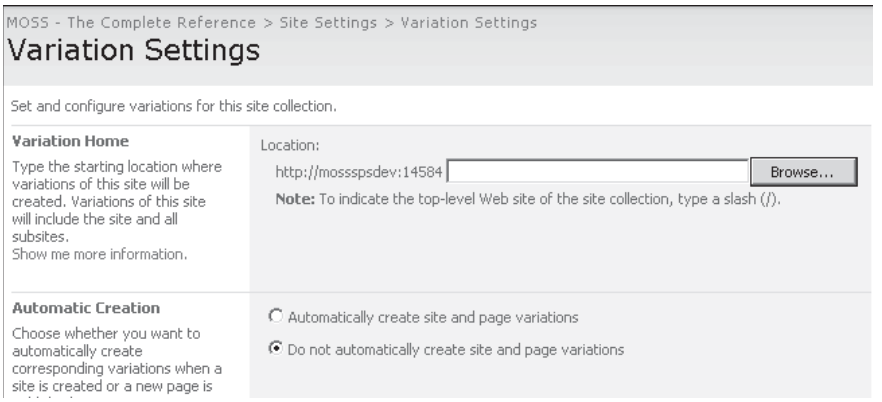


This option allows you to adjust the object cache settings for the site. Objects in this case include site definitions, page layouts, and pages. The more available object cache usually increases performance. Options here include the following:

- **Object Cache Size** This is the amount of space in megabytes to use for caching (the default is 100MB; use more if you can).
- **Object Cache Reset** This enables or disables:
  - **Object Cache Flush** If enabled, it immediately flushes the cache when you click OK on the page.
  - **Force All Servers in Farm to Flush Their Object Caches** This forces a reset on all other servers (particularly important if you have made a significant page change!); it will do this (if enabled) when you click OK on this page.
- **Disk Based Cache Reset** This enables or disables Forcing the Server to Reset the Disk-based Cache.
- **Cross List Query Cache Changes** This enables setting either immediate checking for changes in cross-list queries (found in cache) or setting a time (in seconds) to let the query remain in the cache.
- **Cross Lists Query Results Multiplier** This sets the multiplier into use when calculating cache refresh for cross list queries. A higher number is needed for sites with unique security between them (like 10), a lower number (such as 3) is better for sites that have similar security.

Variations (MOSS)

- **When you need it** You can use this option to create site variations (including localization, language, and mobile).



This option allows you to define variations (or version), which can handle publishing sites to customers using different languages. Through this option, you can define a series of site variations, setting one as primary. Changes to this primary site can be automatically or manually propagated (copied) to target variations. Since this process can require outside services like a translator, this propagation can initiate a workflow. Variation settings has the following options:

- **Variation Home** This sets the starting location where variations of the site will be created.
- **Automatic Creation** This enables or disables automatically creating site and page variations.
- **Re-create Deleted Target Page** This enables or disables re-creating a page if it is deleted but later republished.
- **Update Target Page Web Parts** This enables or disables carrying web part changes over to variations.
- **Notification** This enables or disables sending an e-mail to owners when a new site or page has been published.
- **Resources** This sets where to get resources for the variations (either used from the existing page (the source) or copied to the variation (duplicated).

Variation Labels (MOSS)

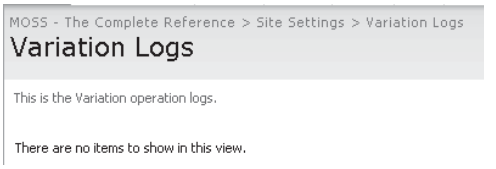
- **When you need it** You can use this option to add or remove variation labels.



This option allows you to create variation labels to be used on variations of the site. The primary pages list those already created; to create a new one, click the New Label link. When the New Label page opens, you can specify the following options:

- **Label Name** This is the name of the label.
- **Description** This is a detailed description of what the label is for.
- **Display Name** This is the display name (what the end user sees for the name so it should be localized).
- **Locale** This is the locale for the variation label.
- **Hierarchy Creation** This sets the creation hierarchy of Sites and Pages, Only Publishing Sites, or Root Site Only (note the hierarchy *cannot* be changed once saved).
- **Source Variation** This sets the source and the publishing template to use.

Variation Logs (MOSS)

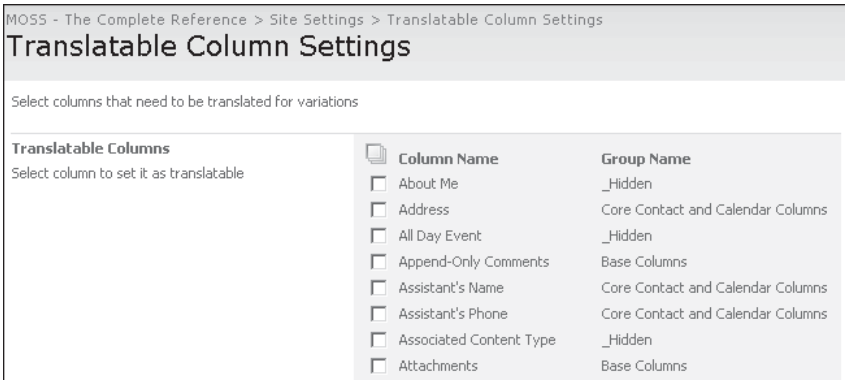


- **When you need it** You can use this option to view variation publishing logs.

This option allows you to view logs generated by the variation publishing process. If no logs are available, this will be indicated as previously.

Translatable Columns (MOSS)

- **When you need it** You can use this option to set columns to be translatable.



This option allows you to pick columns that are translatable for variations. As you can see, this includes most of the core content and user profile information. Check the desired columns and click OK to save.

CHAPTER 9

Exploring the SharePoint User Interface

The SharePoint User Interface is the heart of what users know as Windows SharePoint Services and represents the application side of the house. From a user's perspective, this *is* the SharePoint product and the only part of the technology they see (or will see). Out of the box, SharePoint "the Application" provides many out-of-the-box user features, such as the ability to create sites, create and design pages, utilize web parts, make lists, and so on. Always a key element in the product, the User Interface (UI) has always been intended to be simple to use and focused on enabling users to manage and handle their own affairs without needing any special technical knowledge.

The fact that SharePoint has matured from an application into a technology will probably be lost on most users, but in reality, WSS and MOSS are two of the best examples of applications built using the Windows SharePoint Services Platform. The light version (WSS) provides most of the core features (sites, lists, and so on), whereas the full version (MOSS) simply adds additional features (more site types, enterprise search, the Business Data Catalog, Forms Services, and more).

SharePoint for Users

The primary management of SharePoint is handled with the SharePoint 3.0 Central Administration, which handles creating applications and top-level sites: the part of SharePoint that is for the limited Farm Administrator community. Administrators at this level control the more fundamental elements, such as search scheduling, managing system resources, and so on. Once things are set up, however, Central Administration plays a very small role in the scheme of the operation.

For the user community, Windows SharePoint Services (the light version) begins with a top-level site created by an Administrator. From here they have full access (if they have the rights) to the main features of SharePoint sites. With full access, users can:

- Create/delete sub sites
- Create/delete Lists and Libraries

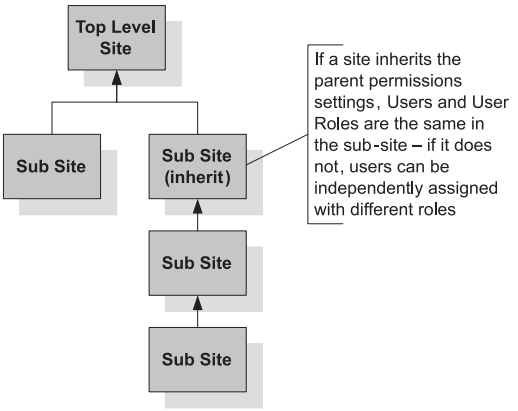
- Create/design/delete Content Pages
- Create/design/delete Web Part Pages
- Manage Site Users
- Search sites (locally)
- Create and use List and Library Workflow
- Use the standard set of web parts (covered next)

MOSS is essentially an application built on top of WSS and it can be used to create a variety of sites from WSS team/meeting sites to Portal and Publishing (intranet/internet) sites (these site types are the same under MOSS and WSS). Portal and Publishing sites enable multiple sites to be anchored to a central site, providing common (global) features such as navigation, search, and more geared towards multi-user enterprise level functionality. MOSS features include:

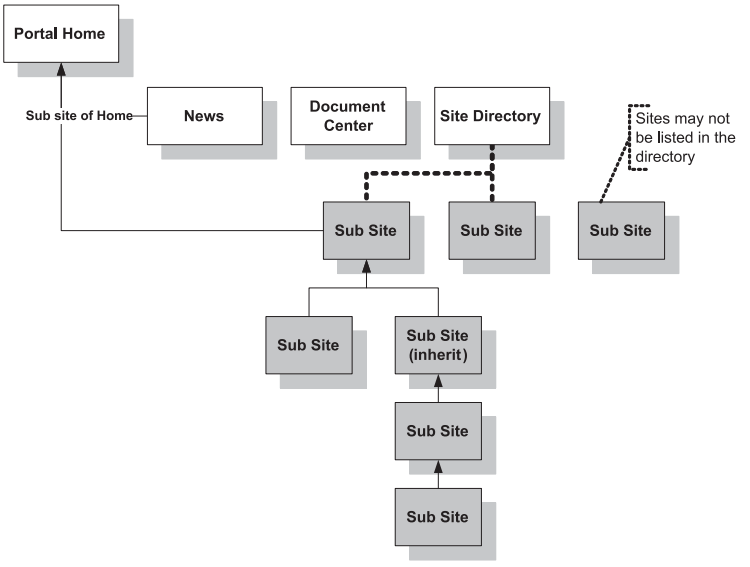
- The capability for creating Portal sites
- Use of the advanced set of web parts (covered next)
- Shared Services that include:
  - User Profiles and My Sites
  - Audiences and Content Targeting
  - Portal and Enterprise wide search (Office Search)
  - Excel Calculation services
  - The Business Data Catalog and Key Performance Indicators (KPIs)
  - Business Intelligence Dashboards
  - InfoPath Forms services
  - Enterprise Content Management
  - Additional Usage and Auditing Statistics

From a purely functional perspective, users see roughly the same things when using either product since much of the UI is common. In fact, creating and managing content is identical with both products. There are a few structural differences worth noting, however:

- **WSS Sites** These are single page sites that allow you to create pages within them and sub sites below them. These are based on Site Definitions (templates) that prescribe the content (lists/libraries) that are automatically added. Sub sites are loosely connected to the parent site and can either inherit the same permissions (users) or have different ones. Sub sites can also be parents and have sub sites beneath them. WSS Sites cannot share content or search with each other (side to side or with the parent), but can be connected from an information architecture perspective using links.



- **MOSS Portal Sites** These are also based on single page sites but provide an anchor site which connects multiple sub sites. MOSS sites are also based on templates and include several additional enterprise-type sites, such as the Portal style for Collaboration and Publishing and a few Enterprise Site Definitions (the Document Center, Search Center, and so on). Specific to the MOSS Collaboration template is the Site Directory-type site, which provides a directory of sites created within the Portal. Sites created within the Portal are created as sub sites of the root, but depending on settings, may or may not have to be listed. Independent WSS sites remain stand-alone, but can join or leave a portal. Also, their association with a Portal enables their content to be searched and allows them to publish links and content "up" to the Portal (they do not become part of it and maintain separate data).



Where to Start

This is always the toughest thing to begin with if you are new to the product or having to train someone on how to use it. There is a lot to SharePoint's features and nuances to the UI (based on user roles) that make it sort of hard to just say "Start here." You could start with the Site Administration, but in reality, users rarely see that and shouldn't need to (that's also why I covered it under Administration rather than here). I've seen every angle used to explain it but the best approach is to learn from how it is actually built, starting with sites and working down.

To begin with, let's start with an overview of what actually makes up a site within SharePoint. As you likely already know from Administration, sites are based on an Application and a Site Collection (representing a Virtual Directory and a Website in IIS). SharePoint sites are virtual and have no physical foot print—they are made up of Site Definitions, which includes an XML file that defines it as well as an aspx page used to create the page view—the rest is all in the database. Things added to a site (lists, pages, and so on, as explained next) are also virtual, and are also made up of the XML and aspx code used to render them. On the other hand, the definition, page placement, and all of the data they contain is stored in the database.

Site Components

The primary components (except the Application that hosts it) that make up a site are as follows:

- **The Site Collection** This is a core site that does not have a user interface; from a technical perspective, this is an IIS Site created in a Virtual Directory that has no pages in it. Because it is provisioned as a SharePoint site, it is not functional (even if you did put a page in it, it would not render). Users don't know about this site, because it is created using Central Administration; however, they can be impacted anyway, because the settings for this site are applied to all sites created in it. While you won't need this just yet, this is the SPSite object in the SharePoint Object Model.
- **Top-level sites** A top-level site is a base SharePoint Site and is where the User Interface (UI) begins. In WSS, the default site is a single page site and may have a few lists and libraries already created (this depends on template type selected). In MOSS Portal sites (collaboration and publishing), the top-level site is the Home Page with a few sub sites already created in it (News, Document Center, Site Directory, and similar). All sites created in the portal are sub sites of the top site. For reference, these sites are SPWeb objects within the SharePoint Object Model.
- **Sub Sites** A sub site is a child of a parent top-level site that may or may not inherit the permissions (users) of the parent. Not inheriting permissions enables a site to have a unique set of users, but it still belongs to the same collection. The functionality provided is the same as a top-level site and even sub sites can have sub sites. For reference, sub sites are also SPWeb objects and have a ParentWeb property to indicate the site that owns it.
- **Pages** Pages within SharePoint provide the website-like content for SharePoint and come in three flavors: Content Pages, aptly named, simply hold content (HTML, Xml, text, and so on), much the same way any HTML page would; Web Part Pages, which are defined-layout pages providing Web Part Zones for controlling the

list view layout and positioning of web parts (note that there is also a content web part that enables you to put simple content on a web part page); and in MOSS only, there are Publishing Pages, which are intended to be used for publishing content in a portal environment (these even have a special library where they are stored). Pages themselves are composed of a Master Page as the base and then an aspx page that inherits from it and adds SharePoint components and aspx code. These are stored in a SharePoint library within the site and when requested, the page is assembled from all of the components. The content section of the page is then rendered inside the virtual page, like an IFrame.

- **Lists/Libraries** Lists and libraries are the core content of SharePoint and the majority of its data store. Lists come in many forms, from Announcements to Calendars to user defined Custom lists. Lists themselves are made up of Rows (Items) and Fields (Columns); field columns can be defined to hold any kind of data (called Metadata in SharePoint), from a text or choice field to a lookup for data in the site. In a very real way, lists can be considered database tables, which happen to have pre-built front-ends that allow you to add, edit, and delete items within them. You'll find out how cool this concept is when you begin developing around them. While all lists can have file attachments, Libraries are lists intended for storing Documents, Forms, Pictures, and Pages. The only difference between them is that attachments in a library are actually indexed by the search service, whereas normal list attachments are not. The list data is the metadata for SharePoint; for example, a document library may store a document, but by default, it only stores the title (name). List columns added to that library can be used to further describe the document and/or otherwise tag it with extra information (even a bar code can be used). The additional information in the list (not the document) not only makes search more powerful, but it also provides a way to know what a document is without having to open it. For reference, there are two SharePoint objects here: SPList, which represents all lists; and SPDocumentLibrary, which is a superset (inherits) object providing controls specific to files. On a technical note, keep in mind that you access a library using the SPList object, and then recast it as an SPDocumentLibrary. It is worth noting again that everything in SharePoint is a list; even the users in the site are maintained in the background using a SharePoint List.
- **List Views** List views are a funny thing. They don't really exist, except to serve as what you use to display list content on pages. I separate list views out because I've never seen them explained as an entity before, and they really should be explained as such. All lists can have multiple views of the data within them. For example, a URL Link list might contain the link, the name, what it's used for, and so on. To add this list to a SharePoint page to display the content, the user drags the list to the page; SharePoint creates a List View Web Part and renders the information on the page. The List View Web Part then becomes its own entity; the view it provides can change, it can be sized, moved, etc. If a new view is added to the list (say a reduced amount of information or styled differently), the List View Web Part on the page can be changed to use it. What often confuses people is that you can *delete* a List View Web Part, but doing so *does not* delete the underlying list. In addition, you can drag a list or library to a page multiple times and create additional Views, each using the same data but still a separate entity. Again, the point here is that these are Web Parts that provide a view, not the actual list or library. Lastly, you can also close a view on a page or minimize it; this does not delete the view (or have any impact

on the list), it simply hides it. For reference, List Views can be predefined in Site Definitions (for example, the Team Site Template provides a pre-defined set of List Views when the site is created). From the SharePoint Object model, you're dealing with a few things: Views in the List are SPList.Views, while the List View Web Parts are WebPart objects (part of the SPWebPartCollection of the WebPartPage object).

- **Web Parts (Solutions)** Web parts (referred to as Solutions) represent the application functionality added to SharePoint, and can provide any kind of functionality that can be built using the .NET Framework. Originally the only way to extend SharePoint's out-of-the-box functionality, these still play a major role, though they're no longer the only game in town. There's a large set of core web parts provided with WSS and additional ones added with MOSS. As I indicate in the name, in technical terms, Web Parts are called Solutions internally. Web Parts are WebPart objects (part of the SPWebPartCollection of the WebPartPage object).
- **Workflow/Features** Workflow provides the ability to perform a pre-defined process for a list or library item. This includes standard processes like document approval, notification, and so on. Features are custom add-ons to SharePoint, either built custom or purchased. The out-of-the-box functionality provided in both WSS and MOSS are based on features.

Site Templates

All sites within SharePoint are created from Site Templates, which provide the core layout for the different kinds of sites (Team, Workspaces, and so on). These are based on Site Definitions, which are defined in XML and include one or more aspx files. In structure, all sites are identical, but what make them unique are the page layout that is defined and the lists, libraries, and web parts that are included by default. This is even true of the Portal sites, such as Collaboration and Publishing.

It is the use of these templates that enable SharePoint sites to be virtual with no actual physical presence. SharePoint stores everything about the site including lists, libraries, data, and even page layout in the database. Using Global Unique Identifiers (GUIDs), SharePoint uniquely tags everything from a site, to a page, to a view on a page. Storage and settings for lists and web parts is also stored using GUIDs. When you make a request to view a site, SharePoint locates the information in the database using the GUID of the site, uses the Page GUID to get the page, and then uses the various GUIDs to get the data and layout information about web parts and lists. The data is then "married" to the XML definition and aspx files to render the page.

In addition to the base templates provided, users can create their own. They can do this by simply arranging a site and adding the items they want, and then saving the entire site as a template for re-use. Custom template definitions can also be created using SharePoint Designer and Visual Studio.

SharePoint pages are based on Master Pages from the .NET Framework. These enable a single master page to set the navigation, branding, colorization, and shell of the site. Pages within Site Templates inherit from a master page, which then enables the pages so that they are only concerned about functionality—in effect, so that they are "on top of" the functionality provided by the master. A single site can also support multiple master pages to support multiple formats (this is typical in a publishing site). For those familiar with Microsoft Content Management Server (MCMS), this is not a new concept, and it's one of the reasons MCMS was merged into the SharePoint platform.

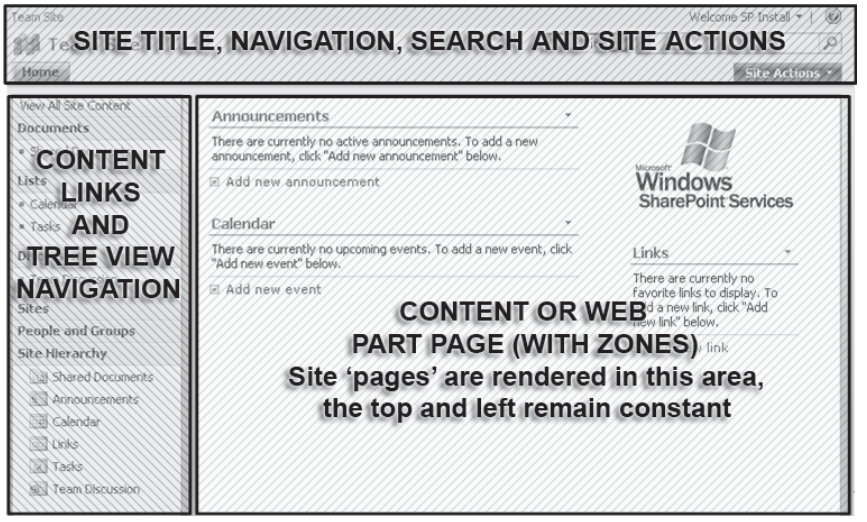
Out of the box, SharePoint provides several templates for both WSS and MOSS; these are covered in detail in Chapter 15. A few important notes on Site Creation and Templates:

- Site templates are assigned when the site is created and can't be changed later on. You can, however, revert to this original definition at any time.
- Site Master Pages can be changed on the fly.
- Sub sites created under the parent (or in a portal) can use any template available.
- Sites created under a portal automatically go under a virtual directory (/sites/).
- User My Sites are top-level sites, just like a team site.

Exploring Sites

From the user perspective, life in SharePoint begins with going to a top-level site so that's where we will dive in. Administration left off at creating the first top-level site, so actually starting to work with the site itself is the next step. Based on the template that was selected, the top-level site will be configured according to the settings. If using WSS, the site will be one of the Collaboration (Team) or Meeting types which are all basically the same, just organized a little differently and with different lists. If you are using MOSS, the site will be either a portal type or one of the enterprise sites (of course, any template can be used in MOSS). Either way, let's take a look at both formats.

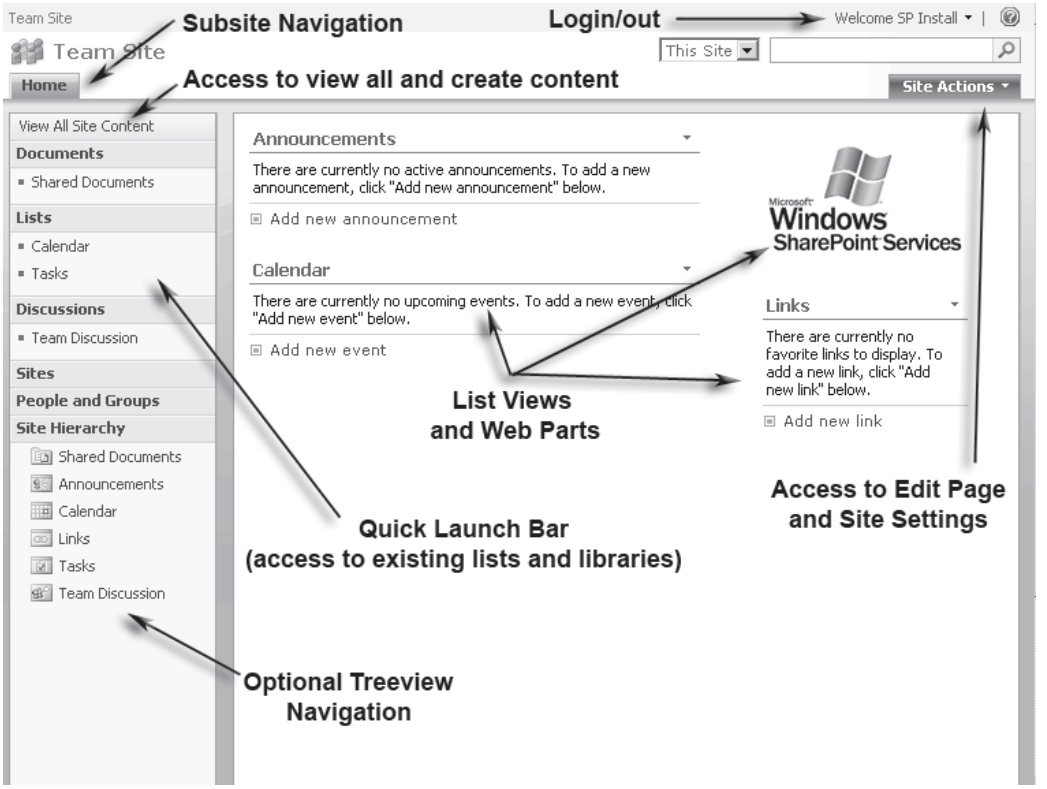
The basic construction of any SharePoint site is made up of the same sections: a Navigation section at the top, a pseudo-Navigation bar (also called the Quick Launch bar because it contains content links) on the left, and a central page portion that is used to display content and web parts as shown:



As you can see, a SharePoint page can be considered a framework with a pre-defined layout, much the same as the frames you use in HTML pages. The top and left portions of the page contain SharePoint controls (navigation, quick launch, and so on), while the main (content) portion of the page is open and is used to display content and/or web parts. The main portion of the page is of most interest to users; on a single page site, this represents the homepage that is displayed when a user enters. Swapping this main portion with different

HTML/aspx code (a content or web part page), gives the appearance of multiple pages in the same site (the outer portion of the page remains the same). Users can design these pages by dragging and dropping list views and web parts on them and then rearranging them to suit their taste and information needs.

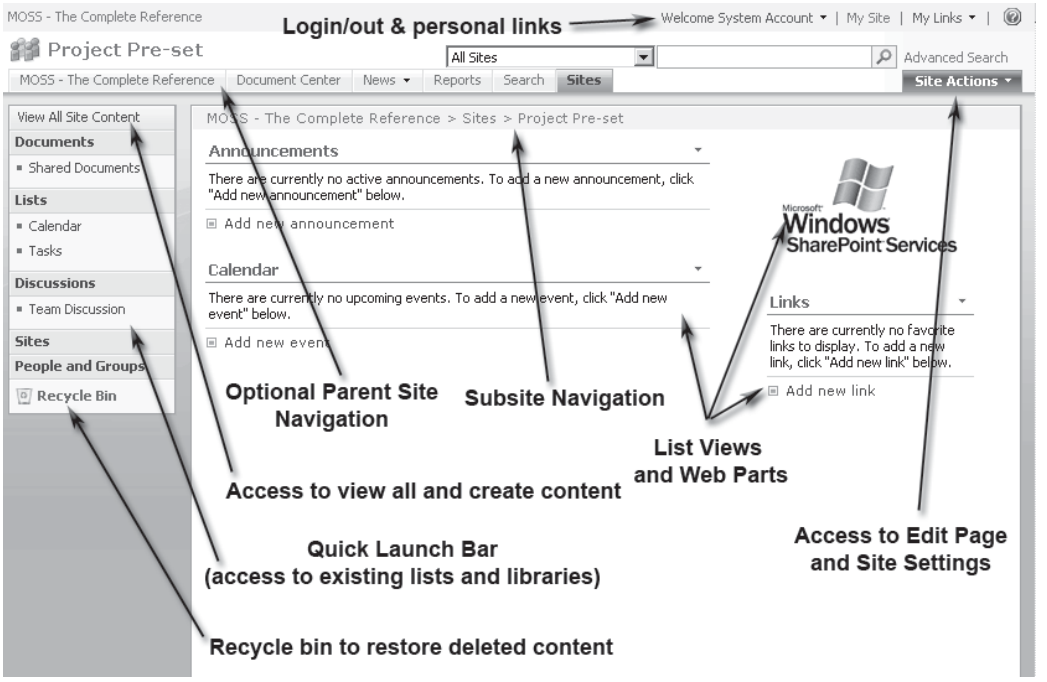
Of all the kinds of site types, the least distinction seems to be between site pages; it only matters which controls are on the page. A WSS site page, for example, has a Search Control that searches local content. A SharePoint site page has the Office Service Search Control that provides access to the application wide search. Again, the actual content portion of the page is defined by whatever the user wants. Supported by all SharePoint Products (and accessible to Microsoft Office Site Creation), the Collaboration and Meeting type site definitions all provide a similar layout:



The basic composition of the page is the same—individual lists (and web parts) appear on the content portion of the page if they are defined by the Site Template (or added by the user). As shown next, this happens to be the default Team Site definition and is showing the optional tree view navigation. Note that the content here—the lists (Calendar, Tasks, and so on) and the document library (Shared Documents)—is automatically created at the same time as the site, but they do not have any actual content (items) in them. One exception is

the Picture Web Part displaying the WSS Logo (this is a web part that will display any image, but in this case it happens to point to an image installed with SharePoint). Notice also that there are existing List Views on the page (Announcements, Calendar, and Links), though these are *views* of the lists pre-created with the site. Also by default, the lists created appear on the Quick Launch Bar on the left; they appear here because the lists themselves have this option turned on (lists don't have to be shown on the page or at the Quick Launch). For historical reference, this is essentially the same template that Team Services used way back when.

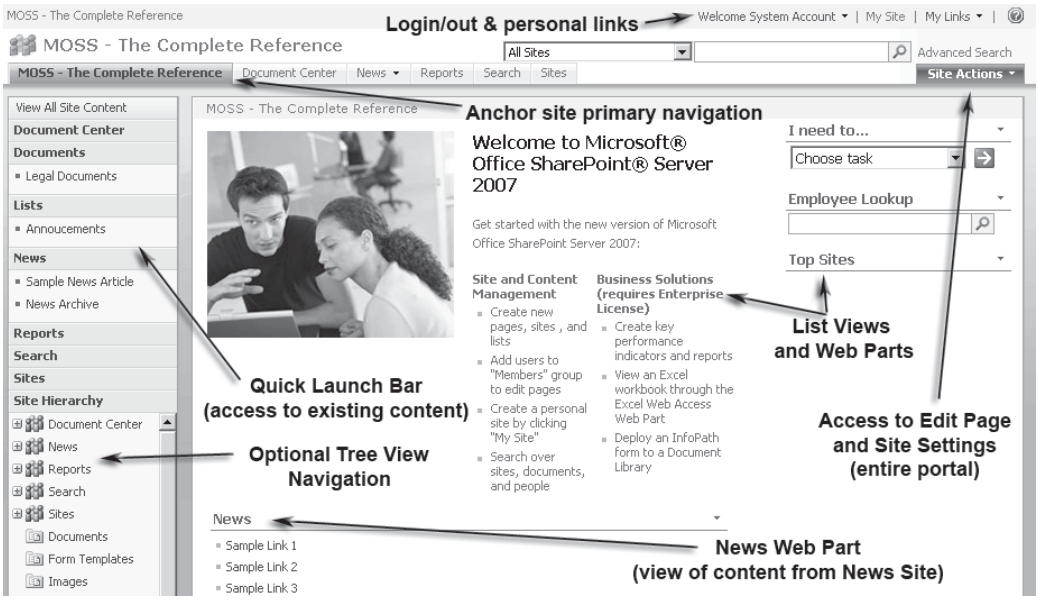
When a WSS site is created under MOSS, it is pretty much the same although the Login/out and personal links are shown and the site can optionally include the Navigation bar from the MOSS site.



As you can see, there's nothing really different other than the additional controls (search and navigation). Note that this is also the Team Site template, although the optional Tree view is not displayed in the view shown in the previous image.

MOSS sites provide the Portal capabilities of SharePoint, just like common navigation across all sites, common search, and similar. The two primary templates out of the box are the Collaboration and Publishing templates, with collaboration (in this case) targeting use for corporate intranets and publishing targeting use for public-facing or otherwise read-only content sites. The structure of these sites is based on an Anchor site and then one or

more sub sites created below it. In a MOSS Site, the home page itself doesn't look remarkably different from a Team Site.



As you can see, the page is laid out the same but there are some additional lists, web parts, and sites already created with it. This includes the Site Directory, which is used to list sites created beneath the main site. Note that when I refer to sites "under" another, the relationships between sites are more linear than hierarchical, and are based on permissions, not content.

NOTE *WSS only supports Collaboration and Meeting sites; MOSS supports all.*

Creating Sites

As a part of the main installation and configuration, a top-level site is usually created by an administrator and the users are not involved. In the Basic installation method, SharePoint automatically creates a top-level site; WSS uses the Team Site template and MOSS installs the Collaboration Portal. In the Advanced installation method in WSS, an administrator creates the site using the STSAdm command-line tool; in that configuration, a different template is used when creating the first site. For a MOSS, *Advanced* installation, any site template can be used. For the average user, creating sites means creating sites under the parent site they are given access to.

Creating sites can be done either through the UI or through one of the Microsoft Office Publishing products (Word, Excel, Visio, and PowerPoint), which create Document Workspaces (note that all Office products integrate with SharePoint, but not all can create sites). From a "what is it?" perspective, a site is just another kind of content, and is basically just creating another website to store content. Content created in sites stays in that site, and while it can certainly be linked to anywhere in the site structure, the content it owns is associated only with that site.

To create a sub site of a WSS site is a little different than MOSS; in WSS we create sites under the main site and can choose to include it in the sub site navigation or not. In MOSS, we create sites under the /sites/ root in the site and (optionally) list them in the Site Directory. Either way, the mechanics are the same.

NOTE *You should practice the actions here as I explain them and take notes on the way you prefer to do things. There is always more than one way to accomplish the same thing in SharePoint, just a matter of convenience.*

Creating WSS Sites

Creating a WSS sub site is done for a number of reasons: to segregate content, to provide a site for a team, to create a department site, and so on. Using the parent site navigation, WSS can actually be used to create a Portal-like site. WSS sites are almost autonomous in that they link but don't share with others. Here are a few things to keep in mind:

- WSS sub sites are loosely linked together; the top-level site is considered the parent, sub sites are children beneath it and have a sibling relationship between them (that is, related by parent only); the root site is an SPSite and the top-level site is an SPWeb; sub sites are SPWeb objects under the parent (part of the SPWeb.AllWebs collection).
- Sub sites can have their own sub sites (meaning that they become a parent for those sites). For example, a department site may have a document site and a project site underneath it.
- Permissions within sites (site groups, rights groups, users, and user roles) are either inherited from the parent or unique to the site. Content in each site is completely independent of the other and so is search.
- Deleting a parent site deletes all sub sites (and their content) created under it.

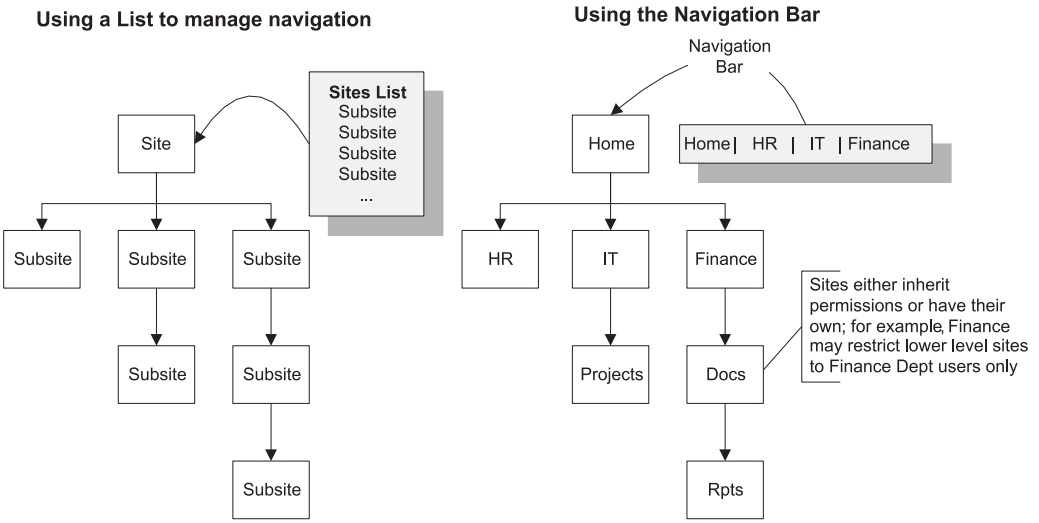
To create a WSS site, the typical trait of SharePoint having more than one way of doing things shows itself as there are actually three ways in which to create a site: two using the UI and the third using the STSAdm tool. Since the STSAdm tool is covered elsewhere, in this section I'll focus on the two options available using the User Interface.

Site Topology in WSS

Site topology (or organization) in WSS is primarily based on security and navigation—you want to ensure that things are secured and that users don't have to click too many times to get to where they need to go. You don't have too many options, since site construction is based on an "upside down tree" and, of course, there is a limit to the width of the navigation.

The key to a successful topology in WSS is to reduce the number of clicks it takes to get to information, particularly since search is limited to a single site. Without a global way of looking for things, you have to pretty much lay it out for the users. You could, of course, list all of your sites on the parent site navigation bar, but naturally there's a limit to how long this can be and it's simply not practical in most cases. When the number of sites will exceed seven or more, you can adopt the site directory approach used in MOSS. By setting up the parent site as a launch page, you can use a simple SharePoint Link list to provide multiple ways of organizing sites with grouping, cross references, and so on. You can repeat this

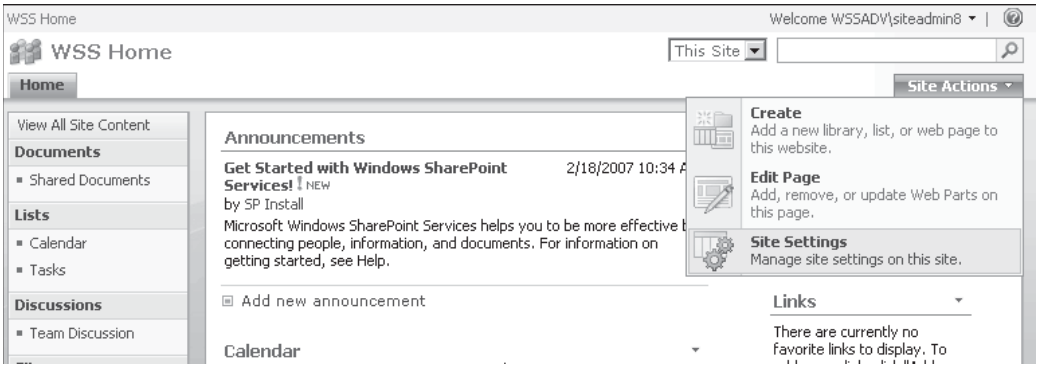
for sub sites, establishing new trees—each with its own launch site that in turn lists the sites below it. A conceptual view of the two approaches looks like this:



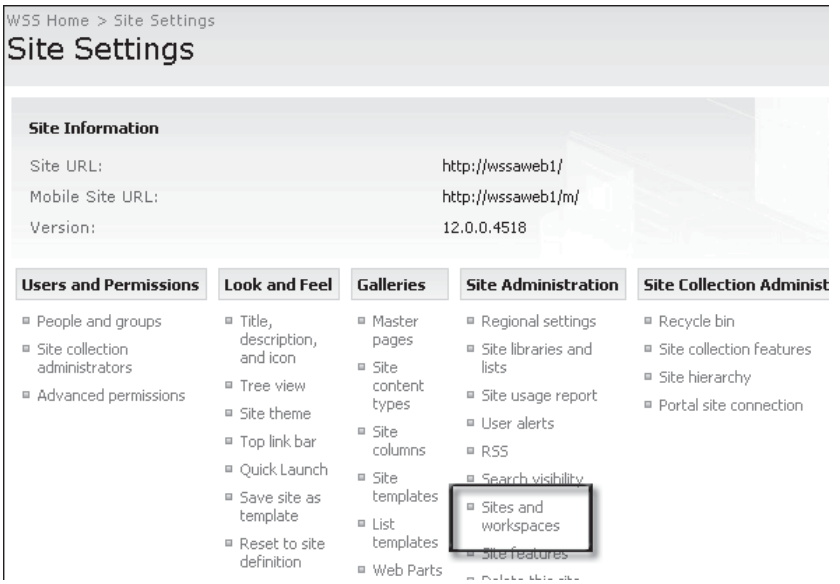
Navigating to Create a Site from Site Administration

The first method in the User Interface available for creating a site is using Site Administration:

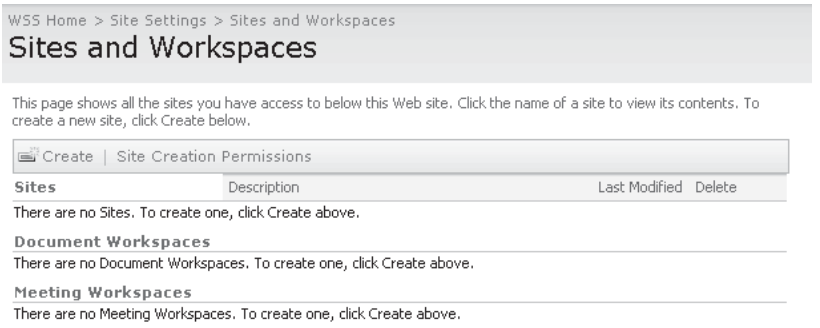
1. Navigate to the site's home page.
2. Click the Site Actions menu and from the drop down, select Site Settings.



3. When the Site Settings page opens, click Sites and Workspaces under the Site Administration group.



4. This will open the Sites and Workspaces page, which lists all of the existing sites (by category) and provides you a link to Create (a site) and Site Creation Permissions.

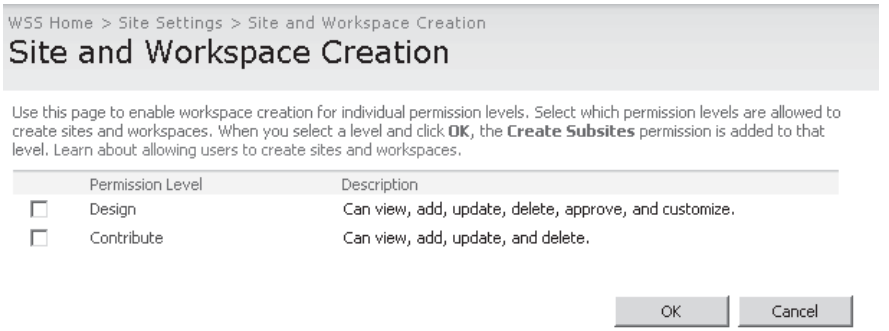


5. Clicking Create here will open the New SharePoint Site page (see next image).

Setting Site Creation Permissions

As a quick side note, you have the ability to set Site Creation Permissions for designating which Site Groups (roles) you want to grant this right to (the only way to get to these

settings is on the Sites and Workspaces page through Site Settings). Clicking the Site Creation Permissions link will display the page.

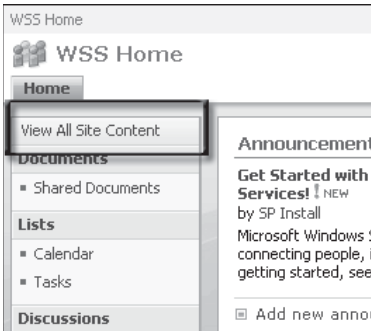


To add to one of the groups shown, simply click the check box next to the name and click OK (note that it is typical for site designers to have this right, but not always typical for contributors). This setting goes back to the governance model you set up front, as to who can do what in a given site. If none of the groups shown are selected, only the Administrator's rights and those users who are assigned the specific right can create sites.

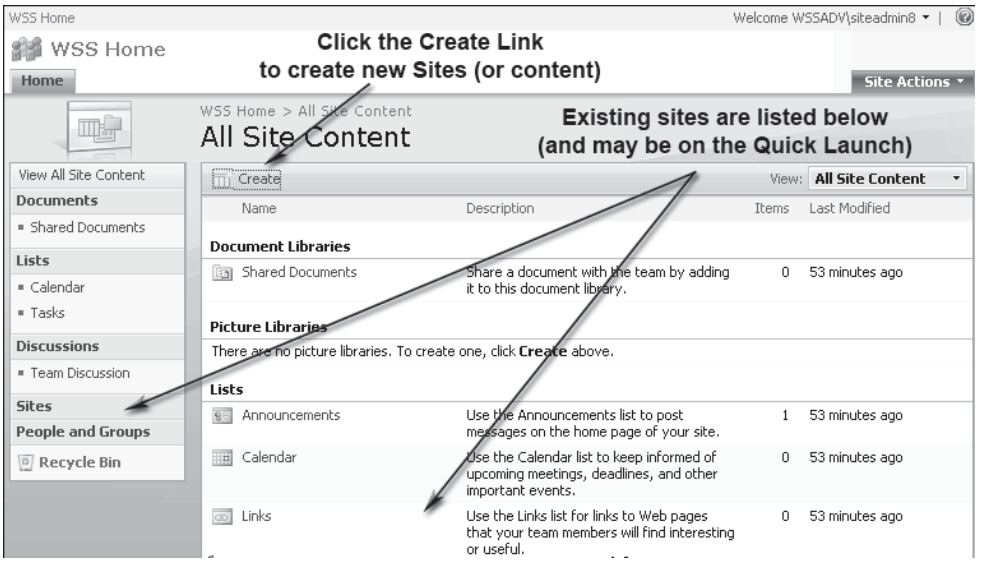
Navigation to Create a Site from the Content Page

The second method for getting to the Create Site page is from the Content Page, with two differences: this method does not give you access to setting Site Creation Permissions, and it requires an extra click to get to the create page.

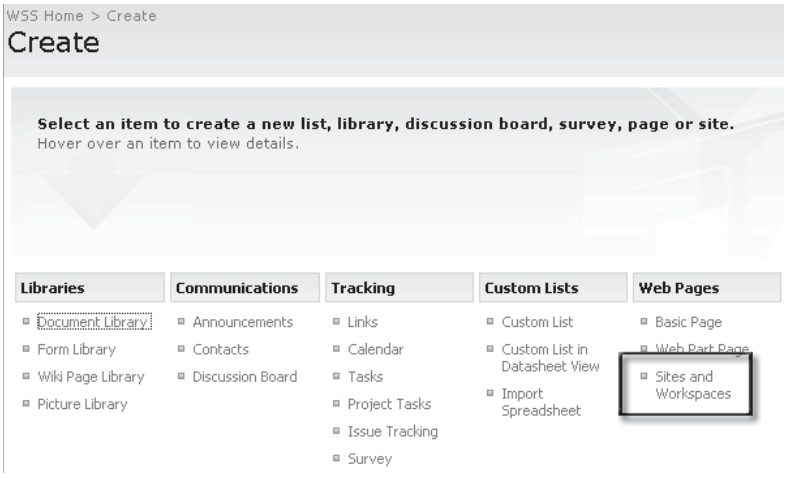
1. Navigate to the home page of the site.
2. Click the View All Site Content link in the left-hand Quick Launch Bar.



3. This will display the All Content page.



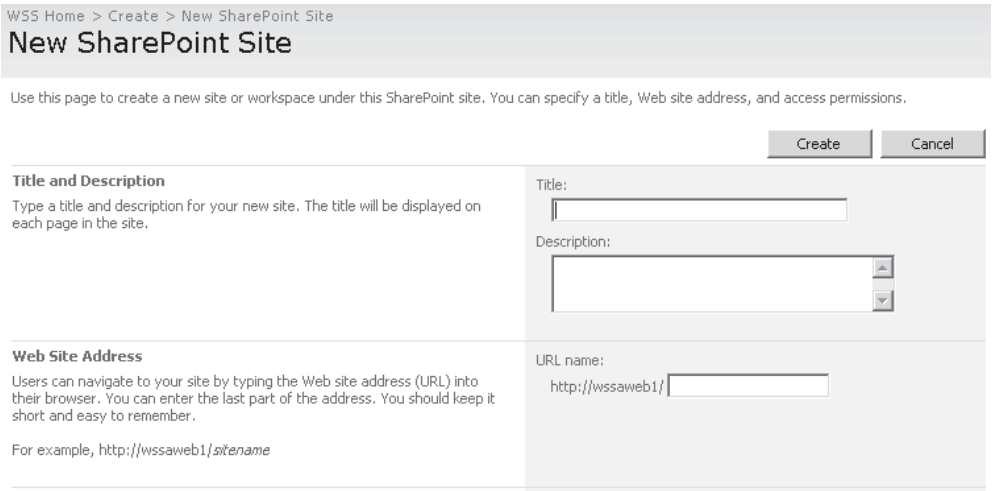
4. From this page, any existing content on the site, lists, libraries, and sites are shown (existing sites are listed at the bottom of this page and may also be listed on the Quick Launch Bar). You can also create any new content (including sites) from here; click the Create link to get to the Create page:



5. As you can see, there is the same link for Sites and Workspaces, but on this page it's listed under the Web Pages group. When you click this it brings you directly to the New SharePoint Site page (it does not display the Sites and Workspaces page, as shown in the previous image).

Creating a Site

When the New SharePoint Site page is shown, you can specify the setting for the site and choose which template to use. On the New SharePoint Site page, the first two sections allow you to set the title, description, and URL to use for the site.

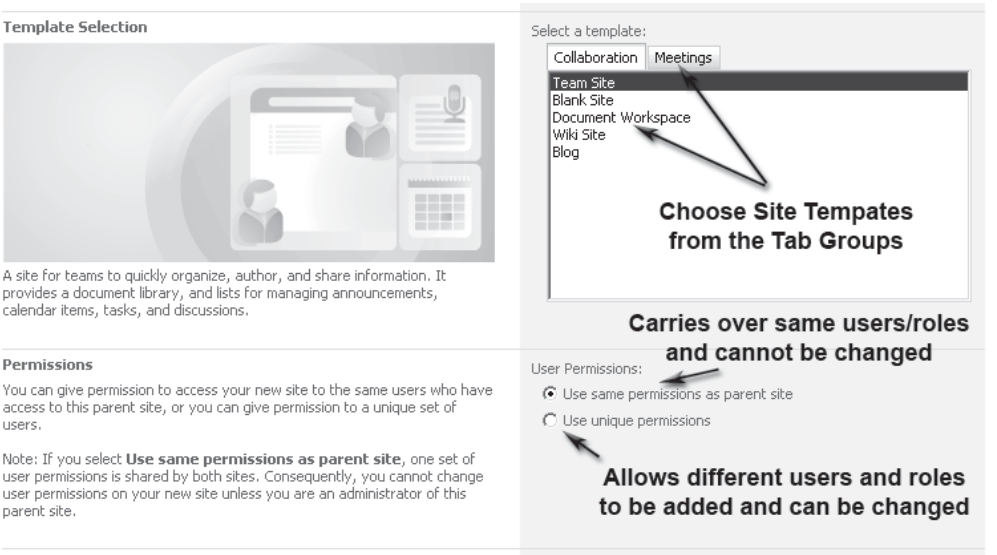


These options are as follows:

- **Title and Description** These fields set the Title of the site, as well as the name and description that appears on the home page. These fields, of course, are freeform and can be set to whatever you like, but you should heed a few warnings. First, don't use special characters in titles, especially punctuation such as periods, semi-colons, and the ampersand character (&), because these characters often cause issues in HTML due to the fact that they are reserved. Also, SharePoint uses Global Unique Identifiers (customer 32-character numbers that are guaranteed unique, called GUIDs) to identify sites. This means that it is actually possible to create two sites with the same name (SharePoint keeps them separate because the GUIDs are different). Since this can obviously be confusing, you should check the sites you already have before creating a new one. And, to minimize the confusion just in case you do use a similar name accidentally, be sure to enter a good description of what the site is for and for whom.
- **URL** The URL of the site is somewhat important. While users rarely use URLs by typing them in, they can have a big effect on content for a few reasons. First, URLs should never have any kind of special characters in them and I further suggest that they not start with a number. Second, the length of the URL can create issues when accessing content (the maximum is 255). In SharePoint terms, access to a document, list item, or site is based on the URL. Each layer down—from the root site to the sub site—adds to the length of the URL as does the name of the library. Add the document's filename length along with that and the URL can get quite long; so long in fact that you cannot access the document itself. For example, if you had a document named MyDocument.doc in a document library called Documents, and it was four sites down, the URL would be something like http://<server>/Site1/Site2/Site3/Site4/Documents/MyDocument.doc. My advice is to use initials of the

site (like shortening Business Intelligence CRM Reporting to *bicrmr*) or some other method to keep your URLs as short as possible. In any case, be consistent.

In the next sections, you can select the Site Template and set the permissions for the new site. Here you set the following:



- **Site Template** The site template selection will list all sites that are installed in the site collection. This includes those that are out of the box (as shown), but may also include additional site templates as downloaded from Microsoft, purchased by a third party, added by developers, or saved by users. As a quick cross reference, sites can be created and then modified to add lists, change views on the home page, add pages, and so on. They can then be saved as a complete template (including content), and used to create another site. Note that new sites created from saved templates are not clones but copies; there is no association to the original site once a template has been created.
- **Permissions** These allow one of two settings:
  - **User Same Permissions As Parent Site** Checking this means that the users and the permissions granted are from the site above and will not be changeable in this site. For example, to add a user means adding it to the parent.
  - **Use Unique Permissions** Checking this means that the creator will be the only user (an administrator) in the site; users and roles can be added to this site as if it were a parent. Using unique permissions is typical for protecting the content it holds or protecting it for some other reason such as the need for limiting access to select project members.

The last two sections deal with the kind of navigation you want on the sub site:

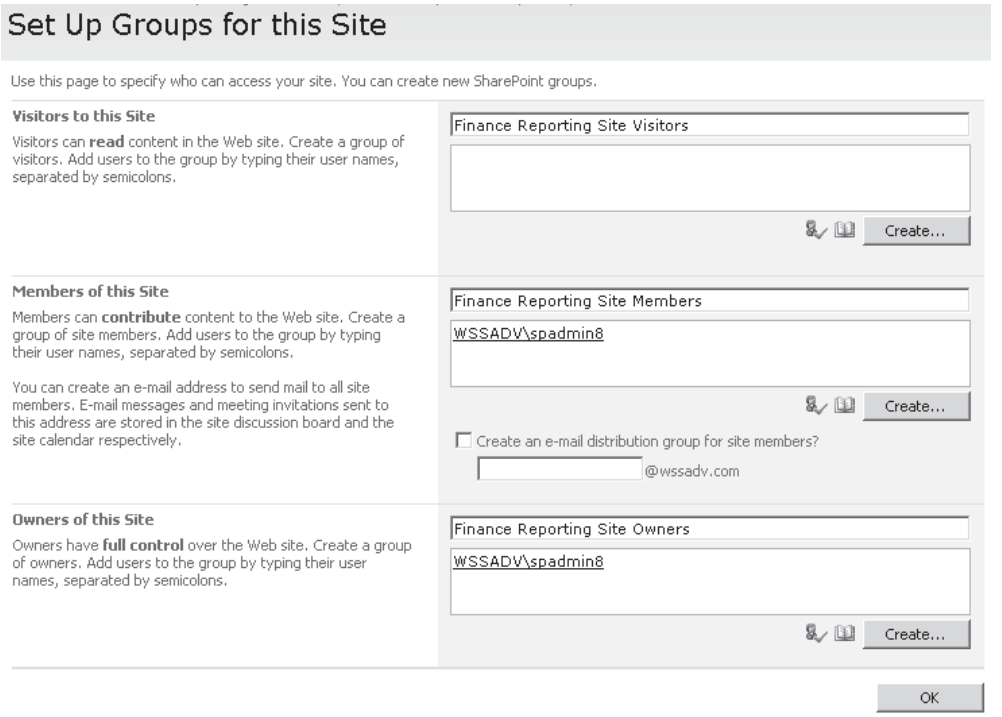
- **Navigation** This provides two options:
  - **Display This Site on the Quick Launch of the Parent Site** When set to Yes, the site title will appear as a link on the Quick Launch bar on the parent site under

the Sites group. If set to No, it does not appear. This can always be changed later (through Site Settings) to include it if necessary.

- **Display This Site on the Top Link Bar of the Parent Site** When set to Yes, the site title will appear as a link on the Navigation bar on the parent site (like a Portal). If this is set to No, it does not appear. This can always be changed later (through Site Settings) to include it if necessary.

- **Navigational Inheritance** If this is set to Yes, the navigation from the parent site will be carried down to the sub site. If this is set to No, only the site navigation will show, with no links up to the parent.

Click OK to create the site. If you selected to inherit the permissions from the parent site, the site is ready to go. If you selected to use unique permissions, you will be brought to the Set Up Groups for this Site page where you can assign the users and groups for the site.



Note that the only defined user when you create a site that does not inherit from the parent is the account used to create the site (in the previous example, the user shown was SPAdmin8).

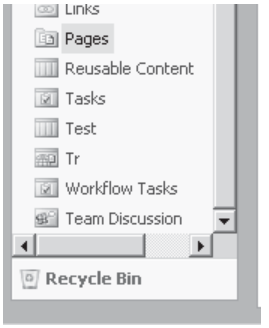
Using the Recycle Bin

Just for the sake of argument, I left this section for last in the UI overview because I think it deserves special attention. While a small piece in the scheme of things and something WSS and MOSS users will come to depend on (but never appreciate), the SharePoint Recycle Bin is a major new feature in this version.

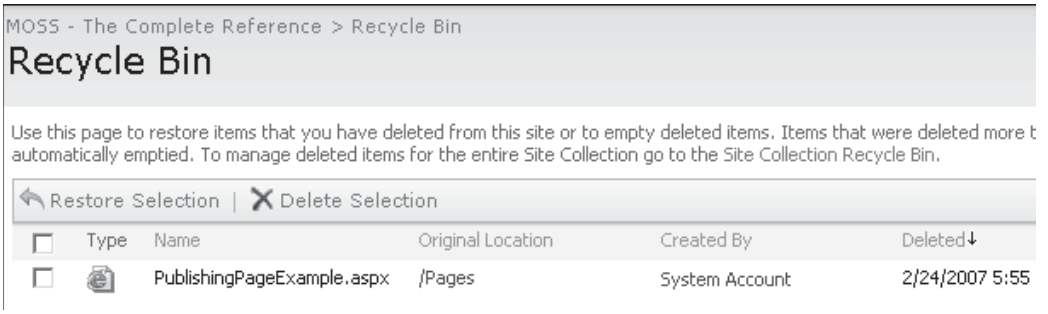
As is typical of the Windows recycle bin, it will store and retain any item or document deleted for up to 30 days from the date it was deleted. Users can restore items they've deleted without having to restore from a backup. While a very cool (and necessary) tool, there are a few things it cannot do:

- **It cannot restore a site** If you delete a site, you're looking at restoring from a backup. But wait, you say, a deleted site is temporarily available and can be retrieved—this is true, but the problem is that there is no way to restore it (yet). Microsoft did not provide a tool for it, but the data is there. Using a Site Definition and the data, the site can be restored and I'm sure a third-party tool is right around the corner.
- **It cannot restore a portal** Delete the collaboration site (as one of my clients did just after everything was working perfectly) and you are history without a full backup. In a portal environment, everything is indeed gone unless you feel like attempting to restore what you do have and rebuild to where you left off. Note that if you do have a database backup, you can recover documents from the documents table in the database by writing a windows application that extracts the binary file streams and saves them to the file system.

The Recycle Bin is available on every site and page and is shown in the lower left of the page.



As items, documents, and other items are deleted, SharePoint keeps track of where they were and stores them in a holding area. Items not recovered are dumped after 30 days (be aware of this if you delete a bunch of stuff and see storage increase!). If the user realizes they've deleted something by accident, they can simply click on the Recycle Bin, which will list the items they can restore.



As you can see, you have the option of restoring item(s) to their original locations or deleting them. Note that restoring an item that has been replaced with a duplicate name will overwrite the item in the list with the one from the bin.

Summary

As you work through not only the parts of the interface covered previously, you will learn that there is a whole lot to it as you get into more in the following chapters. The operational aspects are common, however, as all lists and libraries work the same way. The same is true for navigation and look and feel; while themes and changes to the Master page might change how it looks, the functionality is embedded and consistent. As you begin to work with the SharePoint User Interface, I encourage you to check out everything out of the box at least once or twice; after all, you can't break anything. When you think of extending SharePoint, make it a point to leverage the tools SharePoint already provides and save yourself some coding!

10
CHAPTER

Working with MOSS Sites, Office Workspaces, and Site Pages

Working with MOSS sites is pretty much the same as working with WSS, but with a different set of site templates available, content management, and more reason to use sites for content organization. Instead of using site for a limited purpose (like a Team), MOSS sites are usually intended for a wider audience; for example, the Document Center is a site designed specifically as a corporate-wide document store. MOSS sites are intended to be portal-like sites, in which common navigation, search, and content targeting is very effective, such as the Collaboration Portal designed for use in corporate intranet environments. All sites within a MOSS site collection are children (sub sites) of the primary site (Home). Sites beneath Site Directory sites (explained next) are sub sites of the directory, but their parent is still the home site.

Note that this chapter will also cover another kind of site (workspaces), which are created by Office (or SharePoint) and Site Pages, and are used in both WSS and MOSS.

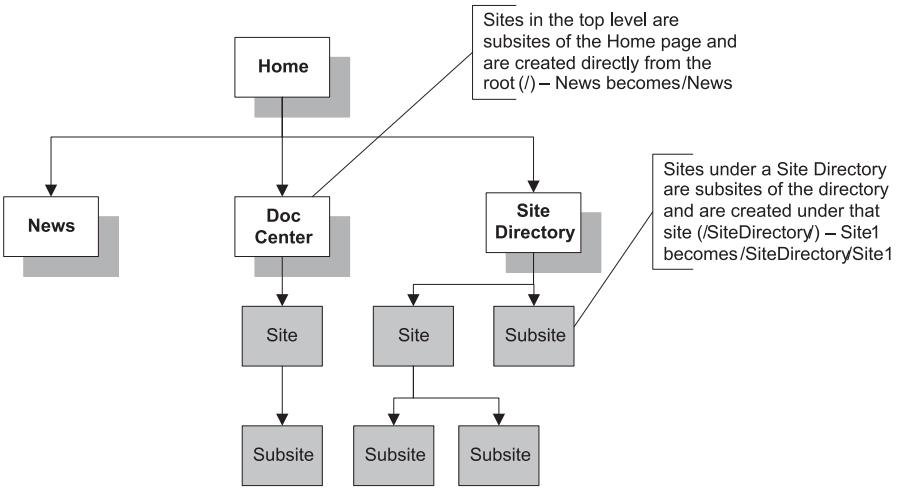
Site Topology in MOSS

Creating MOSS sites is a little different than WSS sites since the topology is not just top-down, but rather both horizontal (top-level sites grouped under the portal) and top-down (each top level can be a site collection). Because of the normal complexity of a MOSS site, including content (information targeting) and enterprise-wide search, there are many more options and decisions to make when it comes to choosing a site type.

If you are using a WSS site, the organization is rigid in a hierarchy, starting with a top-level site with sub sites beneath. MOSS Portal site types (Collaboration and Publishing) can be used to create a complete topology (or site map) of how you want things organized; in other words, a top-level site that can group other top-level sites. This organization allows you to control the items (and how many) that you want to appear in the site navigation. The more general-use the item, the more likely it will be at the top level. Also, the topology of a MOSS site is by no means fixed; sites can in fact be moved around to change both the navigational and information structure at any time.

In addition to the structure for navigation, there is also a pre-built tool for managing sub sites. This tool is for creating (and listing) sites that are not a part of the primary topology (such as private department sites, team sites, project sites, and similar).

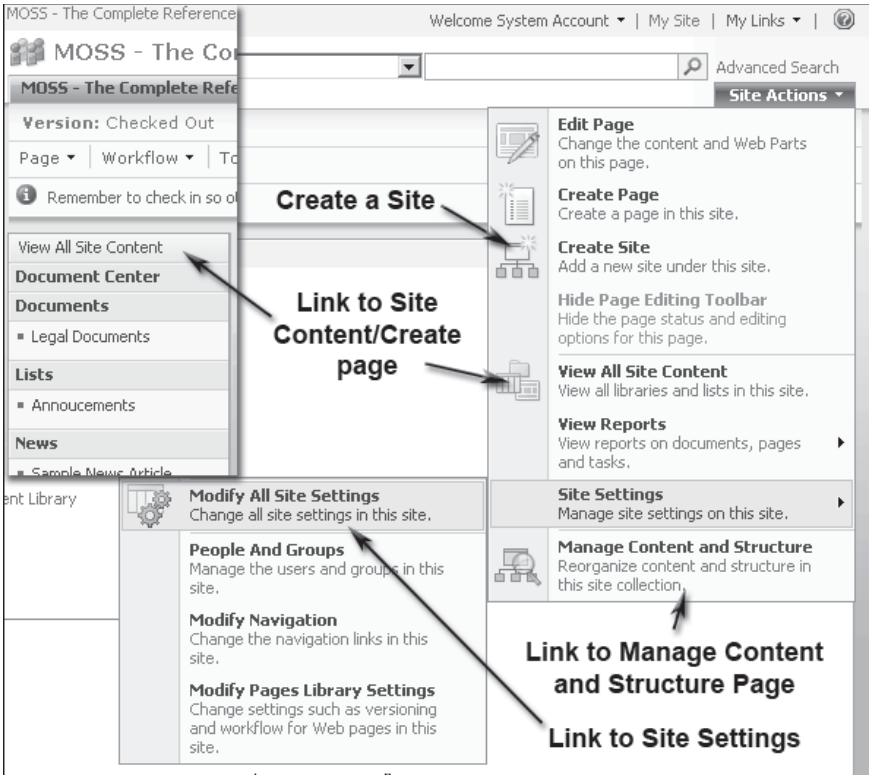
One of the MOSS Enterprise Site templates is called a Site Directory. The Site Directory is a special template type that has a three-tab page layout with preset web parts. Behind the scenes is a list called Sites that contains the information about sites (name, URL, and so on). Sites may or may not be on this list (depending on site creation rules set in administration) and non-SharePoint or external sites may be listed as well. Sites created under a Site Directory are not placed in the top navigation (though a link can be manually added), but are usually added to the Sites list. Sites created under a directory become children (sub sites) of the directory site and assume its permissions and settings; there can also be more than one Directory site for segregating sites by type. URLs for sub sites also include the parent site to which they belong; for the default Site Directory, this adds a path to the site URL of /SiteDirectory/. Use of a Site Directory is completely optional. A conceptual overview of the MOSS topology looks like this:



- MOSS sites consist of a single parent site (I like to call this an anchor site), which has sub sites (top level or collaboration sites) created within it.
- Sites created under the Home (anchor site) are sub sites of that site.
- Sites created under a Site Directory are sub sites of that directory site, but the parent is still the parent of the Site Directory.
- Sites can opt out of search (which is a great feature for protecting sensitive content).
- There can be any number of Site Directory sites and any number of sub sites (and further sub sites of those) beneath them.
- You cannot delete a site if it has sub sites.
- Deleting the Portal site deletes everything underneath it (be careful and have backups!).
- Deleted sites cannot be restored except from a backup; while they are temporarily available in data storage, there is no facility to restore them (yet).

Creating MOSS Sites (Portal Level)

There is always more than one way to do things in SharePoint, and like WSS, creating sites at the Portal (or Home site) level in the User Interface can be done in the same way as with a WSS site (through the Site Settings and the Create Content page). There are also other ways, including through the Site Content and Structure page. Since much of this would be a repeat, I'll dispense with the step-by-step walk-through and note only differences (I will cover creating sites in the Site Content and Structure page section). To create a site at the top level, you have several options directly from the View All Site Content link above the Quick Launch bar and the Site Actions menu on the page.

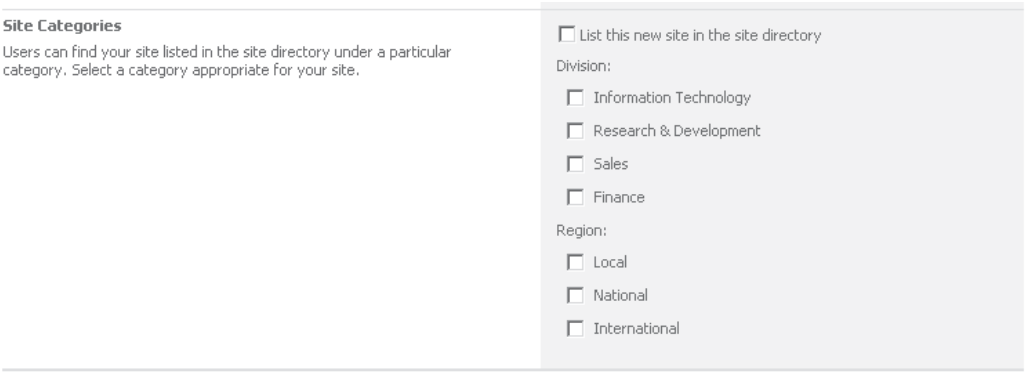


Obviously, the easiest selection here is to simply choose the Create Site link from the Site Actions menu; the other methods are simply ways to get to the same place, but they're more difficult to do. Options for creating a site are basically the same as WSS, with a few minor exceptions:

- **Site Template** The site template selection will list all sites that are installed in the site collection. As you see on the MOSS side, there are additional site templates that include the Enterprise sites (Document Center, and so on), as well as the Publishing Sites.

- **Permissions** These allow one of two settings:
  - **User Same Permissions as Parent Site** Checking this means the users and permissions granted are from the parent site and will not be changeable in this site (to add a user, for example, means adding it to the parent).
  - **Use unique permissions** Checking this means that you as the creator will be the only user (an administrator) in the site; users and roles can be added to this site as if it were a parent. Using unique permissions is typical when you want to protect the content it holds or when you need to limit access to select project members.
- **Navigational Inheritance** When set to Yes, the navigation from the parent site will be carried down to the sub site; when set to No, only the site navigation will show.

A different section is shown for MOSS sites linked to the Site Directory of the site. You have the option of listing a new site in the Site Directory automatically and sites listed can be categorized depending on the settings of the Sites List in the Directory (two default categories—Division and Region—are simply examples and should be changed).



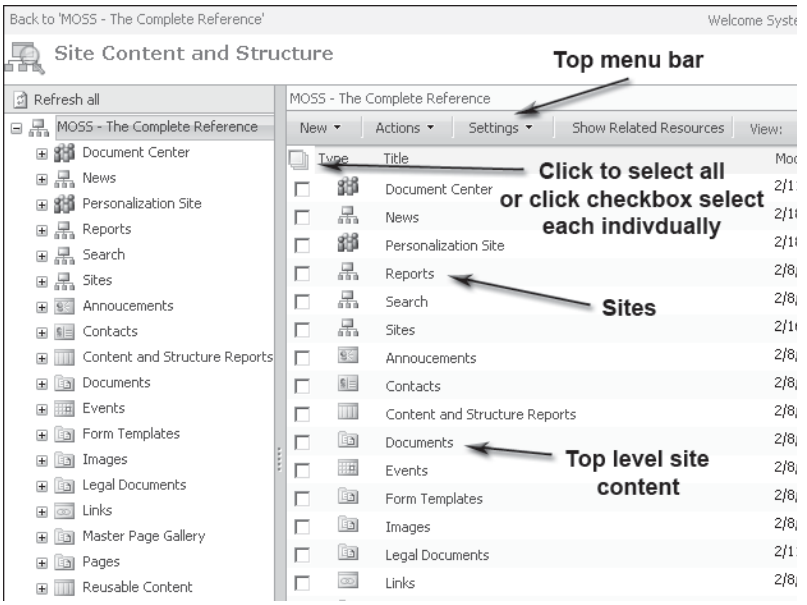
Click OK to create the site. If you selected to inherit the permissions from the parent site, the site is ready to go. If you selected to use unique permissions, you will be brought to the Set Up Groups for this Site page where you can assign the users and groups for the site.

Setting Site Creation Permissions

As with all SharePoint sites, you have the ability to set Site Creation Permissions to designate which Site Groups (roles) you want to give this right, though it's typically given to site designers and administrators (you get to these settings via the Sites and Workspaces page through Site Settings). Clicking the Site Creation Permissions link will display the page. To add the right to one of the groups shown, click the check box next to the name and then click OK. Note that if none of the groups are selected, only Administrators and users granted the specific right can create sites.

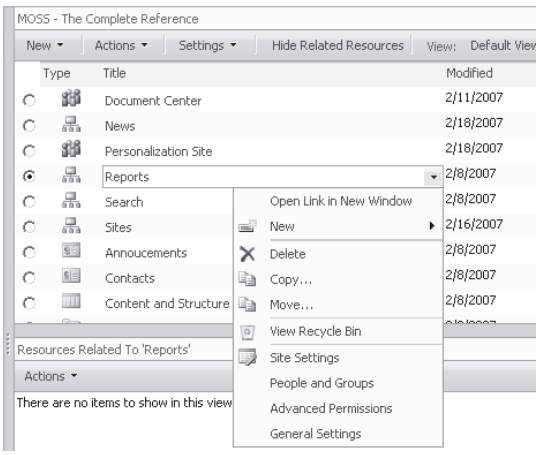
Using the MOSS Site Content and Structure Page

Unique to MOSS sites is the option to manage the site structure and navigation through a single place. This option is available from the Site Actions menu, as shown previously; simply navigate to any portal-level page and select Site Actions | Manage Content and Site Structure.



As you can see, the entire contents of the site are displayed in a tree view fashion on the left and with individual check boxes on the right, allowing any number of items to be selected (Note that when Related Resources are shown as explained below, you can only select one item at a time). The top menu bar provides access to creating and otherwise manipulating content/sites:

- **New** This creates a new Site, Page, or List (Page and list creation are covered next).
- **Actions** This provides the ability to Copy, Move, and Delete sites and content (items cannot be copied on top of themselves but can be copied to another site).
- **Settings** This allows you to bookmark the page and set it as a favorite.
- **Show Related Resources** When clicked, this will split the screen (as shown next) and list any related resources (Business Data Catalog dependencies, and so on) with the selected site or list.
- **Items** This allows you to select how many items to list on a given page.



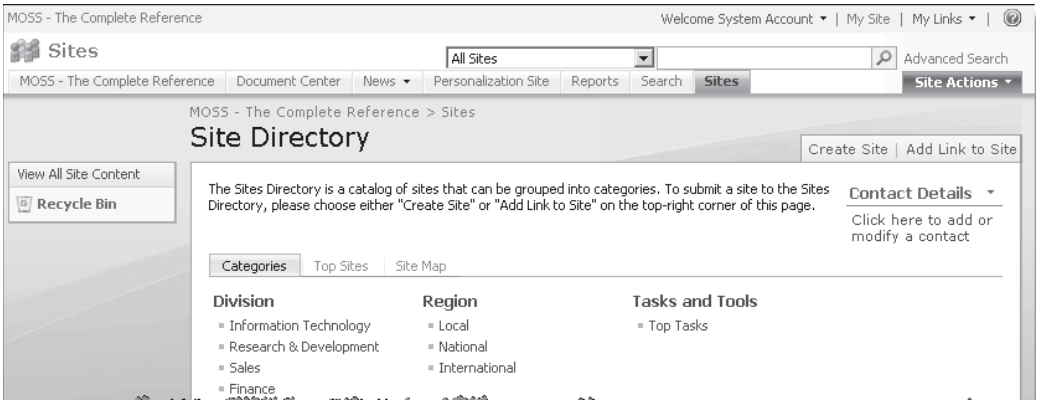
For each item there is also a drop-down menu (shown with the Show Related Resources option on).

From the drop-down list, you can select any of the same actions available in the top menu bar and other options as appropriate. For example, sites have links to access people and groups, site settings, and so on. I'll not go into any more detail on the options here as it should be pretty obvious what New, Move, Copy, and Delete do (and you can usually undo a mistake if you make one). One warning, however: while content is recoverable, sites are not—they do go into the recycle bin but there is no way to get them out. Deleted sites can only be restored using a SharePoint backup.

Creating MOSS Sub Sites (Site Directory)

Creating sites through the Site Directory is easy, and the out-of-the-box view it provides of listed sites is a handy way to organize them. The core of the Site Directory is the Sites list; this is a SharePoint list that is used not only by the Site Directory, but also by other parts of SharePoint when creating a site, as you saw previously. Two categories (choice columns) are provided by default—called Division and Region—and can easily be changed (in the list) to reflect the way you want to categorize your sites. The list and the views can be completely customized (even incorporating custom XSL). I will cover this under Web Parts Pages in the following sections.

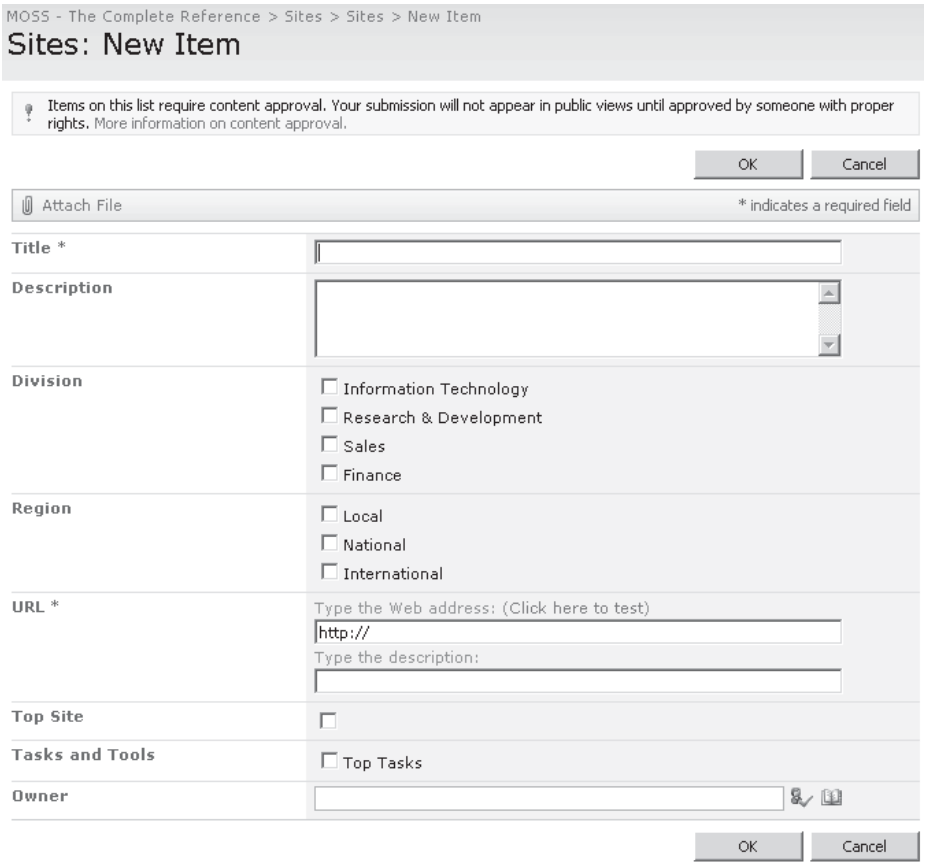
Sites created are automatically indexed for search (crawled) if they are listed in the Directory, but this listing is optional. Sites do not have to be listed and some sites, like workspaces, are not by default; this is because you don't usually list temporary sites or index content that is a work in process. By default, the Site Directory site is called Sites (with a path of /SiteDirectory), and when you click the Sites link on the Navigation bar, it opens to the default view.



As you can see, there are three tabs displayed: Categories, Top Sites, and Site Map. Each of these are simply different views of the same Sites list (actually web parts, which you will learn about next). Of note here are the categories Division and Region, with a few choices for each. The names and the choices are completely arbitrary and are intended to be changed before you begin using the site. From this page, you have two options:

- **Create Site** This brings you to the New SharePoint site page covered previously (the options are exactly the same). Remember that sites created here are placed under the Site Directory site (which is under the Parent home site). While individual permissions can be different, site collection wide features and settings apply to all sites.

- **Add Link to Site** This allows you to add a link to any site or URL (notice that for consistency, categories can be selected just like any other in the directory).



Creating Workspace Sites from Microsoft Office

As you probably know by now, all of the Microsoft Office products interact with SharePoint in some way. Word is probably the most used, since documents can be saved directly to a library in a site, can be used to create a Site Page, and workflow can be managed from there. Also, as you'll see here, Word can be used to create a Workspace site in SharePoint.

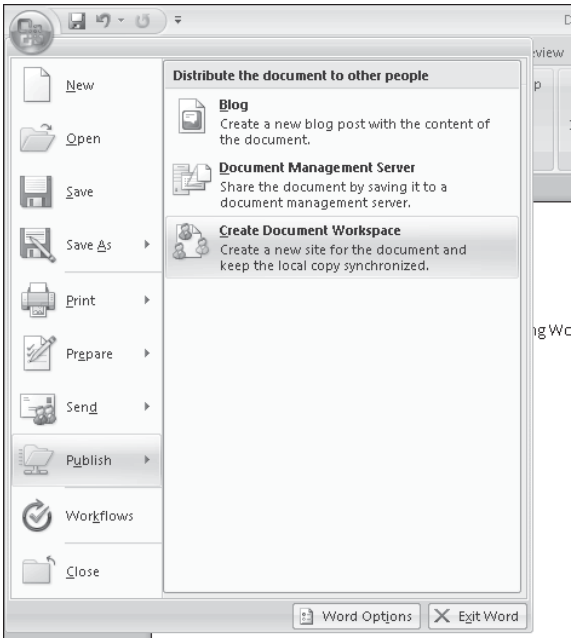
Integration with both WSS and MOSS and Office is quite extensive, with nearly all operational functions in a SharePoint Workspace site managed by Office and Outlook. For example, from within Word, a Document Workspace site can be created, document(s) stored, members added, tasks created, and even workflow started without leaving Word. The same is true for Outlook; meeting sites, tasks, and so on, which can be managed without going to the actual site. Workspace sites use one of the standard workspace templates: a Document Workspace for Word, Excel, Visio, and PowerPoint and a Meeting Workspace for Outlook. All of these work exactly the same as any site created in SharePoint. Office provides external access as well; a document can be saved directly to a site using its URL. As an example,

you'll create a Document Workspace from within Word (of course, it's no surprise that there is more than one way to do this):

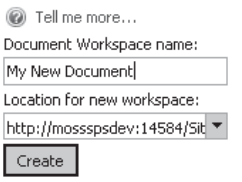
- **Using the Document Management Task Pane** This allows the creation of a site, the storing of the document, and adding users as members. Use this if you are not sure of the members up front.
- **Sending an E-mail** When a shared attachment is sent in an e-mail message, a document workspace is automatically created. The sender becomes the administrator of the site and any recipients on the e-mail message automatically become contributing members. Use this method to create a site and add users at the same time.

To show you an example that doesn't require e-mail (you might not have that available), you'll use the Document Management task pane method to create a site. To begin, do the following:

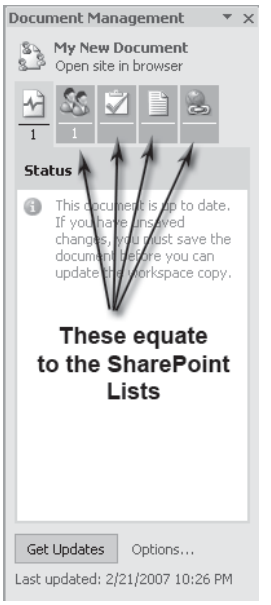
1. Open Microsoft Word, create a new document, and enter some text.
2. Click the Task button and then select Publish | Create Document Workspace.

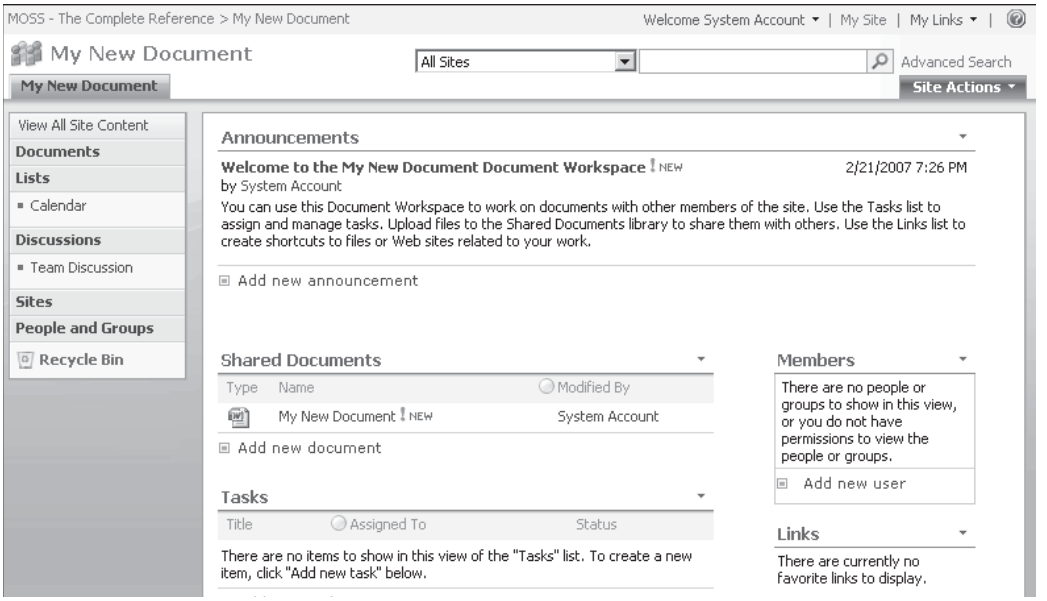


3. When you select this, the Document Management panel will open. From here you can choose the name of the document and location to save it. By default, this is set to My Site (the current users My Site). You can stay with the default, but more typically you would put it under the Site Directory, with the format for the URL something like http://<server><:port>/SiteDirectory (no / at the end), as shown in the following image:



4. Click the Create button to start the process. If you have not saved the document, you will be prompted to (the document must be saved to disk at least once before it can be loaded into SharePoint). While the site is being created, you'll see a status bar; if any problems occur (like an incorrect URL or a name that is already in use), you will be notified. Note that you may see the file upload process briefly as well (this is when the physical file is loaded into the SharePoint library). When the process has completed, the Document Management panel will refresh and access to all of the site tools (library, tasks, and so on) will be available.
5. As you can see, you now have access to Members, Tasks, a Document Library, and Links—virtually everything you might do in a site you can do from here. Of course, if you were to navigate to the site created, you'd see that everything is matched up to what is shown in Word.





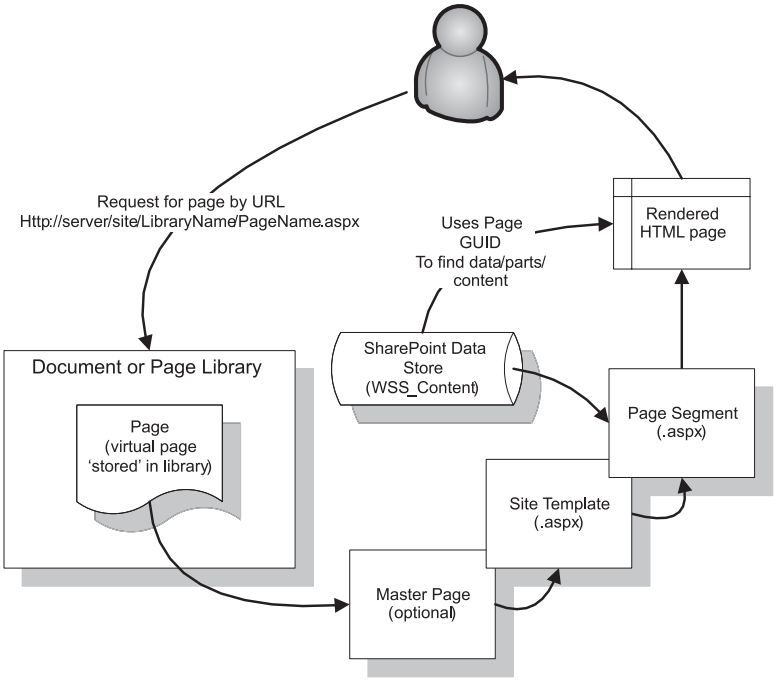
Creating Pages

SharePoint Pages allow you to create multiple independent pages in a SharePoint site. These pages incorporate the site Master Page and are effectively code snippets displayed on top of the master. These pages may have content areas and web part zones; the layouts and the contents they hold are all stored in the database (via a document library).

WSS and MOSS support two page types called a Basic page and a Web Part Page. MOSS adds an additional one, called a Publishing Page, which is used for content management. When pages are defined, they are created as page library list items (a special library type in MOSS) on the site. These are not physical pages, but rather data from the list that is combined with a page layout page when SharePoint renders the page. These are in the database and their content (as list metadata and content) is searchable.

Basic pages and Web Part Pages are actually documents stored in regular document libraries. In most WSS site templates, the default document library called Shared Documents is where pages are usually stored. Access to these pages is by link: in a link list, in a content web part, and so on (they are not included in navigation). In MOSS sites, a library called Pages is specifically for use with Publishing pages; these libraries contain metadata (item fields) that are used to control the page publishing process (approved, published, and so on). Pages of this kind can be used for both content display and to create custom home pages, such as Welcome, Redirect, and Link.

So what is a page? Pages are simply pre-defined page layouts that are either used for flat content (Basic Pages) or layout of web parts and List Views using Web Part Zones (Web Part Pages). Publishing pages are geared towards content publishing (versus application functionality) though they also can have Web Part Zones. Pages are accessible in a site by the URL that identifies them in the library where they are stored. When a request is made for the page (for example, clicking a link to the URL), the page is assembled from either the aspx file template (Basic/Web Part Pages) or the Page Layout definition (Publishing pages) and content stored in SharePoint. Using the Global Unique Identifier (GUID) assigned to the page, SharePoint assembles content, web parts, and layout, and then renders an HTML page to the user. The general flow looks like this:



Creating a page is a simple process and is done by simply navigating to the site you want to put it in and then choosing one of the ways to create it. In WSS, you have two options: one is using the Site Actions menu and then selecting the Create link, while the other is clicking the View All Site Content link in the Quick Launch bar and then clicking the Create link. Either way it will open the Create page, and from there you can click either Basic page or Web Part Page under the Web Page group.

In MOSS you also have two ways to go: you can use the Site Actions menu to select Create Page for creating a Publishing page (only), or you can create any page type by clicking the View All Content link in the Quick Launch bar, and then on the Content page, by clicking the Create link to open the Create page. You can then choose the page type by clicking Basic page, Web Part Page, or Publishing page under the Web Page group.

Now, before you can actually create a page, you'll need a place to store it. Basic pages and Web Part Pages are always stored in Document Libraries (never a page library). In most WSS templates, a library called Shared Documents is created by default. You can also create one specific for page storage. In MOSS, Basic pages and Web Part Pages work the same way; they must be stored in a document library on the site. MOSS Publishing pages, however, must be stored in a Page Library in the site (automatically created in MOSS sites).

Accessing Pages

Accessing pages once created can be a bit of a mystery. As I mentioned previously, Basic and Web Part Pages have to be accessed by links. MOSS Publishing pages can be included in navigation and can even be used to replace the home page of a site. There is no best way to make pages available; it really depends on how users will navigate them.

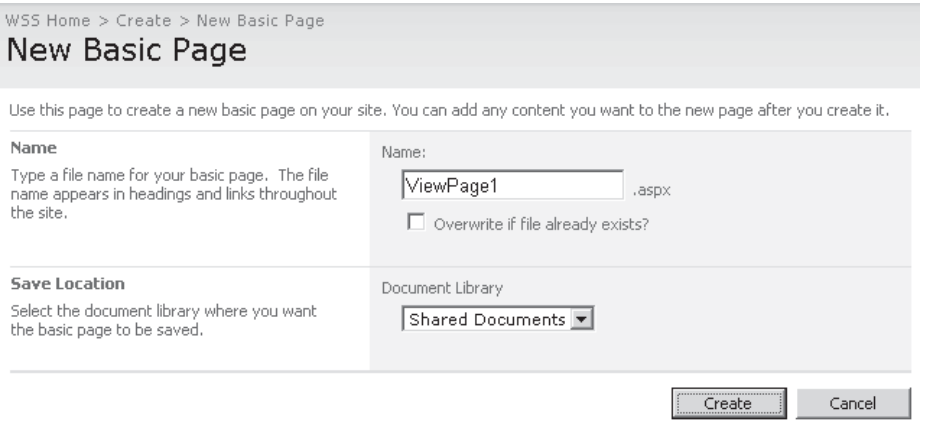
All pages can be linked to and used within content (like the Content Editor Web Part) to provide an information flow (like WIKI), listed in a Links List, or by creating a view of the library that holds the pages. As long as it's a valid link, you are good to go. Publishing pages in MOSS can be added to the top navigation, the Quick Launch bar, and even the Site Directory when created. Be aware that access to a page is based on the security where it is stored (if a user doesn't have access to the site or library, they won't have access to the page).

Creating a Basic Page

So the first question you might ask is what is a Basic page for? Basic pages are a way to provide simple content pages that display text and images, much the way a simple HTML page does. The content in these is generally static, but can be changed at any time. These pages are "live," in that any change to content is shown immediately (unless Item Approval is turned on in the library where stored). These are best used for content that doesn't need publishing controls.

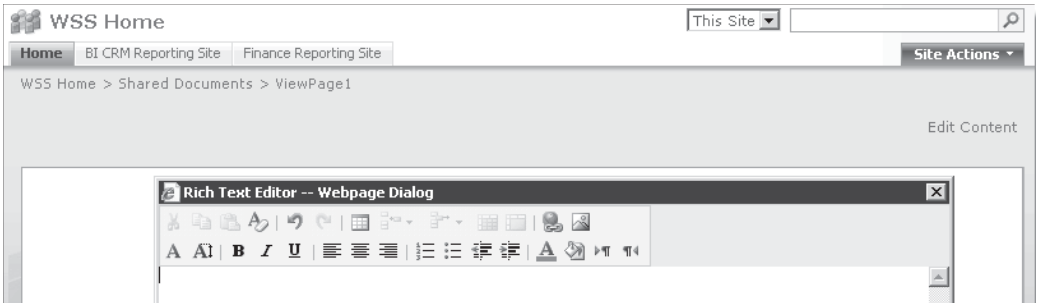
In WSS, Basic pages can be used to create a knowledge base site, new employee introductions, access to self-learning materials—you name it, anything that is more static in nature (note this is also true in MOSS using the WIKI site template type). Creating a Basic page is the same in WSS and MOSS; simply click the View All Site Content link on the home page of the site, and then on the Content page, click the Create link. I'm assuming you have a document library for storage (if you don't, jump to Chapter 11, create one, and then come back here). If you are using one of the WSS standard templates, a library called Shared Documents should already be created. In MOSS sites, a library called Documents is created by default, though you

should also note that another library called Site Collection Documents is available and is accessible from all sites. To create a page, click the Basic Page link. This will open the New Basic Page page.

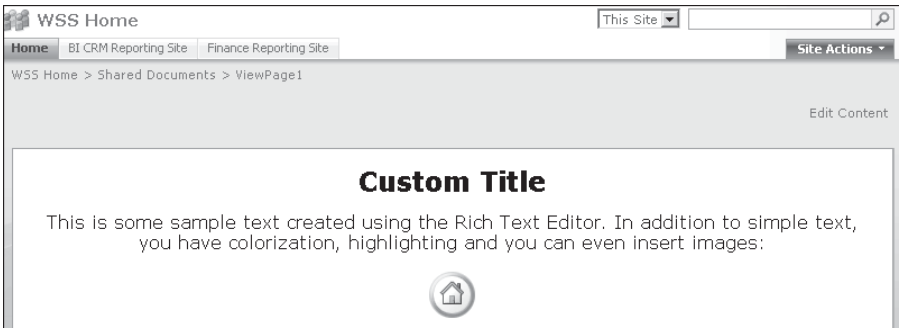


On this page, you'll specify the filename (I've entered as it as ViewPage1); this will become ViewPage1.aspx when stored in the library. Just as with sites, the name becomes part of the URL, so make sure you don't use special characters or spaces. Next, select the Document Library in which to store the page from those available in the drop-down list, and remember that a document library can store any document (documents, images, forms and pages).

Note that the Overwrite check box controls how the page is created. If it's checked and the same name is found, the new one will overwrite it—and unless versioning is turned on in the library, any existing content will be deleted if this happens. When you click OK to save the page, it's opened in editing mode and a Rich Text Editor dialog box is displayed.



With the Rich Text Editor control, you can do most anything you can do with Word (albeit scaled down), such as insert tables, add images, add links, format text, and so on. Once you enter something in the editor and click OK, it will be saved and rendered on the page like so:

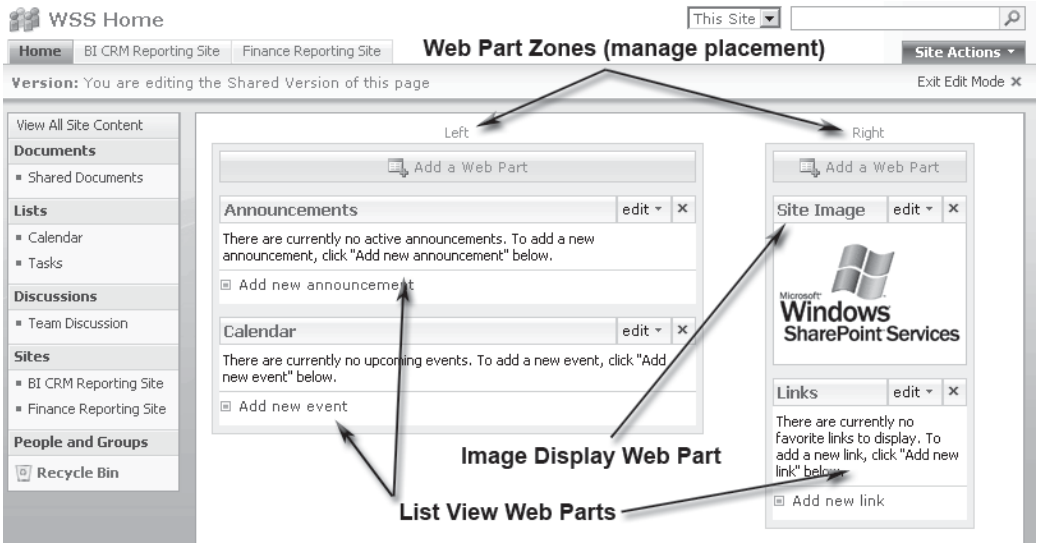


Ta-dah! Instant page! To make any changes later, you can use the Edit Content link. As I mentioned, these changes are immediate—an error is there for all to see the second you save it!

Note that in WSS, pages have a few states: View, Edit Shared, and Edit Personal. When you exit Edit mode, the changes are immediately published. In MOSS, pages have publishing controls that allow you to view and approve changes before publishing.

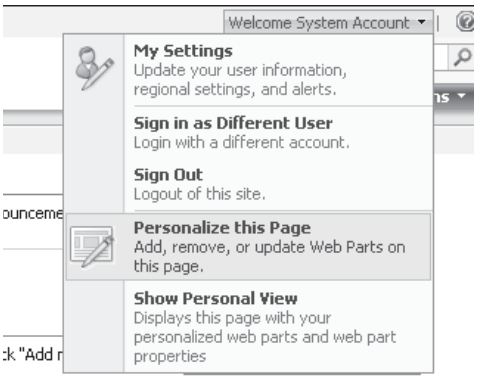
Creating a Web Part Page

Web Part Pages have a special use in SharePoint and have been the original foundation for the actual look and display of content to users since its inception. In fact, every WSS Site Template is based on a Web Part Page. Web Part Pages are structured into specific formats which control the layout of Web Parts and List Views on a page. These pages are made up of web part zones that set the layout for users, evident when in Edit Page mode.



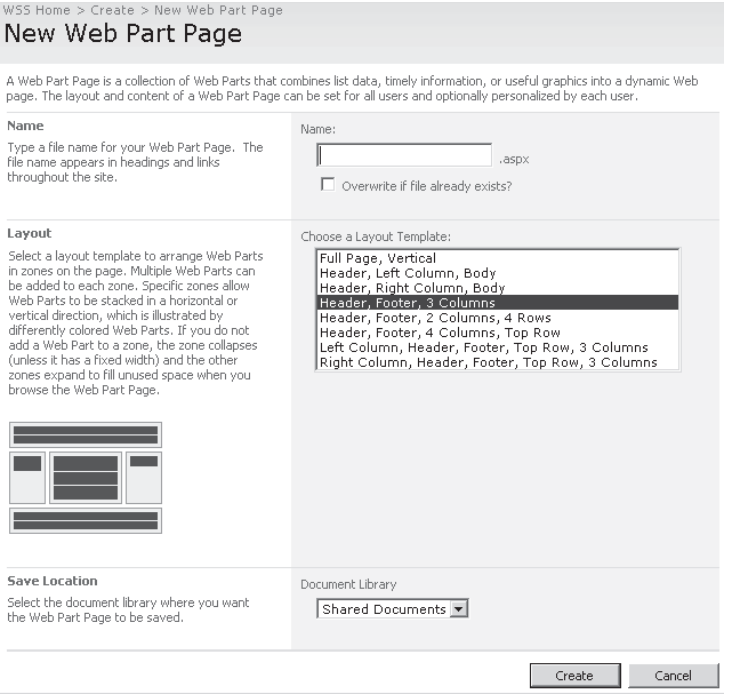
As you can see, there are defined Web Part Zones. Zones are "containers" that hold web parts; web parts can be controlled by height, width and view displayed. Web Parts, Content Web Parts (the content editor and XML web parts), and List Views can be freely dragged between zones to rearrange the look—and even minimized to provide more space. Try it out!

Web Part Pages also have two states: a Shared View, the default view of the page that everyone sees, and the Personal View, which (if they have the right) allows users to modify how the page looks just for them. Web parts also have settings and include either Shared, Personal, or both. The idea is that the end-user can manage what and how they want to see information. Since they can only modify personal settings, Shared Settings allow a certain level of control; for example, a web part can be set so that it cannot be minimized or moved. Accessing the view is done from the personal menu at the top of the page.

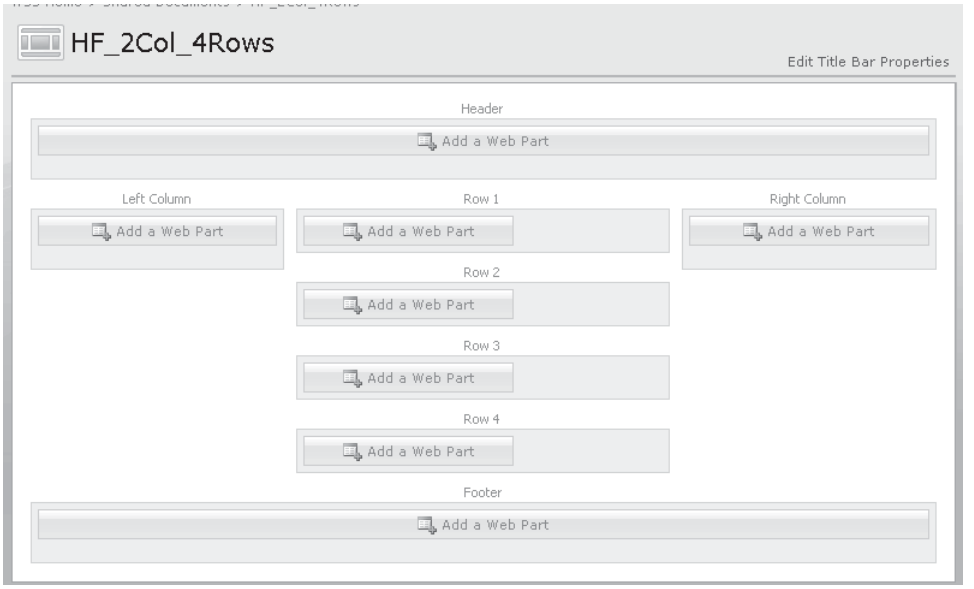


Each user can set their own particular view, while at the same time the Shared View remains the same for everyone else. One thing to note here is that sometimes personalized views can get out of whack; fortunately, however, personal page views can be reset back to the Shared View.

Creating a Web Part Page in WSS and MOSS is the same as a Basic page and must be stored in a document library. To create one, click the View All Site Content link on the home anchor page of the site. Then, on the Content page, click the Create link. To create a new page, click the Web Part Page link, which will open the New Web Part Page page.



This is the same as a Basic page; you enter a name, description, and where you'd like to store it. But, as you can see, there are many pre-defined layouts to choose from. When you click OK to create the page, it will be displayed in Edit mode and the Web Part Zones are displayed.



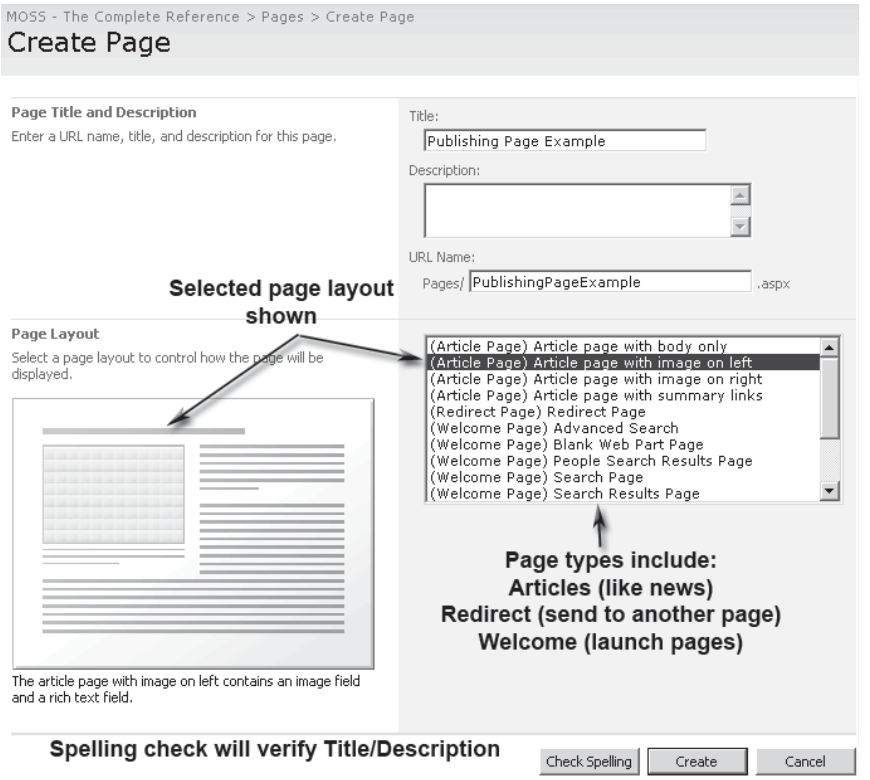
Notice there are a few structural differences. This is because the page is actually wide open, and because things like the Quick Launch Bar are not included. This can seem weird for a user when he/she goes to a page and suddenly finds that the navigation is missing. For consistency, you might create a Link List on the left to emulate a Quick Launch. You might also use straight HTML and links to create a toolbar, using a Content Editor Web Part. On the advanced side, pages can be edited and modified using SharePoint Designer and added navigation controls.

Creating a Publishing Page (MOSS)

Publishing pages in MOSS are built specifically to provide the enterprise content management capabilities in SharePoint. A hybrid, they can have both content and web parts, but are specifically for use in MOSS and are stored in Pages libraries and use Site Content Types.

Publishing pages are basically the same as Web Part Pages, but they have a few additional features to them as well. For one, they are stored in library called Pages; the mechanics of storing the page itself is the same, but the definition of the page fields and metadata in the Pages Library are controlled by Site Content Types. Also, the pages themselves are based on the SharePoint Master Pages, which allow them to inherit branding, styles, and navigation. For MCMS users, this should be familiar, because these are the replacement of Page Layouts with Content Editor Controls. Creating a Publishing page

in MOSS is simple—just navigate to the site you want to create a page in, and then from the Site Actions drop-down list, select Create Page.

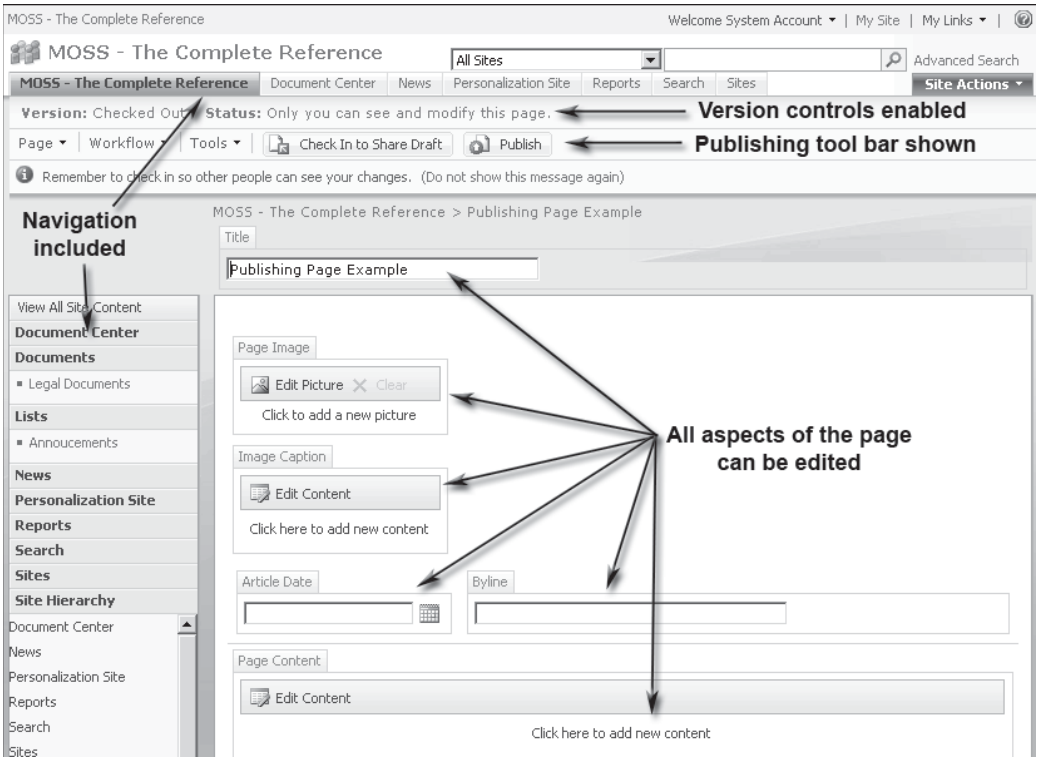


From here, you specify the Title, which automatically becomes the name of the page (spaces are OK and the name can be changed), the description, and any Page Layout you decide to use (there are several included with MOSS and new ones easily created using SharePoint Designer or Visual Studio). Pages are not tied to a specific layout and can use more than one. The default Page Layouts (included out of the box) are

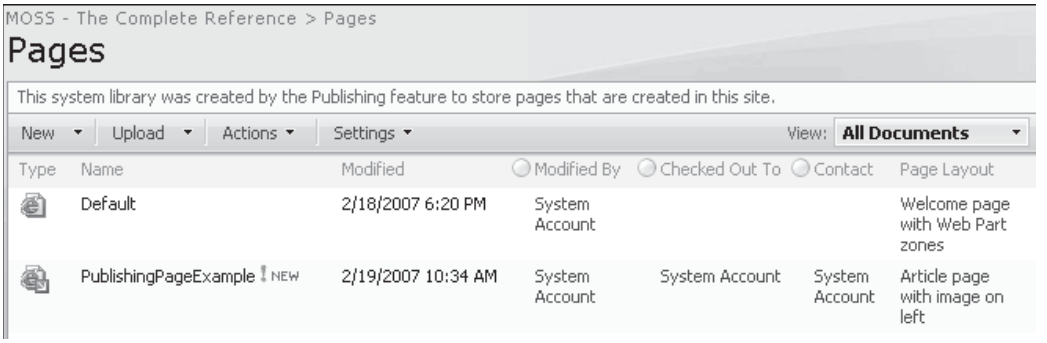
- **Article Pages** These are basic news-like articles or any place where simple content will do; good uses are for News, Department Home Pages, reference pages, and so on.
- **Redirect Pages** These allow you to redirect a user to a different part of the site (very useful in extranets).
- **Welcome Pages** These provide entry pages into the site that can be set as the home page for users when they enter. The Welcome page simply front-ends access to the site; formats include a simple splash to a complete search center.

You will also notice that there is a Spell Check button on this page; clicking this checks the title and description, and if there are any misspellings, you are prompted to fix them.

Once you click OK to create the page, you are brought to the page in editing mode. From here, you can see that the publishing tools are available, as well as the versioning for the page control, and the content holder web parts are pre-laid out.



This is the normal user editing mode, where the predefined layout of the fields and web parts allow you to simply enter the content for the page (the content is saved and when rendered, reuses the layout for display). Click Check-In or the Publish button and the page is saved and stored in the Pages Library of the site. If you navigate to this (View All Site Content | Pages), you can see the list of pages available.

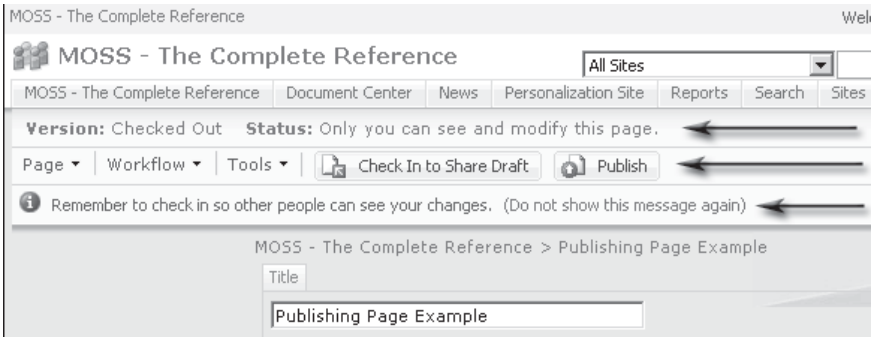


Notice that there is already a page called Default; this is part of the template and the assigned Welcome page for the site. To view a page, click on the name. To edit the properties or the actual page in SharePoint Designer, click the name and select an option from the drop-down.

Publishing Pages, Editing, and Workflow (MOSS)

In the previous examples, I kept things simple by just saving a new page and publishing it immediately, which is how Basic and Web Part Pages work (visible "as is" to users when created). However, the Publishing and Workflow capabilities provide much more control over the content: who authorizes it and when it is either published or rescinded. Workflow is a deeply integrated component in SharePoint and is available for use on lists, libraries, and as you will now learn, pages as well.

Workflows available out of the box include simple approval, multiple approvals, and so on. In the context of the page, this is Content Management; you can create a page and then start a publishing approval workflow to have the content reviewed and approved before letting the public see it. Kicking off a workflow can be done when created, edited, or even started manually, depending on the settings. The basic publishing process starts when creating a new page; after opening, the page is in Editing mode. At the top of the page, you can see the Publishing Menu.



The Version line indicates that the page is checked out (this is automatic when editing) and that the status is Only You Can See and Modify This Page. This is because the page, while created, has not been published. On the Publishing menu, you have several options:

Page Options:

- **Save** This saves the page, but the page remains in edit mode (and checked out).
- **Save and Stop Editing** This saves the page and returns to view mode (page is still checked out).
- **Check-In** This checks in the page from editing and allows for check-in comments.
- **Discard Checkout** This releases checkout status without any changes.
- **Page Settings** This provides several options for modifying page details including:
  - **URL Name** This shows the current name and can be changed.
  - **Title** This shows the title of the page and can be changed.
  - **Description** This shows the description of the page and can be changed.
  - **Page Layout** This shows the current page template selected and can be changed.
  - **Audience Targeting** This allows you to select the User Audiences the page should be targeting.

  - **Page Contact** This shows and allows you to change the contact user (and image) that is responsible for this page.
  - **Historical Detail** This is a quick summary display that shows the page's publishing history (reference only).
- **Delete Page** This deletes the page (and any content on it).
- **Add Web Parts** This is only available on a web part page, puts the page into edit mode, exposes the web part menus, and brings up the Add a Web Part dialog window.
- **Modify Web Parts** This is only available on a Web Part Page, puts the page into edit mode, and exposes the web part menus.

Workflow Options:

- **Publish** This publishes the page immediately.
- **Cancel Approval** This rejects the page or last edit.
- **Cancel Scheduling** This cancels the publishing schedule of this page.
- **Unpublish** This retracts the page from the site.
- **Approve/Reject** This approves or rejects a page (started from workflow).
- **View Page Tasks** This allows you to view the current tasks that are part of the workflow on the page.
- **View Workflow Status** This shows the current status of the workflow on the page.
- **Start a Workflow** This starts a new publishing workflow for this page. Note: If no workflows are available, either none have been installed or the features have not been enabled on the parent site.

Tool Options:

- **Spelling** This allows you to check the spelling in the content.
- **Preview in New Window** This opens up a preview of the edited page for viewing.
- **Check for Unpublished Items** This checks for items that are waiting to publish (on timer jobs, and so on).
- **Submit a Variation** This allows you to submit a variation of the page (for localization, and so on).
- **Update Variations** This will update variations from the primary page changes.
- **Quick Deploy** This forces quick deploy publication of the page.
- **Version History** This allows you to view the editing and check-in/out history of the page.
- **Compare Text Changes** This allows you to compare content from versions.
- **View Page Status** This shows the current status of the page (state) in the publication process.
- **View Recycle Bin** This allows you to view/restore items that have been deleted from the site (including pages).

- **Edit Item** This allows you to edit the page contents.
- **Check In to Share Draft** This checks in the page as a draft, meaning it is saved, but not visible. You or an administrator can later publish the page.
- **Publish** This checks in the page and publishes it, making it immediately available to users.

The version status line is always used to indicate the current state of the page in the editing context. When editing, this indicates that the page is checked out and not visible to anyone but the editor. When items are checked in but not yet published, it creates a minor version (like 0.1); publishing creates a major version like 1.0, often used to start a workflow.

The status of the page is always shown below the publishing menu and will warn of any issues or conflicts. For example, if two users try to save the same page, SharePoint will display a message stating, "The page you are attempting to save has been modified," and will provide the user the option to Preview Existing Version, Save and Overwrite Existing Content, or Exit Without Saving.

As a last word on workflow for publishing pages, the workflow steps may be different but the process of using workflow is the same as for lists and libraries (which you'll have plenty of experience with very soon). Page workflow publishing is based on workflows attached to a Site Content Type (as you will learn) or attached to the Pages library itself. Note that workflow is not preset in site template types. In the Publishing with Workflow Site template, all of the Pages libraries have a parallel workflow pre-defined to submit to a pre-defined Approvers site group. In the Collaboration Portal, the Page libraries have a simple one step approval process.

Summary

From this chapter, you should have a good understanding of the way sites are organized and the operational side of using SharePoint pages and when you might use one over the other. You should also have a general idea of the publishing process and how SharePoint supports content management. In Chapters 12 through 15, all of the concepts here will be covered in much greater detail.

CHAPTER 11

SharePoint Lists and Libraries

Ah lists, lists, wonderful lists—the true heart and soul of SharePoint. Lists represent the core content in SharePoint. Lists can be thought of as database tables that SharePoint uses to store items. For example, for a project your items might include the project list, the task list, the member list, and so on. In a meeting, you have an agenda (a list), a calendar (a list), and attendees (again a list). In addition, you have libraries (a type of list) for images (such as the Picture Library), documents (Word, Excel, Text, Visio, PowerPoint, PDFs) in Document Libraries, and Form Libraries (for InfoPath and XML forms). In short, these are all lists that have attachments.

When you think about it, the list is a natural thing you use every day. An event calendar is a list of events. Tasks are lists of to-do items. SharePoint lists are really the equivalent of data tables in a database, and it happens to provide all of the tools users need to maintain that data. I've used lists as a "cheap" form; anything you'd typically collect from a form can be emulated using a list. Since columns can be multiple data types, including lookups to standard information like users, you can provide a complete entry form with drop-downs, field validation, required fields, and so on. Add versioning and approval processing and you've got a mini-application.

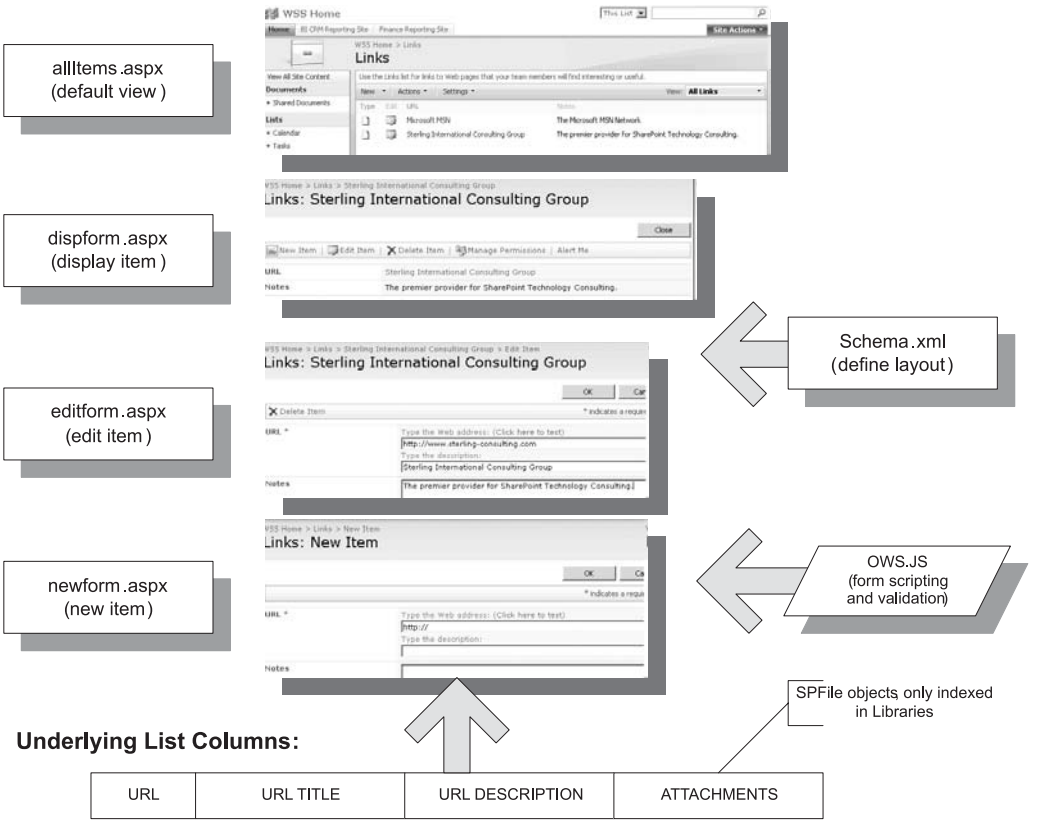
There's not much to a list in technical terms and a list is something used in a typical application anyway; SharePoint just happens to provide them for us. From a platform point of view, lists are accessible via the SharePoint Object Model programmatically, so something that would take weeks to code can leverage SharePoint and accomplish the same thing in minutes. You can see the complete list of out-of-the-box list types later in the chapter in Table 11-1.

Anatomy of a SharePoint List

A SharePoint list is composed of only a few elements: a few aspx pages that handle display (these could just as easily be InfoPath forms) and a Schema.xml file that defines the layout and display of the list. This same construction applies to all lists; the only difference between a list and a library is that library attachments (files) are indexed for search. A list exists at the Site level and may or may not have a link to it on the Quick Launch bar (this is just a setting and only used if users will regularly go to the list itself). A component of lists is that List View Web Parts can be created automatically by adding the list to a Web Part Zone. The List View Web Part is simply a view of the list data (not part of the list). You can add more than one List View Web Part for the same list on the page with the same or different view.

Because they are simply views, deleting a List View does not delete the underlying list. However, if you delete a list, it will delete any List View Web Parts associated with it.

When you break down a list, you'll find it really starts out with a row (an item in SharePoint) and fields (columns in SharePoint), just like any database table. Think of the list name as the table name, items as records, and columns as the data fields. By default, a list must have at least one column (same as a database) and columns can be several different kinds of data types. Once defined, the rest is already handled by the aspx forms. If we look at it from a high level, it's something like the image shown next. (Note that several background fields like Created By are part of every list and not shown).



A list in SharePoint (an SPList in the SharePoint Object Model) is mapped to certain tables in fields within the SharePoint database to create a virtual table. The list itself is really a database row in the Lists table of the database. When created, a Global Unique Identifier (GUID) is assigned to separate it from others. Column names used in the list are linked to generic fields in the Userdata table and tied to the list GUID. Documents attached to lists are stored in the Documents table, again linked to generic database fields and tied to the GUID of each list row, which is also tied to the list GUID.

As for what's in a list specifically, let's look at what a list is made of: columns of data that represent metadata and that create some kind of an entity. All those glorified terms aside, a list is made of

- **List Items** These items (or rows) are defined by columns. An item represents some kind of an entity (or a thing, like a task). List items are assigned unique IDs within the list (an identity column in the database). Attachments to list items are indexed by the item GUID, but are actually stored in a different table (creating two data rows for a single list item); only attachments in libraries are indexed for search. In the Object Model, items are SPListItem objects and are all a part of the SPListItemCollection. Attachments to items are SPFile objects (all of these have backward references to the list and the site where they reside).
- **Columns** Columns define what an item is in terms of data (or tagging). Columns can be of several data types, from a simple string or number to a data lookup. Columns can also be a reference to a column in another list, Site Columns, or Site Content types (as you will see in Chapter 14). Columns equate to SPField objects in the SharePoint Object Model and the collection of them to an SPFieldCollection object (derived from SPListItem.Fields).
- **Views** Views are different ways of showing list items. When a list is created, a default view of All Items is created, and as you might have noticed, a corresponding aspx page is created. When new views are created, a new aspx page is created and stored with the list. In the Object Model, views are SPView objects and all views are an SPViewCollection (derived from SPList.Views). View options include:
  - **Columns and Column Order** This allows you to select which columns to display and in what order.
  - **Sorting** This allows you to sort on any column(s) in either ascending or descending order.
  - **Grouping** Grouping provides a tree view type effect on a grouped column. For example, if you grouped on a Customer Name column, the list would then group all items by name and allow them to expand and collapse so you can view items per customer.
  - **Filtering** Filtering allows views to be set to filter on one or more columns. For example, you could set a filter on a Region field to limit it to "Eastern."
  - **Style** A few styles are provided that control the way items are displayed (there is a News format, a boxed format, and so on).
  - **Limits on Display** This is the maximum number of items to display on a single page and prevents excessive numbers of items being returned.

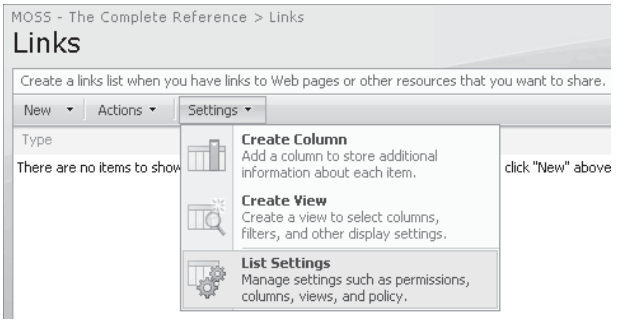
Just like sites and other things in SharePoint, SharePoint lists are based on templates. Once created, you can modify them to suit your needs by adding columns and so on. Lists made from templates are unique; if you change the underlying template, it does not change the lists already created from it. Lists and libraries can be saved as templates (with or without content) with custom columns, views, and so on. Templates can even be exported to another site.

The Administrative Side of Lists

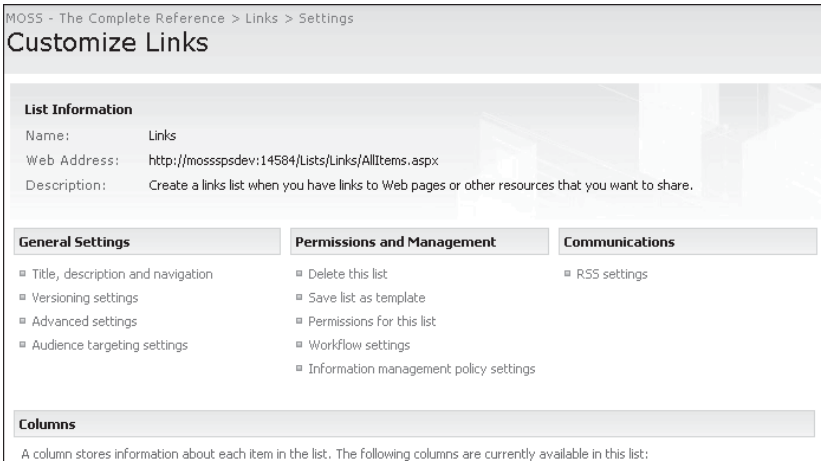
Lists have three places where you can adjust settings and control the ways that the list and its items work (as well as who has access to them). In the Lists Settings, you set aspects about the list itself and on individual items you can manage line item security (this must be enabled in the list settings). The other settings are the List View Web Part Settings that control how list data is displayed.

The List Settings allow you to access the primary settings (title, description, and security) as well as maintenance options such as deleting or saving a list as a template. Accessing a list's settings is done by navigating to a view of the list itself accessible from the site's Content page accessible via the View All Site Content link. When you create a list in a site, a list really becomes its own entity; the functionality and views of the data are provided by the list, not the site. With the forms provided, you can add and update items, while the List Views (not web parts) provide different views of the list data. By default, there is always view created called All Items, which provides the main view when the list is created. Of course, you can create any number of additional views, and all are selectable from the list. Just to be clear, when you are inside a list or a library, you will see the list name at the top of the page. When you are dealing with a List View Web Part this is a view, and the Site or Page title is displayed.

To manage a list's settings, open the site where the list is, and click the View All Site Content. Then on the Content page, find the list you want to view and double-click on it to open. When opened, the menu bar shows the Settings menu; use this to click on The List Settings link.



Notice that you have other options in this menu, including Create Column (add a column to the list) and Create View (create a new view of the list); these are quick links to those functions (also available from the List Settings page). The main list settings break down into three sections.



List Settings

These options are the general settings for the list and its security as follows:

General Settings

- **Title, Description, and Navigation** This allows the changing of the title, description, and enables/disables the list appearing on the site Quick Launch Bar.
- **Versioning Settings** This enables settings for
  - **Content Approval** This enables/disables content approval for submitted items; if enabled, users can submit, but only the list owner or an approver can approve the items.
  - **Item Version History** This enables/disables creating a new version each time a duplicate item is submitted. Versions can be "major" or "minor" and you can limit the number retained. (Keep this in mind for PowerPoint!)
  - **Draft Item Security** This sets the security level for draft items, including:
    - **Any User Who Can Read Items** At this level, all readers can see drafts.
    - **Only Users Who Can Edit Items** At this level, all users with contribute rights can see the drafts.
    - **Only Users Who Can Approve** At this level, only approvers and the users that originally submitted the items can see the drafts.

Advanced Settings

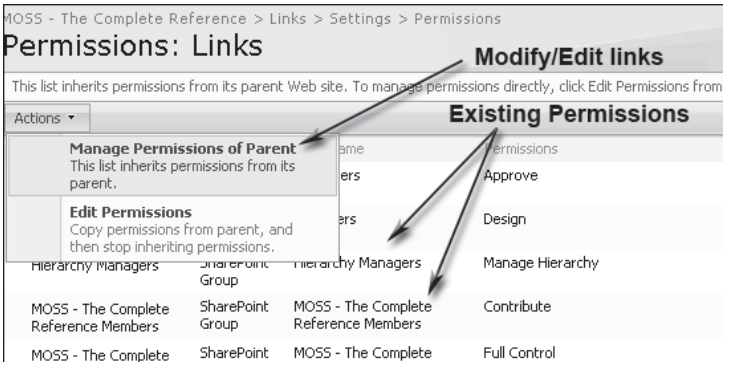
- **Content Types** This allows you to enable or disable management of content types. If enabled, the list can use Site Content Types (covered in Content Management). Content types allow for multiple item types in a single list.
- **Item-level Permissions** These apply to List Items and include one of the following:
  - **Read Access** This allows users to see All Items (also known as items added by anyone) or Only Their Own, which means that users can only see the items that they have created. Administrators can see everything.
  - **Edit Access** This sets the Item Editing options, which includes one of the following:
    - **All Items** Users can edit any item.
    - **Only Their Own** Users can edit only items they created.
    - **None** Users cannot edit any items (even their own).
- **Attachments** This enables/disables attachments to this list; if enabled, you can also choose to include an icon on the List View to indicate the document type (if known).[1] Unless the item in question is a library, attachments are not indexed.

[1] **It is possible for a common document type to be in SharePoint yet not be identified with an icon (this is typical for an Adobe PDF file); these are defined in a file called DocIcon.xml located in <drive>:\Program Files\Common Files\Microsoft Shared\web server extensions\12\TEMPLATE\XML; images (icons) specified in this file are expected to be in the SharePoint Hive Images folder.**

- **Folders** This enables or disables folders in the list. These provide "Windows Explorer"-type functionality with "folders" in which to organize items.
- **Search** This enables or disables items in the list to be included in a search; enabled by default, disabling it means this list and its contents are not indexed. Examples of when you use this include when you are disabling a search on nonessential items or are doing something temporary such as a one-time event calendar. It can also be used for security if the information in the list is sensitive.
- **Manage Item Scheduling (Libraries only)** This enables/disables the scheduling of items using content types that include start and end dates.
- **Audience Targeting Settings** This disables or enables the Audience Targeting feature for the list. (Audiences are part of MOSS and are based on user profiles; members of a targeted audience see the data, nonmembers do not.) Audiences are not security-related; a nonmember can still see the content but it is not highlighted for them.

Permissions and Management

- **Delete This List/Library** As it implies, this will delete the list and move it to the Recycle Bin (it can be restored for up to 30 days from date of deletion).
- **Save List/Library as Template** This will save the current list as a template to the List Template Gallery inside SharePoint; users can access it when they open the Create page. List templates will appear along with the out-of-the-box ones listed by their type (library, list, and so on).
- **Permissions for This List/Library** This option allows you to set specific permissions for this list, overriding the default settings applied to the sites. (Lists inherit the site security by default.) Permissions are indicated by which user groups have been assigned which rights, and can be modified by the drop-down menu shown.



- **Managed Checked-out Files (Libraries Only)** This allows overriding and taking ownership of files that are checked out. Used by administrators, this can be used to forcibly check a document in so that it can be accessed.
- **Workflow Settings** This option allows you to create new and access existing workflows for the list. See the full usage of workflow below.

- **Information Management Policy Settings** This option allows you to specify an IRM auditing policy for this list. The auditing policy sets the information about the list and items within it are tracked (recorded), so they can later be reviewed for auditing and usage reporting. The default option is None, but you can also select a policy that is already defined (done in Application Management by an administrator), or you can define a custom one for this list. If you choose to define a new Policy, you will enter the following information:
  - **Policy Name** This is a descriptive name that describes what the policy is for.
  - **Policy Statement** This is a description of the policy, including its purpose, and so on, so users know what to expect.
  - **Labels** This enables tagging items with labels to add additional information as needed (for example, marking items as Confidential). These are added on to the metadata for the item and can be viewed like any other column. When selected, you can format the label (height, width, font, and so on). The label can also include place markers for data like {Date} to indicate the current date.
  - **Auditing** This enables auditing of specific actions within this list. It also adds to the logging and auditing records for reporting. Each kind of event can be logged (for example, check-in).
  - **Expiration** This enables you to set an expiration policy for the items in the list; for example, a meeting list might drop off records for past events automatically. This can also be set to invoke a workflow when an item reaches the expiration limit.
  - **Barcodes** This enables/disables bar coding of items. You have the option of prompting users for one before saving or printing. Just like Labels, the Barcode and the Barcode value can be used like any other column.

Communications

- **Incoming E-mail Settings (Libraries only)** This provides the settings for a document library to receive e-mail and e-mail attachments:
  - **Incoming E-mail** This enables/disables whether the library can receive e-mail and the e-mail address (use a name similar to the library itself).
  - **E-mail Attachments** Sets of these include:

    Save All Attachments to Root Folder This means that any attachments are stored in the top folder (root).

    Save All Attachments in Folders Grouped By E-mail Subject This directs attachments based on the Subject (like an Outlook rule).

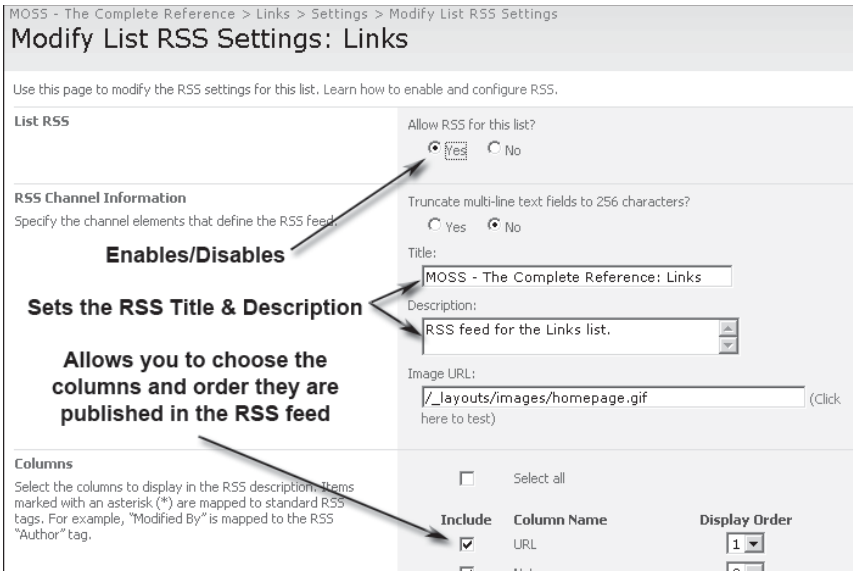
    Save All Attachments in Folders Grouped By E-mail Sender This redirects attachments based on the Sender (like an Outlook rule).

This also sets:

Overwrite Files With the Same Name This enables/disables duplicate files from being received.

- **E-mail Message** This enables/disables saving the original e-mail message.
- **E-mail Meeting Invitations** This enables/disables saving meeting invitations.
- **E-mail Security** This sets the security level for e-mails received, either based on the library permissions or from any sender.

- **RSS Settings** This option allows you to enable and specify the settings for this list to be used as an Really Simple Syndication (RSS) Feed. An RSS Feed is basically XML that can be accessed on the Internet, and it's often used for news tickers and similar items. SharePoint lists provide this same capability, as shown:

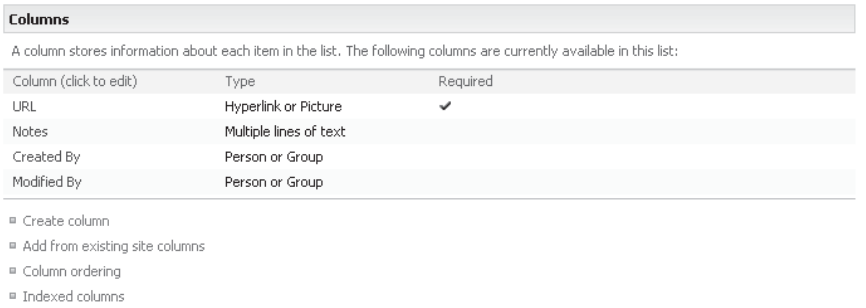


As you can see, you can specify the title, description, and so on, as well as specify which columns are to be included in the feed. The Item Limit section (not shown) sets the values for Maximum Items to Include, which is set to a default of 25 (this is the max number of items that will be published to RSS), and Maximum Days to Include sets the number of days an item will be displayed (7 means 7 days from post date).

List Columns

List columns are the actual data that is contained in a list item: in either a list or a document library. In lists, they are the data itself, whereas in libraries, they are the supplemental metadata that goes with the document (often referred to as tagging). Columns within items (or rows) are the same as using a database table. Each item represents a row and each column a field in that row.

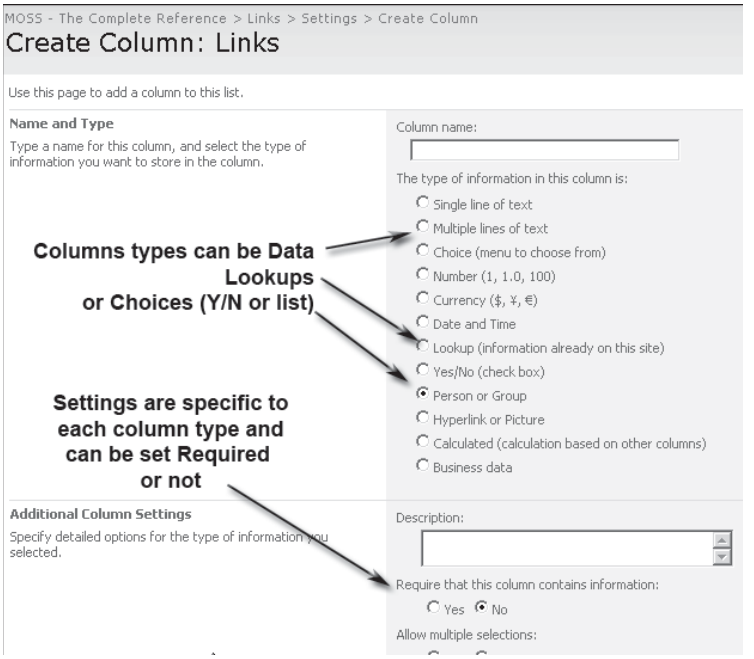
Columns can be most any kind of data and can inherit from Site Columns. There are also some special fields used for Publishing (such as the Full HTML Editor type). Columns can have conditions like "required," and can be text, numbers, or dates, or can be cross lookups to other data in the site (such as a Site Column, the Site Users, or another list). Columns can also be made as a choice (yes/no) or a list of choices. From the List Settings page, you can view the columns in the list:



Though not shown here, you may see another section called Content Types; this only shows if the Manage Content Types in the list's Advanced Settings have been enabled. Column options are broken down in the following sections.

Create Column

Create Column allows you to add a new field to the list items. Click the Create Column link to display the Create Column page:



From here, you can specify the name and description of the column and the type. Avoid spaces and special characters and use descriptive names; for example, "Review Status" is much better than "Status." The column can be any one of the following field types:

- **Single Line of Text** This is a single text field up to 255 characters.
- **Multiple Lines of Text** This allows multiple lines of text using a rich text editor; this is good to use for descriptions, comments, and so on.

- **Choice (Menu to Choose From)** This creates a drop-down list selection of the manually entered choices (typical for static options like Pending, Approved, and Rejected).
- **Number** This is any numeric value, with or without decimal places.
- **Currency** This is a momentary value.
- **Date and Time** This is a date, or a date and time selection. It can be defaulted to the current date.
- **Lookup (Information Already on This Site)** This allows you to look up a column in another list or library in the site or site collection. This is a great way to use custom lists to create rich functionality in others; the lookup provides a drop-down column in the list that uses it, eliminating manual entry—any list available can be used.
- **Yes/No (Check Box)** This is a simple True or False flag.
- **Person or Group** This provides a drop-down list that allows selection of a user or group. This is very handy when you are creating something like an issues list where the item must be assigned to someone. By using this lookup, the user can select from a list and not have to type in the person's name or login.
- **Hyperlink or Picture** This provides a hyperlink column and can also be used to display an icon from the URL specified (like an image stored in the Site Collection Images library).
- **Calculated (Calculation Based on Other Columns)** This allows you to create calculated columns (sum, dif) based on values in other columns (these columns cannot be edited).
- **Business Data** This allows access to Business Data columns provided by the Business Data Catalog. These are like any other column in SharePoint, but are derived from external data sources (such as databases or web services).
- **Publishing Field Types** There are several additional field types used for Publishing (not shown); these are available only through Site Content Types intended for use with publishing pages.

Below the field types are the options for the fields in general as well as for the specific type you selected. For example, if you selected a Choice field, you will have a place to enter in those choices and to optionally set a default choice. For numbers, you can set the limits (high, low, max digits, and so on). All of this is similar to using a database such as Access, SQL Server, or even Excel.

Also at the bottom, you will see a check box that says, Add to Default View. If this is checked, the column will be automatically added to the default List View and simply appended as the last column in that view (to change this you have to edit the view as shown below). Once you have set the options, click the OK button to create the column. Adding a column only affects new items and existing items will remain unchanged. If it's a required field, it will be required when creating a new item, but also when editing an existing item. Also, if the field is to be mandatory for all items, it will be necessary to edit the existing items to fill the data.

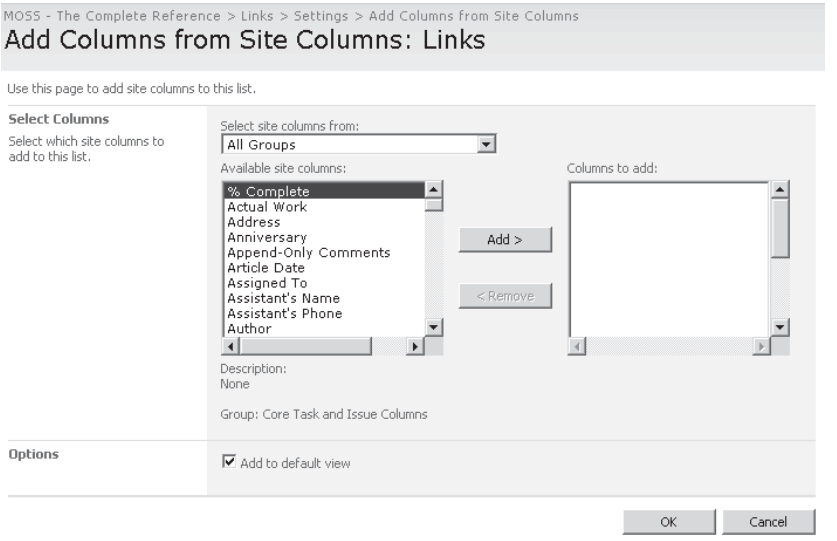
Edit Columns

Editing the properties of a list item column (or deleting one) can be done from the Edit Column page by simply clicking on the name of the column under the Columns section. The Edit Column page looks basically like the Create Column page and allows you to change the name and description. You can even change some data types to others in some cases (not always, as it may conflict with SharePoint's layout; the workaround is to remove the column and add it back). A few notes about editing columns:

- If you remove a column and re-create it, you will need to reset the order of the column.
- Deleting a column removes the data in all list items and cannot be restored.
- Modifying a column data type may cause problems if existing data does not have the same format (like strings or numbers) or it may delete existing data.
- Not all column data types can be changed (for example, lookups cannot be changed).
- The name of a column is set when created, renaming changes only the name shown.

Add from Existing Site Columns

Site Columns (covered in the Content Management chapter) are columns defined at the site level (above lists). This allows Common Column definitions to be used across all lists and libraries in the site; it is defined in the site, and then inherited by lists and libraries. Data can also be added to the column, which makes it available in all lists that use it. From the List Settings, clicking the Add From Existing Site Columns link will display the list of columns you can choose from:

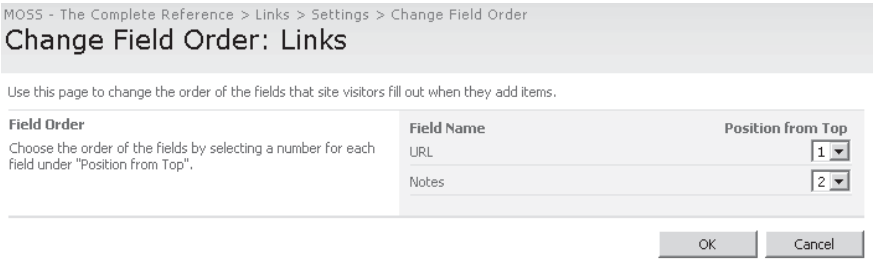


As you can see, there are quite a few columns already there (and several more in MOSS). From this page, you can select any number of site columns in the Available Site Columns

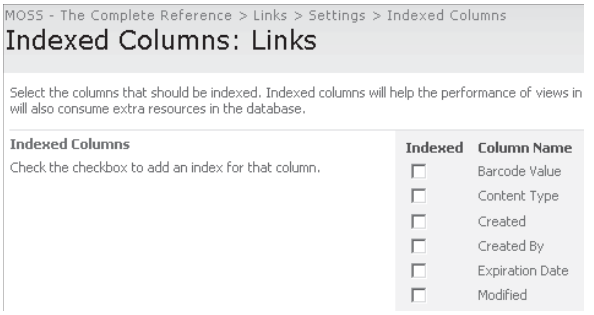
window by clicking on the name (control-click to select more than one) and then clicking the Add button. To remove an item from selection, click on the name under the Columns to Add Group and click Remove. Notice that you have the option here to include the column(s) on the default view (just like adding a regular column). When you have made your selections, click the OK button; the column(s) will be added, and you'll be returned to the List Settings page.

Column Ordering

Column ordering sets the order in which fields appear in the list and sets the order of fields that appear in the New, Edit, and View forms. This does not affect List Views as they have their own order (shown below). The order is based on 1 being the first field displayed from top down in the Item View and left to right in the List Views. Using the drop-down next to each field, you can set the order. Clicking the Column Order link will display the Change Field Order page:



Indexed Columns

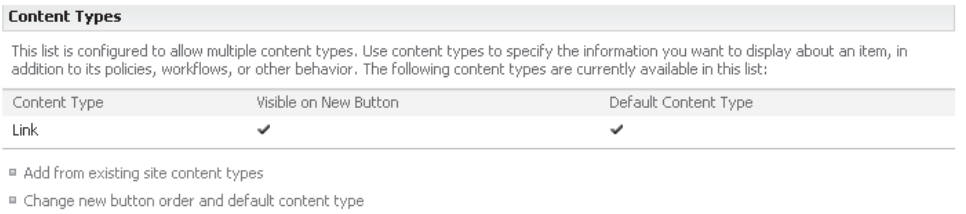


Another example of the similarity of lists to database tables is the fact that list columns can be set as indexes to increase performance. If you click on the Index Columns link, this will display the Index Columns page, which shows all of the columns (in the list) that can be indexed:

Notice that not all of the columns are available, and additional ones are not included in the default view. The ones shown are those that can be indexed (such as the Barcode column). Setting a column as an index may or may not help the performance overall, because while it can add speed, it also increases memory usage and database storage. The best recommendation I can provide is to focus on how your views will display the list data and decide what will be the most common ordering.

Content Types

Content Types allow groups of columns to be joined together to create profiles of data. While you get into this with the Business Data Catalog (BDC), suffice it to say that instead of adding a single site column, this is like adding a group of site columns that have been combined to create some kind of functionality or "a list definition without a list." Whether you see this or not depends on the List Settings; if Manage Site Content is enabled (under the General Settings | Advanced Settings option) this is shown, but if not, this group is omitted:



In this example, you can see that a content type of Link is included in this list. The link happens to be two columns: a URL and a Title. Instead of having to define both of these, the Link Content type gives you the combined functionality you need in a single definition.

List Views

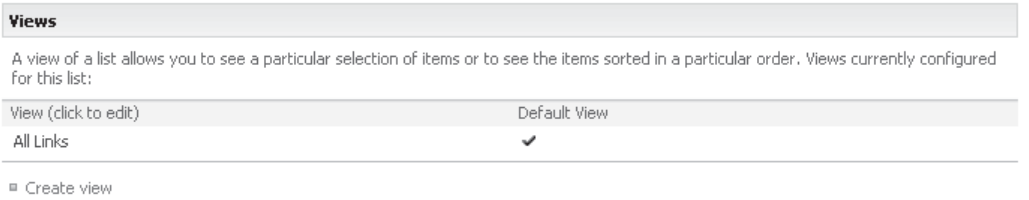
List Views provide a dual role in lists. Within the list itself, they provide ways to slice and dice the list items, allowing you to filter, sort, and group the view. When viewing a list, the different views become a sub-navigation on the left-hand portion of the page allowing the user to choose any view they want. Conveniently enough, these same views are also available to the List View Web Parts on a page, because they can be set to use a specific view.

The key to views is to remember that they are just views of the list data. Regardless of the type of view you create, the fields required for creating new items and other such details remain the same; views don't change the underlying list. In reverse, however, changes to a list will affect the list's views—if a column is removed, it is removed from the views as well. If a view is grouping on a deleted item, everything will appear under an empty group. If you are communicating with other web parts (covered below), connections can be broken if the joining column(s) are deleted.

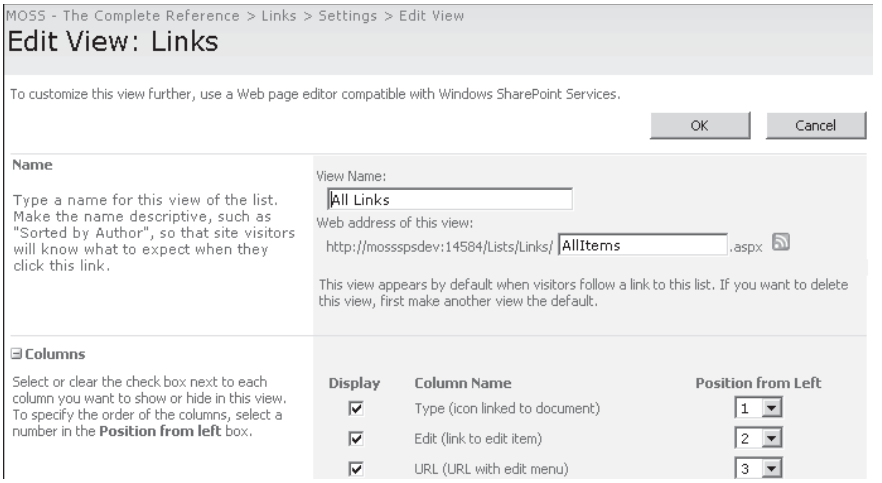
To handle views, SharePoint actually maintains an aspx page that is used for the display. When a list is created, a default view called All Items (items may be links and so on depending on the list type), and subsequently, a file called allitems.aspx is created for the list. When you create a new view, a new aspx file is created (with a matching name). One view must be designated as the default, because this is used repeatedly. As you saw when you added columns, there is always an option to Add to the Default View. When a list is created, the default is the All <whatever> View; you can change this to any view you create.

Create/Edit View

Using views in a list is done from the List View itself; to access these for editing or to create new ones can be done via the List Settings page at the bottom of the Views section:



Creating or editing a view uses the exact same page; you can create a view by clicking the Create View link or edit a view by clicking on the name of the view in the list. Clicking the All Links View in the example will open up the existing view definition:



As you can see from here, you can modify (if editing) or enter (if creating) the name and the URL for this particular view (note that if you change the URL a new aspx page is created to match it). If you're using MOSS, the next section sets the audience (only shown when being created) as either a Personal View (available to users to customize) or a Public View (preset for all users).

Below that, there are check boxes, column names, and a position drop-down, which, if checked, include the column in the view. The position drop-down sets the order (from left to right) of the column in the view. Note that columns that indicate "linked to" will provide drop-down menus when being viewed by the user. For example, Linked to Edit means that a drop-down menu with Edit as one of the options will be enabled, and when a user is viewing the item, moving the mouse over the name shows the drop-down menu. Columns selected are cumulative; the more columns, the more width the view will take. While a wide view of data can be useful, consider making many short views and then using multiple List View Web Parts for arranging on a page.

The next few sections deal with the look and feel of the actual view; be aware this is not a full-blown style sheet. However, it does provide limiting data, presentation, presorting, and grouping information:

- **Filter** This is made up of the following options:
  - **Show All Items in This View** This basically shows every item in the list.
  - **Show the Items Only When the Following Is True** This allows you to set up anything from a simple to very complex filter of the data, based on an "if", "and/or" decisional tree with columns against values you choose. Conditional options include the basics, such as Is Equal To, Is Not Equal To, and so on, but

they also include comparisons such as Contains and Begins With. Any number of conditions can be created.

- **Sort** This enables/disables sorting by users. If enabled, users can click on column titles in a view to sort by that column. This also allows you to set the default sort for as many columns as you like (either ascending, as in 1, 2, 3, which is typical for most lists, or descending, as in 3, 2, 1, which is typical for calendars).
- **Group By** This provides grouping in the display by one or more columns in the list. An example would be a column called Region. When displayed in the list, Region would appear as a highlighted line with all items that matched that region underneath it (this grouping display can also be expanded and collapsed by default). Note that the column must be able to be used for grouping, because if it isn't, SharePoint will not display it. To facilitate the grouping, you can also set a limit on the number of items to display (think "top five").
- **Totals** This option allows you to create total columns based on columns and/or item counts in the List View.
- **Style** This provides View Style options that provide different ways to view items; for example, Boxed will display each item with a box around it. These are useful but not too sophisticated; for advanced stylization, List Views support using alternate XSL.
- **Folders** This controls the way folders are shown in the list (if enabled):
  - **Folders or Flat** This is the option to show items inside folders or show without.
  - **Show This View** There are three options within this option:

    When In All Folders is selected, the view is available when "inside" of a folder.

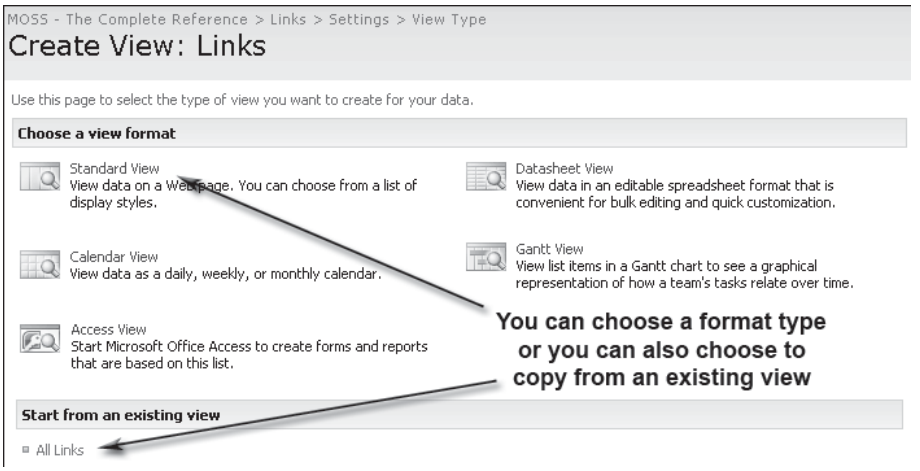
    When In the Top-level Folder is selected, the view is only accessible in the top folder.

    When In Folders of Content Type is selected, a link to this view is shown only when in folders that have a specific content type.
- **Item Limit** This sets the maximum number of items to display (paging) and the maximum of total items that can be returned (ideal when large lists create performance issues).
- **Mobile** This enables/disables the view to be used for Mobile Applications and can be set as the Mobile Default View (the mobile addresses are shown).

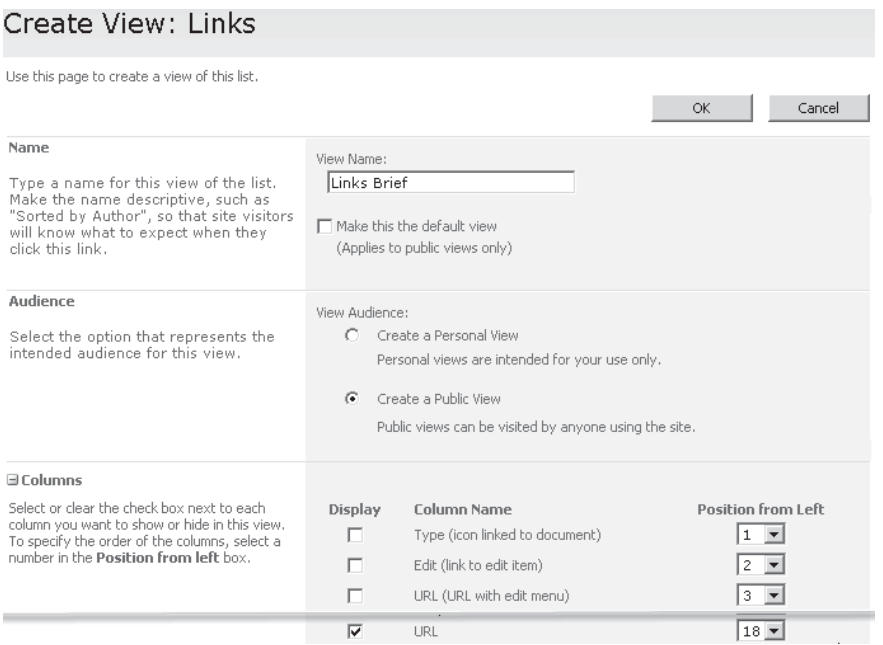
When a view is created, you have a few choices in terms of the kind of view to create. The Standard View is a simple list, just like the All Items View shown previously. Other views include a Datasheet View (like a spreadsheet) or the Gantt Chart View (only useful for task-type lists). The view type you choose merely determines the front-end presentation that the user will get. As an example, in a Links List, I've created a view that shows only Link and does not show the Notes column. The steps to do this are as follows:

1. From the List Settings page, click Create View.

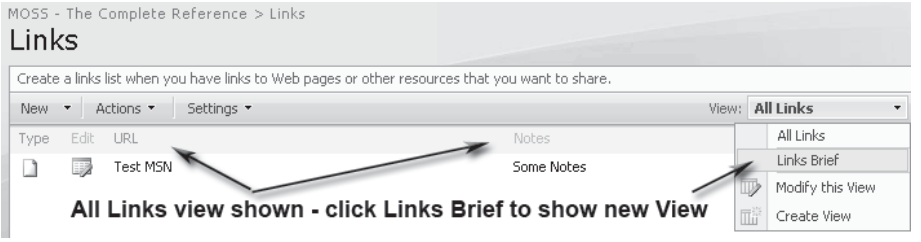
2. From the View Type page, click Standard View.



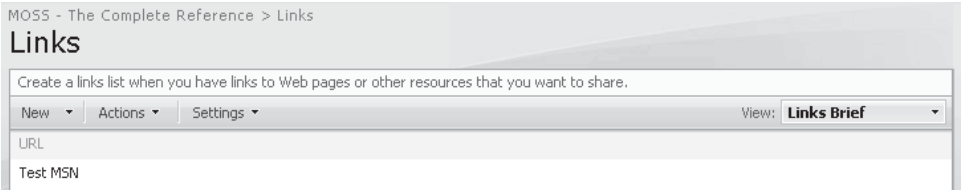
3. Set the title of the view, such as **Links Brief**, and leave the URL as set.
4. Set the type of view whether personal or shared (this is a one-time setting).
5. In the column choices, uncheck all shown, and then scroll all the way down to the end of the list of columns and select the URL column. This column is the flat display version of the URL and does not have menu (using the View column URL, specifically the URL with Edit menu, provides a drop-down menu that includes the editing of the item. If making a display-only version of this, you do not want the Edit menu available):



6. In the drop-down selection of the column order, notice that it shows 18. Because this is the only field in this view, this setting does not matter (in fact, if you open the view for editing after it is created, you'll find that the number is reset to 1).
7. You can set any additional options you like and then click OK to save the view; when you return to the List Settings page, click the link (in the breadcrumb navigation) at the very top of the page to return to the list. When you see the List View page displayed, you will see the new view listed in the Views drop-down:



8. Selecting the new view will redisplay the list using the new view:



As a quick summary on views, you can see there is quite a bit of flexibility out of the box. You can choose to include columns or not, make them sortable, filter them by a specific column, and so on. The views are also available when you are creating List View Web Parts. The best part about this is that you can create as many views as is useful to limit what data users see. "Less is more" when it comes to targeting information.

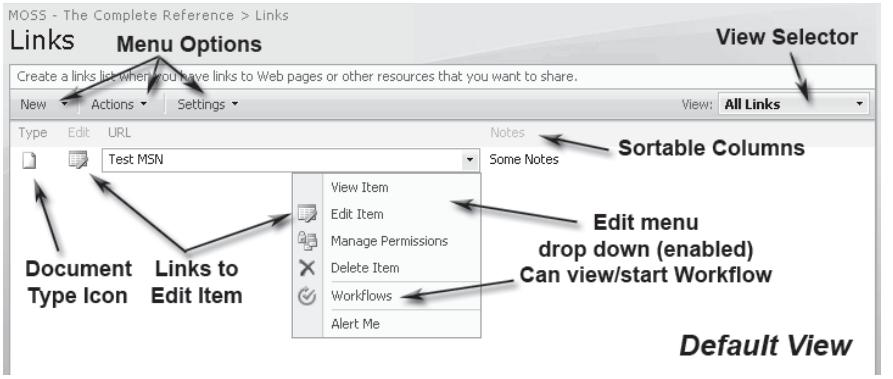
Using a SharePoint List

All of the SharePoint lists operate in the same basic way and provide their own user interface as appropriate (Libraries have an Upload option, for example). You can see the complete list of list types in Table 11-1. Key points here are:

- The list UI is based on creating, viewing, editing, or deleting an item.
- All list items (if enabled in the list) can have attachments.
- Attachments in regular lists are not indexed.
- All library items are items with attachments but these attachments are indexed.
- Views can be changed at any time, but one must be designated as the Default.

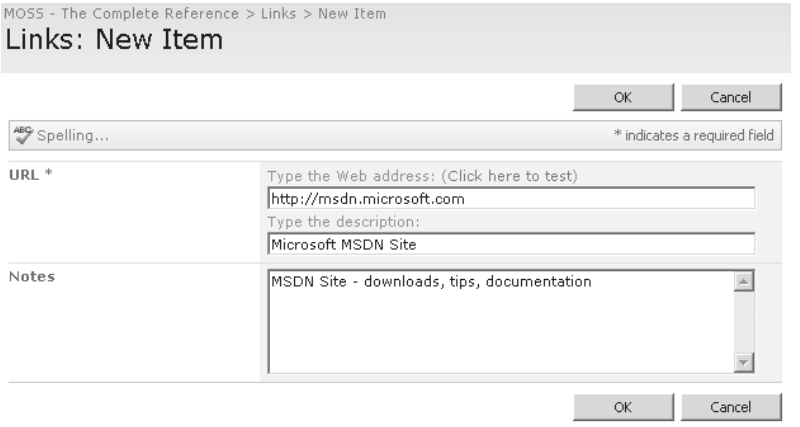
So let's return to the Home Page (default view) of the Links list shown in the previous examples. To get there from the List Setting page, you can click on the breadcrumb link.

Or, from the Site Home page, click the View All Site Content and then click Links on the Content page:



This opens the designated default view. The view of the items also provides menu options:

- **New** The new menu provides links to create a new item or create a new folder (New folder only shows if folders are enabled). To create a new item, click the New menu and then select Link from the drop-down menu. This will open the New Item page, which shows all of the enterable columns in the list (a red * next to an item indicates it is required):



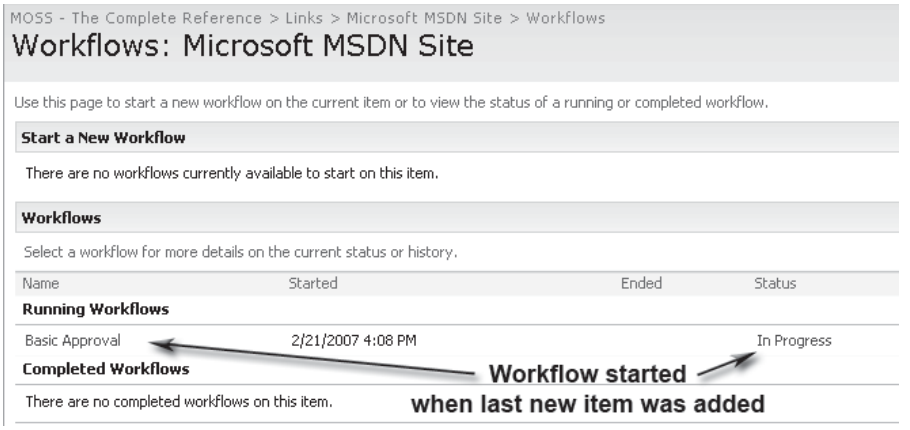
- **Actions** The Actions menu provides a host of things you can do with this list:
  - **Edit in Datasheet** This reverts the list to a spreadsheet (row/column) format for editing. It also allows multiple items to be changed, but with all changes saved at one time (easier than editing a single item at a time).
  - **Change Order** This allows you to change the order of the list items and reorder them to taste (very useful in a links list!).
  - **Export to Spreadsheet** If enabled, this will export the data in the list to an Excel spreadsheet that can then be saved as a workbook file; note that lists can be created by importing a spreadsheet as well.

- **Open with Access** This will open the list in Access (if available), which provides the rich editing interface. You can work with the list as if it were a database table.
- **View RSS Feed** As the name implies, this will open the list as it is shown in RSS.
- **Alert Me** This option allows users to set an Alert on a list (or an item) to notify them when something has changed.

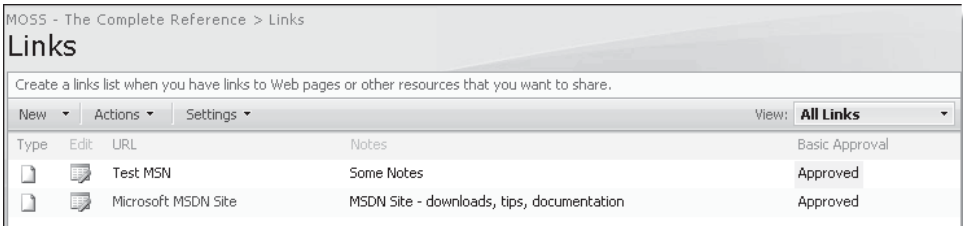
- **Settings** The Settings menu, covered previously, gives you the options to Create Column, Create View, or Open the List Settings.

Also on this page, you have access to the View Selector, which allows you to change views and sort columns by clicking on the heading (if enabled). You also have the Item Edit drop-down menu (if enabled in the view), which provides access for viewing, editing, and deleting, as well as two other options:

- **Manage Permissions** This allows for the setting of permissions (rights) for groups and users for this line item. The permissions assignments are inherited from the list by default, and the settings here enable you to modify them for this item.
- **Workflow** This provides a current view of the workflow, alerting you of any that are active, and if there are any available to start manually:



If you click on the existing workflow, this will display the Workflow Status page:



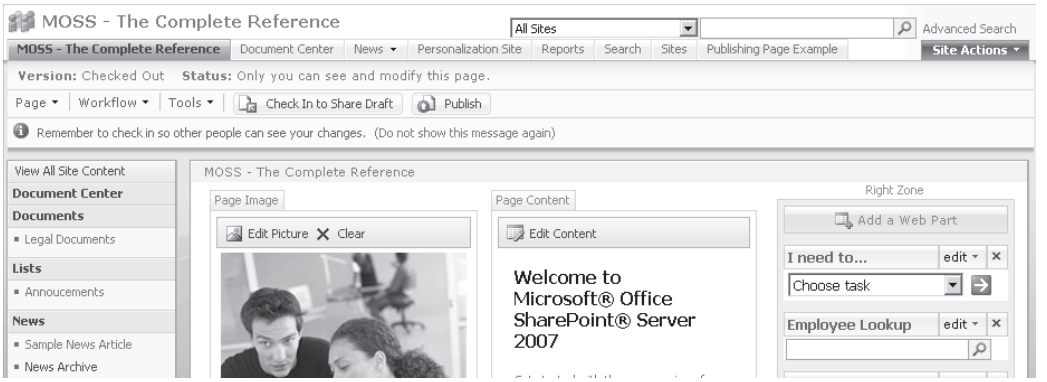
From this page, you can manage the task, cancel the workflow, and so on. Be aware that all of this information is derived from the Task and Workflow History lists assigned when the workflow was created.

For detailed reference on using SharePoint workflows, refer to Chapter 16.

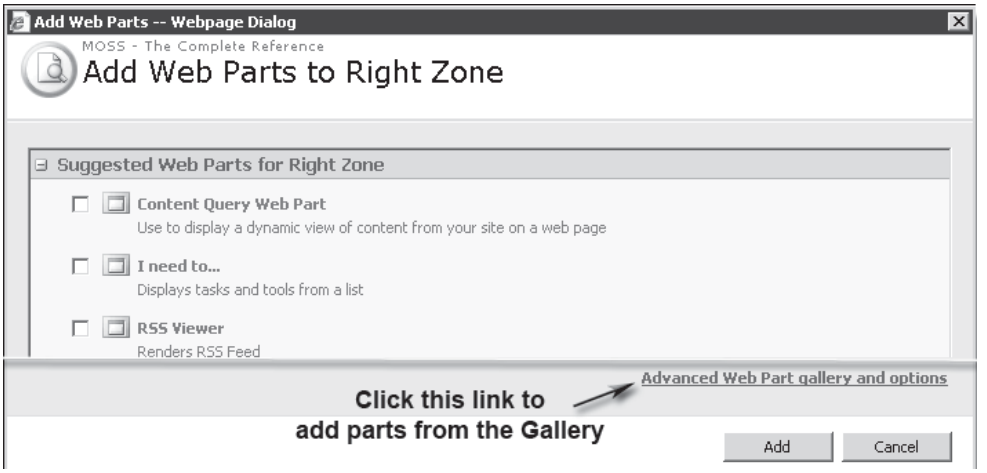
Creating a List View Web Part and Setting the View

So now that you know the guts of a list, it's time to see how you can use a list to provide content on a web page. In the previous example, a view was created called Links Brief. This shows only the URL of the items and does not provide an Item Edit menu, so all the end user can do is click the link to follow it (typical for display only). A List View can be created once the list is created (the list can be empty). The step-by-step for adding the List View Web Part is as follows:

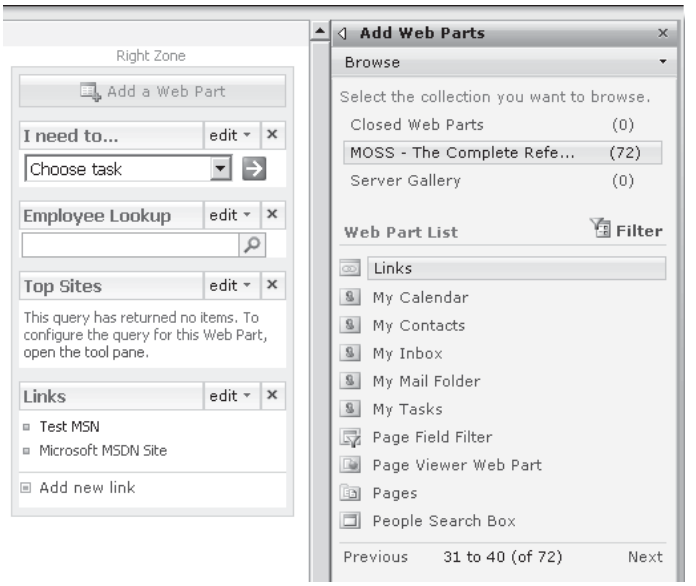
1. Navigate to the home page of the site.
2. From the Site Actions drop-down menu, click on Edit Page. This will redisplay the page in editing mode:



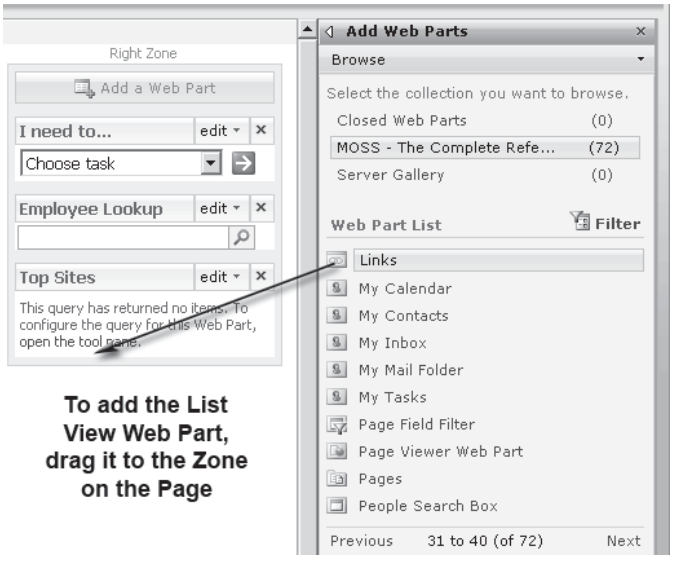
3. As you can see, this is the portal page for the Collaboration Site template (MOSS). In addition to the Web Part Zones being displayed, you can see that the Publishing menu is also displayed (though not shown in WSS). To add a web part, click Add a Web Part link in the zone you want the list displayed in. This will open the Add a Web Part dialog:



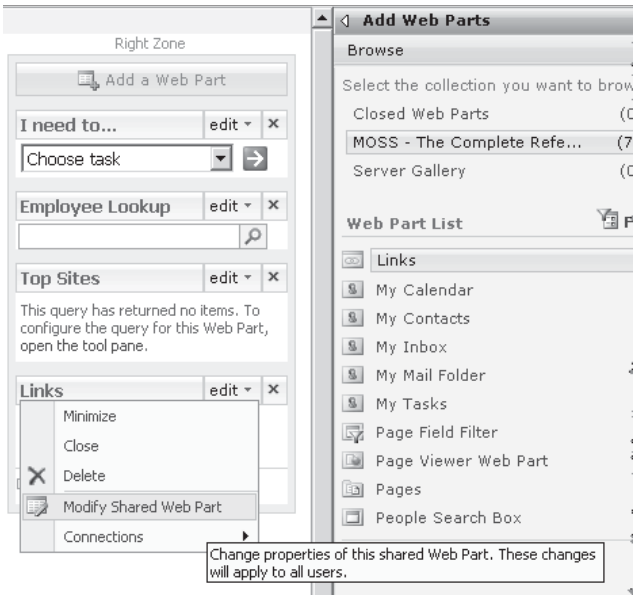
4. In WSS sites (and team/workspace sites in MOSS), the Links list will be shown in the Add a Web Part dialog (you can click the check box next to the name and then click OK) under the Lists and Libraries section. If you are in a MOSS site, this section is not shown and you may not be able to find it. In this case, you need to use the Advanced link at the bottom of that dialog (using Advanced works for both WSS and MOSS). Click that link to open the Web Part Gallery panel.



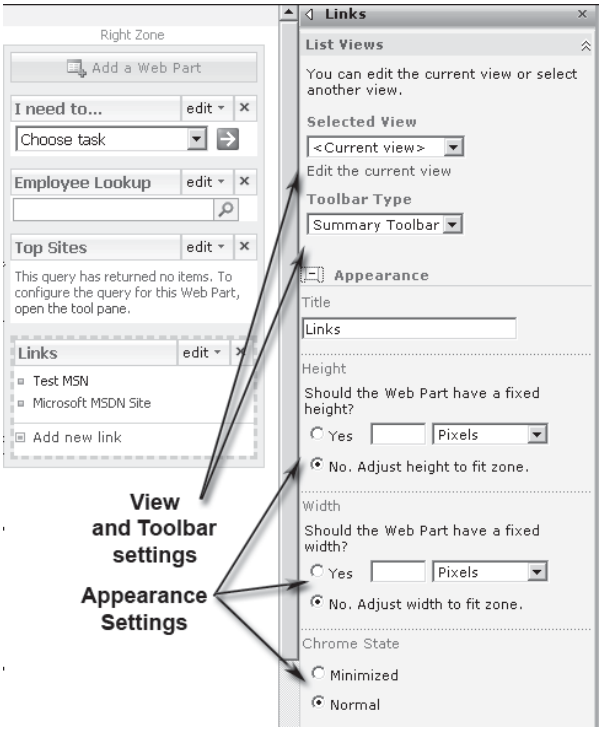
5. There are three Galleries: one that holds parts that have been closed (not deleted) from the page; the Site Gallery, which holds all out-of-the-box lists and web parts; and the Server Gallery, which is for third-party or custom web parts. The Site Gallery is selected, which is what you want—you can see under it that there is a list of web parts (I've paged a few times to get the Links list). To add the List View Web Part to the page, click on the Name in the Web Part List and drag it to a Zone on the page. When you are inside of the zone, a line will appear; release the mouse and it is added:

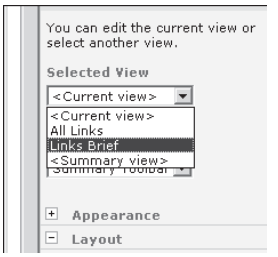


6. As you can see, the Links list, as shown, shows a view you've not seen. This is a default view built into the web part called the Summary View, and it varies slightly based on the list type. In the Links List type, this happens to be a square image and a URL link. Notice also that it has an Add New Link link at the bottom, which indicates that the toolbar for this web part is enabled (also meaning that contributing users can add links). To alter the view, you need to open the web part properties. This is done by clicking the Edit menu on the web part and then selecting Modify Shared Web Part.

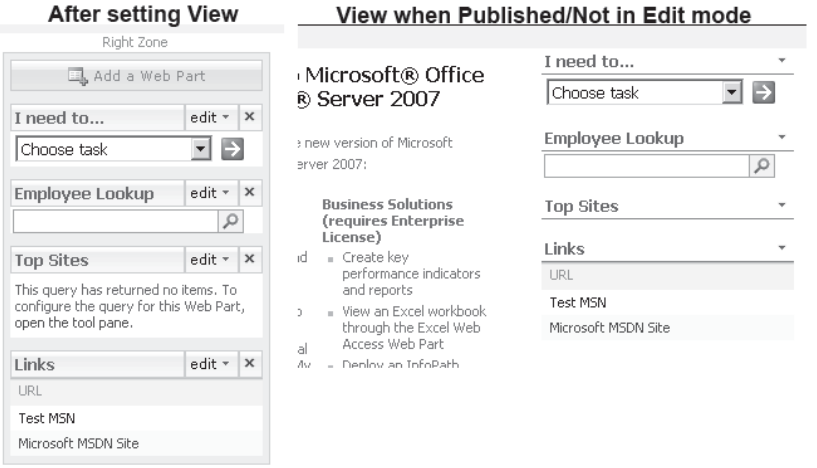


7. When the Web Part Properties panel opens, it will look something like this:



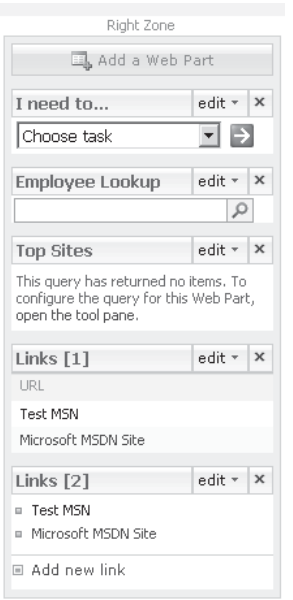


8. There are three standard groups that are always included with web parts. I've expanded the Appearance group here to show some of the additional settings, such as title, height, width, and chrome. (Chrome refers to the web part title bar; it can be either shown or hidden and can either create a box around the outside of the part or not.) Other settings include Layout Settings (the zone it's in and the order displayed) and Advanced (sets whether the view can be minimized, moved, and so on). For now, we're interested in the View and Toolbar Settings, but first we need to change the view to the one created.
9. By the way, the <Summary View> you see is the generic default view based on the list type (not the default in the list itself). When you change the view, you receive a warning message that says, "Switching to a different view removes changes you have made to this view, and may disable Web Part connections that depend on columns in this view." This is warning you that any properties you have changed will be reset and that if you've used connections (explained later), they might be broken. Simply click OK, and then change the toolbar selection from Summary Toolbar to No Toolbar.
10. Scroll down to the bottom of the Web Part Properties panel and click the OK button. (Here's a quick description of the options offered here: OK applies the changes and closes the panel; Cancel discards any changes; and Apply simply applies the changes but leaves the panel open.) When the page refreshes, you will see the List View Web Part displaying the view selected. When you publish the MOSS page or exit editing on the WSS page, the List View will appear flat on the page and the zone will disappear as shown:



11. Now as a final demonstration to prove that the List View Web Part is merely a view of the list, click Edit Page from the Site Actions menu again. When the page enters edit mode, click the Add a Web Part link.

12. Then, on the Add Web Parts dialog, scroll to the bottom and click the Advanced link. When the Web Part Gallery panel opens, page down until you find the Links list again and then drag it to the page. When displayed, you will see that a) the titles of both web parts have changed adding a [1] and [2] to the titles and b) that the List View Web Part just added [2] shows the default Summary View and Summary toolbar (the default settings):



13. While for most practical reasons you don't need to have more than one view, this demonstrates that you can have the list as a content source to provide different views on the page. You can even communicate between them, using one to filter the data in the other but using the same list.

Using Libraries

Libraries (all types) are used to store documents within SharePoint. Libraries themselves are simply lists with attachments, except that the actual document content is indexed by the search engine. While good for searching, the Item/File relationship can be very important; however, the List Item and columns represent a crucial part of the "metadata" for documents in SharePoint (in this case, metadata means background information that identifies what the document is and what it's for).

This is important because the true metadata of a document depends on the application. In Word, for example, you can select Document Properties and add supplemental information

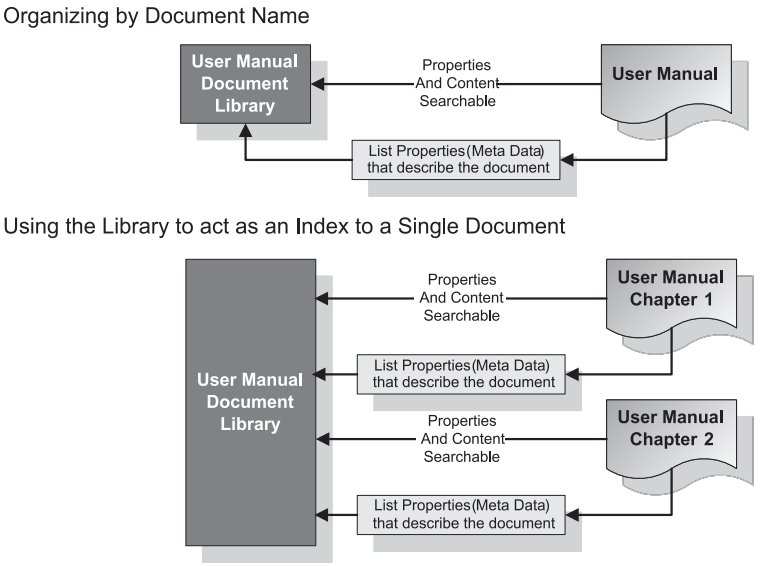
about the document: who authored it, keywords for searching, what category it belongs in, and so on. (Excel, Visio, PowerPoint, and even PDF files also include document metadata.) If such information is included, SharePoint's search engine will pick it up, though users are rarely disciplined enough to add it. The SharePoint List Item that goes along with the document makes up for this. By adding columns to a library that help in identifying and describing a document without having to open it, you've just saved yourself and your users a whole lot of time.

Using document libraries also requires some discipline; many users are familiar with only standard file systems and some find it very hard to equate how to convert from files and folders into libraries and folders. In reality, SharePoint libraries are flat because they, like lists, are actually pseudo-database tables. To help users (though I don't agree with this), Microsoft added the visual use of folders inside of a document library to help users adjust.

I prefer to use the List Item (metadata) to manage and organize documents since it is cleaner, doesn't promote duplication (folders allow users to upload a document to multiple folders in the same library), and doesn't require knowing the "method" used in naming or usage. My experience is that 90 percent of the time folders are enabled, and the users spend countless wasted hours desperately trying to reinvent the same mess they had on Network file shares. As a best practice, think through organizing documents by their metadata and do not use folders.

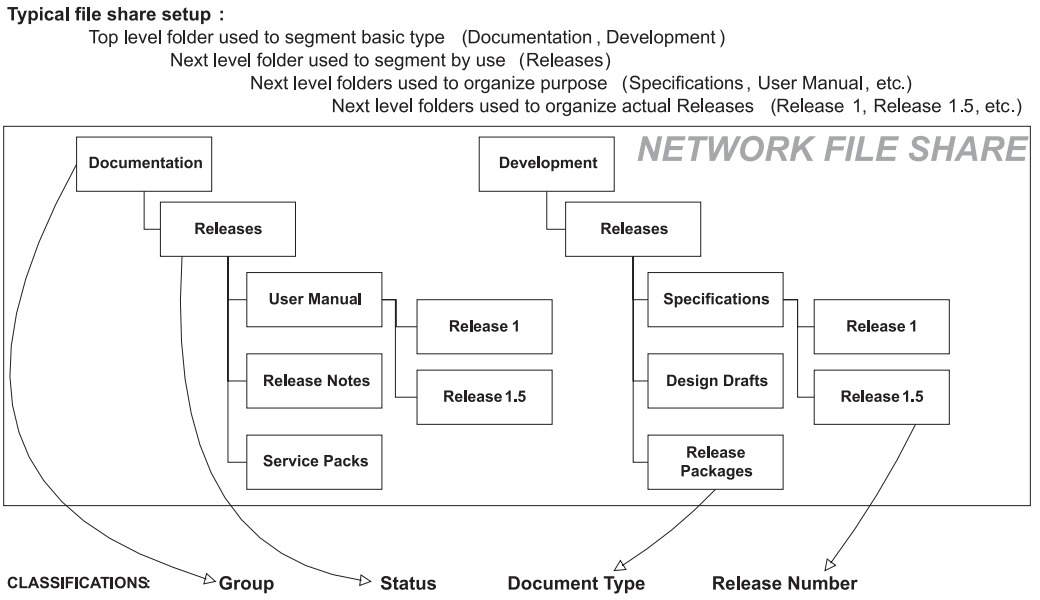
There are two aspects to document storage: the first (and toughest) is how to convert the thinking of folder organization into document libraries, as shown in Figure 11-1.

The second aspect of this is looking at how to use data to create relationships between documents instead of folders and libraries as shown:



Using either method simply provides different views however, each needs Meta Data (List Columns) that describe the documents and make them easier to filter on and search.

I highly suggest you work with users on the previous principles and avoid using the features that will cause pain down the road.



MORE ABOUT META DATA :

* Document Libraries are “lists” with documents as attachments
* The Document Library “List” itself is considered the “Meta Data” about the documents it holds
* The List Columns are used to describe a document and its context
* The better the descriptive information, the less likely the document would need to be opened to know what it is
* Meta data is searchable and more reliable than simply the contents of the document
* Meta data enables you to describe documents that have no searchable content (Images, etc.)
* Meta data enables Views and Filters to be used to quickly locate related documents

FIGURE 11-1 Converting folders to document libraries

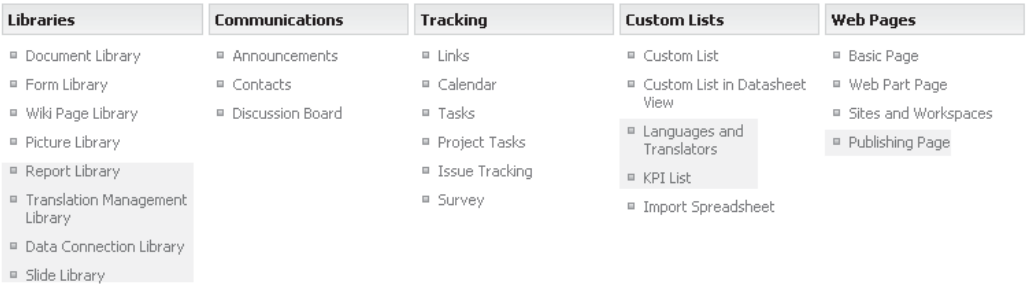
Migrating Documents to SharePoint

A word of advice on SharePoint documents in general. If you are implementing SharePoint and starting to move away from local drives and file shares, you should take a pragmatic approach to this. The natural inclination is to move everything lock, stock, and barrel up to SharePoint and then figure out how to sort it out later. This is a very bad move.

Rather, you should work with users to be sure they understand the consequences of uploading documents, how to organize them, and so on, before setting a policy for what gets uploaded. Also, while you'd think it wouldn't be necessary, make sure to qualify for your users what should and should not be uploaded to SharePoint. While SharePoint sites can be very secure, users should be very aware of what is considered appropriate and inappropriate material.

SharePoint List Types

WSS and MOSS both include the same core set of SharePoint lists, including the Custom List which is used to build virtually any list. MOSS has a few additional lists that are part of the Enterprise Features (like the KPI List). All the lists work the same and provide add, update, and delete options. Views add nice ways to display the information, but the core functionality is the same. Creating lists is always done from the Create page; the direct route to this is the View All Site Content link above the Quick Launch Bar (or from the Site Actions menu). On the Contents page, click the Create link to display the Create page (MOSS specific items are indicated):



There is no best use for a specific type of list; if it does what you need it to do (or can by adding a column or content type), great but you can always create a custom list if needed. For storage, be aware that lists support any kind of document (if not prohibited); you can easily upload and Excel and Word documents into the same library. Lists fall under certain groups: libraries are for storing documents, lists for list information, discussions for discussions, and so on. In all cases, the uses for these lists are exactly the same; the only difference is the columns that each list includes. The out-of-the-box lists and intended use for each are shown in Table 11-1.

Library Type	Used For
LIBRARIES	
Document Library	Used to store any kind of document.
Form Library	Used to store forms (InfoPath) documents.
Wiki Page Library	Used to store Wiki aspx pages; this has additional metadata specific for Wiki pages; the library provides a custom view that appears much like a new site.
Picture Library	Stores images and provides a picture viewer view. These are best for using pictures that will be used as content in the site. Note that these lists are ideal for using images on a website since images are accessible by the relative URL. This means that they can be accessed in links, content editor parts, and so on—all within the confines of SharePoint's security.
Report Library	Used to centralize reports and key performance indicator information used to communicate metrics, goals, and business intelligence information.
Translation Management Library	Used to hold documents to be used in a managed translation process; it has a built-in workflow for managing the translation process, subfolders for storing versions, and so on.
Data Connection Library	Used to store the data connections created for the site; data connections (or profiles) define external data sources and can be used to connect to those sources. As files, security and connection information can be embedded in the definitions.
Slide Library	Stores images and provides a "slide show viewer" view. These are best for using libraries that will themselves be the content.
Pages Library (not shown)	This is a system-type library created for content management and use in MOSS only. This library is created by the system automatically; you cannot create or save this as a template.
COMMUNICATIONS	
Announcements	Used to provide a date-based announcement list and items can be set to expire.
Contacts	Provides a contact list similar to the one in Outlook (and they can be linked). This is used to contain non-SharePoint user contacts.
Discussion Board	A general discussion thread list that allows queries and responses based on subject.

Table 11-1 Summary of SharePoint List Types

Library Type	Used For
TRACKING	
Links	Provides a simple list of URLs with friendly titles and a variety of views; best used to provide common or subject-related links (great for development sites, project sites, and so on).
Calendar	Provides a full calendar list that allows scheduling of events, meetings, and anything else based on from/to dates. Includes all day events and recurring events.
Tasks	Provides a task list identical to the one in Project. It allows creation of tasks and assignments to users with due dates, percentage completed, and so on. Used most often with Workflow Tasks.
Project Tasks	Provides a task list identical to the one in Project. It allows creation of tasks and assignments to users with due dates, percentage completed, and so on. This is used most often with Project tasks.
Issue Tracking	Provides an issue tracking list with date posted, issue description, and resolutions.
Survey	Provides a multi-question survey format with single or multiple responses and provides export of the results to spreadsheet format.
CUSTOM LISTS	
Custom List	Creates a blank list with only one column called Title by default; intended for use to create any kind of list desired that is not already available.
Custom List in a Datasheet View	Creates a blank list with only one column called Title by default; intended for use to create any kind of list desired that is not already available.
Languages and Translators	Creates a list that can be used to add translators (individuals) to be assigned to specific language translation workflows (users get assigned when the workflows are kicked off).
KPI List	Lists to provide the data to KPI display web parts to measure the metrics set (high, low, etc.).
Import Spreadsheet	Imports an Excel spreadsheet creating a SharePoint List.

Table 11-1 Summary of SharePoint List Types *(continued)*

Summary

In this chapter, you saw a complete overview of lists and libraries provided by SharePoint. You should now have a much better idea of where SharePoint actually stores content and documents and how it presents this storage through lists and libraries. In addition, you should have an idea of the power of building your own lists and how lists can provide application-like functionality. Finally, you should also have gained an understanding of the relationship between lists and list view web parts. As you move into the next chapter on SharePoint's web parts, you'll begin to see the entire picture of SharePoint's out-of-the-box features and content capabilities.

12
CHAPTER

Web Parts in SharePoint

Web parts are an interesting phenomenon in development. Originally built for SharePoint 2003, they proved so useful that they are now part of the .NET Framework. The whole concept behind web parts is quite simple; they provide a way to build in application functionality that is hosted by an application or framework. The typical details such as security, page layout control, and so on, is handled by the host (such as SharePoint) itself. Since web parts have the full complement of .NET functionality, they also have the ability to pull data in from external sources. In the SharePoint platform, they can also access all of the data in SharePoint including accessing specific items such as user details. The results are mini-applications that provide functionality in their own right and because they are hosted, they can work both inside and outside of SharePoint.

The real beauty of web parts is what they give to the users. The benefits include drag-and-drop capability for designing their own pages, the ability to put functionality where they need it, and the ability to leverage SharePoint to control otherwise straight web-based applications.

A good number of web parts are provided out of the box with WSS and many more with MOSS (including Filters and those used with the BDC), all of which will be covered here. You will also find that there are several third-party web parts available. Of course, you can also build your own (covered in detail in Chapter 20).

SharePoint Web Parts

SharePoint includes a host of web parts, including a core set in the WSS Platform. One example is the Content Editor Web Part. MOSS adds quite a few additional ones such as Filter Web Parts (a set of communicating web parts), special ones to support KPIs and BI dashboards, and a whole group that work directly with the Business Data Catalog (BDC). One thing to note: while I do use the term *web parts,* that name is losing significance. Web parts built specifically for SharePoint are considered SharePoint Solutions, and they are installed using the Solution framework.

Web parts loaded in SharePoint are grouped by categories and suggested use. *Categories* are simply groupings of web parts as they are shown in the Add a Web Part dialog box. In MOSS, *suggestions* are used for suggesting which web parts might be used in a particular zone (for example, if you add a web part in a right-hand zone, SharePoint will show a list of suggested web parts for that zone). These are suggested simply as a guide to choose the part that you need and where it will display best; you are not limited as to which part you put where.

Common Web Part Properties

All web parts have properties that control how they appear on the page, but they also give you a way to customize them with user-based choices. Web part properties are actually specialized settings stored with the web part. Whenever the page is called, it will apply the settings before displaying. Each web part has its own settings; even if a duplicate web part is on the page (as shown with the List View Web Part previously), each instance has its own settings. Properties can also have three states:

- **None** This is the value set for the property and is never saved (this is rarely used).
- **Shared** These are the property values set and are used as the default settings for all users.
- **Personal** Users can personalize this part and save their own settings. (The part will then render them in the way they have specified. Everyone else sees the shared view unless they have their own settings.)

In many cases, web parts will use a combination of storage; a web part might allow users to change the title, but not the database connection. Custom properties in web parts can be used to supply all kinds of settings and options and can make them more easily available to the user.

There are a number of common properties that every web part in SharePoint inherits; a bit of overhead needed by SharePoint to control where a part can be placed, the ways it is displayed, and so on. These common properties fall into three categories: Appearance, Layout, and Advanced as follows:

- **Appearance** This controls the way the part looks on a page and includes the following:
  - **Title** This sets the title for the web part as it will be displayed on the page (in the web part title chrome or title bar).
  - **Height** This sets a specific height for the part or allows it to expand and contract within the zone.
  - **Width** This sets a specific width for the part or allows it to expand and contract within the zone.
  - **Chrome State** This sets the state of the web part displayed on the page. If it's minimized, it means that only the web part title bar is displayed and the user can restore it. Normal (the default) will display the part in the normal mode.
  - **Chrome Type** This sets the border properties of the web part, which include:

    Default This shows the web part contents with a web part title bar (and web part menu).

    None This shows the web part only; no title bar or border.

    Title and Border This shows the web part with a web part title bar (and web part menu) and a border around it.

    Title Only This shows the web part with a web part title bar only.

    Border Only This shows the web part with no web part title bar with a border around it.

- **Layout** This sets the placement of the web part on the page:
  - **Hidden** This allows hiding of the part on the page.
  - **Direction** This sets the display direction for the part: None (use default), Left to Right, or Right to Left.
  - **Zone** This is a drop-down menu selection to specify the zone that you want the part to be displayed in.
  - **Zone Index** This sets the index (appearance count) of the part within the zone (0 means first).
- **Advanced** This sets the controls and enables/disables user options, including:
  - **Allow Minimize** This enables/disables the web part from being minimized on the page (when minimized, only the web part title bar, chrome in this case, is shown).
  - **Allow Close** This enables/disables closing the part. If closed, it will be put into the Page Gallery (not deleted).
  - **Allow Hide** This enables/disables hiding the part. If hidden, the part is still on the page, just not shown.
  - **Allow Zone Changes** This enables/disables the ability to drag the part between zones.
  - **Allow Connections** This enables/disables connections between this part and other parts on the page.
  - **Allow Editing in Personal View** This enables/disables whether users can customize the view of the web part in their personal views.
  - **Export Mode** This sets the type of export that is enabled for this part. Export All means that if this part is exported, all properties will be included (this does not mean the data; just the properties). Non-sensitive Data Only does not include properties marked as sensitive (such as database connection strings).
  - **Title URL** This sets a link for the title of the web part. In List View Web Parts, this defaults to the default view of the list. In other web parts, this is whatever link is desired. For example, a KPI Web Part might have a direct link to the Reporting site that holds the metrics.
  - **Description** This is a description for users about what the part does.
  - **Help URL** This can be used to set a link for help when the user clicks Help in the web part title bar Edit menu.
  - **Help Mode** This includes Modal (pop-up), Modeless (new window), or Navigate (redirect).
  - **Catalog Icon Image URL** This sets the URL to an icon to be displayed for the web part in the Web Part Gallery.
  - **Title Icon Image URL** This sets the URL to an icon to be displayed in the web part title bar next to the (text) title.
  - **Import Error Message** This sets the text to display to a user if the part cannot be added to a page.

Accessing Web Part Properties

Accessing web part properties is always done the same way. You start on the page where the part is displayed, and then use the Site Actions menu to select Edit Page. All web parts on the page will then be displayed with a web part title bar Edit menu (located next to an *X*, which closes the part). Clicking the Edit drop-down menu allows you to select the Manage Shared Web Part Properties link. Clicking this will open the Web Part Properties tool pane on the right-hand side of the screen.

Exercise: Connecting Web Parts

While it might seem a little out of order, this is a good place to introduce web part connections in all their glory. While I've mentioned (but not explained) connections previously, they really come into play when using web parts like Filters, which makes now just as logical a place as any. Connections are a way that one web part can "talk" to another. The beauty of this on page communication is that one web part can react to another even though there is no true relationship between them, such as when you start using the Filter and BDC Web Parts. In the case of web parts that manage data, like the Links List created previously, web parts can pass values between them, creating complex relationships and master/detail type displays—even between completely unrelated lists.

Communications between parts are based on fields within the lists or on other conditions. When making a connection, you have the following options:

- **Provide Row To** This means the web part will provide the list item to another.
- **Provide Data To** This means the web part will pass data to another.
- **Get/Sort/Filter From** This means this part should react to what is in another list.

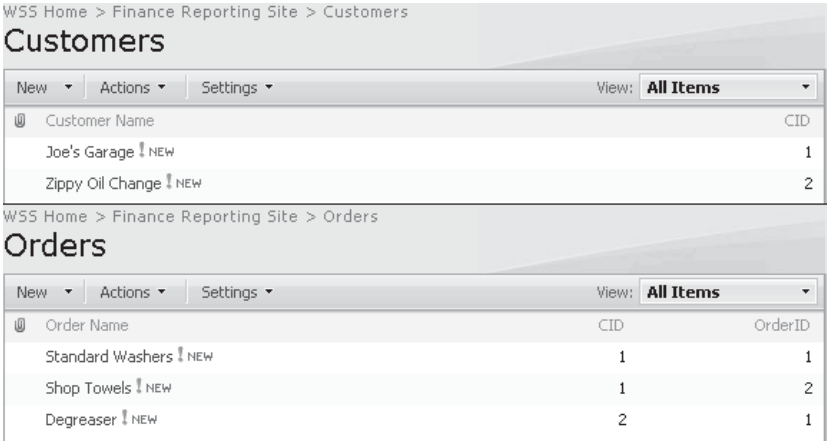
Exercise: Creating the Custom Lists

To demonstrate inter-list communications, you'll need two custom lists. Here's a step by step on creating the custom lists:

1. Navigate to any SharePoint site, click the Site Actions menu, and then click the Create link.
2. When the Create page opens, click Custom List under the Custom List group.
3. On the Create Custom List page, enter the title as **Customers**.
4. Leave Show on Quick Launch selected and click OK to create the list.
5. When the Customers List View appears, you'll see that only one column (in the view) is shown, called Title (other columns are created but not shown in the default view). Using the Settings menu from the toolbar, select List Settings.
6. When the List Settings page appears, scroll down to the Columns section and click the Title column (this will open it for editing)
7. Change the column name from Title to **Customer Name**, and then click OK to save.
8. When you return to the List Settings page, you'll see that the name is now Customer Name.
9. Click the Create Column link.

10. On the Create Column page, set the name to be CID, and then set the column type as a number (give the page a second to refresh to display the number field options).
11. Scroll down and change the number of decimal places from Automatic to 0. Leave all other settings as they are, and then click OK to save the column.
12. Return to the List View (use the Breadcrumb link at the top of the page).
13. On the toolbar, click New, and then select New Item (or simply double-click New).
14. On the New Item page, enter a customer name of **Joe's Garage**, and then enter the CID as **1**, and click OK to save the item.
15. On the toolbar, again click New, and then select New Item.
16. On the New Item page, enter the customer name as **Zippy Oil Change** and CID as **2**, and click OK to save the item.
17. Return to the Site page using the Breadcrumb link at the top of the page.
18. Click the Site Actions menu, and then click the Create link.
19. When the Create page opens, under the Custom List group, click Custom List.
20. On the Create Custom List page, enter the title as **Orders**.
21. Leave Show on Quick Launch selected, and then click OK to create the list.
22. When the Orders List View appears, click Settings in the toolbar, and then select List Settings.
23. When the List Settings page appears, scroll down to the Columns section, and then click the Title column (this will open it for editing).
24. Change the column name from Title to **Order Name**, and then click above to save the change.
25. When you return to the List Settings page, you'll see the name is now Order Name.
26. Click the Create Column link.
27. On the Create Column page, set the name to be **CID,** and then set the column type as Number (give the page a second to refresh to display the Number Field options).
28. Scroll down and change the number of decimal places from Automatic to 0; leave all other settings as they are, and then click OK to save the column.
29. Click the Create Column link.
30. On the Create Column page, set the name to be **OrderID**, and then set the type to be a Number. Change Number of Decimal Places from Automatic to 0, and then click OK to save the column.
31. Return to the List View (use the Breadcrumb link at the top of the page).
32. On the toolbar, click New, and then on the New Item page, enter the order name as **Standard Washers**, set the CID to **1** and the OrderID to **1**, and click OK to save the item.
33. On the toolbar, click New. Then on the New Item page, enter the order name as **Shop Towels**, set the CID to **1** and the OrderID to **2**, and then click OK to save the item.
34. On the toolbar, click New. Then, on the New Item page, enter the order name as **Degreaser**, set the CID to **2** and the OrderID to **1**, and then click OK to save the item.

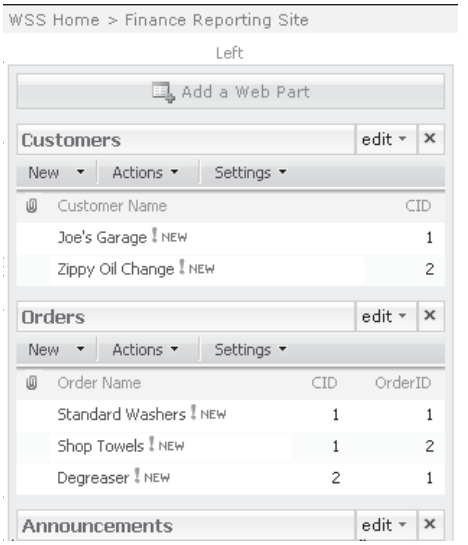
When you are done, the Customers List and Order List Views should look like this:



Exercise: Creating the List View Web Parts

The next step is to create web parts from these lists (connections are between web parts, not the lists themselves):

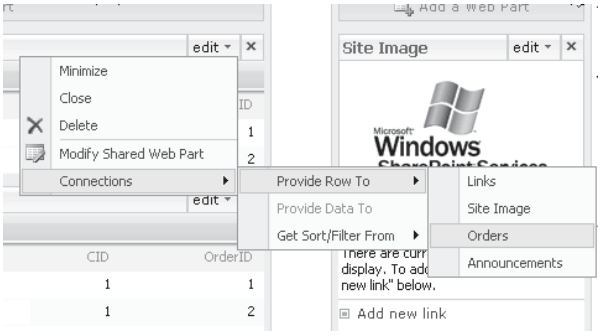
1. Navigate to the Site home page.
2. Click on the Site Actions menu and select Edit Page.
3. When the page enters edit mode, click on Add a Web Part in Any Zone.
4. In a WSS Site (or MOSS team/workspace site), you should see your custom lists under the Lists and Libraries section. Click the check box next to both of the lists and click the Add button (this adds two List View Web Parts at one time); in MOSS portal sites, you must use the Advanced link at the bottom of the Add a Web Part dialog page. When the Web Part Gallery panel opens, you will need to find the first list (Customers) and drag it to the page. Then page through the list (using the next link) to find the second list (Orders) and drag it to the page as well. With either method, the two List View Web Parts should be on the page and look something like this:



Exercise: Connecting the Web Parts

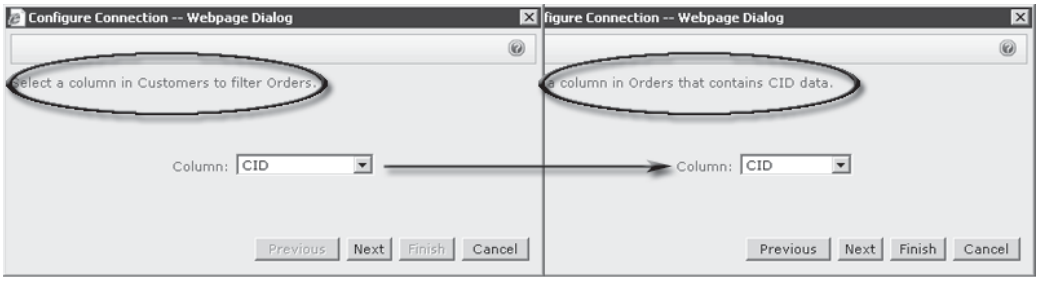
Now that you have two List View Web Parts, you can have one provide information to the other. In this example, you're going to connect the Customers View to the Orders View by way of a filter. When a Customer item is selected, the Order View will respond by showing only the orders for that customer (note that the field names as CID are not required; just convenient).

1. From the Site Page, you should be in editing mode and be able to see the Edit menus shown on the web parts (if not, use the Site Actions menu and select Edit Page).
2. On the Customers List View Web Part, click Edit on the web part toolbar, and then select Connections | Provide Row To | Orders as shown:

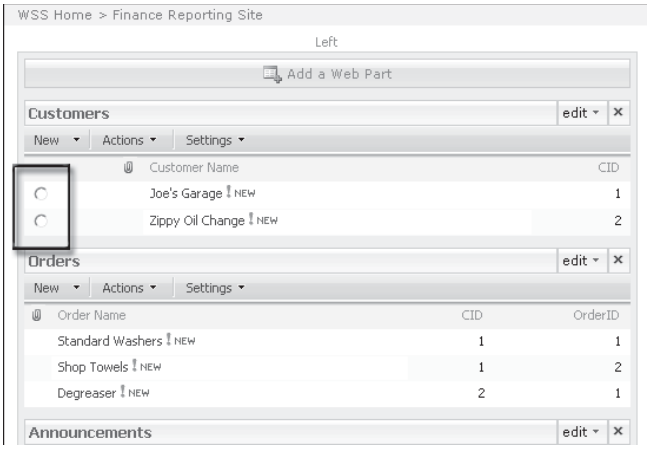


What this connection is really saying is, "When something is selected in Customers, pass something from the Item to the Orders List View Web Part." Of course, this is only half of the equation; you need to tell SharePoint how to make the actual connection so that SharePoint will display the Connection Configuration. This means asking what column in the list (Customers) you want to use to pass to the other list, and in this case, you want to pass the CID (Customer ID) as the "key" unique value:

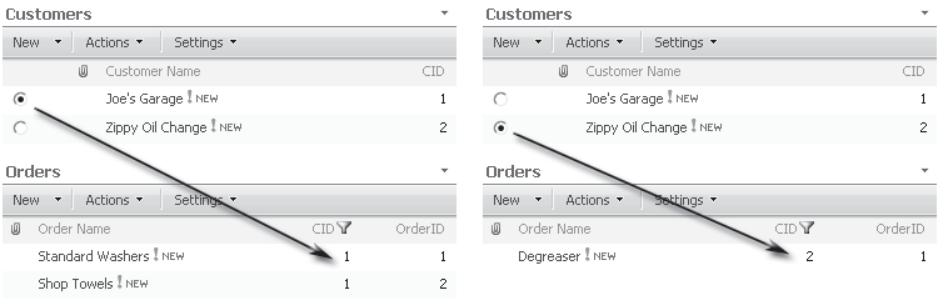
3. Set this and click Next to display the other side of the equation. In this case, SharePoint wants to know the column in the receiving web part that matches up (creates a key) with the sender. Again, in this case, this is CID because each order has a customer ID associated with it:



4. When you return to the page, you'll notice radio buttons have been added:



5. To finish this off, publish the page or exit edit mode to return to the normal view.
6. Click either radio button in the Customer View, and you'll see that the Order View now changes:



While this is a very basic example, I'm pretty sure you can get the idea of how powerful this can be. Remember that the lists are not what's really being used here; the connection and the application-like functionality is provided by connecting the two List View Web Parts (you'll really get into this when you starting using Filter and Business Data Catalog Web Parts). Also be aware that the connections don't stop at just one to one. You can have any number of web parts communicating back and forth. For example, you could easily create a Customer to Order to Order Detail to Shipping Items combination—all with no code!

Also as a side note, you may recall that when changing the view for a List View Web Part, you get a warning about breaking connections. The reason for this should be clear to you now. If a view in Orders was selected that didn't include the joining column (CID), the connection just made would break.

Core Parts

The core web parts in SharePoint provide the basic types of content in a SharePoint site. This content includes the Image web part, which is used to display images on the page

(you've see this on every page in SharePoint; the logo shown on WSS pages and the splash image on the MOSS Portal Home page both use this part). The basic web parts come in two flavors: List View Web Parts used to display list content on pages (as covered previously) and Content Display Web Parts, which provide stand-alone functionality. The core set is shown in Table 12-1.

Web Part Name	Purpose	Typical Usage
Content Editor Web Part	Provides a container that can contain HTML, XML, and script code that is executed in line with the page; the rendering is contained within the confines of the part.	The Content Editor part allows you to add any kind of HTML, XML, or script (like JavaScript) to a page. This provides the most flexibility in adding functionality since it does not need to be displayed. A sample use is in displaying simple text and HTML or JavaScript to create a picture viewer.
Form Web Part	Provide custom-code, HTML-based form for providing values through web part communications to other web parts.	This allows you to create an HTML form that can interact with other web parts to supply them with data or selections.
Image Web Part	Displays any web-viewable image providing alignment and background color.	Used to display an image on a page, images from any valid URL (be careful on permissions here; if the image is not accessible, it can keep the page from loading).
Page Viewer Web Part	Renders a website page (regardless of type) within an IFrame on the page.	This allows you to display a page within a page. Based on a specifying URL, the page is rendered within a user-sizable frame. Functionality on the inset page remains, but if SharePoint refreshes the page, you return to the original URL.
Relevant Documents Web Part	Provides an individual user's view of recent documents they have created, modified, or have checked out.	This gives users a personal view of the recent documents they have worked with. This is very similar to any Office application like Word (or My Recent Documents in Windows).
Site Users Web Part	This renders a summary list of users that have access, are members, or from a specific Site Group.	Used to provide a quick-reference members list that can be used to quickly find people that belong to a site.
User Tasks Web Part	Renders a list view of the individual's assigned tasks from the task lists in the site.	Used to give each user a snapshot of all tasks they are assigned to (active) in the particular site (all task lists are included).
XML Web Part	Renders an XML document (via local or remote URL) with optional XSL applied.	Used to display local or remote XML files or pasted in XML and allows an optional XSL style sheet to be applied (as a file or pasted).

Table 12-1 Core Web Parts Purpose and Usage

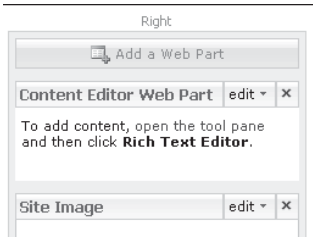
Content Editor Web Part

The Content Editor Web Part provides free-form content within SharePoint sites. The source of the content it displays can come from a few places: hand-entered text via a rich text editor, in code using HTML and JavaScript, or pulled from a URL. Content Editor parts are self-contained, and can actually be exported with content. When you create one and add something to it, it's stored in the database. If you add another one to the same page, it is considered new and has no content (this is opposite of what happens with a List View Web Part). These parts are great because they can use rich text to create sharp content, but also because they can use HTML source and JavaScript. Any code added (whether it renders something or not) is executed just as if you coded it into the page (handy for JavaScript Image Viewers, and so on). Be aware that code added here is rendered after the page, so you are limited to client-side only scripting.

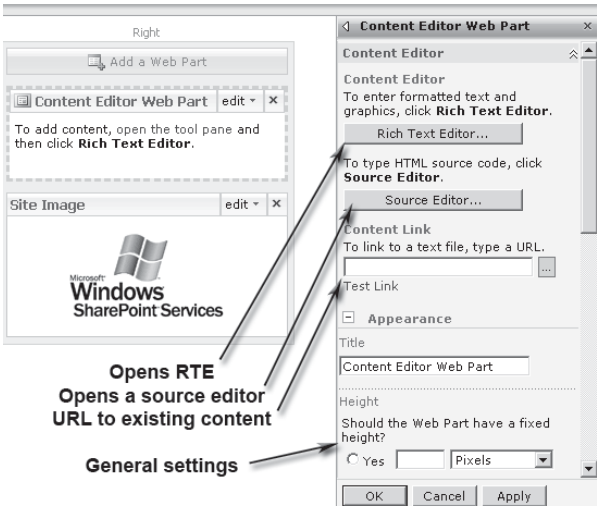
Creating a Content Editor Web Part

Here's how to create a Content Editor Web Part:

1. Navigate to any page in a SharePoint site, and then click the Site Actions menu. From the drop-down menu, select Edit Page.
2. When the page redisplays in edit mode, click Add a Web Part (any zone is fine).
3. When the Add a Web Part dialog opens, scroll down the list to find the Content Editor Web Part. In WSS, this can be found under the All Web Parts category under the Miscellaneous group. In MOSS, you'll find this under the Default category; click the check box next to the name, and then click Add to add it to the page. When first displayed, it looks like this:

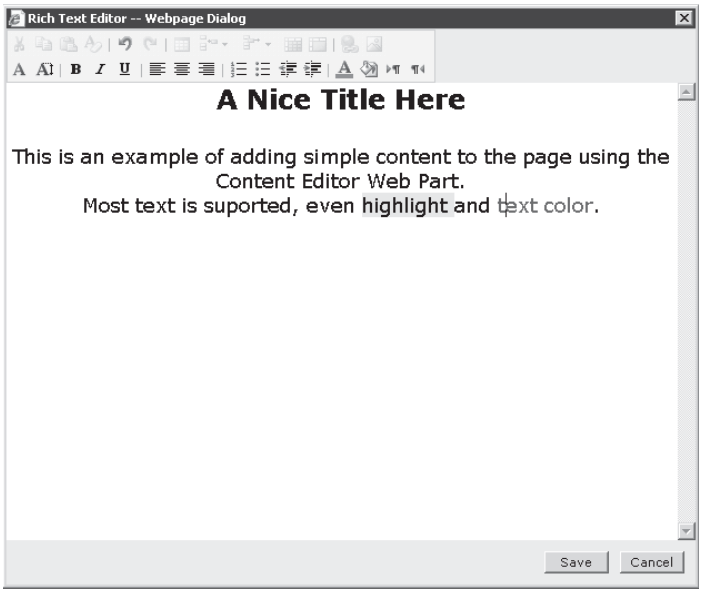


4. Notice that the part detects that it has not been set up, so it provides a link to Open the Tool Pane. Clicking this link opens the Web Part Settings panel that shows you the options for this part (note that you can also get to the settings the normal way by clicking on the web part Edit menu and selecting Modify Shared Web Part.

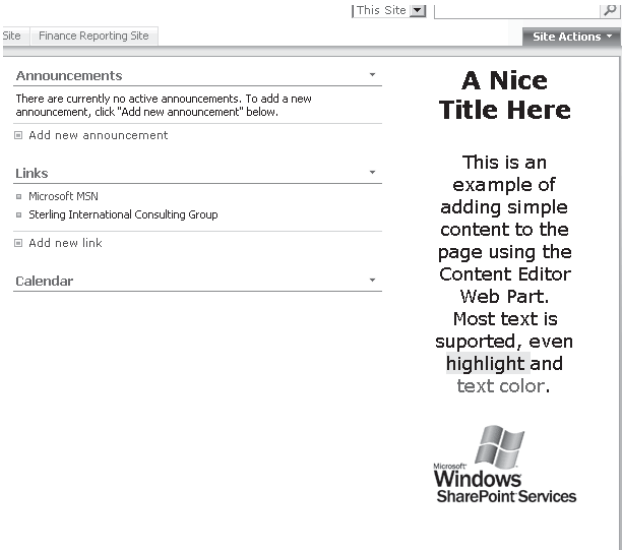


5. To keep this example simple, you'll simply add some text using the rich text editor. Click the Rich Text

Editor… button in the Web Part Properties tool pane, which will open a new pop-up window:



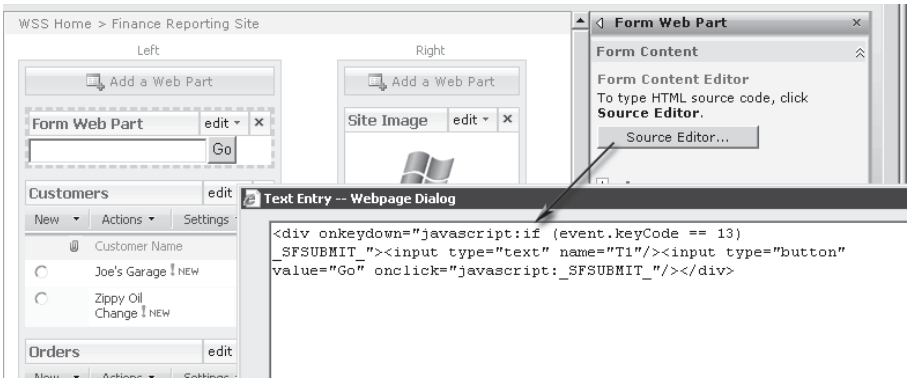
6. Notice that with the text entered, it supports most formatting. You'll also notice that you can include images with the text as well. (Note that if you use images, make sure they use the relative URL to an image stored in a Picture library to avoid security issues.) Click OK to save the text.
7. When you return, you will see the text displayed in the part. Under the Appearance options, scroll down and change the Chrome Type to None (this will hide the web part title).
8. Click the OK button at the bottom of the Web Part Properties tool pane. This will close it and return to the page. In WSS, click Exit Edit Mode, and in MOSS, click Publish. The end result should display the text without a border or title so it looks like part of the page:



To edit the content, simply re-edit the page and select Modify Shared Web Part from the Web Part toolbar. Note, if you delete the part, the content in it is also deleted (permanently); deleted web parts are not saved in the Recycle Bin.

Form Web Part

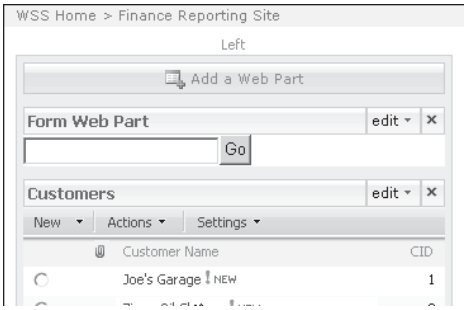
The Form Web Part provides a simple HTML form setup that allows you to create custom forms for providing data to other web parts via connections. By default, this is first created with a text box that can be used by users to pass values to other web parts on the page (through connections) and a Go (submit) button that will refresh the page and submit data to other web parts (they can supply values but can't receive them). If using a simple text box is enough, you don't have to do anything but set up a connection. If you want to get advanced with it, however, there is a Source Editor that can be used to actually paste in HTML:



Creating a Form Web Part

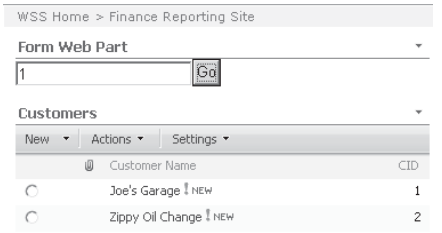
To create a Form Web Part, you should do the following:

1. Navigate to any page in a SharePoint site, and then click the Site Actions menu. From the drop-down menu, select Edit Page.
2. When the page redisplays in edit mode, click Add a Web Part (any zone is fine).
3. When the Add a Web Part dialog opens, scroll down the list to find the Form Web Part, and then click Add to add it to the page. When first displayed, it looks like this:



4. At this point, the part does nothing. If you click on Go, it will simply refresh the page; to make this part useful, it must connect to another part. To demonstrate this, this Form part can be connected to the Customer list used in the connection example previously. Instead of clicking on the Customer list, the Form can be used to set the customer ID automatically. To make the connection, click Edit in the web part toolbar and select Connections | Provide Values To | Customers. On the first Configure Connection dialog, leave the field selected (T1) and click Next.
5. On the next page, change the column to be CID, and then click Next or Finish.

6. In WSS, click Exit Edit Mode, and in MOSS, click Publish when the page redisplays. It should now look like this:



7. Now you can try out the Form part. Simply enter **1** in the field and click Go; the result is shown by the filtered Customer View.

   Notice that the item in the Customer list is not selected; the form provides only a filter. You would still need to select the customer to display the Order View correctly.

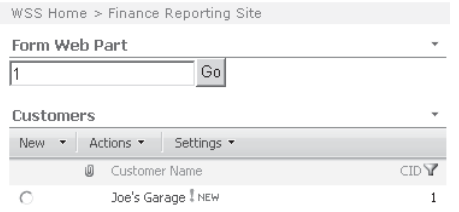


Image Web Part

The Image Web Part is used to display an image anywhere on a SharePoint page (demonstrated on the home page of WSS Sites). Properties of this part allow you to set the URL to the image and attributes for how to display it, including alternate text, background color, and horizontal and vertical alignment. The best way to use images in this part is to load them into the Site Collection Library (or any one you create), which brings them under SharePoint security and control. The image can then be referenced by the relative URL (/Images/Picture.jpg), which is just the way SharePoint prefers it. You can use other sources (http if the image is publicly readable or a virtual directory in SharePoint), but you should never use http when referencing images stored within SharePoint because it causes a "round trip" to the server, which can be very problematic for VPNs and other remote logins. Remember that security of the library where the image is stored applies when displayed by the part; if the user doesn't have access to the image library or the http or file image path, either the image will not display or it may even display a login prompt or force the user to an Access Denied page.

Creating an Image Web Part

Here's how to create an Image Web Part:

1. Navigate to any page in a SharePoint site.
2. If you already have an image uploaded to a picture library you can use that; if not, do the following:
   a. On the Site page, click the Site Actions menu, and then click the Create link.
   b. On the Create page, select Picture Library.
   c. Name the library whatever you like (something like Content Images is useful) and put in an optional description.

d. Uncheck the Show on Quick Launch option, and then click OK to create the library.

e. When the library opens, click the Upload button, and then browse to upload an image.

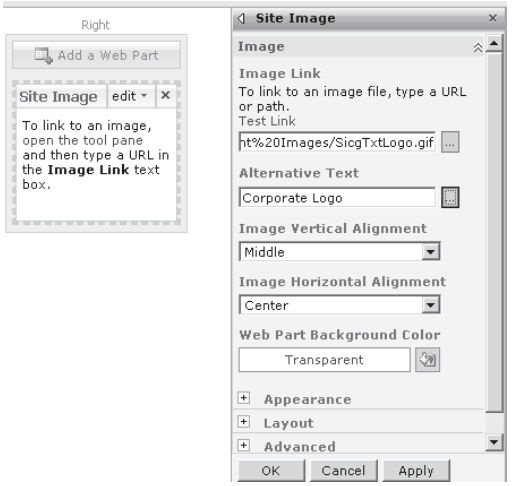
f. When the image is uploaded, you can enter the additional information, such as the date the picture was taken, and so on, if desired. Then, click OK to save it.

g. When you see the image in the Library View, you can capture the URL you need. Double-click on the image to open the Properties View, right-click on the image, and select Copy Shortcut:

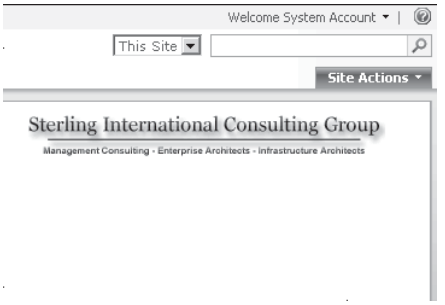


When you copy the URL, it will be the full URL, including http. It will look something like this: http://wssaweb1/Content%20Images/SicgTxtLogo.gif. The Relative URL you need to use to reference this image is /Content%20Images/SicgTxtLogo.gif.

3. Click the Site Actions menu, and then from the drop-down menu, select Edit Page.
4. When the page redisplays in edit mode, click Add a Web Part (any zone is fine).
5. When the Add a Web Part dialog opens, scroll down the list to find the Image Web Part, and then click Add to add it to the page.
6. In the Web Part Properties panel, specify the URL to the image. This should be the Relative URL without the http for SharePoint images, but it can also be the full URL to any website image (if public or open), or any web-displayable type, including .jpg, .gif, .bmp, .tif, and so on. Specify the URL to the image, and then set the alignment and display options:



7. Click the OK button in the Web Part Properties panel to save the options and return to the page. Click Publish in MOSS or Exit Edit Mode to see the result (note that the Chrome Type must be set to None to hide the web part title bar.



Page Viewer Web Part

The Page Viewer Web Part does exactly what it sounds like it would; it's used to display a web page, Windows Folder, or an actual file in a sub window on a SharePoint page. For example, if you had an existing web application that you used often, you'd include it on your own page. In most cases, the functionality of the page operates normally (even things like logging in). However, the part is not in control of the page; a page refresh will cause the Page Viewer to default to original URL specified. (This might log you off or change the page you are viewing.) Technically speaking, for developers, this part uses an IFrame within the page.

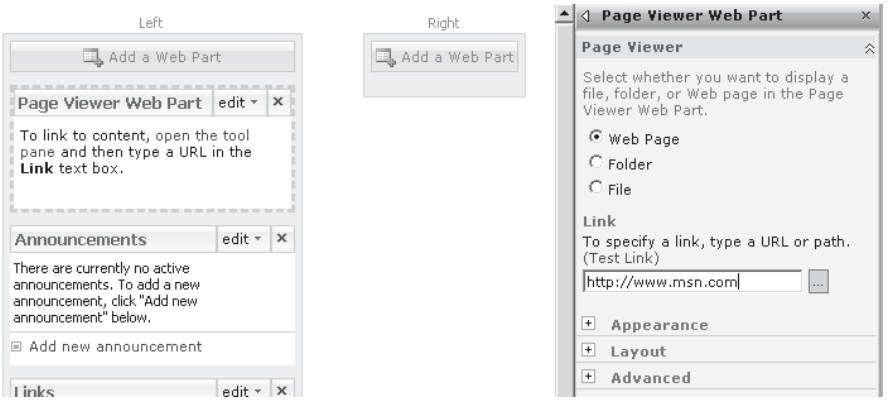
You can use this part for some pretty ingenious things. For example, it can enable you to access any web-based application you have or can be used to display a common news page. With the folder option, you can view the contents of a folder on the server (or file share, but make sure users have the privilege to read it!). You can also display a file stored in the file system or a document stored in SharePoint.

Creating an Page Viewer Web Part

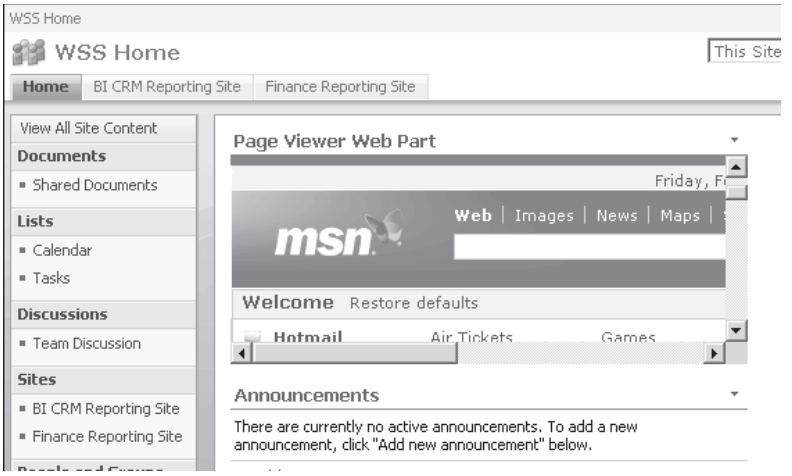
You can create a Page Viewer Web Part by doing the following:

1. Navigate to any page in a SharePoint site and click the Site Actions menu. Then, from the drop-down menu, select Edit Page.

2. When the page redisplays in edit mode, click Add a Web Part (use the left-hand or larger zone for better viewing).
3. When the Add a Web Part dialog opens, scroll down the list to find the Page Viewer Web Part, and then click Add to add it to the page.
4. Click the link to open the Web Part tool pane (or select Modify Shared Web Part from the Edit menu).
5. In the Web Part Properties tool pane, click the Web Site radio button and specify the URL to the website (for example, http://www.msn.com):



6. Click OK in the Web Part Properties tool pane to save the options and then return to the page. Next, click Publish in MOSS or Exit Edit Mode to see the result (you must be connected to the Internet for this to work):



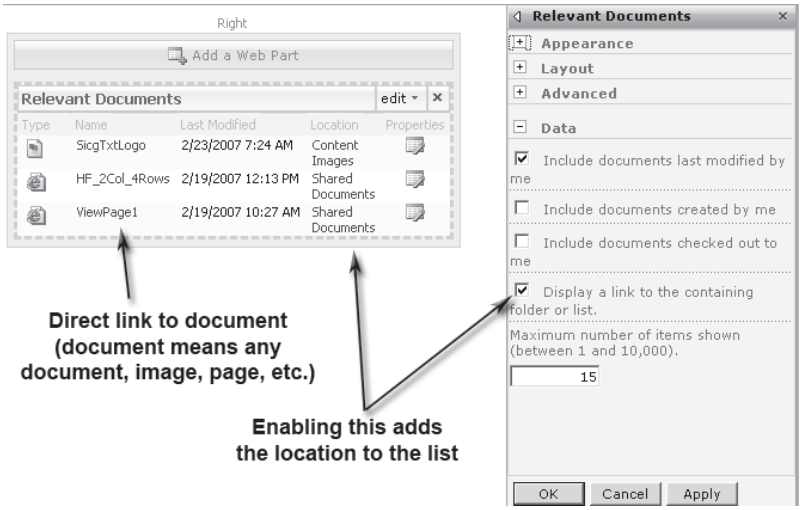
Relevant Documents

The Relevant Documents Web Part is actually a piece of application functionality. This scans content in the site and based on who the user is, looks for items that have been recently uploaded, modified, and so on, and then displays a summary list. This web part has a

special Data section in the web part properties that provides settings for the part, in this case, allowing you to select which documents should be shown, whether last modified, last created, or last checked out. The Display a Link to the Containing Folder or List option adds the Location column shown (this link brings the user to the list or library that holds the item) and the Maximum Number of Items to Show limits the items shown in the list. These options allow you to use this creatively. For example, if you add two parts to the same page and set one to Show Modified and the other to Only Checked Out (and set the titles appropriately), you can provide the user with a quick list for each type.

Using the Relevant Documents Web Part

1. Navigate to any page in a SharePoint site and click the Site Actions menu, and from the drop-down menu, select Edit Page.
2. When the page redisplays in edit mode, click Add a Web Part (any zone is fine).
3. When the Add a Web Part dialog opens, scroll down the list to find the Relevant Documents Web Part, and then click Add to add it to the page.
4. Click Modify Shared Web Part in the web part Edit menu to open the Web Part Properties panel. Next, scroll down the panel to expose the Data section:



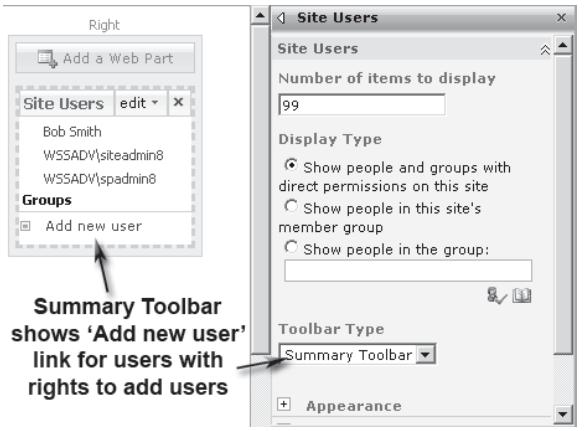
5. Click OK in the Web Part Properties tool pane to save the options and return to the page; click Publish in MOSS or Exit Edit Mode to see the final result.

Site Users

The Site Users Web Part provides a list of users that belong to the particular site. There are only a few options for this part: Number of Items to Display Limits; Total Number Shown in the List; Show People with Direct Permissions in This Site (this means all site users); Show People in This Site's Member Group (this means only users that are members); and the Show People in Group option, which allows the list to be limited to a specific site group. Like others, you can use multiple copies of this web part on the same page and use different settings to split out the users shown.

Using the Site Users Web Part

1. Navigate to any page in a SharePoint site and click the Site Actions menu, and from the drop-down menu, select Edit Page.
2. When the page redisplays in edit mode, click Add a Web Part (any zone is fine).
3. When the Add a Web Part dialog opens, scroll down the list to find the Site Users Web Part. Then click Add to add it to the page.
4. Click Modify Shared Web Part in the Web Part Edit menu to open the Web Part Properties panel. The Site Users section should be shown:



5. Click OK in the Web Part Properties tool pane to save the options and return to the page; click Publish in MOSS or Exit Edit Mode to see the final result.

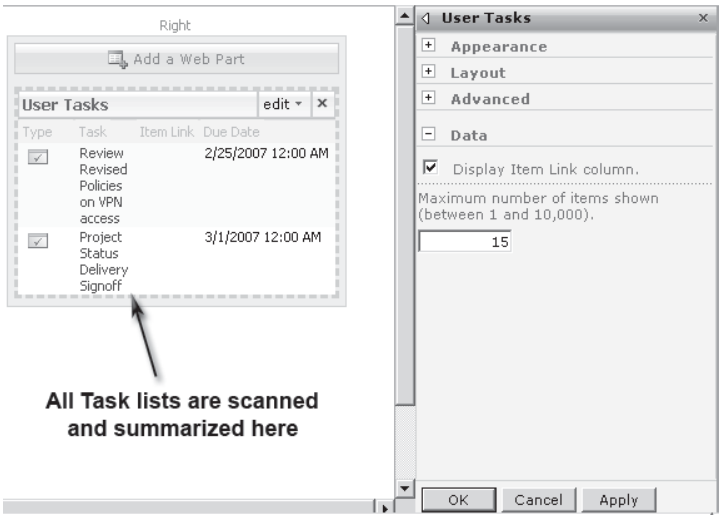
User Tasks

The Users Tasks Web Part scans all of the Task Lists in a Site Collection for a specific user and provides a handy summary with links. (Note that there's a minor bug; the option to Show the Item Link adds the column, but it is not always shown).

Using the User Tasks Web Part

1. Navigate to any page in a SharePoint site.
2. Click the Site Actions menu, and from the drop-down menu, select Edit Page.
3. When the page redisplays in edit mode, click Add a Web Part (any zone is fine).
4. When the Add a Web Part dialog opens, scroll down the list to find the User Tasks Web Part, and then click Add to add it to the page.

5. Click Modify Shared Web Part in the web part Edit menu to open the Web Part Properties panel. Next, scroll down the panel to expose the Data section:



6. Click OK in the Web Part Properties tool pane to save the options and return to the page; click Publish in MOSS or Exit Edit Mode to see the final result.

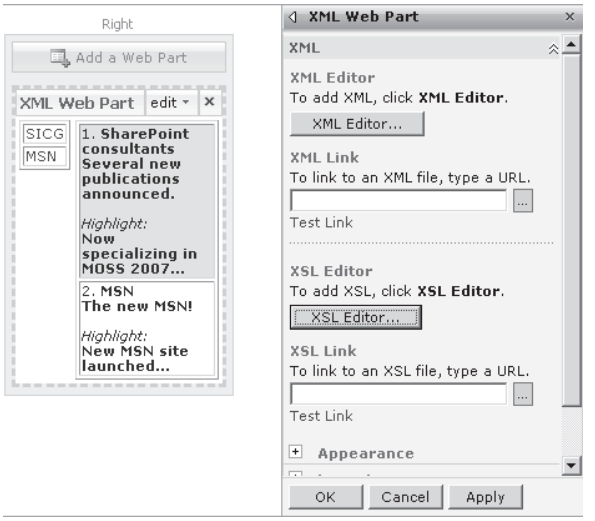
XML Web Part

The XML Web Part is one of the most flexible content-type parts available since it allows you to display any XML with or without a style sheet (XSL). This means that content from all kinds of sources can be used, provided it is well-formed XML. This includes both news sites and content-producing sites. XSL allows you to transform the data into any format. In addition to allowing you to select existing XML and XSL files by URL, this part also provides a Text Editor for both.

Creating an XML Web Part

1. Navigate to any page in a SharePoint site.
2. Click the Site Actions menu, and from the drop-down menu, select Edit Page.
3. When the page redisplays in edit mode, click Add a Web Part (any zone is fine).
4. When the Add a Web Part dialog opens, scroll down the list to find the XML Web Part. Then click Add to add it to the page.
5. Click Modify Shared Web Part in the web part Edit menu to open the Web Part Properties panel. At the top, you'll see the options to point to files or to use the

editors (these editors are simple text; I advise using something else to create your XML and XSL and then paste the code). The sample XML and XSL is available on the website for this book:



MOSS Web Parts

MOSS adds a number of web parts to SharePoint for facilitating the enterprise features and for providing additional functionality to support the needs of a multisite portal (roll ups, filters, and so on). Access to web parts in MOSS is the same as WSS; the difference is that there are a lot more web parts listed and a few additional groups. Parts included with MOSS all provide application functionality like the Filter Web Parts, and most are for use everywhere. However, some are for a specific purpose; Outlook Web Access (OWA) parts, for example, are only intended for the My Sites.

The list of MOSS Web Parts and details are shown in Table 12-2. Note that these are available in any site, but I won't cover the Business Data Web Parts and Dashboard Parts (covered in the online Business Data Catalog, Web Chapter 1) or the OWA parts (part of My Site, covered in Chapter 13).

For the following examples, I'm going to assume you now know how to add a web part to the page and that you know how to use the Edit link on the web part title bar to open the Web Part Properties tool pane. If you're not sure, revisit the beginning of this chapter. All of these examples will assume you are using a Portal Site page (in a MOSS site), except for the Site Directory Web Parts shown using a Site Directory site type.

Site Aggregator Web Part

The Site Aggregator Web Part provides a toolbar/tab display of the content of selected sites in a summary list of all of the content in it. This is handy for creating a summary list of documents in a site without having to open up individual libraries. Any number of new tabs can be added for additional sites. This part is also in available in My Sites for personal use.

Web Part	Description
Default	
Content Query Web Part	This is used to display views of existing content in the site.
I Need To... Web Part	Like Users Tasks, this provides a drop-down menu list of tasks.
RSS Viewer Web Part	A viewer tool for RSS feeds (of any source).
Summary Link Web Part	An advanced link list that enables grouping and styling.
Table of Contents Web Part	This displays a hierarchy navigation of the site.
This Week in Pictures Web Part	This displays images using an Image library source and provides a link to a slideshow of all images.
Filters	
Business Data Catalog Filter Web Part	This filters contents of web parts using values from a BDC data source (MOSS BDC only).
Choice Filter Web Part	This filters contents of web parts using a set list of values.
Current User Filter Web Part	This filters contents of the web parts to the current user's rights and content.
Date Filter Web Part	This filters contents of web parts by a specific date selected by the user.
Filter Actions Web Part	This is a generic filter tool.
Page Field Filter Web Part	This filters contents of web parts using page information.
Query String (URL) Filter Web Part	This filters contents of web parts using values from the URL Query String.
SharePoint List Filter Web Part	This filters the contents of web parts using a SharePoint list.
SQL Server 2005 Analysis Services Filter Web Part	This filters contents of web parts using a list of values generated by a SQL Server Cube.
Text Filter Web Part	This filters contents of web parts using text entered by a user.
Search Web Part	
Advanced Search Box Web Part	This provides multi-selection (advanced) search.
People Search Box Web Part	This is used to search People (User Profiles and Site Groups).
People Search Core Results Web Part	This is used to search People (User Profiles and Site Groups) and detailed User Profile information.
Search Action Links Web Part	This displays search action links.
Search Best Bets Web Part	This displays items that are ranked higher in searches.
Search Box Web Part	This is a simple search box for generic searches.
Search Core Results Web Part	This enables search results and properties returned.

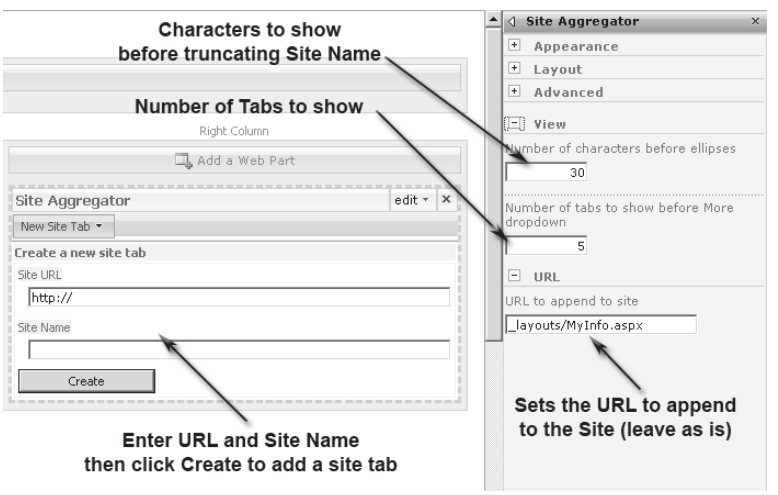
Table 12-2 MOSS Web Parts

Web Part	Description
Search High Confidence Results Web Part	This displays keywords, best bets, and high confidence results (content providers can help push results to this list).
Search Paging Web Part	This provides search with the ability to page through results (best for general searches that will return lots of content).
Search Statistics Web Part	This displays search details: the total returned, time taken, and so on.
Search Summary Web Part	This provides a display of Did You Mean? when it detects misspellings in search terms.
Site Directory Web Part	This is for Site Directory Site use only.
Categories Web Part	This displays categories in the site list.
Sites in Category (Site Map) Web Part	This displays Sites in the site list by Category.
Top Sites Web Part	This displays sites set as top sites in the site list.
Miscellaneous	
Contact Details Web Part	This is used to display details about a user designated as the owner/contact for the page or site.

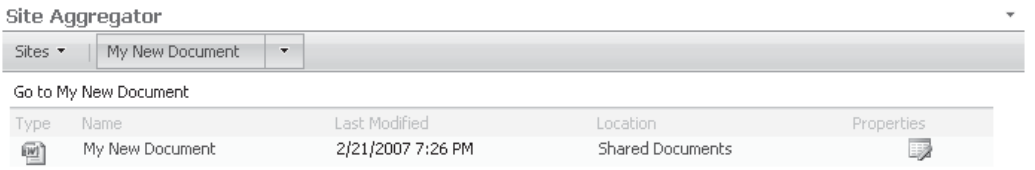
TABLE 12-2 MOSS Web Parts *(continued)*

Creating a Site Aggregator Web Part

1. Navigate to a page in the site, select Site Actions | Edit Page, and then click Add a Web Part in Any Zone.
2. In the Add a Web Part dialog, locate the Content Rollup group, and under it, click the check box next to the Site Aggregator Part. Next, click the Add button to add it to the page. When first opened, it looks like this:



3. Notice the settings for the web part. The Number of Characters Before Ellipses will set the maximum length of a site name to be shown; the Number of Tabs to Show sets the number of site tabs to display across (if there are more tabs than this number, a More link will be shown). The URL to Append to the Site (as shown) points to a special page within SharePoint that creates a list of content based on who the user is. If you remove the URL shown, the link itself will be used, and instead of a summary, this part will act like a Page Viewer and display the home page of the site.
4. To add a new site tab, enter a URL and site title, and then click Create. URLs should be entered as the full URL to the site; however, the page name should be omitted. For example, http://<server><:port>/SiteDirectory/My Document/default.aspx, the home link for the site, should be entered as http://<server><:port>/SiteDirectory/My Document/.
5. Add as many sites as desired, and when complete, click the OK button in the Web Part Properties tool pane to save the settings and return to the page (to keep the Properties tool pane open, click the Apply button instead).
6. As an example, I've added a link to a document workspace that has a single document in it. When clicking the site tab, you can see that the content is aggregated and displayed something like this:



Content Query Web Part

The Content Query Web Part allows users to create custom queries of sites and content in sites to display a summary of the content. Queries can be set to choose specific content types (pages, documents, and so on), can be filtered by any columns, or even set to use audience targeting to provide the query with more relevance to the individual user. A good use for this is to allow you to segregate sites and/or content into areas of interest, and if needed, filter the summary. Queries can be based on the current site, the site collection, and even a different site collection. For example, one use might be to display a table of contents of Corporate News articles on a Division site. This part has a number of options.

Query

- **Source** This includes one of the following:
  - **Show Items From All Sites in This Collection** This creates a summary listing (with links) to all sites or items in the current collection.
  - **Show Items from the Following Site and all Sub Sites** This creates a summary listing of a different site collection.
  - **Show Items from the Following List** This replicates list data from anywhere else in a new display (that can be filtered); note that this does not affect the original list.

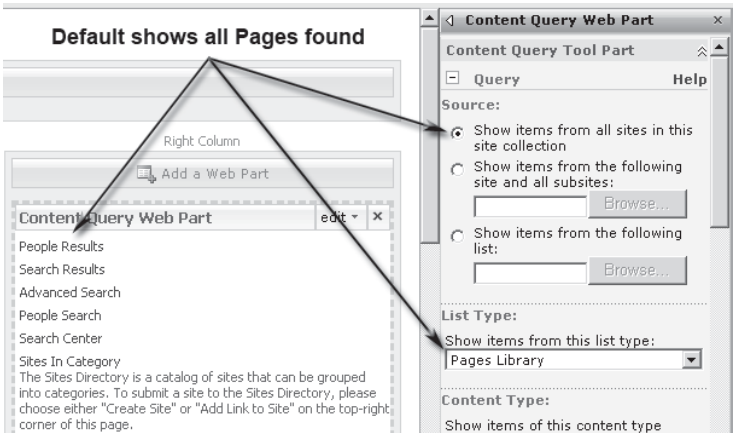
- **List Type** This allows you to select the type of list to use, such as the Pages Library, which is a list of all pages (any list available in the site can be selected here).
- **Content Type** This allows you to select both a Content Group (this includes Page Content Types, Document Content Types, and so on) and a specific Content Type (document, page, and so on). There is also a check box where you can exclude or include Child Content Types.
- **Audience Targeting** This enables/disables audience filtering on the content shown and gives you the option of including items that are not targeted (so if the list is targeted but not necessarily an item in the list, it is still shown).
- **Additional Filters** This allows the setting up of up to three filters based on content columns, for example, filtering the list to only approved items.

Presentation

- **Grouping and Sorting** This allows you to select a column to group on, sort on a column, and specify the maximum number of items to show (the default is 15).
- **Styles** This allows you to choose from different display styles.
- **Feed** This enables/disables this view to be used as an RSS Feeder. If enabled, you can specify the name and description of the feed.

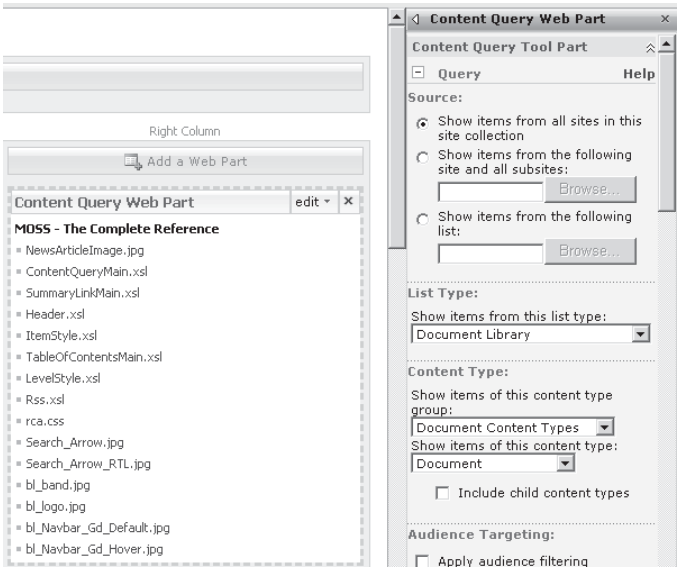
Creating a Content Query Web Part

1. Navigate to a page in the site, select Site Actions | Edit Page, and then click Add a Web Part in any zone.
2. In the Add a Web Part dialog box, locate the Default group. Then, under it, click the check box next to the Content Query part, and then click the Add button to add it to the page. When first opened, it looks like this:



NOTE *By default, this part is set to list all sites and pages as shown.*

3. Choose the options you want for the list by the content and filtering desired, and then click OK in the Web Part Properties tool pane to save all changes.
4. As an example, I've set this query to look in Document Libraries for Document Content Types (I've also set the style as bulleted and grouped by the site). This picks up any documents (and images) found:



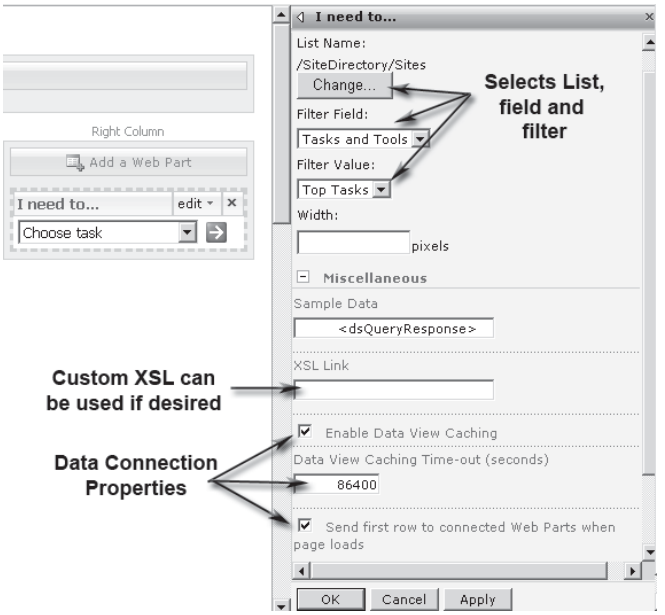
I Need To... Web Part

The I Need To… Web Part is a convenient little tool that allows pointing to a specific list (like tasks), and, filtering by the user, shows a list of items in the drop-down menu. By default, this points to the Sites list, which has a link to Create My Site, but it can be pointed to any other list including Task lists. For example, if you had a general Task list for the site and then wanted to display a list of tasks that had not been started (for that user), you would point this part to the Task list, select the Column as Status, and then filter that by the status of Not Started. When a user enters the site, the I Need To... Web Part drop-down menu would list all tasks that had not been started. The drop-down menu link brings the user directly to the item.

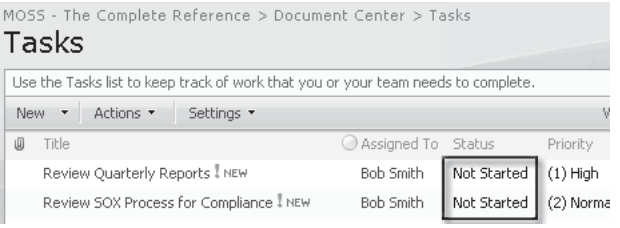
Creating an I Need To... Web Part

1. Navigate to a page in the site, select Site Actions | Edit Page, and then click Add a Web Part in any zone.

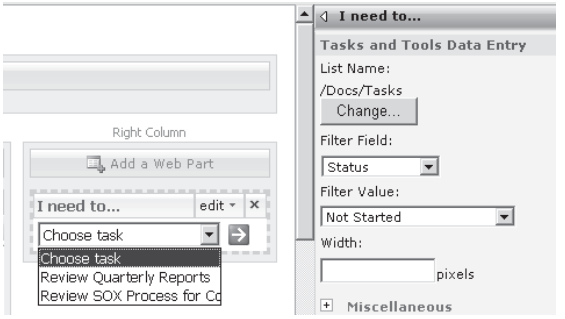
2. In the Add a Web Part dialog, locate the Default group. Then, under that, click the check box next to the I Need To… part and click the Add button to add it to the page. When first opened, it looks like this:



3. As an example, setting this part to look at a specific Task List for Tasks (for the user), and then choosing the Status Field filtered by a status of "Not Started," will display all of the tasks for the user when they view the page. This starts from the tasks as shown:



The web part reflects this in the drop-down menu (shown in edit mode so you can see the settings).



Note that you might have to play around with the settings until the ones you really want are set. This part seems to have a problem when selecting a list, then a column; it often resets to point to a different list or changes fields (quite annoying). Eventually, you will get the settings correctly saved. Just be patient and observant of what the settings are before you exit editing mode.

RSS Viewer Web Part

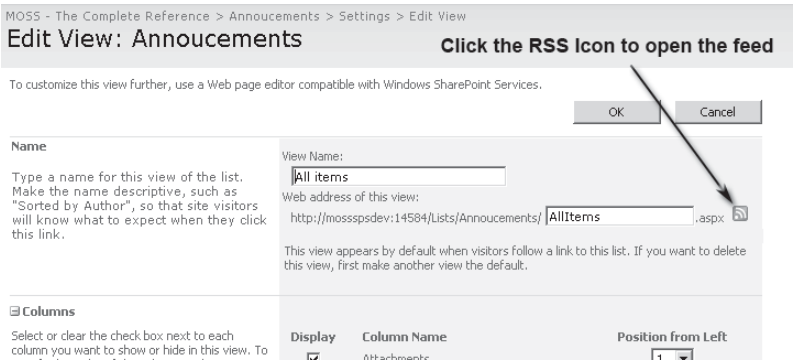
Really Simple Syndication (RSS) is one of those things that snuck up from a little known play-around tool to becoming nearly as big as blogs. RSS is based on a Syndicator (a server) that delivers an XML (or sometimes HTML) page of information in a list format or a news format. An RSS Reader (a client browser or application) handles the display. Many include custom XSL (Styles) that allows custom display of the feed data.

SharePoint 2007 adopted RSS due to its popularity and proliferation on the Web (and in fact, the Smiling Goat RSS Feed Reader Web Part was one of the top downloads for the 2003 version). All SharePoint lists are stored in the database and can be retrieved in XML format—and as it happens, this XML is just right for RSS. Viola! All SharePoint lists can be enabled as RSS Syndicators. This means that when you create a News site in your portal and start pumping out news, you can syndicate this as an RSS feed to any other site that has an RSS Reader.

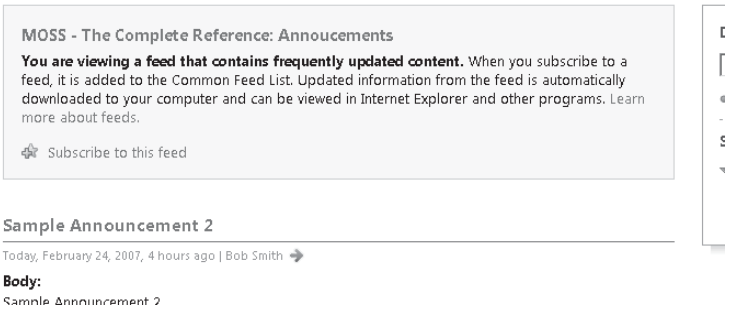
The RSS Viewer Web Part happens to be the out-of-the-box RSS Reader and can be used to read and display any RSS feed, whether it be SharePoint or any one of the thousands of feeds available (news, sports, you name it, check out rssfeeds.com). For developers, note that nearly all of the MSDN and Technet sites provide RSS feeds. However, be aware that this part doesn't work with RSS feeds in a non-anonymous SharePoint Site. This part is intended for reading public feeds and doesn't support authenticated feeds (also known as sites that require logging in). Sterling makes a version that does support internal RSS (go to www.sterling-consulting.com).

Creating an RSS Viewer Web Part

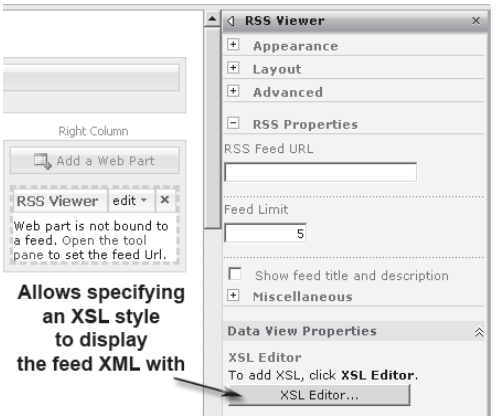
1. Before you add an RSS Viewer Web Part, you'll need the URL to an RSS feed. If you are using an external RSS feed, skip to step 3. To use a SharePoint list RSS feed (it must be anonymous), you will need the URL for the desired view of the list. In the list that will be the provider of the feed, you must enable the RSS feature. This is set under List Settings | RSS Settings; if the RSS Settings link is not there, that means RSS has been disabled in Central Administration. To capture the RSS URL, edit the desired view (of the list):



2. Click on the RSS icon to open the feed. You will see the default view (using the default XSL):

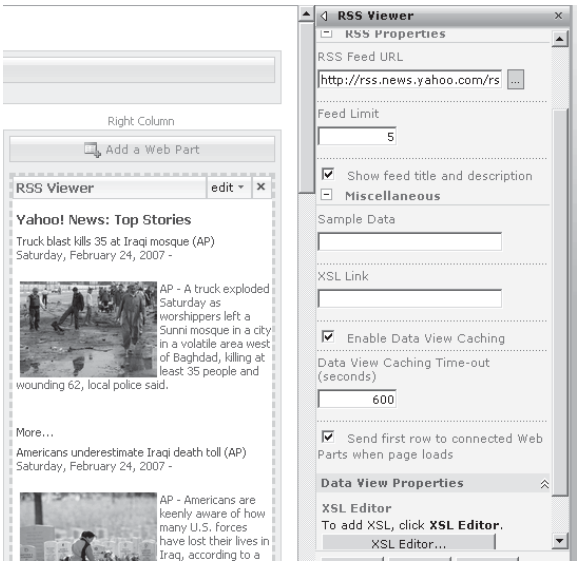


3. Simply copy the URL in the browser to use in the RSS Viewer Web Part (FYI: This link includes the GUID of the list; since it is pretty hard to type in a GUID, copy and paste is the best method).
4. Navigate to a page in the site, select Site Actions | Edit Page, and then click Add a Web Part in any zone.
5. In the Add a Web Part dialog, locate the Default group. Then under it, click the check box next to the I Need To… part, and click the Add button to add it to the page. When first opened, it looks like this:



6. Click on the link shown to open the Web Part Properties tool pane. Here you can specify the RSS feed URL, the maximum number of items to display, and enable or display displaying the feed title and description (in SharePoint that is the name of the List and List Description). Under the Miscellaneous section (not expanded), you can set an XSL style sheet link, enable or disable caching, and set a timeout limit (good when feeds fail). A button is also provided if you want to manually enter the XSL code.

   As a quick example, here's a view of the Yahoo Top Stories news feed (from URL http://rss.news.yahoo.com/rss/topstories). This one happens to include images (note that your SharePoint RSS feeds will look very similar):



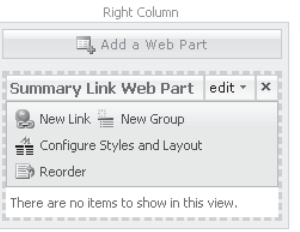
Note that whenever SharePoint goes to get RSS data, you get to see the Knight Rider progress bar indicating it is loading; users will see the same thing when they navigate to a page that has an RSS Viewer. Be aware, however, that if there is a problem with getting the feed (no response or slow response), this can lock up the page momentarily. If you find that a particular feed is causing problems, you should not use it or reduce the wait time.

Summary Link Web Part

The Summary Link Web Part is an advanced form of the Links List and provides several additional options such as adding images and tool tip to links. In addition, you can specify groups, which you can use to organize links, and you have several different out-of-the-box display styles available (there are quite a few so I encourage you to check these out).

Creating a Summary Link Web Part

1. Navigate to a page in the site, select Site Actions | Edit Page, and then click Add a Web Part in any zone.
2. In the Add a Web Part dialog, locate the Default group, and under it, click the check box next to the Summary Link Web Part. Next, click the Add button to add it to the page. When first opened, it looks like this:



Unlike other parts, this one has no special settings in the Web Part Properties. All of the options are shown on the toolbar, and are as follows:

- **New Link** This adds a new link to the summary and provides the following options:
  - **Link type** This link can be to an item (URL) or a person (a user in the Site); if a person is selected, a new User Selection box is shown, allowing you to select the user.
  - **Title** This is the title of the link to display in the summary list.
  - **Description** This is a description of the link to display in the list.
  - **Link URL** This is a link to item/site/document and so on.
  - **Open Link in New Window** If checked, the link will open a new window when clicked; otherwise, it is opened in the same page.
  - **Tooltip** This is text to display to the user when the mouse is placed over the link.
  - **Image URL** This specifies the link to an icon or image to display in the summary (remember, this should be a relative URL to a picture stored in a library in the site or an image link that is publicly available to all users in the site).
  - **Image Description** This specifies the alternate text to use in place of the image (shown if the image cannot be loaded).
  - **Groups** This allows you to set the group for the listing.
  - **Style** This allows you to set the style for the item display.
- **New Group** This allows you to add new groups; these are optional and can be used to group links in the summary display.
- **Configure Styles and Layout** This allows you to configure a variety of styles and layouts for the link summary (including images, not including images, and so on).
- **Reorder** This allows you to reorder the list.

Once you have added the links desired, save the page (via either Exit Edit Mode or Publish the Page), and the Summary Links are displayed with the grouping/format selected:

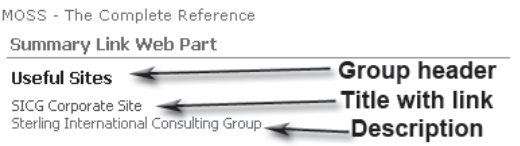


Table of Contents Web Part

The Table of Contents Web Part does what you might think it does; from a starting site location (a Portal Site, Site Collection, or Site), this part displays a table of contents to the sites and content below it. In one sense, this provides a site map that makes it easier to navigate around; on the other, it can be customized (via properties) to make it a quick map to content.

This part is not very sophisticated but does have several web part properties that control how much and how to display the data. The settings are divided into three sections:

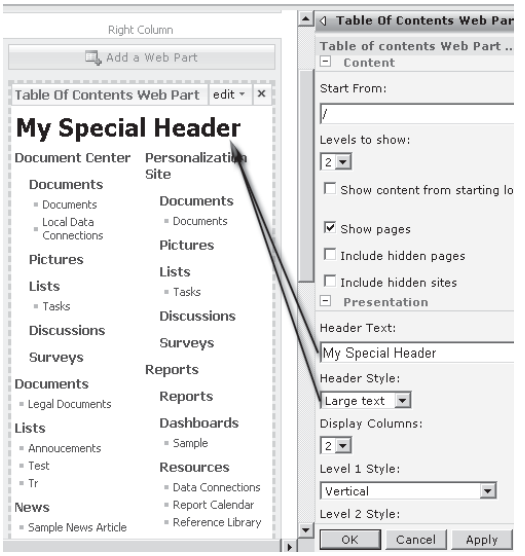
- **Content**
  - **Start From** This specifies the URL to start from, and is either the root of the site (/) or the URL to a site collection or site.
  - **Levels to Show** This indicates the number of sites deep.

- **Show Content From Starting Location** This enables/disables showing content from the start from site; if enabled, lists and libraries from the top site will be included.
- **Show Pages** This includes pages in addition to sites.
- **Include Hidden Pages** This shows all pages even if hidden.
- **Include Hidden Sites** This shows all sites even if hidden.

- **Presentation**
  - **Header Text** This is the title to display at the top of the TOC Web Part.
  - **Header Style** This provides a few style options on how the title is displayed.
  - **Level 1 Style** This sets the way the primary (first level) of the TOC is displayed.
  - **Level 2 Style** This sets the way the second level of the TOC is displayed.
  - **Level 3 Style** This sets the way the third level of the TOC is displayed.
- **Organization**
  - **Sort Contents As They Are Sorted in Navigation** If enabled, all contents are sorted as they are in the site.
  - **Sort Contents Using the Following Settings** This allows you to choose the sort yourself.

Creating a Table of Contents Web Part

1. Navigate to a page in the site, select Site Actions | Edit Page, and then click Add a Web Part in a zone that accommodates the display (in most cases, the center or top zone).
2. In the Add a Web Part dialog, locate the Default group, and under it, click the check box next to the Table of Contents Web Part. Next, click the Add button to add it to the page; when first opened, it looks something like this (the items listed will obviously be different):



NOTE *I put this in the right-hand zone just to keep it visible; while you can do this, for large sites, you should use a zone that will expand enough to see the list without scrunching or overlapping names.*

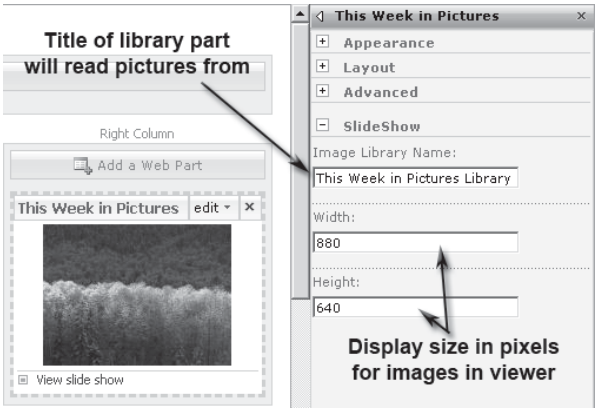
This Week in Pictures Web Part

The This Week in Pictures Web Part is a simple slideshow part that reads pictures from a picture library. When created, this part looks for a picture library in the site called "This Week in Pictures Library" as the source; if it does not find this list, it will warn you. In the Web Part Properties settings, you can specify a different list name (the name of the list is the title), and you can specify a specific width and

height for pictures displayed. On a page, this displays a thumbnail of the last picture loaded into the library (by date) and provides a link to open the slide show.

Creating a This Week in Pictures Web Part

1. Navigate to a page in the site, select Site Actions | Edit Page, and then click Add a Web Part in any zone.
2. In the Add a Web Part dialog, locate the Default group, and then click the check box next to the This Week in Pictures. Click the Add button to add it to the page; when first opened, it will check for the This Week in Pictures Library. If not found, it will display a message indicating this; if that library is found (and has pictures in it) it would look vaguely like this:



3. When displayed on a page, this part will scan the library and display a thumbnail of the last picture in the list (it assumes that the last is the latest). It also provides a Slide Show link, which opens a new window that allows the user to scroll through to display all of the images in the list:

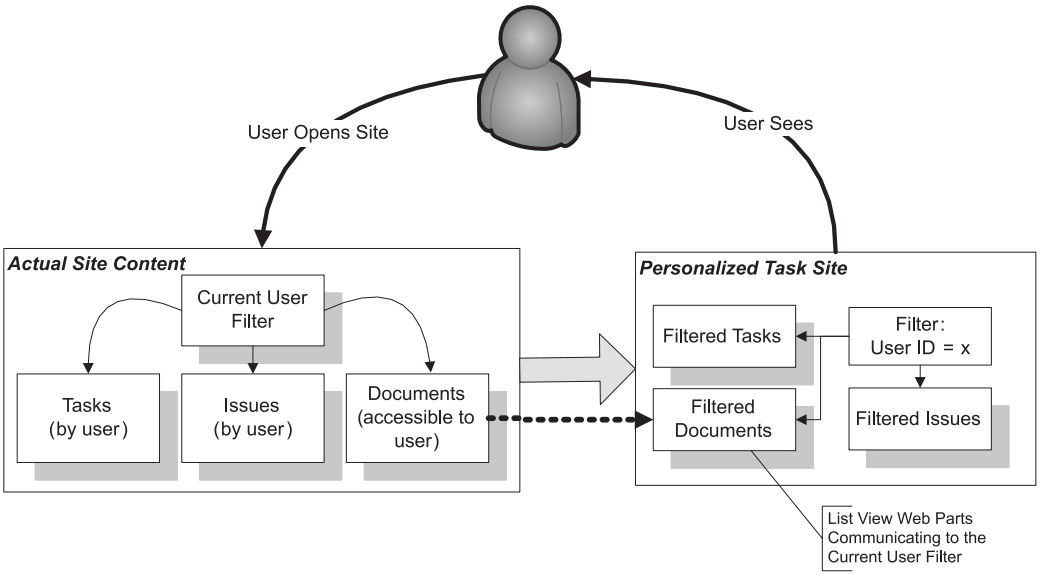


Filter Web Parts

As you have already seen, web parts can communicate with each other by passing value(s) to filter information shown by other web parts (including List View Web Parts). The SharePoint Filter Web Parts provide end users with a way to create these filters themselves using a host of different data sources. For example, the Business Data Catalog Filter Web Part allows users to filter or pass values from a BDC data source to filter other parts on the page (for example, if you were using Siebel data to filter your SharePoint Lists). The Choice Filter allows the user to create hardcoded choices to pick from and the Current User Filter allows filtering by the current user logged in.

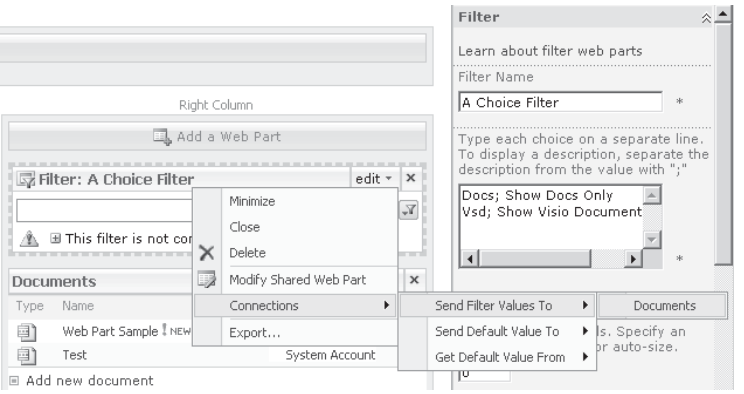
All of these filters work the same way, and each represents some form of data (Users, BDC, SharePoint List, and so on), which is then used to filter (by web part communication) other parts on the page. One of these is a little different; the Current User filter is a Context filter that knows who the current user is and can apply this to other parts (this also allows you to set a specific user). There is also one additional part called the Filter Actions Web Part, which, unlike the filters, gives a check box that enables the end user to Apply Filters on the page. This gives users the chance to make several selections and then apply them to see the result. By the way, Filter Web Parts are generally invisible on the page; the exception are those that accept user input (the Choice Filter for one).

This may not sound like much on the surface, but a savvy user could create pretty complex application functionality out of these. For example, using a simple Current User Filter Web Part, an entire site could be personalized so that every user that enters sees only their own information (think of a shared site that takes a bunch of common information, and with the filter, produces a My Site like information view). Conceptually, a picture's worth a thousand words:



Keeping the above in mind, you can see how the rest of the filters might fit in, including the Date Filter, the Text Filter, and so on. All these filters work the same way; use some data source, whether it's the user ID, a choice, an attribute to the page, another list, or even the BDC and SQL Server 2005 Analysis Cubes to pick a value to filter other parts on the page.

All Filter Web Parts work with communications; they can pass a set value, the default value, the selected value, and so on, and then target the specific list:



Note that each of these parts, while they work exactly the same way, do have some specific Web Part Properties that apply to the way they produce the filtering data (such as setting a default value, forcing the user to choose a value, and so on). This is a bit much to list here, and a lot of it is redundant, so you should check out each filter to understand the options available. A last note on the Apply Filters part: in addition to letting the user apply filters on demand, the user can also save the settings as their default so that when they return to the site, the previous settings are saved (this is an example of Web Part Personal storage where each user can save their own settings).

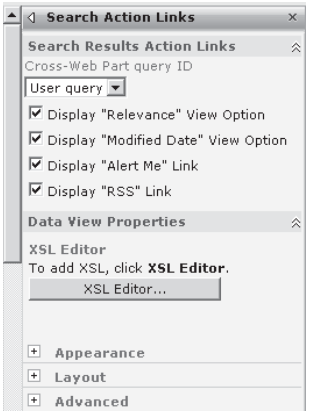
Search Web Parts

The Search Web Parts is another group of web parts that provide a combination of functionality. The primary Search Web Parts are the Advanced Search Box, the Search Box, and the People Search Box. Each of these represent the typical search functionality you would expect on a page. The Search Box can be dropped anywhere on a page, and it's just like the one included in the Site Templates by default. When used, users are redirected to the search results page to show the data (by the way, the results page is completely customizable).

Other search parts provide add on functionality to the search. For example, the Search Paging part allows you to apply a paging feature for search results, whereas the Search Statistics Web Part will display statistics alongside the search that indicate how long the search took. All of these are designed to allow you to add a search on a page where you need to and to let you create your own customized search and search results pages with the options, paging, and details you want. Search boxes are available in different styles based on the one provided at the top of the page, including:

- **Advanced Search Box Web Part** This provides the advanced search options that allow for a parameterized search (item contains, is equal, and so on).

- **People Search Box Web Part** This is a search box preset to the People Search Scope.
- **Search Box Web Part** This provides a simple search box with search scope.



Each of these returns results to their own search pages; these can all be customized with the add-on parts and with additional styles, and so on. The additional search parts add information or drill-down search functionality; for example, the Search Action Links Web Part enables you to add additional items to the display, such as the RSS feed link (if one is found).

Notice that you can even apply unique XSL to the information displayed by this part. The summary of web parts used to modify the results is as follows:

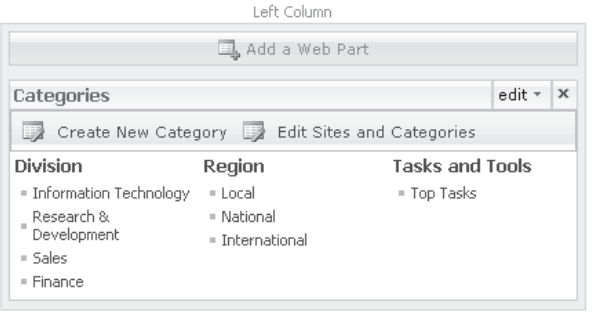
- **People Search Core Results Web Part** These are search results returned by the People Search Web Part.
- **Search Action Links Web Part** This adds links to search results.
- **Search Best Bets Web Part** This returns search results for Best Bets (highlighted items).
- **Search Core Results Web Part** These are search results returned by the basic Search Box Web Part.
- **Search High Confidence Results Web Part** These are search results for higher ranking items.
- **Search Paging Web Part** This creates a paging control for search results.
- **Search Statistics Web Part** This shows statistics on the search items (such as search time, and so on).
- **Search Summary Web Part** This shows a selected summary of a specific query.

Note that all of these web parts also have a Cross Web Part Query ID, which allows you to select which Query to apply the particular part to (this enables you to have more than one search on the page and more than one way to display each).

Categories Web Part

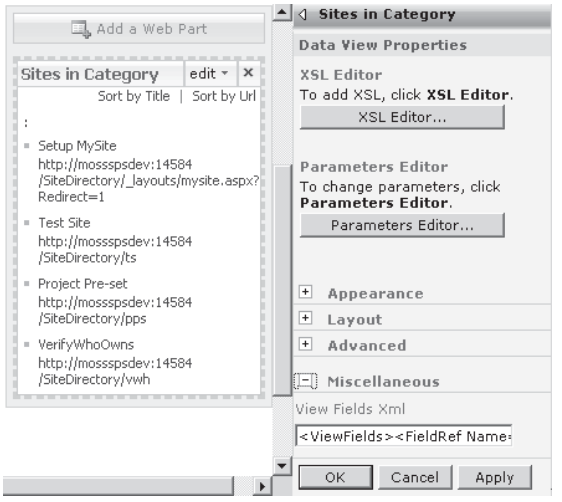
The Categories Web Part is a web part that is included in the Site Directory template (the default view when you enter a Site Directory site) and provides a categorical list of sub sites (the categories shown are defined in the Sites list in the site). There are two categories defined by default (Division and Region) that are simply Choice Fields in the Sites list. You can rename or delete them, change choices, and even add additional categories (any choice field becomes a category). This web part allows you to change categories in the settings, but you can also go to the list and modify it yourself. To view this part, simply navigate to the Site Directory. When the main page is displayed, switch to edit mode (Site Actions | Edit

Page) to reveal the toolbar as shown below. Note this has the same properties as the Table of Contents Web Part (as discussed previously).



Sites in Category Web Part

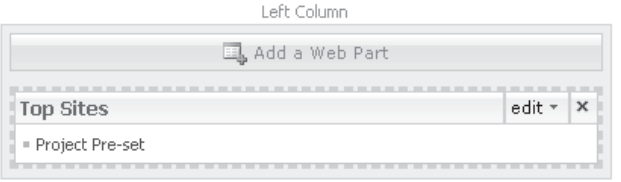
The Sites in Category Web Part is a simple list display that allows you to create a view of the Sites list in the Site Directory that you can style using XSL. You can also modify the XML query parameters. Be aware that this part requires configuration; without formatting with XSL and/or a custom query, this part is of limited use to users and the display is very plain:



Top Sites Web Part

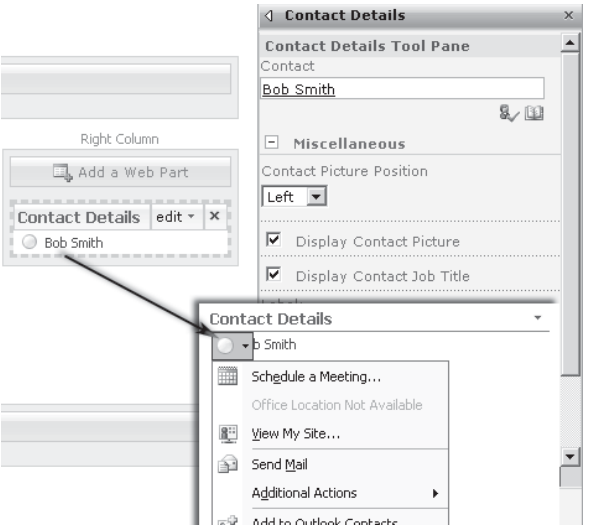
The Top Sites Web Part is a web part that is included in the Site Directory template and is available on the tab selection on the Site Directory site. This web part simply provides a list of all items found in the Sites list (sites) that have been marked as Top Sites, in order to

highlight them here. Other than display properties, this has the same items as the Table of Contents Web Part above. To view this part, simply navigate to the Site Directory, and then click on the Top Sites tab in the page toolbar. If nothing is displayed, no items in the Sites list have been marked. To change the display settings, switch to edit mode (Site Actions | Edit Page) to reveal the web part toolbar:



Contact Details Web Part

The Contact Details Web Part is used to create a page display of a user to be designated as the owner or contact of the particular site or page being viewed. When added to the page, it provides a link to open the web part properties to set the contact. Once the user has been set and the page saved, user presence is enabled, which provides a drop-down menu of ways to contact the user (this includes Live Meeting and Instant Messenger if enabled). Clicking the user's name sends the user to the Contact's My Site:



Summary

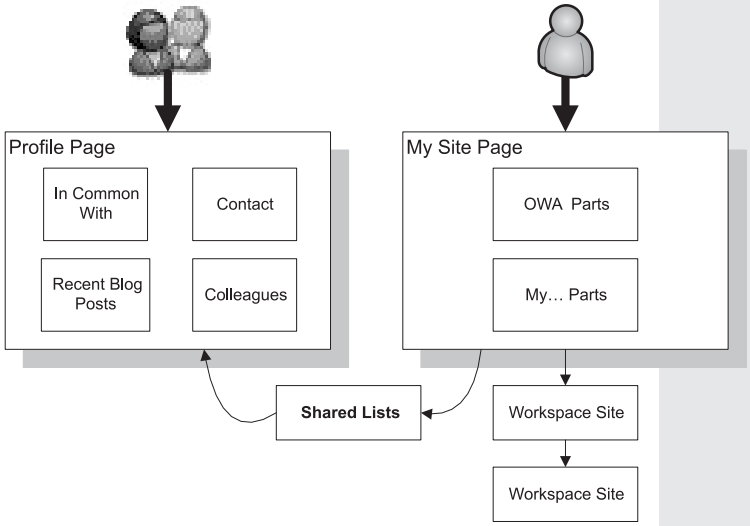
This chapter covered the web parts that are provided by both WSS and MOSS out of the box. As you begin to work with SharePoint, you'll begin to develop your own (as shown in Chapter 20) to create application-like functionality for users. Also, be aware that there are many third-party web parts available, some free and some for purchase. A simple search for web parts on the Microsoft site will provide a great starting point. Chances are if you need something specific, someone else did too.

13 CHAPTER

The MOSS My Site

The MOSS My Site is a personal site for individual users, and for all intents, works exactly the same as any other WSS site, In fact, the My Site is considered a top-level site. There are two distinct pages to this site: the My Site Page (My Home), which a user sees when they go to their personal site, and the Profile Page (My Profile), which all other users see. Users can store documents, links, and pictures (subject to the My Site Policy for storage of 2GB), get their alerts, and even use Outlook Web Access tools like the Inbox, Calendar, and so on, allowing the user to bring nearly everything they deal with on a daily basis into a single site. I've often encouraged folks to set this site as their Desktop link in Windows so that it's the first thing they see every day.

The My Site also works like any other site; outside users can be added to the site and lists and libraries can be shared with other users via the Profile Page (this makes them accessible to visitors). Users can also use another feature called the Personal Blog, which allows users to enter comments and observations, either as an ongoing commentary, an information source (tips and tricks), or as a personal diary of events. Users can either set this as private, or they can use the Recent Blog Posts Web Part to show the latest entries on their Profile Page. The Profile Page is a business card or a contact page of sorts; whenever a user's name is displayed in a MOSS site (as a link) or appears in Search Results, this is a link that brings the user to this page. When navigated to, the user is detected and sent to the appropriate page:

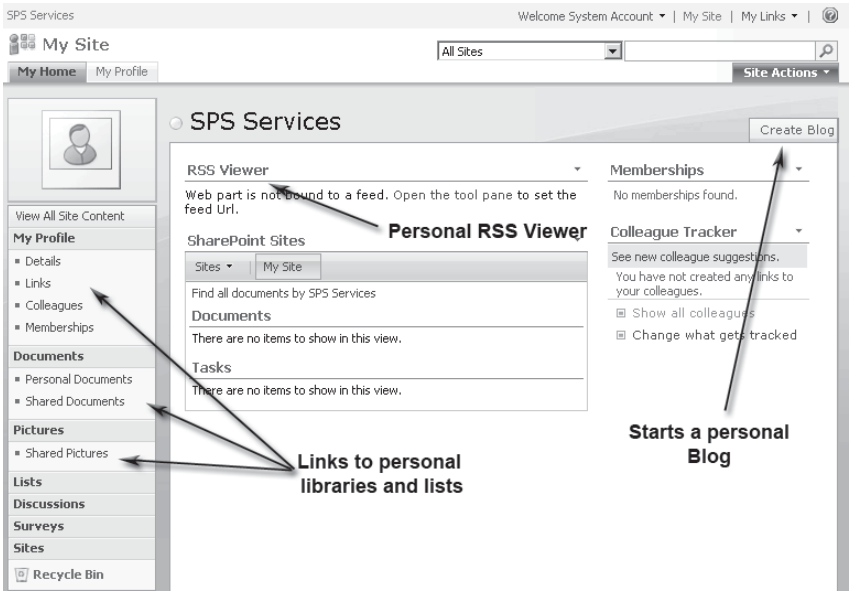


By default, My Sites are not created automatically for users. To create one, the user must be a member of the portal site (with rights) and must login to the site and click the My Site link in the Personal Settings links. When a user does this, their My Site is automatically created in the Shared Services Provider My Site Host associated with the portal site. Also, there are two parts of the User Profile data about a given user: data brought over from Active Directory and the several items that are specific to SharePoint. Users can usually maintain all of their SharePoint Profile information, but keeping AD profile information up to date is not automatic. The Shared Services Provider must be configured to import User Profile information on a regular basis.

Note that as I mentioned in Chapter 6, the layout of the My Site My Home Page is dictated by the settings of the System Account My Site; web parts added and removed are automatically set as the soft template for all My Sites. Because of this, I highly suggest you use the System Account My Site to set the template, but do not allow anyone to use it for personal reasons.

My Home Personal View

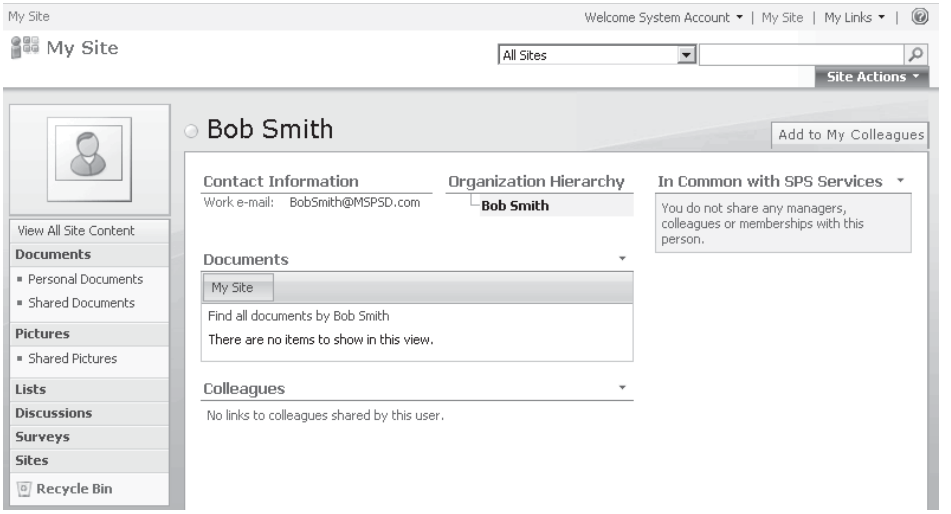
The view the user gets in their My Site is pretty basic when first created; by default, none of the OWA parts are included (since they require setup) and neither is the Blog. Out of the box, only a few personal lists, libraries, and a shared library are created:



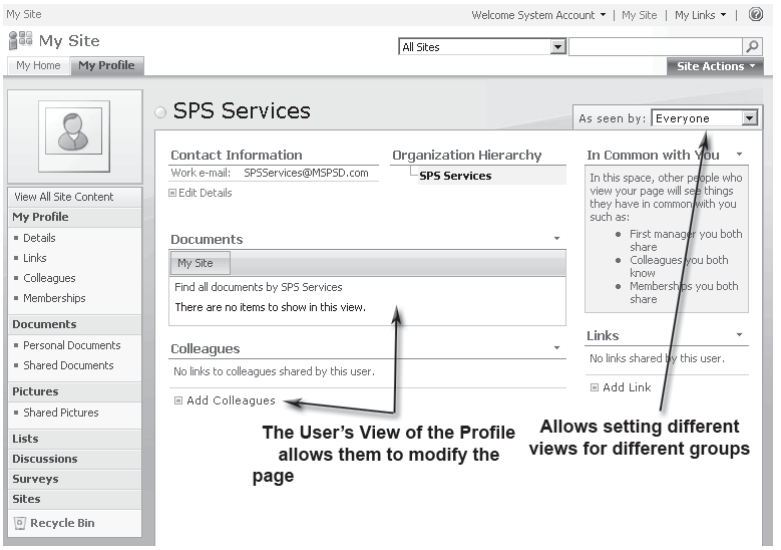
As you can see, this looks pretty much like a Team Site without all the lists, and you're right—the only difference is that for this site, the user is the administrator and only members and the lists already created have "My" as part of the name (My Pictures, for example).

My Profile Public View

When outside users come to visit someone else's My Site, they get the Profile Page. This provides a picture of the user (if available), as well as contact and colleague information that I liken to a business card (a shingle site is another term). In addition, this page provides access to content and documents that the user has shared. This gets to the lowest level of collaboration; the user can make it available through their My Site Profile Page without having to create a workspace.



The content and items shown on this page are controlled by the My Site user. When a user goes to their own Profile Page, they can customize and modify the layout to decide what others can see (selectable by groups, manager, and so on), and they can add/edit items to lists:



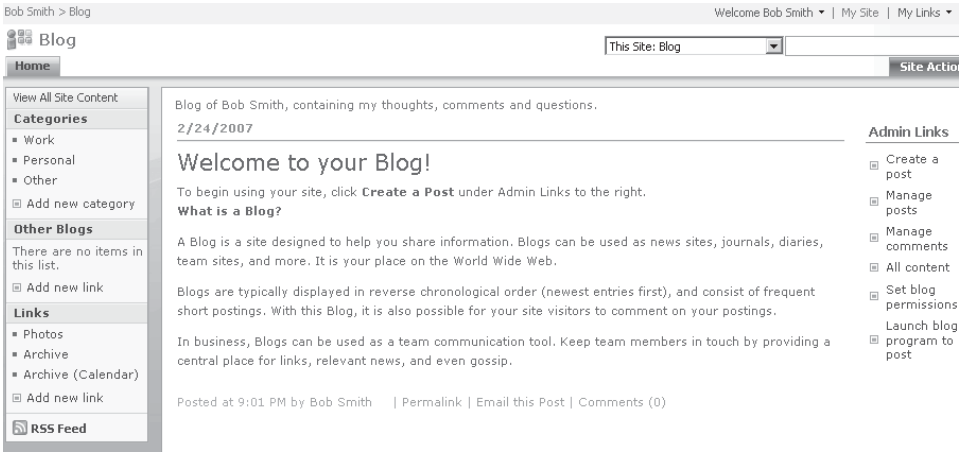
My Site Web Parts

In addition to the additional parts that MOSS gives you, there are several that are added for use specifically in the My Site and are intended for personalization to the User level. For example, the Outlook Web Access parts all rely on Microsoft Exchange and OWA and are pointed directly to the user's information in Exchange. Others are intended to be roll ups of information elsewhere in the site, which allows users to create a one-stop location for seeing information across the board. In addition to those available for SharePoint, the My Site Web Parts include the following, as shown in Table 13-1.

Note that because many of these parts are identical to the basic lists that have already been covered (as well as the process of adding them to the page); the following descriptions will focus only on the part's purpose and its settings.

Creating the User Blog (My Blog)

Blogs are new for SharePoint and have been included as both a new site template and as a web part for use in sites. This functionality has also been extended to My Site and has, therefore, included the ability to create a blog and then maintain it on a personal level. When a user's My Site is created, necessary components (like the Recent Blog Posts Web Part) are included, but are not shown and blog list is not created. To indicate that it hasn't been created, there is a Create Blog link just under the Site Actions menu on the My Home page of the site. When the user clicks this, the blog is created and displays a default message explaining how to use it:



As you can see, the administrative options allow the user to control the blog entries; also notice that there is an RSS icon shown (if enabled, Blogs can provide RSS feeds).

My Calendar Web Part (OWA)

The My Calendar Web Part connects the web part to the user's Calendar folder in Exchange. Using the same control as Outlook Web Access, this displays the calendar within the My Site. Users can view, add, and remove items from the calendar and link SharePoint Calendars to their Outlook Calendar. Web Part Property Settings include specifying the Mail Server address, mailbox (e-mail) name, and choosing the view for default (Daily, Weekly, and similar).

Web Part	Description
Blog	
My Blog	A personalized blog for either personal or public use. This provides the same functionality available in the standard Blog Web Part.
Outlook Web Access (OWA)	My Site Use Only
My Calendar	This uses OWA to connect to the users Outlook Calendar.
My Contacts	This uses OWA to connect to the users Outlook Contacts.
My Inbox	This uses OWA to connect to the users Outlook Inbox.
My Mail Folder	This uses OWA to connect to the users Outlook Mail Folder.
My Tasks	This uses OWA to connect to the users Outlook Task list.
Content Rollup	
Colleague Tracker	This displays a list of colleagues (associated to the My Site user) and recent changes to their profiles and is intended for either the Profile or My Site Page.
Colleagues	This displays a links list to colleagues related to the My Site User, which is intended for the Profile Page.
Get Started with My Site	This displays a link list that provides links to various hints and tips about using the My Site (local help for new users) and is intended for the My Site Page.
In Common Between Us	This shows what the current user and the site user have in common (such as same department and so on) and is intended for the Profile Page.
Memberships	This displays all of My Site User's site and distribution lists memberships and is intended for either the Profile or My Site Page.
My Links	A personal link list for links entered, as well as those added from content in the site. This is intended for the My Site Page.
My Pictures	A personal library for images. This is intended for the My Site Page.
My SharePoint Sites	A display list of documents authored by the My Site user and any other sites added.
My Workspaces	This displays a link list of all sites created under the My Site and is intended for the My Site Page.
Recent Blog Posts	This displays the most recent blog posts by the My Site User and is intended for the Profile Page (though useful in the My Site Page if the blog is a diary).
SharePoint Documents	This displays a list of documents authored by the user where the user is member and is intended for the My Site Page.

Table 13-1 MOSS My Site Web Parts

My Contacts Web Part (OWA)

The My Contacts Web Part connects the web part to the user's Address Book in Exchange. Using the same control as Outlook Web Access, this displays the address book like a contact list within the My Site. Web Part Property Settings include specifying the Mail Server address, mailbox (e-mail) name, and choosing the phone list to view.

My Inbox Web Part (OWA)

The My Inbox Web Part connects the web part to the user's e-mail folder Exchange. Using the same control as Outlook Web Access, this displays the user's e-mail Inbox folder with a similar look and feel to Outlook's. Web Part Property Settings include specifying the mail server address and mailbox (e-mail) name, as well as choosing the view: Two-line Summary, Messages (All), By Sender, Subject, or Conversation.

My Mail Folder Web Part (OWA)

The My Mail Folder Web Part connects the web part to the user's folders within Exchange. Using the same control as Outlook Web Access, this displays the user's folders in an Explorer format. Web Part Property Settings include specifying the mail Server address and mailbox (e-mail) name, as well as choosing the Exchange folder name (usually Inbox).

My Tasks Web Part (OWA)

The My Tasks Web Part connects the web part to the user's Tasks folder Exchange. Using the same control as Outlook Web Access, this displays the user's tasks in a list format. Note that this list is from Exchange and not from the site; users must include their tasks in Outlook before they will appear here. Web Part Property Settings include specifying the mail server address and mailbox (e-mail) name, as well as choosing the view: By Due Date or Subject.

Colleague Tracker

The Colleague Tracker Web Part displays a list of colleagues associated with the My Site user based on workgroup and user profile information. This will display the colleague's contact information and list any recent changes to their profiles based on the part's settings. This part also has an option to list potential colleagues based on their profiles. This part has several settings that determine what is tracked about colleagues; users can enable/disable tracking on changes or upcoming Anniversaries, Profile Property Changes, Membership Changes, New Documents, Out of Office Messages, and Blog Changes.

Custom Properties include settings for:

- **Show Colleagues Without Changes** This enables/disables showing colleagues with no recent changes.
- **Only Show Colleagues in My Workgroup** This limits the colleagues to the user's immediate workgroup.
- **Number of Columns to Display** This displays from 1 to *n* number of columns horizontally.
- **Remind Me to Update My Colleagues In** This sets a ticker reminder for keeping colleague information up to date.
- **Specifies the Maximum Number of Items to Display** This sets the maximum number of items to display on the page.

Colleagues

The Colleagues Web Part displays a list of users that are colleagues, as added by the My Site User. A simple way to think of it is as a list of people the user is associated with. There are only two settings for this part: the number of columns to display in the list and the number of rows.

Get Started with My Site

Loaded on the My Home Page by default when the site is created, this web part is basically a link list that provides links to various hints and tips about using the My Site. It's also a kind of local help for new users who need to get acquainted with using the My Site and the tools within it. This is intended to be used temporarily and removed by the user when it's no longer needed. There are no settings for this part.

In Common Between Us

This part is a fixed part on the Profile Page, and when shown, it compares the user profile information of the viewing user to the user profile of the My Site user. Anything in common, such as manager, department, and so on, will be shown, but if there is nothing in common, it will indicate that as well. There are no settings for this part.

Memberships

This displays a list of the My Site User's Sites and Distribution List memberships; this can be used on either the home or Profile Pages. There are only two settings for this part: number of columns to display in the list and the number of rows.

My Links

This is a personal links list used in the regular fashion but it is also used to store links that a user selects from within a site (using the Add to My Links option) and can also be shown on the Profile Page. Links can be added here directly or from any page in the site. The user's personal menu links allows the user to add the link from the current page or item directly to this list. There are no settings required for this list.

My Pictures

This is a personal Picture Library for storing images and it can be shown on the Profile Page. There are no settings required for this list.

My SharePoint Sites

This part is basically a different named version of the Site Aggregator Web Part and functions exactly the same way.

My Workspaces

The My Workspaces Web Part displays a link list of all sites that have been created under the My Site. This part has only one Web Part Property; under the Miscellaneous section there is a check box option to Show Detail, and if this is checked (the default), the list will display the date and time the site was last accessed and when it was created. This can be used on either the home or Profile Pages.

Recent Blog Posts

This web part is included on the My Home Page by default and displays the most recent blog posts by the My Site user. Initially, this part is hidden on the page and is not displayed until the user creates their blog; once the Blog list is detected, this part will become active. Settings for this part include adding custom XSL to manage the display and cache settings for connections to the Blog List.

SharePoint Documents

This part is basically a differently named version of the Site Aggregator Web Part and functions exactly the same way. Settings for this part make it display a list of documents authored by the user where the user is a member (like My SharePoint Sites but only documents).

Creating Sub Sites in the My Site

As a Site Collection, the My Site has all of the same features that would be found in a top-level site in either WSS or MOSS. Users can create new lists, content types, and even manage users. They can also create any number of sub sites (workspaces) under their My Site and can use any of the same templates available in WSS and MOSS (except the Portal Site Templates, Collaboration, and Publishing). Users can use Workspace sites for any purpose, such as developing a document, and can invite others to work with them as members. Creating a site is done exactly the same as with a WSS Site:

1. From the My Home Page, click the Site Actions menu and click Create.
2. On the Create Page, click Sites and Workspaces.
3. Enter the title, description, and URL for the site, and then pick the template.
4. Set the navigation options for Quick Launch and the Navigation Bar, and then click OK to create the site. The site will automatically appear in the My Workspaces list.

User Personal Settings, Login and Out, and My Links

The Personal Settings menu and My Links menus are available from every page in a site (usually the top-right corner) and provide the user with access to their personal settings, the ability to change logins, and so on. There is a slight variation in this menu, because whereas in WSS you can set a personal view, in MOSS you have a different option to request access to the current site.

The links available in these menus are as follows:

- **Sign In as a Different User and Sign Out** These both operate as you would expect. If you choose to log in as a different user, you will be prompted to log in with new credentials (account and password). If you choose to log out, the browser window will be closed.
- **Requesting Access (MOSS)** This is a simple form that sends an e-mail to the owner of the site or page to request access (this is the same as entering a site you are not a member of).
- **My Settings** In both WSS and MOSS, this link allows the user to make changes to their personal settings. There is a slight variation between WSS and MOSS (both

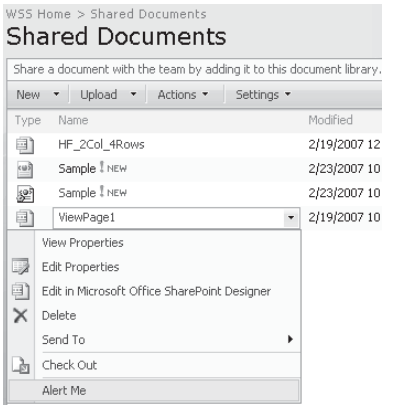
include the same information, but MOSS has several additional attributes from the User Profile).

- **My Links** These do have a few default links:
  - **My SharePoint Sites** These provide a submenu link to all of the sites that the current user is a member of (much like the web part in the My Site).
  - **Add To My Links** This adds the current item or page to your My Site My Links list.
  - **Manage Links** This allows a user to add additional personal links or otherwise manage them in this list (any valid URL is permitted). Note that this is not related to the My Links part in the My Site.

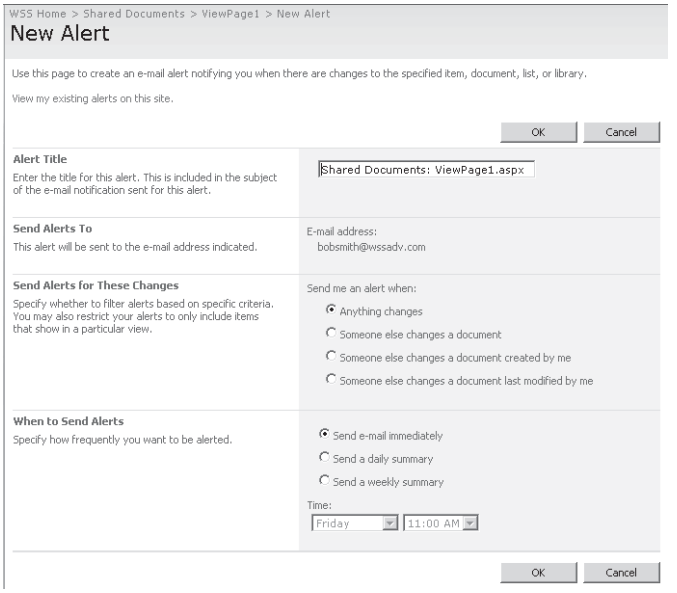
Note that additional links may appear here (others can be added at the Site Administration level and user-created links).

Using SharePoint Alerts

SharePoint alerts are likely one of the strongest collaboration tools provided because they give SharePoint users the freedom to be notified about things both when and how they want without having to go look to see if something has changed. It can also be used to track search results and notify them of an event (such as being assigned a new task, for example). Alerts are set the same way in both WSS and MOSS at the Site, List/Library, and Item level. Just look for the Alert Me link on page and item menus. For example, setting an alert on an item can be selected from the item menu itself:



Once an item is selected, you can set the options on the alert and then indicate the frequency of your notifications, among other things:



Now as cool as alerts are, don't go out of control with it unless you really want to get lots of e-mail. If you are only concerned with a certain item, set the alert at that level, and then if you decide you're getting too many alerts, delete a few. Note that Site Administrators do have the ability to manage and delete alerts for sites, lists, and items (this is not centrally managed).

Summary

This chapter introduced you to the MOSS My Site, personal sites available for all users. My Sites are a way in which users can focus information from multiple sites and Microsoft Outlook into a concise one-page site. In addition, they provide a personal home page or business card–like site for visitors, an excellent tool for truly collaborating between individuals. As I've mentioned, My Sites are great for collaboration; they are also an excellent way to gain user adoption of the system.

14
CHAPTER

Site Columns, Content Types, and Content Management

By this time, you have a pretty good idea about what SharePoint gives you out of the box in terms of lists and functionality as provided by web parts. You should also have a pretty good grasp on what content, sites, pages, and lists are. What you may not have noticed is that you've only dealt with a single site or site page (in the case of web parts). In addition, you'll recall that I've mentioned that sites don't share content between them; they can reference each other via links, but the content is solely based on the individual site.

Of course, there is a need to share content and things like workflow. For this, SharePoint provides Site Columns and Site Content Types. These are independent of lists and libraries and can be used anywhere in a collection. Site Columns are like a list with only one column and can be any kind of data. These can be referenced by any list in any site (useful for providing drop-down lists), which breaks the barrier that keeps sites from being able to share content. Site Content Types (which use Site Columns) are like a definition of a list with no list, a profile of a document, or the definition for a Publishing Page (the latter is what replaces MCMS content management).

Site Columns

As mentioned throughout, lists are made of items (rows) that have columns (fields) that hold data. Columns defined in a list are a part of that list; they can be referenced by other lists in the site but not outside of it. Site Columns, however, are related to the entire site collection and can be used anywhere (all sites, all lists). In a way, Site Columns represent the lowest form of content management since they provide common reuse. Site Columns are inherited by a list; if the Site Column is changed, it can optionally force the change down to any lists that use it (the exception is that deleted data from a Site Column is not removed from the lists that use it).

You might say that a reusable column definition as content management is a stretch, but part of real content management is controlling it at the lowest level and reducing duplication whenever possible. Site Columns give you both of those aspects; since the column's data and format is independent and any type (columns can be static data, lookups to other data in SharePoint, or external data via the Business Data Catalog), they are reusable for eliminating duplicated data and prevent defining the same kind of column over and over again.

Organizing Site Columns is done by Site Column Groups, which are simply names that allow you to categorize the type of column by what it might be used for and to identify it as not a SharePoint Site Column. As you might have guessed, SharePoint uses Site Columns in several of the predefined lists. These columns are already defined under groups that include:

- **Base Columns** These include columns like Start Date, End Date, and URL.
- **Core Contact and Calendar Columns** These include the typical columns for Address, Birthday, Business Phone, and so on.
- **Core Document Columns** These include Author, Date Created, Source, Status, and so on.
- **Core Task and Issue Columns** These include Percent Complete, Priority, Task Status, and so on.
- **Custom Columns** These include Form ID, Form Locale, Template ID, Version, and so on.
- **Extended Columns** These include Company Phonetic, Issue Status, Purpose, Related Issues, and so on.

While groupings are just names, you should choose ones that will be easily recognizable by the end user who will use them; similarity in Site Column names can be distinguished by the Group and can help the user to avoid using the wrong one. Remember, a change to a site column can be pushed down to the lists that use it; if the wrong one is used, it could mess up the data.

Creating a Site Column

Creating a Site Column is done the same way in both WSS and MOSS and is done at the Site Collection (top level) through Site Administration (Site Settings). For this example, we will define a Site Column based on the following:

- **The Problem** A company handles a large number of issues, tasks, notes, and meeting-documents that are tied to generating releases of packaged software. When nearing a new release, technical writers scan through the sites to find all items that might need to be included in the Release Notes for the software. To do this, they need something common across all sites and lists that will allow them to identify the right materials.
- **The Solution** Create a Site Column for Release Document Type to be used in lists to create a common thread (Note that you could just as easily create a column for release number as well).

The steps you'll need to take to accomplish this are to first create the Site Column and then use it in a list, creating the column itself:

1. Navigate to a top-level SharePoint Site, and then if WSS, click the Site Actions menu and select Site Settings. If MOSS, use Site Settings | Modify All Site Settings to open the Site Settings page.
2. Under the Galleries Group, click Site Columns.
3. On the Site Columns Gallery page, you can see the current list of Site Column Groups and Columns. Click the Create link at the top of the page.
4. This opens the Create Site Column page. As you will notice, it is not remarkably different than creating a column in a list. You will see one difference, in that if you set the group, it will be created (you can select an existing one or create a new one); all other settings are the same.
5. Click OK to create the column. When you return to the list, you should be able to find the group and column.

Changing a Site Column

You can change a Site Column, but changing the data type from one to another is not always permitted. Also, if it's a choice field and you remove or change options, the original value is not changed or removed from any existing list items (to preserve integrity in the items as they were entered). Other than that, everything is editable, though you should keep in mind that you cannot delete a Site Column if it is being used.

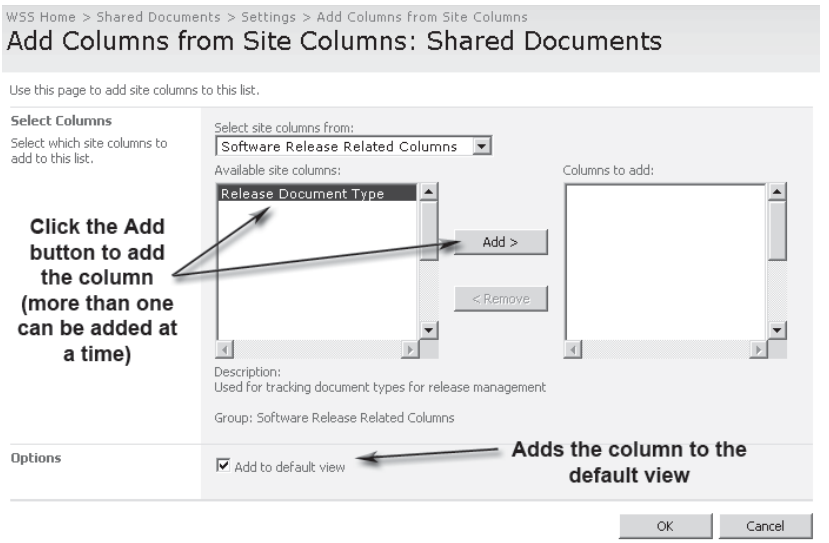
When editing, there is an additional option on the Edit Site Column page: Update All List Columns Based on This Site Column. When set to Yes, changes made to the Site Column are pushed down to any lists that use it (this is usually what you want). If set to No, current users of the column are left unchanged, but any new ones that use it will use the new definition. As mentioned, removing something (like a choice) does not update item data in lists.

Using a Site Column

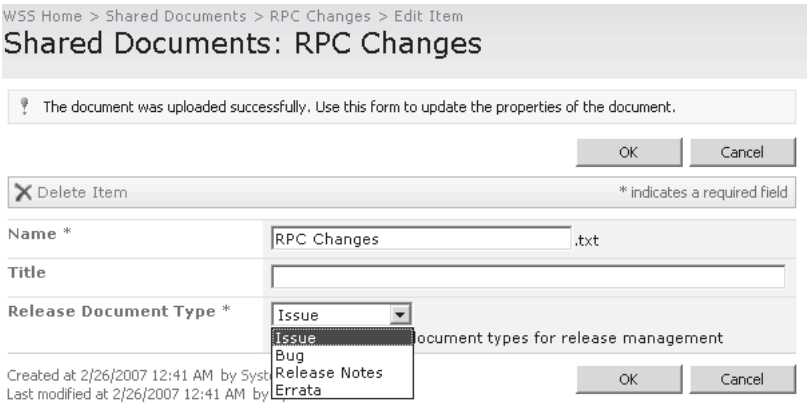
Any list or library can use a Site Column, and they can be added just like regular columns, except no settings are required. To add a Site Column, for example, you will use the column created previously, along with an existing document library. This is a Team Site, so the Shared Documents Library already exists. To add the Site Column to the library (or any list), do the following:

1. Navigate to the page where the document library resides.
2. Click the link to the library in the Quick Launch bar if shown or click the View All Site Content link. Then on the Content page, click the name of the library to open it.
3. On the toolbar, click the Settings menu and then select Document Library Settings.
4. On the Settings page, scroll down to the Columns section and then click the Add from Existing Site Columns link.

5. On the Add Columns from Site Columns page, select the group in the drop-down menu to show the columns in that group in the list. Click to select the column:



6. Click the Add button, which will move the column into the Add Column panel on the right (if more than one column is shown, multiple columns can be added at one time). Leave the Add to Default View option checked (this makes this column visible) and click OK.
7. When you return to the Settings page, you should see the added columns under the Columns Group. Return to the List view using the breadcrumb link at the top of the page.
8. Click the Upload button to upload a new document.
9. Browse to select a document or file (anything will do).
10. When you return to the list, you'll see that the Site Column is shown with the choices:



NOTE *If you wish to see change in action, you can return to the Site Columns page and click on the column name to open it for editing. Add a new option and return here to see the effect.*

One warning on Site Columns; they can be used (inherited) by Site Content Types, but you can't use the publishing type fields (for example, the Full HTML type). If Site Columns are included and then added to a Publishing Page, the page will not render and you will get an error (as a general rule, don't use Site Columns for Publishing column types).

Site Content Types

Site Content Types are a kind of list template without a list. These allow you to create profiles (or an entity) for content that can be used to preset the columns and data needed for a particular type of content. The entity might be an information resource or represent a real thing, like a client profile. In libraries, these are used as document profiles for creating document types in libraries, each with its own metadata. In Publishing, they are used to create Page Layouts; users use these Layouts to create Site Pages that are stored in Page libraries. Another major use for Site Content Types is that they provide global access to workflows across a site or site collection. Whereas workflows can be attached to a list or library, a workflow can be attached to a Site Content Type. When associated with a list or library, the workflow comes with it. Site Content Types also allow a few other attributes that are, again, attached to the type and not the list or library that uses it:

- **Document Information Panel Settings** This allows a customized Document Information Panel (DIP) shown in Microsoft Office applications to be carried with the type. The DIP is a customizable XML file that provides a form for entering list column data for Office documents.
- **Information Management Policy Settings** This allows the setting of policies for the type (auditing, barcode, and so on).
- **Manage Document Conversion for This Content Type** This allows defining conversions for this content type. This is part of Enterprise Content Management (ECM) you will read about later on.

Site Content Types can be defined with any type and number of columns, but it cannot be defined by itself because it must always inherit from a basic type defined in SharePoint (a starting point if you will). For example, to create a document profile you could inherit the SharePoint default Document Content Type (this contains only the file type and the title field). Content types can inherit from others, so you can build very complex document-metadata compositions rivaling the best Document Management systems. At the core, content types can be used simply within SharePoint to define entities that can be used in lists and libraries. This use can also be extended into Microsoft Office using InfoPath and embedding properties (metadata columns) directly into documents (you'll see this next).

Exercise: Creating a Content Type

Creating a Content Type requires setup; this means that you can't just go create one, because you need two things up front: a SharePoint type to inherit from and, optionally, Site Columns that it will use for metadata (I say optional, but there is not much point in duplicating an

existing type verbatim). For demonstration purposes, I'm going to create a document type wherein the full lifecycle (integration with Office and InfoPath) can be shown. The basis for the document will be to serve as a Building Work Order—basically a simple document, but tagged with the corresponding building ID. I know this sounds pretty simplistic, but trust me, this will touch on the entire process.

Exercise: Integration with Office

All of the metadata included in a particular content type is carried over to Microsoft Office documents so that users on that side can specify the required metadata. The Document Information Panel (DIP) is a feature in Office that provides a data-entry form for metadata (this can be customized using InfoPath and utilize InfoPath Views). These properties stay with the document, are visible, and can be updated while offline. A key feature is that the properties are requested from the user up front, instead of when saving, thus bettering chances of users entering the information. In addition, any Office Properties, such as Author, automatically fill in other standard properties in SharePoint (for example, Created By).

When a document is first opened, Microsoft Office detects that the metadata is empty and automatically displays the DIP. Required fields, drop-down menus, and so on, are all supported just as if they were entered in SharePoint, and the metadata itself can also be used directly in the document.

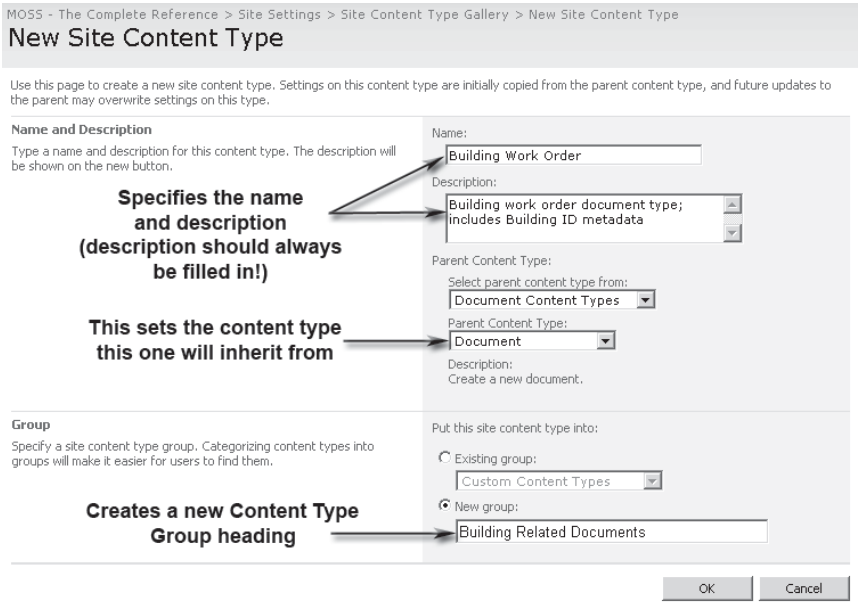
Exercise: Create Site Columns

To begin with, a Content Type is based on a Content Type Template and the one or more Site Columns that extend the metadata. If you base your new Content Type on the default Document Content Type, you automatically get the File and Title columns. This means that a Site Column is needed for a Building ID. Creating a Site Column was covered previously, so I'll cover it only lightly here (and I'm only creating one content type; you can create more if you like):

1. First, navigate to the root site (home page) of the SharePoint site.
2. In WSS, click the Site Actions menu and then select Site Settings (in MOSS, that is Site Settings | Modify All Site Settings) to open the Site Settings page.
3. Under the Galleries Group, click Site Columns.
4. On the Site Columns Gallery page, you can see the current list of Site Column Groups and Columns. Click the Create link at the top of the page.
5. On the Create Site Column page, set the name of the column to be Building ID.
6. Set the type of column to be a Choice.
7. Set the Group to be a new group, and enter the new group called **Building Document Site Columns.**
8. Under the Additional Column Settings, enter a description as a Building ID to which this document is related.
9. Click the Yes radio button next to Require That This Column Contains Information.
10. In the choices field, enter the following choices: **Building 10, Building 11, Building 12**.
11. Leave the other settings as is, clear the default value, and click OK to create.

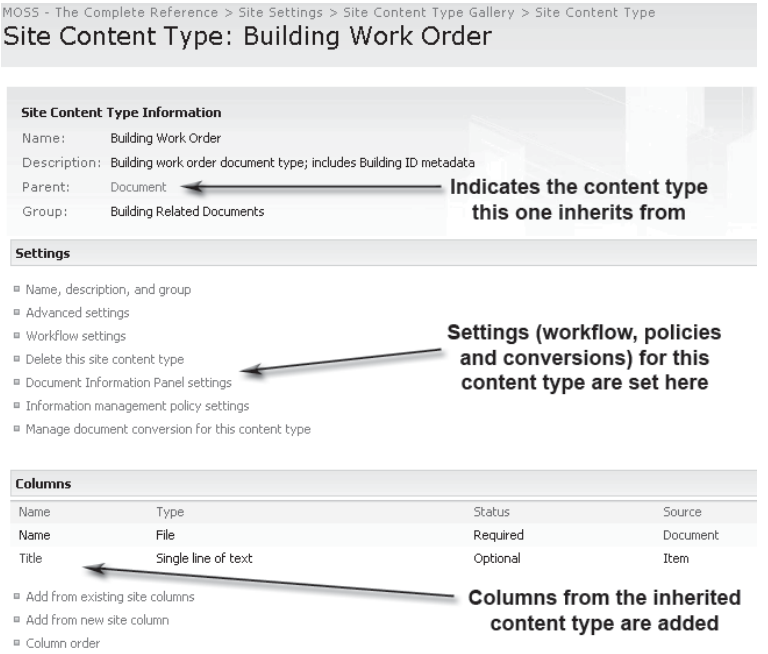
Exercise: Create a new Content Type

1. First, navigate to the root site (home page) of the SharePoint site.
2. In WSS, click the Site Actions menu and then select Site Settings (in MOSS, that is Site Settings | Modify All Site Settings) to open the Site Settings page.
3. Under the Galleries Group, click Site Content Types.
4. On the Site Content Type Gallery page, you can see the current list of Site Content Types (notice that like Site Columns, they also have Groups). Click the Create link at the top of the page; this opens the New Site Content Type page (I've already filled in the settings):



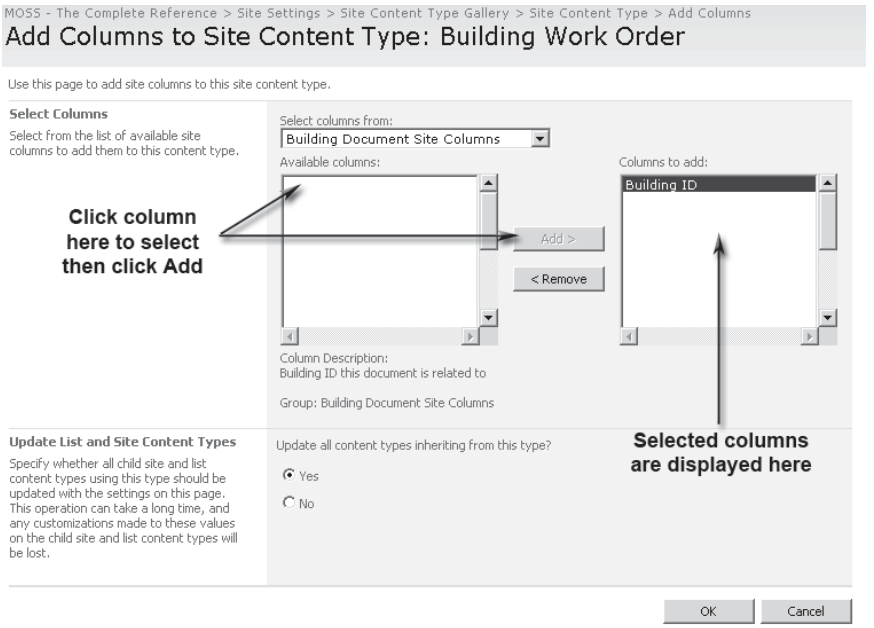
5. Under the Name and Description you supply:
   - **Name** The name of the content type.
   - **Description** A description of the type and what it is used for (this is very important as this is shown to users when this content type is used in a list).
   - **Parent Content Type** This allows you to select the Group and Template this type will inherit from; since you are creating a document, you inherit from the Document content type (SharePoint's default).
6. Finally, you assign this content type to a Group; since this is a new one, you will set the new group to be Building Related Documents.
7. Click OK to create the content type.

8. When the Site Content Type definition page is displayed, you can see that from the inherited template, there are already columns included:



CAUTION *Remember that the columns included are Site Columns; if you modify one of them here (such as rename Title), that modification will also affect every list and content type that uses it.*

9. Next, you need to add the Site Column created previously. At the bottom of this page in the Columns section, you can see the options to Add from Existing Site Columns, which adds from those already created; Create a New Site Column, which brings you through the process shown previously, but automatically adds it here; and Column Order, which allows you to reorder the columns. Note that this affects this Content Type and not the one it inherits from; in other words, the parent stays as is.
10. Click Add from Existing Site Columns.



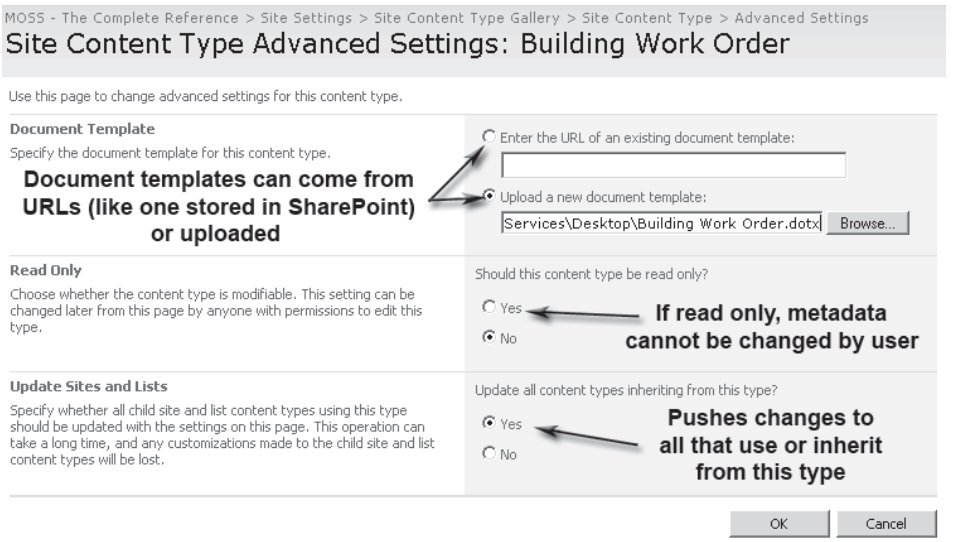
11. Use the Group drop-down menu to select the Group. Next, click on the Site Column name and then the Add button to add it (more than one can be added if there are more). Then, click OK to save the change; when you return to the Site Content Type definition page, you should see the column listed.

Exercise: Assign Content Type Settings

Once the content type has been created you can immediately start using it, albeit with the default settings. Let's take a moment to cover the rest of the settings on the Site Content Type definition page:

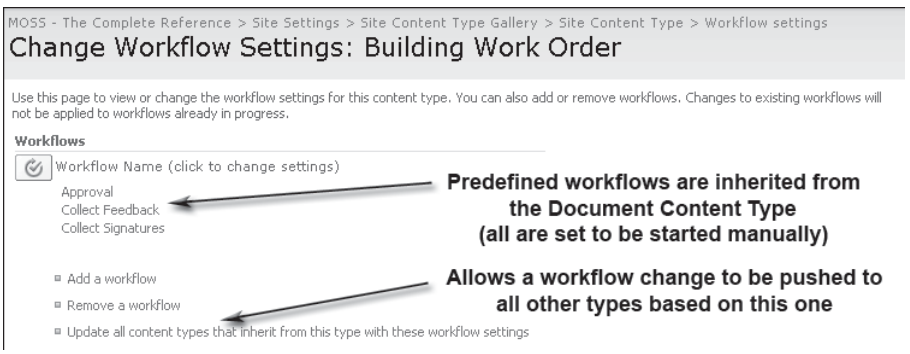
- **Name and Description** This allows you to change the Name and Description of the content type.
- **Group** This allows you to reassign or create a new group for this content type.
- **Advanced Settings** This allows you to set a specific document template to use with this particular content type. In this case, I've created a Word document

template to upload from my desktop folder (this could be any template, even a SharePoint page):

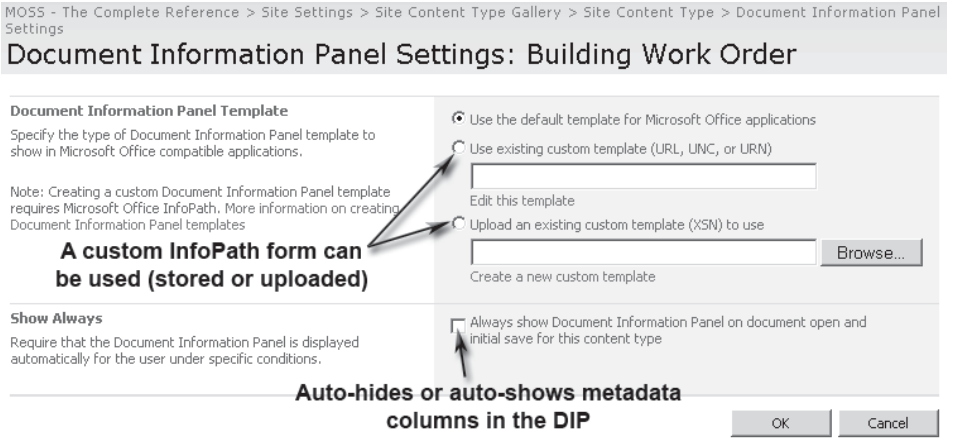


The default template is based on the default Document Content Type, which is the generic template type assigned to the library; assigning one here overrides the template for this Content Type (which gives you a choice as you will see). Additional settings include Read Only, which can set content type columns as read only (but can't be changed by the end user) and Update Inheriting, which means that this change (the Word document template being added) will be pushed to all content types that inherit this one—automatically changing them.

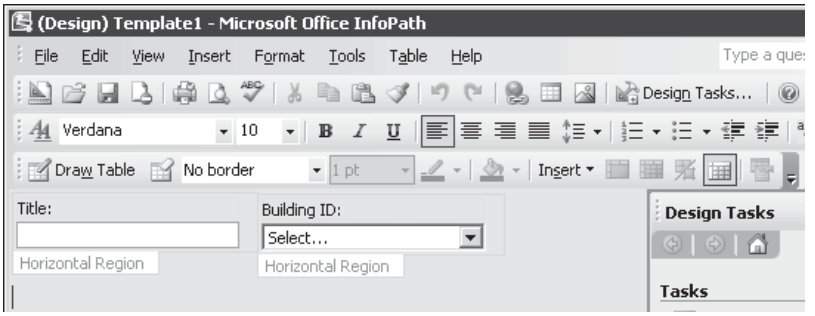
- **Workflow Settings** As you saw in setting up a workflow for a list, a content type is no different; any of the installed workflows can be used. These workflows, as you recall, are templates that then create a new template that will go along with this content type. The settings selected for this workflow are the defaults and can be overridden when a workflow is started by a document using this content type. Clicking this link opens the workflow selection page; as you can see, three workflows are already defined (these are inherited from the Document Content Type):



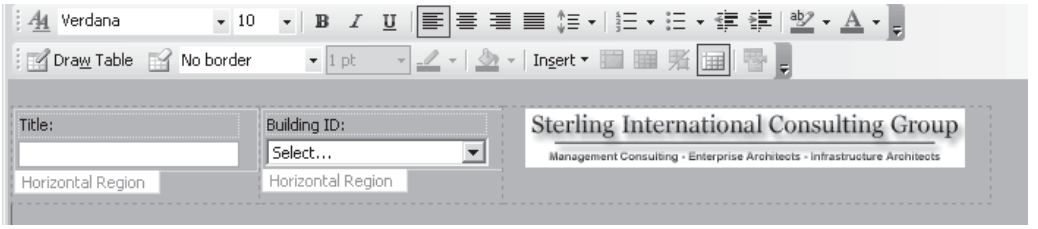
- **Delete This Site Content Type** This will obviously delete the Content Type from SharePoint; if the content type is in use, you must remove it first before it can be deleted.
- **Document Information Panel Settings** The Document Information Panel is a part of Microsoft Office Applications (such as Word) and is displayed when a document from SharePoint has been opened. This panel is really just an InfoPath form that shows the Columns that are associated with the Item (the metadata). Because this is InfoPath, you can completely customize the form itself (add a logo and so on), as well as the layout of the columns in the form. You can also use managed code (with Visual Studio) to extend the InfoPath form to read external data and fill in fields automatically (a good use for read-only content types since they can be set programmatically this way). From here, you can choose to use the default, specify one that has already been created, upload an existing one, or even create a new one:



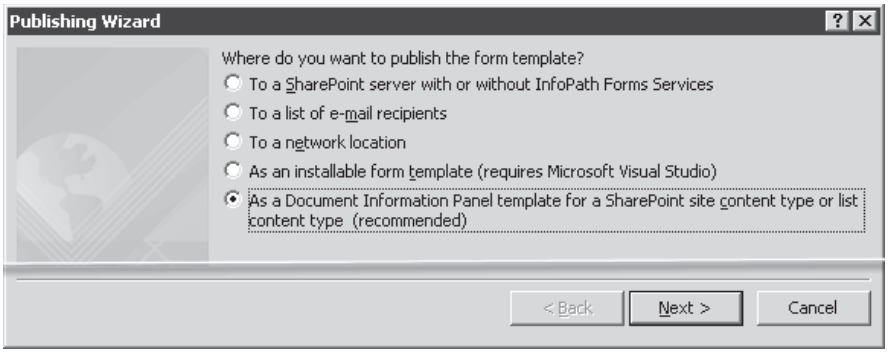
Just as an example, you can create a custom DIP for this particular content type (though to do this you must have InfoPath installed). To create it, click the Create New Custom Template link, which will automatically launch InfoPath. When InfoPath first opens, you will receive a dialog box that will notify you that InfoPath will set up the form for managing properties (columns) and that if you make changes, you must publish the template. Simply click Finish. The current metadata columns will be retrieved and automatically laid out in the new form:



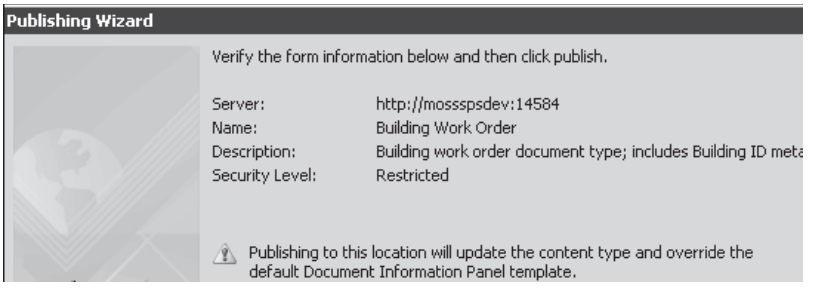
Notice that the fields are automatically laid out and in order (this is the order from the content type itself). Also notice that the Building ID Site Column shows up correctly as a drop-down menu selection. You can now design around this all you like; in this example, I'm going to simply change the background color from light blue to orange, put the fields in a table, and add the SICG Corporate Logo to the page:



Next, the form must be saved locally (to the PC hard drive) before it can be published to SharePoint; click the Save Icon, or from the menu, select File | Save. If enabled in InfoPath, when you save, it will notify you that when you want to make this form available, it must be published. Simply click OK to close this, and then in the Save File dialog, call this **BWOCustomDIP.xsn**. When the save is complete, click on the File menu and then select Publish… This will start the InfoPath publishing wizard. Since InfoPath knows you got here via a Content Type, it automatically selects how you should publish:



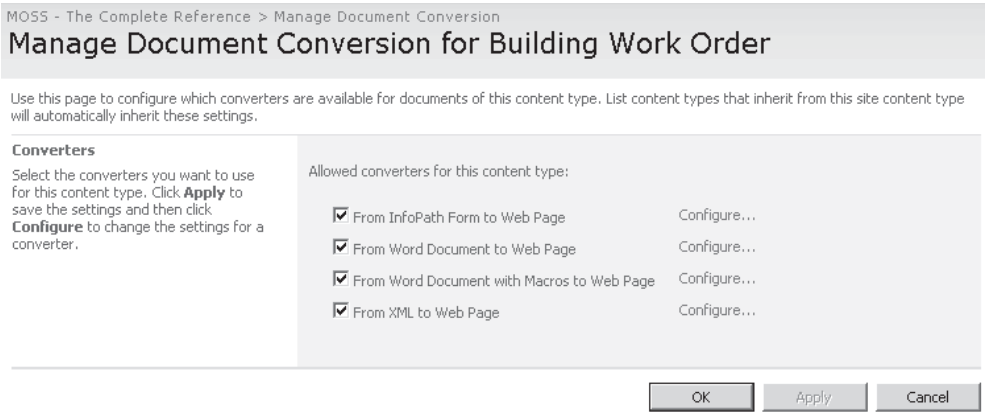
As a Document Information Panel is already selected, click Next to display the confirmation page. Here you can see the content type you are publishing to:



Notice that it also warns you that when you do publish, it will overwrite the default DIP (in this case, the default one inherited from the Document Content Type). Click the Publish

button; you'll see the file briefly uploaded. Then, click the Close button to complete the process and then close InfoPath. Return to the browser that launched InfoPath, and you'll see the link Go Back to the Document Information Panel Settings Page. Click this to return to the Document Information Panel Settings page where you'll see that the uploaded file is now defined. Click OK to return to the Site Content Type definition page.

- **Information Management Policy Settings** Information Management Policy Settings are basically audit controls set on documents and settings such as expiration policies and similar. These can be assigned at the Application, Site, and List level, and as you can see here, for a content type as well. You have the option here of using no policy (the default), creating a new one, or using one that has already been defined. Since this process is identical to IMP settings already covered, I won't go into detail here (refer to Chapter 11 for an explanation of each of these options).
- **Manage Document Conversion for This Content Type** This setting enables/disables the kind of document conversions that can be used with this content type. Providing document conversions are enabled (at the Application level), this will show a list of the available types and allow you to enable/disable them. You can also custom configure one if needed (you may have additional ones installed):

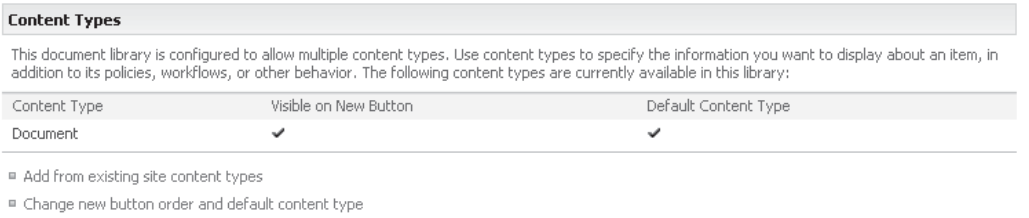


Exercise: Use the Content Type

Now that the content type has been defined, you are ready to use it in a library. So, after choosing a library, you will modify it to change the Advanced settings to allow content types. Do the following:

1. Navigate to the site page in the SharePoint Site where you want to create a library (remember, site content types are available across all sites).
2. From the Site Action menu, click Create.
3. On the Create page, click Document Library.
4. Name the library **Building Documents** and leave the rest of the settings as is (including the template). Click OK to create it.
5. From the list, click Settings on the toolbar and select Document Library Settings.
6. When the Settings page opens, click the Advanced Settings link under the General Settings group.

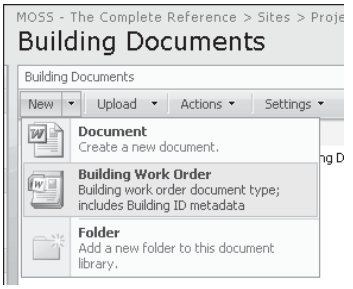
7. In the Content Types section, click the Yes radio button to allow management of content types, and then click OK to save the change.
8. When you return to the Setting page, you'll notice that there is a new group shown:



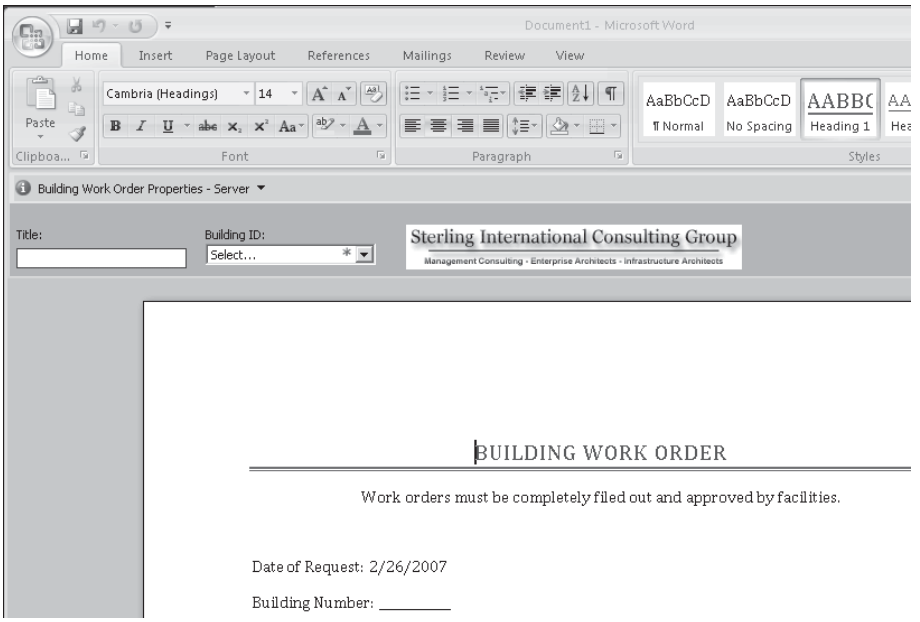
9. As you can see, the default content type already assigned to this library is the Document Content Type. Click the Add From Existing Content Types link, and then, from the Add Content Types page, change the Group to Building Related Documents. Now, click Building Work Order in the Available Site Content Types window, click the Add button, and then click OK. When you return to the Settings page, you'll see that the Building Work Order Content Type is now shown.

NOTE *Because you have two types: the default document and the Building Work Order, both will now appear in the New menu in the Document Library view. The other option here allows you to change the order of which comes first.*

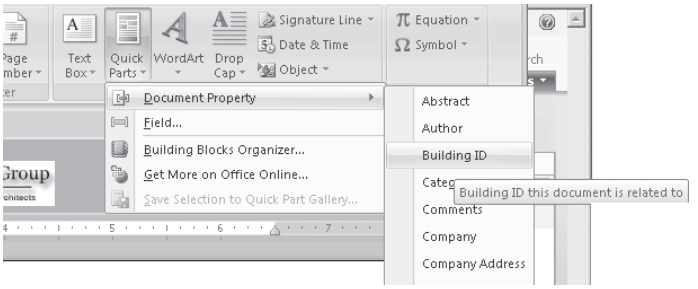
10. Click the Breadcrumb link to return to the Document Library view.
11. Click the New menu and you will now see option in the drop-down menu (notice the description):



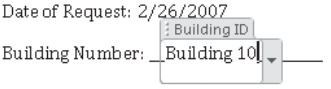
12. If you have Microsoft Word installed, you can click Building Work Order to create a new document. Click OK if you get a warning about files; when Word opens, it will take a second or two to load the Document Information Panel. But when you're done, you'll see two things: the custom DIP created (the default if you didn't create one) and that the document is opened with the predefined template for this type:



13. Note that you can now include metadata directly in the document. Place the cursor anywhere in the document that you want to put the content (in this case, Building ID), and then select Insert | Quick Parts | Properties, and you'll see that the metadata columns are included:



14. Adding this to the document automatically fills in the data (if you change the metadata, it is changed here if the document is opened):



Dublin Core Columns

Before closing out on content types, there's a document management item content Type worth mentioning called Dublin Core Columns (named for a technical workshop in Dublin, Ohio, in the U.S.). Dublin Core Columns are an industry standard metadata definition for documents (http://dublincore.org/). Much like the group standards that evolved with Electronic Data Interchange, the Dublin Core Metadata Initiative (DCMI) set out to define a standard for documents that apply no matter what the document is. The core columns represent fifteen different metadata items (Columns in SharePoint terms); SharePoint includes its own Title field (to remain consistent). Documentation for these columns is available at the site or in the SDK.

Site Content Types and InfoPath

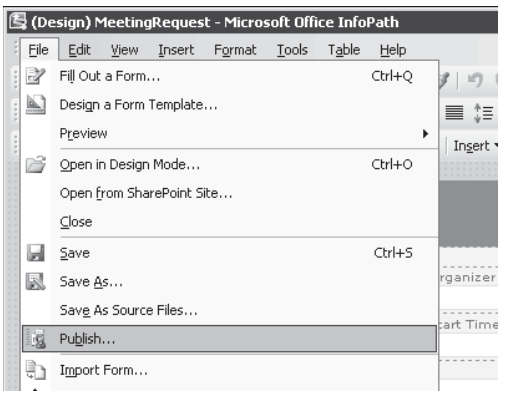
Like Site Content Types, profile documents and other entities in—not coincidentally—InfoPath can be used to create Site Content Types for forms. This gives InfoPath (and Forms Libraries) the same capability as a Document Library. Just as multiple Document Content Types can be used in a single library to provide multiple document types, InfoPath Content Types enable Forms Libraries to provide any number of Form Templates.

InfoPath is used throughout SharePoint in workflows, integration with Office, and so on. Forms themselves, though, have a special place within SharePoint, because they allow for interactive user input as simple forms; can be used for custom reporting, statistics gathering, and setting business metrics; and can be used to supply data to back-end systems through the BDC. InfoPath Content Types provide a unique way in which to deal with forms, since a single form type can be made available to any library in any site.

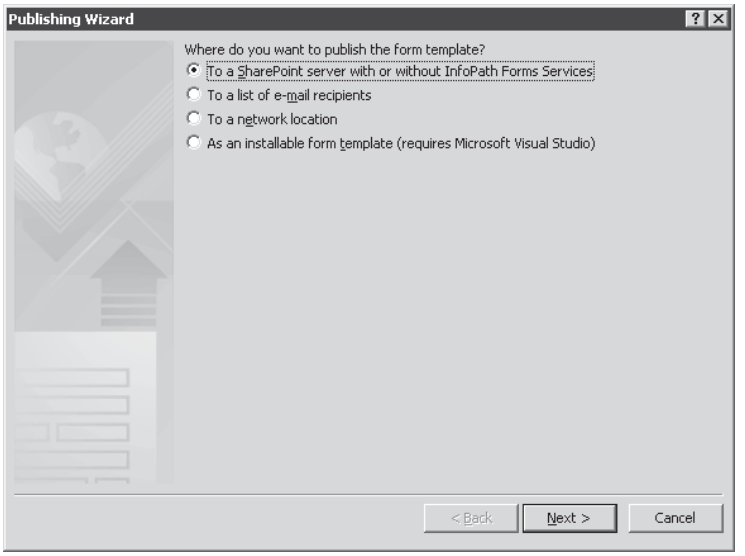
Creating InfoPath Site Content Types

InfoPath, like any other Office application, can save forms directly to a SharePoint Form Library. Any other option, however, won't save the form, but will rather convert it to a Content Type in SharePoint, defining all the columns required, and so on. The easiest way to explain this is by example. To keep this simple, I'm using the Meeting Agenda Form (out of the box with InfoPath). To save a form as a Site Content Type, do the following:

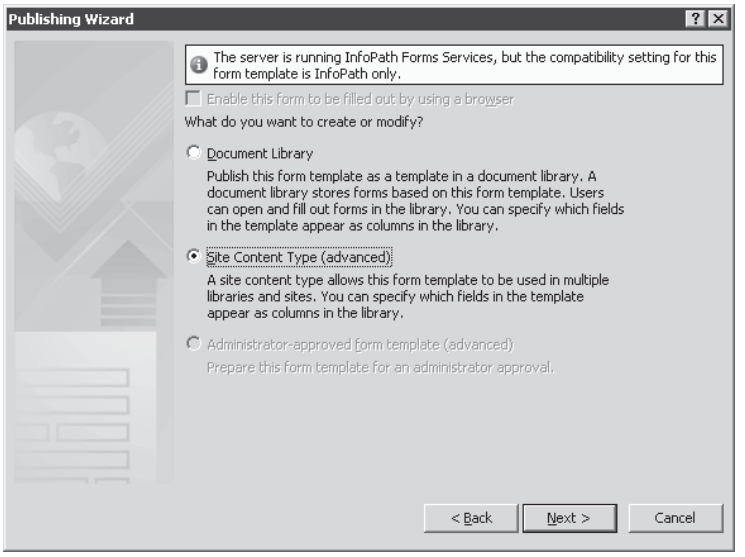
1. Open up InfoPath and either select the default form or create a new one (you must have fields defined before it can be saved as a Content Type).
2. Save the form to the local disk (this is required before a form can be published to SharePoint).
3. From the File menu, select Publish.
4. The first Publishing Wizard dialog box enables setting the type of destination; make sure that To a



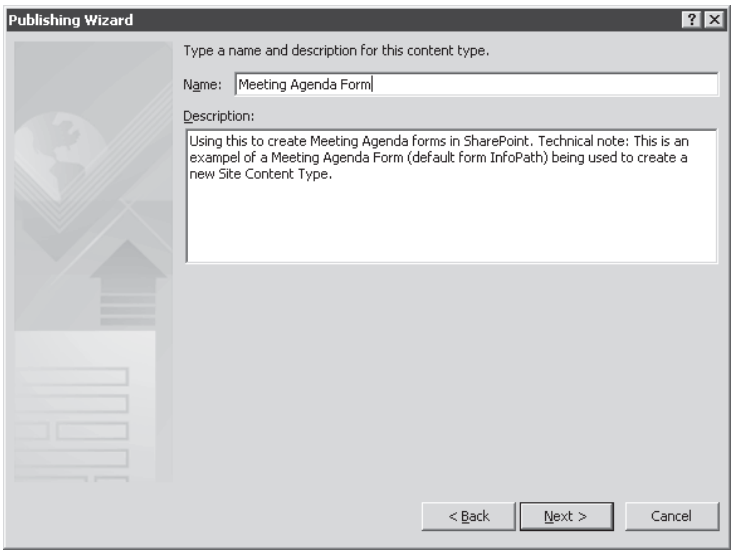
SharePoint Server With or Without InfoPath Forms Services is selected and then click Next:



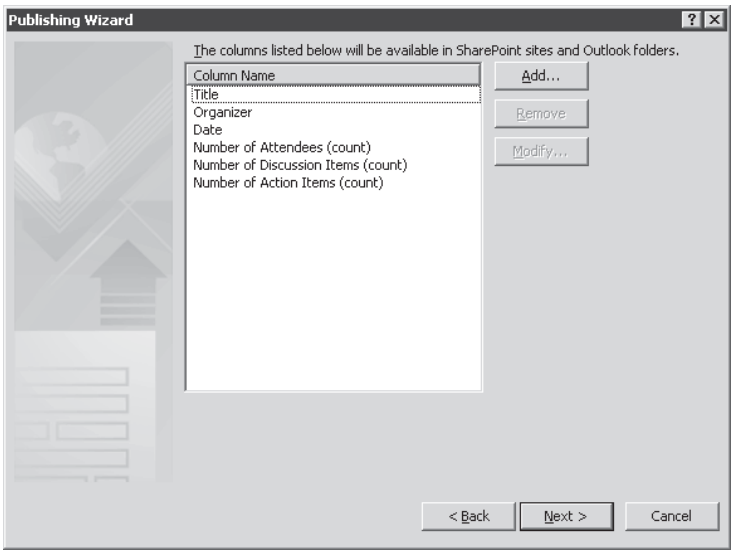
5. On the next dialog box for the Wizard, specify the SharePoint Site URL, and then click Next.
6. On the next dialog after this, the publishing options for Library or Content Type are shown. By default, this will be set to a Document Library (the most common use); select Site Content Type as shown:



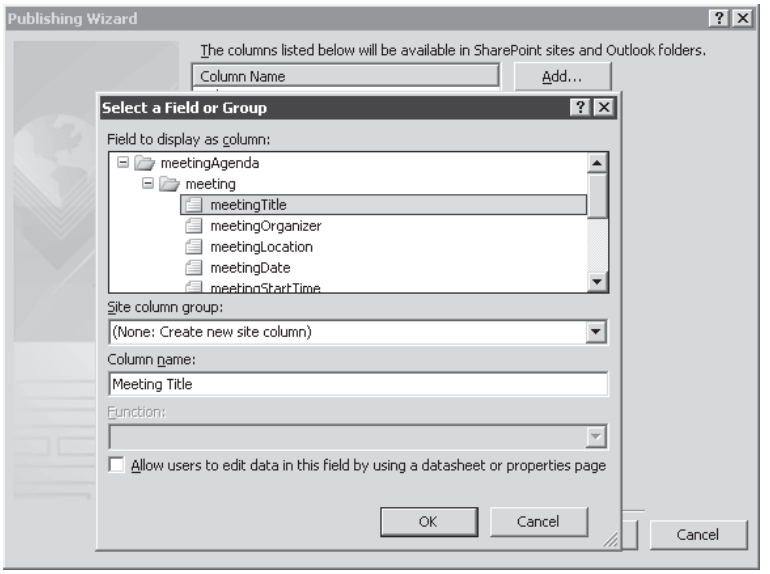
7. On the next dialog box, leave the default setting for New Content Type and click Next to continue.
8. On the next dialog box, enter the title for the new Site Content Type and a description as shown. Click Next.



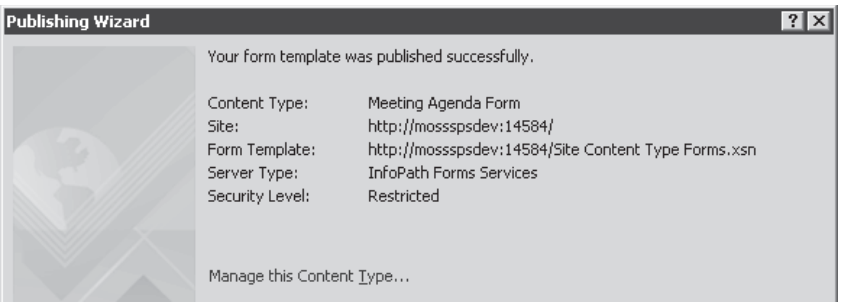
9. The next dialog box specifies the location for storing the file in a Forms Library on the SharePoint Site (I usually set up a specific library for this in the base site). This becomes the storage location for the form that the content type will read from. Click browse and find the library desired, or enter a URL and click Next. The next dialog allows you to pick (and if needed) modify the fields from the Form that will be used to create the content type:



10. Clicking Add will provide a list of all of the fields that are available on the form. For example, to include the Meeting Title, click Add and then select it from the list.

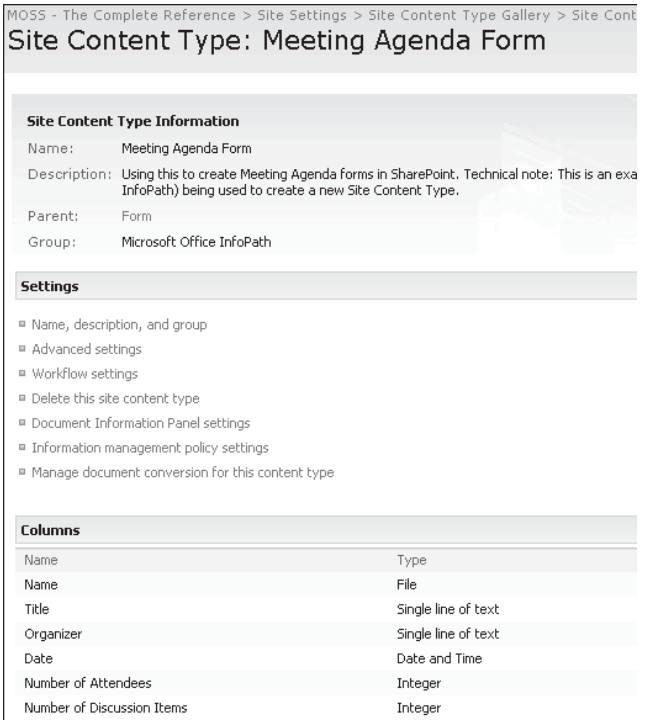


11. Notice that in the Site Column, you can choose to create a new one (the default), or you can choose an existing Site Column. This links the column name to the actual Site Column. Also, notice that there is an option at the bottom of this window, Allow Users to Edit Data in This Field Using a Datasheet or Properties Page; checking this enables end users to edit the contents of this field in the content type (there are uses for this, but if enabled, form logic on the field won't work if the field is changed in SharePoint).
12. Click OK to add the title and then click Next to continue. On the Verify page, click the Publish button to complete creating the type. If all goes well, you will see a few messages (it will attempt to delete any existing site content type and create a new one), and at the end, you'll see the message that it was created successfully.



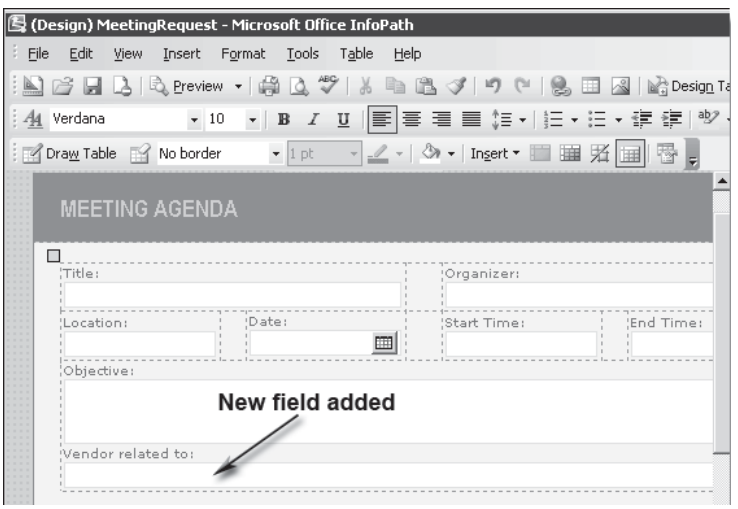
If there is a problem creating the type, you will receive a message like "Getting the library or list items failed." This usually indicates that the library is the wrong kind or that you have insufficient rights to create items here. The fix is to check the rights, and if they are OK, try creating a new Form Library.

13. To manage the content type, click Manage This Content Type… link (while the site opens, you can click the Close button to complete the Publishing Wizard). As you can see, all of the selected fields and so on have automatically been created:

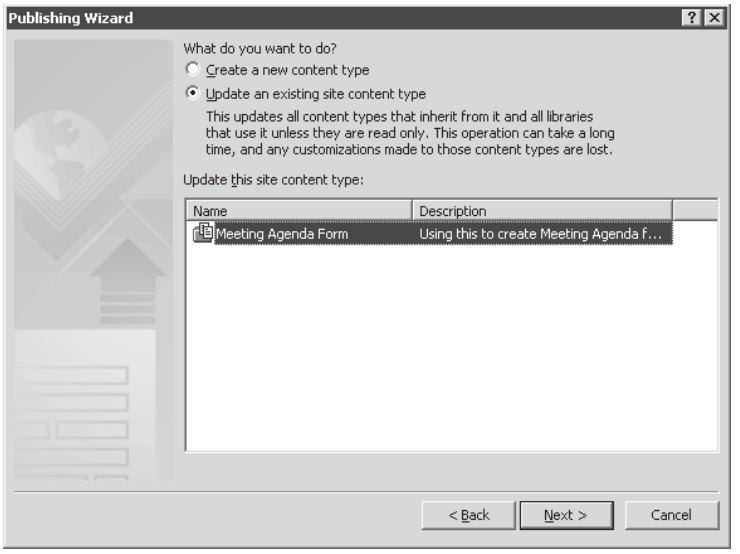


Updating InfoPath Site Content Types

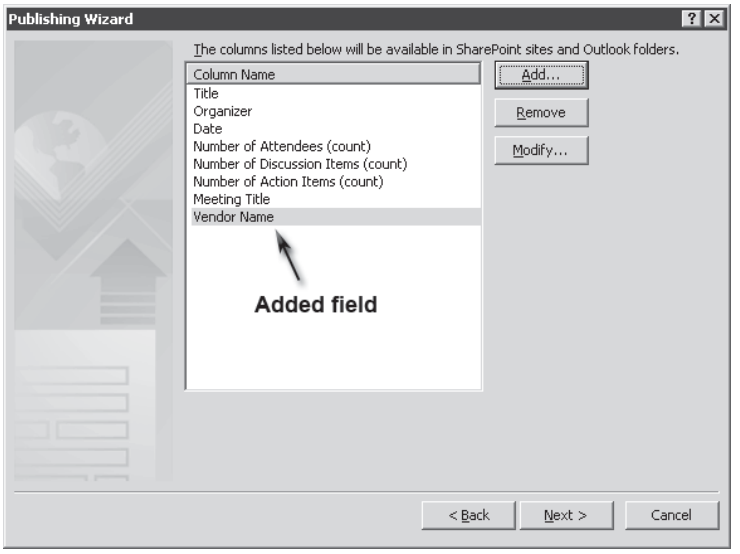
Updating a Form Site Content Type is done directly in InfoPath, much the same as when it's created. For example, to add an additional field, this would be added to the original form:



Once the change is made and the form saved, select File | Publish to publish the form to SharePoint. All of the same options are provided by the wizard, but it detects whether it is an update:



When the field list dialog is shown, the field can be added:



Click Next and then Publish to complete the update. If you view the Site Content Type again, you'll see the additional field.

Enterprise Content Management in SharePoint

In the past few chapters, you've seen what the application side of SharePoint can do and how it can be used to create pages, use web parts, create lists, and so on, including content creation using the Content Editor Web Part and the XML Web Part. Well, welcome to the other side of SharePoint: Enterprise Content Management. ECM has been a buzzword for years, but unfortunately, what it means to folks varies greatly. Wikipedia says:

> "Enterprise content management (ECM) is a widely-recognized IT-industry term for software technology that enables organizations to create/capture, manage/secure, store/retain/destroy, publish/distribute, search, personalize, and present/view/print any digital content (e.g., pictures/images, text, reports, video, audio, transactional data, catalog, code)."

Now that's a mouthful, quite accurate, but still a mouthful. The shorter version is

- You have images that you use, such as logos, pictures, icons, buttons, arrows, and so on. These are digital assets.
- You have text content in the form of news, articles, instructions, and training materials. These are content assets.
- You have documents in the form of Word Documents, Excel Spread Sheets, Visio, and PDF files. These are content reference assets.

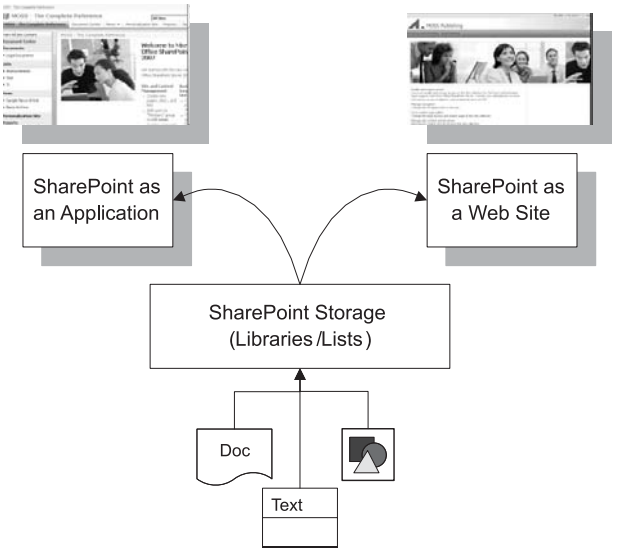
So what do you do with those assets? You can create new documents, create letterhead, create forms, print business cards, print reports, and so on. You also use these assets to create websites for the company, intranet sites for internal use, and perhaps build an extranet site for your vendors or customers. All of this, of course, requires some kind of Content Management (or should anyway).

Management of these assets is what ECM truly is; its purpose is to:

- Create a central location to store all of the stuff
- Organize it so you can find it
- Version it so that you can locate what is most recent
- Make the assets available to use from where it is stored (the repository)

So does SharePoint provide true ECM? In fact, it does, and it does it in two ways. First, SharePoint is at its core a repository for content. Document libraries, Picture libraries, and so on can be used to store documents, images, text, and so on, and lists can be used to hold pretty much anything. All of these stores can be versioned and can provide check in and check out and so on. This satisfies the storage side of ECM; a central place to store everything in a secure, searchable, and scalable place. On the other hand, SharePoint gives you a way to present content, images, documents, and so on, to users, which satisfies the presentation side of ECM.

It does this so well that all functionality of the former Microsoft Content Management Server is now merged into SharePoint, and MCMS has itself been dropped. Microsoft took the features of SharePoint as a repository and an application and added the MCMS features that SharePoint lacked: multiple authentication methods, easily edited content pages, page templates, easy branding, and styling, and so on. A view of SharePoint's content strategy looks like the following:



ECM Components in SharePoint

Enterprise Content Management in SharePoint is made up of more than pages and content, as covered previously. It's also made up of other tools that fully extend it into enterprise class. The components in question include:

- **The SharePoint Repository** The WSS core repository provides all the tools necessary to manage and control any kind of digital assets, documents, and content resources.
- **Master and Layout and Content Web Pages** The entire User Interface can be easily modified and branded; customized layouts allow control of the site look.
- **Content Web Pages** Content pages can be created on the fly, both through the User Interface and via client authoring, and can include complete version control, workflow management, and publishing.
- **InfoPath Forms** Forms provide interactive content and are capable of collecting user input and initiating workflow.
- **Document Management** Documents can be version controlled using minor and major versioning, profiled (using Content Types), and can provide a role-based security model to control access (quite an improvement over Version 2).
- **Records Management** An enterprise feature (MOSS only), SharePoint provides a permanent Record Management tool with individual authorship and update control.
- **Search** Office Server Search (MOSS only) provides extensive search capabilities on content, documents, websites, and even file systems.

Master Pages and Branding

The release of .NET 2.0 brought the development community a new concept in aspx pages called *Master Pages*. Master Pages are background pages that can be inherited by other pages that add on functionality. Changes to the Master Page automatically change the children, but the child pages themselves do not change. Both WSS and MOSS use Master Pages to create the basic frame of the User Interface and to control the basic page layout, including heading, side navigation, main page display, and so on.

SharePoint Master Pages are really nothing more nothing more than aspx pages with links to style sheets and references to a few SharePoint components. Pages within sites inherit this page then add their own functionality, and with this inheritance, they also get the styles set by the Master Page. There's also inheritance in Master Pages and sites; sub sites can either inherit the Master Page of their parent site or use a completely different page. This means that a single page (the master) sets the style for the entire site, and Page Layouts (the actual content display) are only used for layout of content and web parts.

In terms of branding, Master Pages provide complete flexibility for what a site looks like, from the heading down to the styles on the pages (the Chrome, as Microsoft calls it). By modifying or creating a new Master Page, a SharePoint site can be changed to taste. Even the controls (like the Navigation control) have style settings to ensure branding is consistent. Since sites can have their own page, you can easily customize each one. For example, an extranet site for customers can be custom tailored just for them.

In WSS, there is one Master Page for the site, so when pages are created (Basic and Web Part pages) they are rendered inside the main page making branding a snap. Simply change the Master Page and you're done. With Page Layouts and multiple Master Pages, MOSS provides even more flexibility. Master pages are typically made up of the following:

- **HTML** HTML defines the layout of the page.
- **Navigation Component** This renders the navigation (using the SharePoint data source).
- **Images** This includes logos and backgrounds.
- **Search Box Control** This renders the Search Text box and buttons.
- **Login Control** This detects if the user is logged in or not.
- **Editing Controls** This controls the page editing modes.
- **CSS References** This links to styles.

The cool thing is everything mentioned previously is out-of-the-box stuff, and you are certainly not limited in your choices. Any web control you can build can be included in the master page—whether it displays (renders) anything or not—so just think of the possibilities! Another great advantage to the Master Page is having access to the SharePoint object model (though aspx code) and through web components. This means the entire page, style, and content can be altered based on lots of things: the site name, the current date, and even the current user.

A word of caution on adding additional functionality to the Master Page; don't forget that this is run for every single page in the site. If your component calls the database, 1000 hits on a page in the site equals 1000 hits on the database. I suggest you try to stick to plain HTML or JavaScript output and keep server-side work to a minimum. There are some

advanced methods for caching that can help in this area, which I'll cover in the advanced discussion later on in the chapter.

Both WSS and MOSS have a default Master Page (called default.master), which is used for the home page of the site. MOSS has several additional page types:

- Content Pages (customized with custom.master)
  - Welcome (landing) and Article Page Layouts
  - Untemplated (Free-form pages)
- Application Pages (customized with application.master)
  - Forms pages

There are also layout pages that are part of the application interface; these will adopt custom styles, but the pages themselves shouldn't be modified. Note that in addition to pages, sites can be assigned Alternate CSS Settings, which override the master (the SharePoint themes do this, for example).

Building a Master Page

To brand the site and change styles, you have the option of modifying the default master page that comes with SharePoint or of creating a completely custom one from scratch. It really depends on the extent of your design. Also, even if you are not really familiar with coding around assemblies in aspx code, you shouldn't have any difficulties. One of the great things about this is that testing a new master is easy, and if there's a problem, you can always change it back.

The best tool for working with Master Pages is SharePoint Designer; you can easily use Visual Studio, but Designer is faster and simply provides better SharePoint object and style support. I don't have the space here to cover both, so I'll stick to Designer for this example.

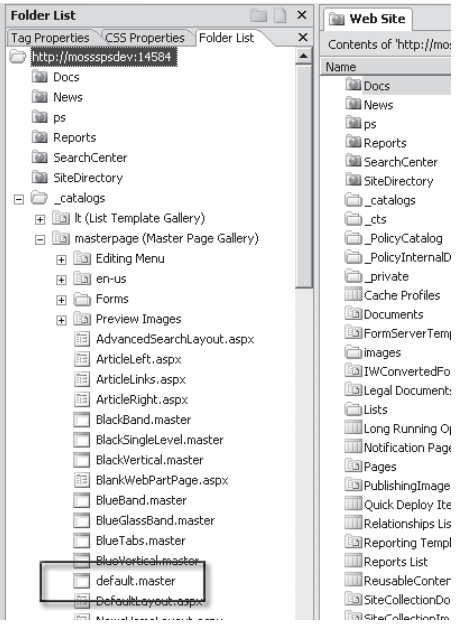
To begin with, you need to decide on a design. If it's reasonably close to what is provided in the default master, you should simply copy it and then modify it to taste, since the work of adding the appropriate controls is already done. You can, however, really start from scratch by using a base Master Page that has only the references needed to run SharePoint pages, and then add whatever controls you want (the minimal Master Page is available from Microsoft's MSDN site: http://msdn2.microsoft.com/en-us/library/aa660698.aspx).

Before getting into the code or Designer, I've found it really is best to test the layout of your design in simple HTML first. You can do this by hand (my preference) or use Visual Studio to lay it out in a web form. There are a few design considerations:

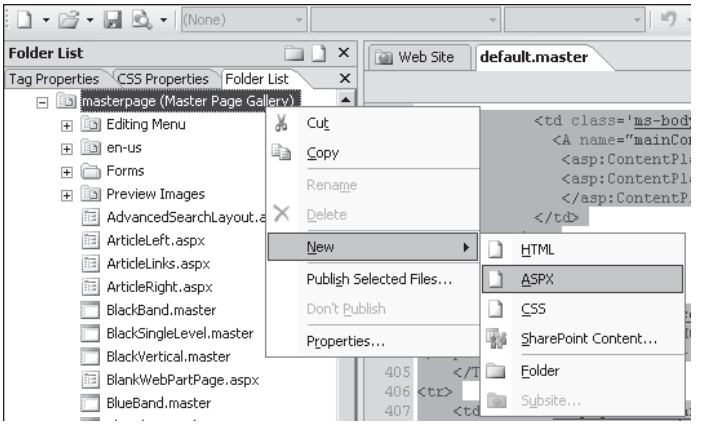
- The SharePoint page margins are set to 0; when working with HTML, you need to add those settings to the <BODY> tag to ensure that what you see here is what you'll see in SharePoint.
- SharePoint pages can stretch to fit the available window size (even my 1900×1200 screen), which means whatever you use in the background has to stretch with it. Using graphics is fine, but make sure they still look right when stretched.
- Always use SharePoint Styles in your code.
- Avoid using anything that could delay the page (RSS feed, for example, is a no-no).

To keep this example complete but simple, I'm going to modify a copy of the default Master Page to change the logo, move the navigation, and change the background color. The following example uses SharePoint Designer to create a new page (this example works the same in WSS or MOSS):

1. Open SharePoint Designer, and then select File | Open Site.
2. Enter in the URL to the site. In this example, I'm using http://mossspsdev:14548/ (the home page of a collaboration style portal).
3. When the site opens, you will see the list of folders, pages, and so on, appear in the Folder List (if you don't see the Folder List, select Task Panes | Folder List). Click to expand the Catalogs folder, and then again to expand the Master Pages folder and locate the default.master file as shown here.



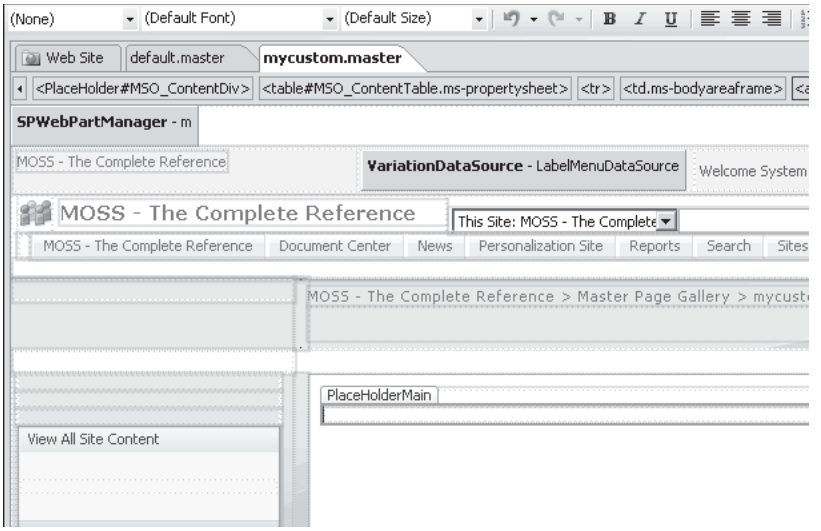
4. Double-click the file name to open it. You will be warned that the page is under source control and has not been checked out, but you won't be changing this file, so click No.
5. When the editor opens, make sure you are in the Code View; if not, select the tab at the bottom of the editor window.
6. In the code window, select all of the code so you can copy it (press CTRL-A and then press CTRL-C). Right-click on the Master Pages folder (in the Folder View) and select New | ASPX:



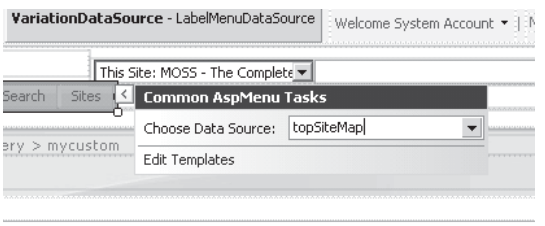
7. The new page will be created under the name Untitled_*n*.aspx, where *n* is some number. There will be a green checkmark next to it (indicating it is checked out).

In the Folder List, right-click and rename the file to **mycustom.master** (you'll see the icon change).

8. Double-click the new file to open it for editing.
9. In the code window, select all of the code put there by default (CTRL-A) and then delete it.
10. Paste in the code from the default.master.
11. Save the file. You can do this via File | Save, or you can click the Save (disk) icon.
12. Change the Editor Mode to Design by clicking the tab at the bottom of the window (note it will be necessary to switch back and forth from Design Mode to Code Mode; when you see this referenced, this means the tab options at the bottom of the editor window).
13. The page should look like the site shown next, with all of the markup and zone lines shown:



14. Now just to show you one of the common controls (Navigation), you can see that this is simply an AspMenu object. The data source in this case is a default one added by SharePoint to load the site navigation:



15. First, you'll want to change the image shown on the page. Click on the icon (the Team icon on the top-left of the page).

16. Click the Code tab in the editor to switch back to Code Mode. The code for the image will be highlighted:

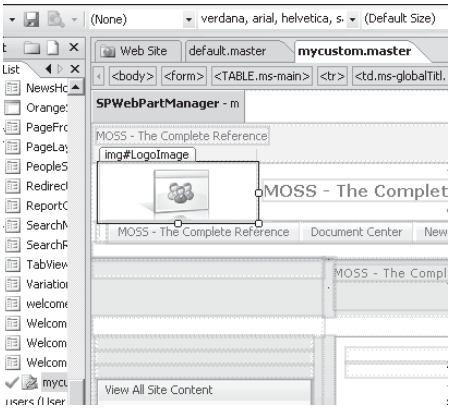
```
<SharePoint:SiteLogoImage id="onetidHeadbnnr0"
LogoImageUrl="/_layouts/images/titlegraphic.gif" runat="server"/>
```

17. Delete this and add this line in its place:

```
<img alt="Logo" id="LogoImage" src="/_layouts/images/PGRPICON.PNG">
```

NOTE *The path here, /_layouts/images/, points to the SharePoint Hive Images folder.*

18. Save the page and switch back to Design Mode; the image change should appear:

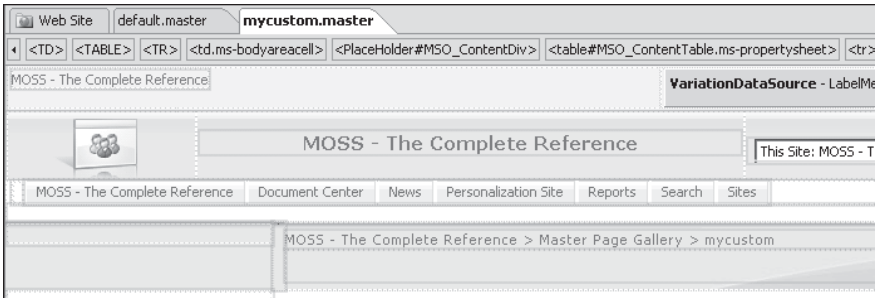


19. Now you're going to change the background for the title portion of the page. Click the site title, and then in the top bar, click the <Table> tab as shown:



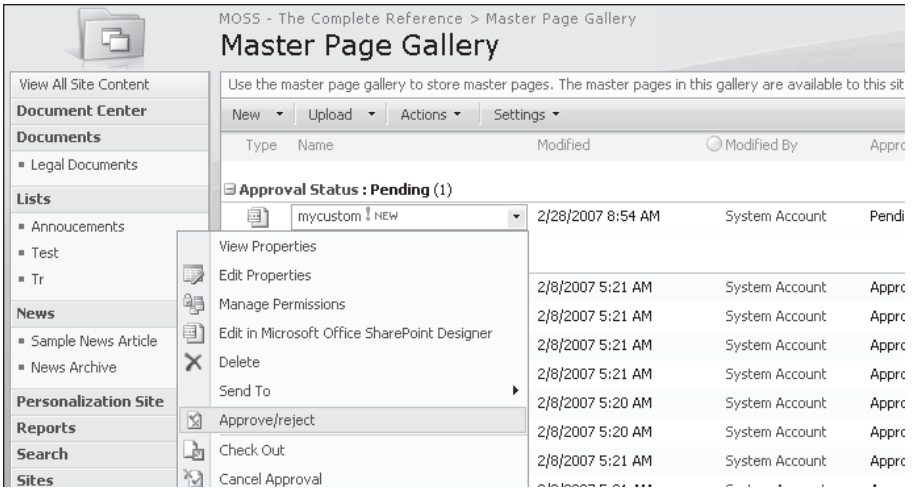
20. Switch back to Code Mode (the table code will be selected); change the <td> class (just above the <Table> from ms-globalTitleArea to ms-WPAddButton.

21. Save the page and then switch back to Design Mode. The title should now be different (the background has changed, and the title is now centered):



So you're done! You've changed the background and page image, but now you'll need to finish:

1. Right-click on the new page name in the Folder List window and then select Check-In. When prompted, select Publish a Major Version and click OK (comments are optional). SharePoint Designer will now pop up a prompt stating that, "This document requires content approval. Do you want to view or modify its approval status?" This indicates that the page must be approved before it can be used. Click the Yes button, which will open up the Master and Layout Pages page (from Site Settings).
2. Locate your page in the list. The status should be set to Pending.
3. Mouse over the name to reveal the drop-down menu and click the Approve/Reject link:

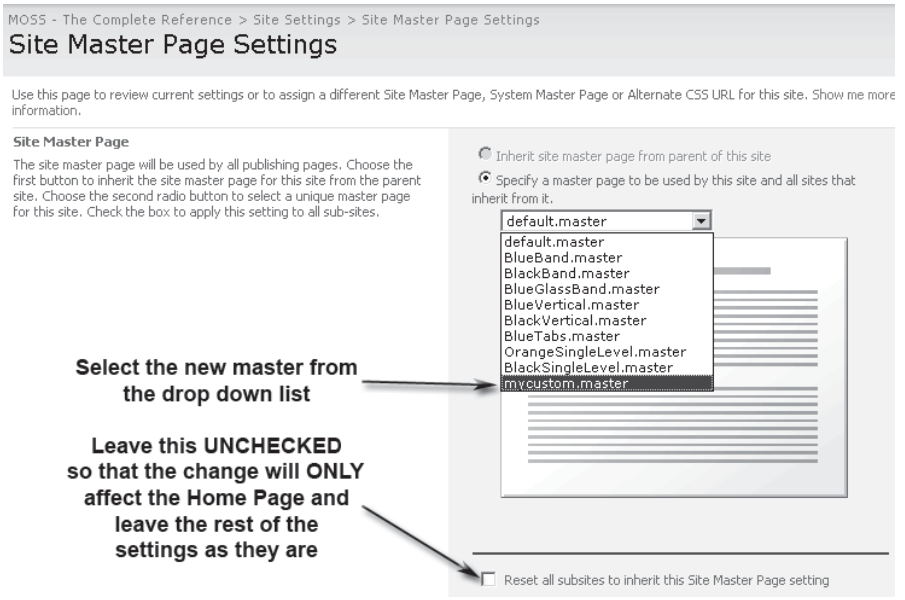


4. On the Approval page, click the Approved radio button and then click OK.

Now, to use the page, do the following:

1. In WSS, select Site Actions | Site Settings; in MOSS, select Site Actions | Site Settings | Manage All Site Settings.
2. On the Site Settings page, under the Look and Feel group, click the Master Page link.

3. On the Site Master Page Settings page, choose the new master in the drop-down menu list; leave the rest of the settings as they are so that this change will *only* affect the home page:



4. Click OK to save your changes, which will return you to the Site Settings page. Then, click on the link in the Navigation bar to return to the home page. The changes should be obvious:



5. If desired, return to the Site Settings page, click the Master Page link, and then reset the master to the default. You can close SharePoint Designer as well, though it will be needed in the next section. Also note that you cannot delete a Master Page if it is in use; to remove a master, you must remove all pages that use it first.

While I didn't cover this here, be aware that all standard controls, navigation, search, and so on, are simply drag and drop to the page. Navigation, for example, is simply an

AspMenu control that uses a data source to figure out what to display (SharePoint includes the default data source for this navigation). You can certainly explore more of the code and try some additional changes. Controls you can add include both ASP.NET and SharePoint controls (in SharePoint Designer, open a page in Design Mode and then click Insert | ASP.NET | Controls or Insert | SharePoint Controls from the menu bar).

Page Layout and Publishing Pages

Previously, I described how Master Pages set the format and the style of the pages in the site with a few simple changes. Obviously though, Master Pages certainly don't create the site. The actual pages in the site are Publishing Pages that have been created by Layout Pages.

SharePoint Layout Pages are based on Site Content Types and are used to provide the page's template. Any number of Page Layouts can use the same Content Type. Users use Page Layouts to create Publishing Pages for both editing and display. Layouts can inherit any Master Page in the SharePoint site providing chrome (style) and with whatever functionality, such as navigation, it provides. Several default layouts are available, including Welcome Pages, Article Pages, and so on, and you can (and will) create your own. Like Master Pages, Layout Pages are simply aspx pages that combine HTML and aspx script, but may also include:

- Field controls that are bound to fields in the content type.
- Web parts that can be used by authors.
- Web part zones to add web parts.
- Custom Server controls.

Layout Pages can do most anything that an aspx page can do, including using server-side controls, accessing the SharePoint object model, and inheriting any Master Page. These are stored in the Master Page Gallery, along with Master Pages, and are selectable when creating a new page.

Field controls on Layout Pages are based on the columns defined in a Site Content Type. This binding is a fancy way of using the definitions stored in SharePoint to create and organize the fields on the page. Users use the layout to create a Publishing Page that is used for both entry (creating/editing) and for display of the actual content. However, they are not connected because a page that is just content stored in the Pages Library can be edited or viewed with more than one layout—even through the Pages Library itself. One thing to note is that a page can only use one Page Layout; the same page content cannot be used in different layouts (this is different from the way MCMS worked).

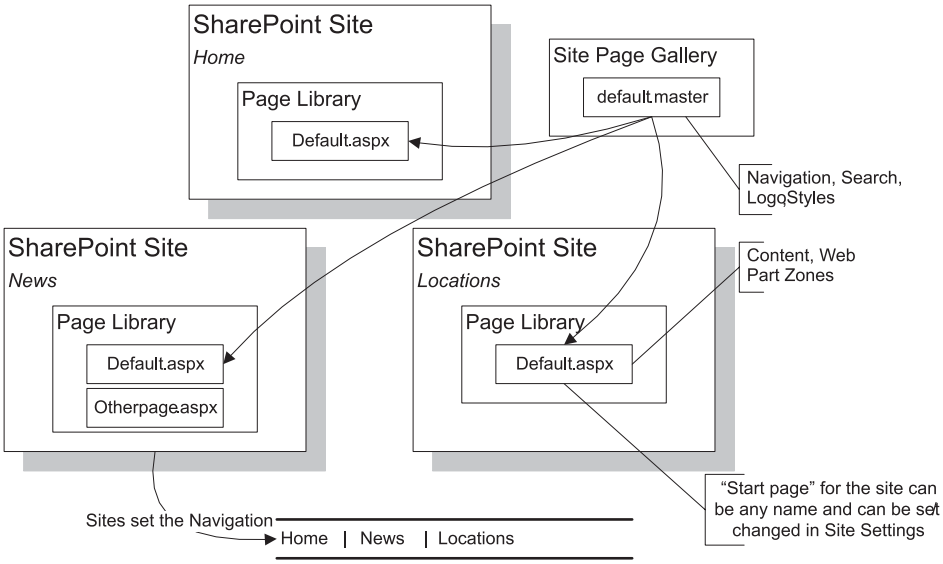
Publishing Pages (created from layouts) are a hybrid of the basic pages in that they provide editing and publishing controls on the page. They support Content Fields for content pages and/or web part zones. When created, Publishing Pages are stored in a special library called a Pages Library, which are set up to facilitate the publishing process. Because they are in a library, they get the full benefits from the platform as well:

- **Versioning** This includes major and minor versioning.
- **Check in/out** This includes content control.
- **Workflow** This includes approvals and sign-off processing.
- **2 Stage Undelete (Recycle Bin)** This includes the ability to restore deleted pages.

- **Content Types** This includes map content to pages.
- **Access Control** This includes role and rights management from SharePoint.
- **Link Fix Up** This includes link and URL repair.

Rendering of Publishing Pages for end users is not complicated; when a request is made to display it, the Master Page and Page Layout that were used to create it are put together with the actual content from the Pages Library (the item column data) and then rendered to the user. If a Page Layout is changed, the change is reflected in the published page.

In MOSS, the combination of sites (each with a page library) and pages stored in each actually builds the hierarchy of the website:



Anatomy of a Publishing Page

Publishing Pages themselves are only responsible for the content and controls within them; they have nothing to do with the actual navigation or other features in the site since this is handled by the Master Page. As mentioned previously, the Master Page sets the frame of the page and includes the standard controls (navigation and so on) as well as sets the style for the rest of the page. A Publishing Page inherits the master and with it, all of the controls, features, and styles, but it only handles the content portion of the page using content fields, web part zones, and HTML.

When rendering a single page, SharePoint determines the Global Unique Identifier (GUID) for the page; when it finds the definition (stored in the library), it begins to assemble the page by first loading all of the controls for the Master Page and then the Publishing Page. Next, all of the appropriate controls, such as fields, web parts, list views, and so on, are loaded from the database (using the GUIDs), and then the whole thing is rendered out into HTML and JavaScript to the browser. The breakdown of all of the pieces involved is as shown in Figure 14-1 (note that this shows the Pages Library used in MOSS; in WSS, there is only a single Master and Default Page).

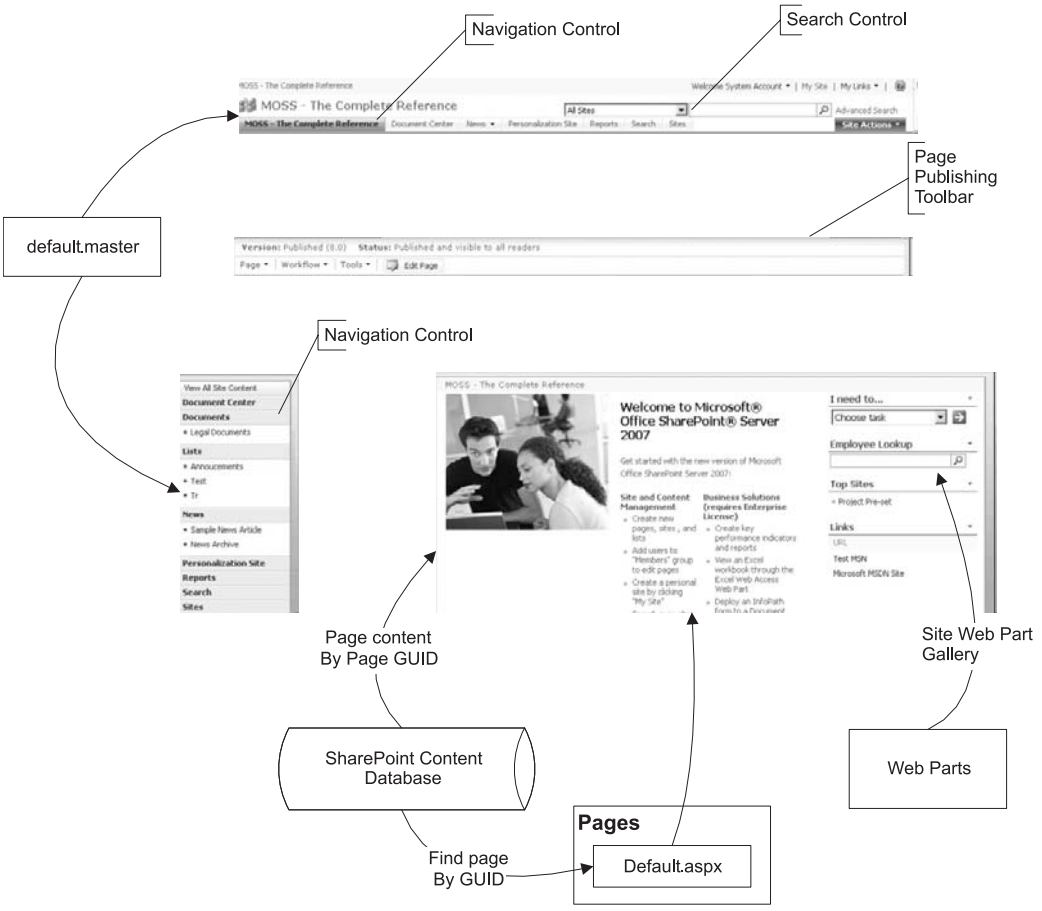


FIGURE 14-1 Anatomy of Publishing Page

While it is pretty obvious that WSS as a one-page site knows to open the Default.aspx page, you might be asking yourself how SharePoint knows. The answer is that in every MOSS site, under the Site Settings, within the Look and Feel group, there is a link to Welcome Page. This option allows you to pick out any page anywhere in the site as the Welcome (or Launch) Page. Initially, this is set to Default.aspx to match the one page added when the MOSS site is created.

Setting up a Page Layout Page

As covered previously, Page Layouts are the templates for creating content pages within SharePoint. The basis for this is a Site Content Type, which defines the content columns and data types. When the user enters data in these fields (in Editing Mode) and saves the page, the content is stored in the Pages Library list columns. Because there's no tie to the layout,

more than one layout can be used to view the content in the list. SharePoint includes a handful of Layout Starter Pages based on default Site Content Types created just for publishing (of course, you will also create your own):

- **Article Page** This is a "news" article with author, subject, body, and so on.
- **Redirect Page** This allows automatic redirection to another site.
- **Welcome Page** This is intended as a Launch page or anchor point for other sites.

Choosing a Content Type

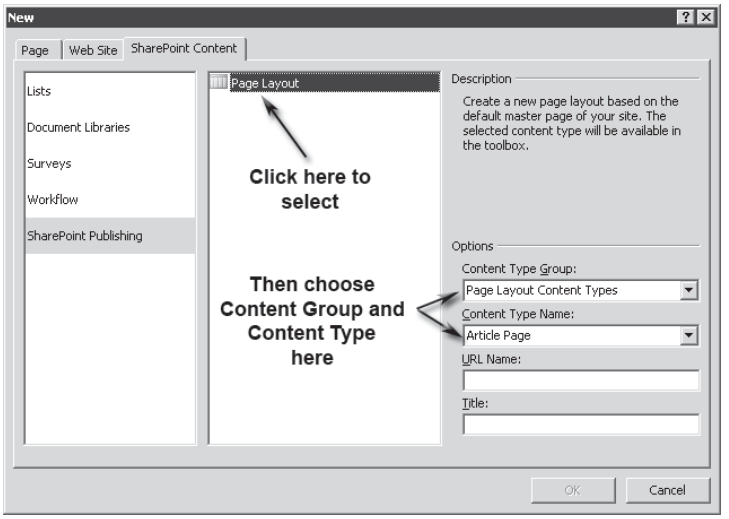
Page Layouts are based on Site Content Types, so you can select from any of the defaults or create your own. As an example, you'll use the Article Page Content Type; using this is the same as creating one yourself, but is quicker for this purpose. To view this content type, do the following:

1. From the home page, select Site Actions | Site Settings | Manage All Settings.
2. Under the Galleries group, click the Site Content Types link, which opens the Site Content Type Gallery page.
3. Scroll down to find the Page Layout Content Types group, and then, under that, find the Article Page Content Type. Once here, simply click the name to open it and view the columns that are included.

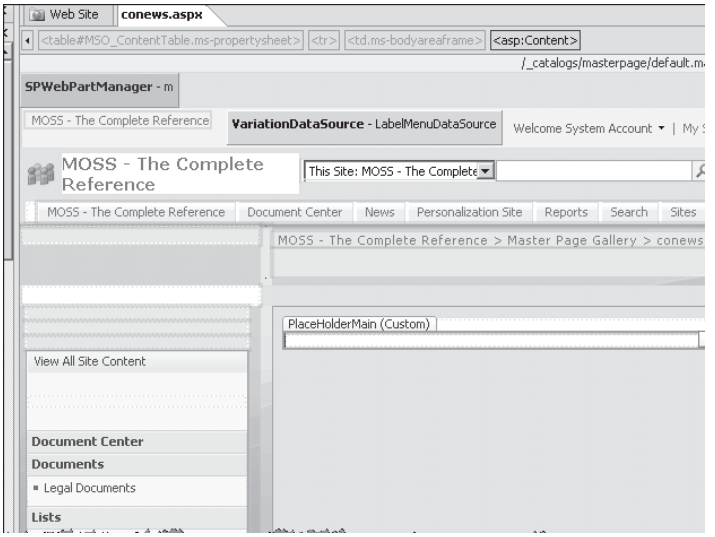
Building a Page Layout Page

To build a Page Layout Page, do the following:

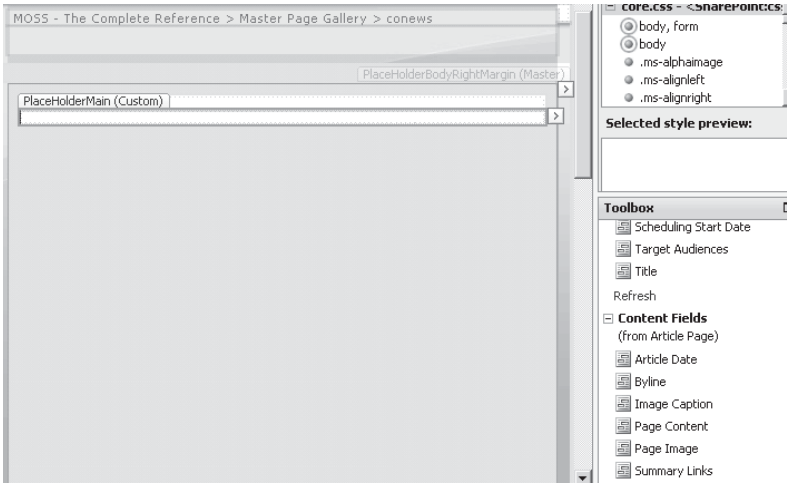
1. Open SharePoint Designer and then select File | Open Site. Enter the URL to the SharePoint site, and then click OK to open it.
2. If the Folder List is not shown, select View | Folder List in the menu tool bar to enable it.
3. Click to expand the Catalogs folder, and then again to expand the Master Pages folder.
4. Right-click on the Master Pages folder, and then select New | SharePoint Content.
5. When the dialog box appears, click on SharePoint Publishing; you'll see that the Page Layout is shown (this is the template for templates). Click to select it, and then you can select the Content Type Group and Content Type for this page:



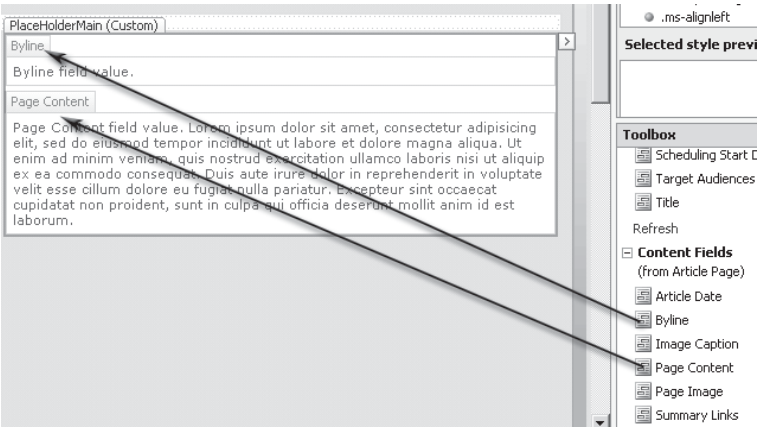
6. Enter the URL name (this becomes the page name) and the Title (this becomes the Title for the Layout Type and is what the users see). Then, click OK to create the page.
7. Depending on the mode you are in, you may see a Loading Master Page message before the page loads. If it loads in Code Mode, click Switch to Display Mode, and you should see something like this (notice that the Master Page for the site was automatically picked up):



Now comes the fun stuff! To design this page, open up the Toolbox panel (Task Panes | Toolbox) and scroll down until you find the Content Fields:



1. To begin designing your page, simply drag the fields over and drop them on the page:



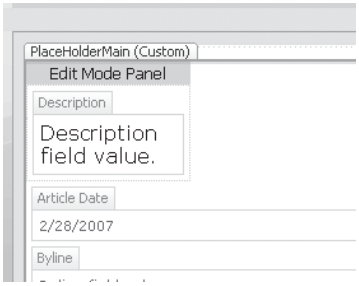
2. Once the fields are on the page surface, they can be moved around the page, and additional zones can be added, and so on.

One last (optional) feature is to provide additional fields (such as a description or comments) that are available to page editors, but that do not render when the page is displayed. This is handled by the Edit Mode Panel, which creates a zone where the fields placed in it are only visible when the page is being edited.

To do this:

1. Scroll the Toolbox to find the SharePoint Controls Group, and then find the EditModePanel.

2. Drag this to the top of the page section (you'll see it creates a boxed-in area).
3. Return to the Toolbox and find the Description field under the Page Fields group. Next, drag it into the box, and the result should look something like this:



At this point, the layout is complete:

1. Select File | Save to save the page.
2. In the Folder List, right-click on the file just created and then select Check-in. Next, select Publish Major Version and then click OK.
3. Click OK when SharePoint Designer asks if you want to check the status of the page. When the Site Master Gallery Page opens, scroll the list to find the new page. You should find it in the Pending status; mouse over the name to show the menu, and then select Approve/Reject. Click Approved and then OK.
4. At this point, the Page Layout is ready to use (test it following the next example).

Web Content Authoring

Web Content Authoring is simply the process that authors use when creating or editing pages in the site. Page Layouts, mapped from Site Content Types, are used to create pages; the content entered on these pages is then stored as items in the Pages library. Editing simply opens the Page Layout prepopulated with the list data. While the examples here have used fields, web authoring is certainly not limited; if the Page Layout provides web part zones authors can add web parts with the normal drag-and-drop functionality. However, there are a few web parts that are specifically used in Publishing Pages:

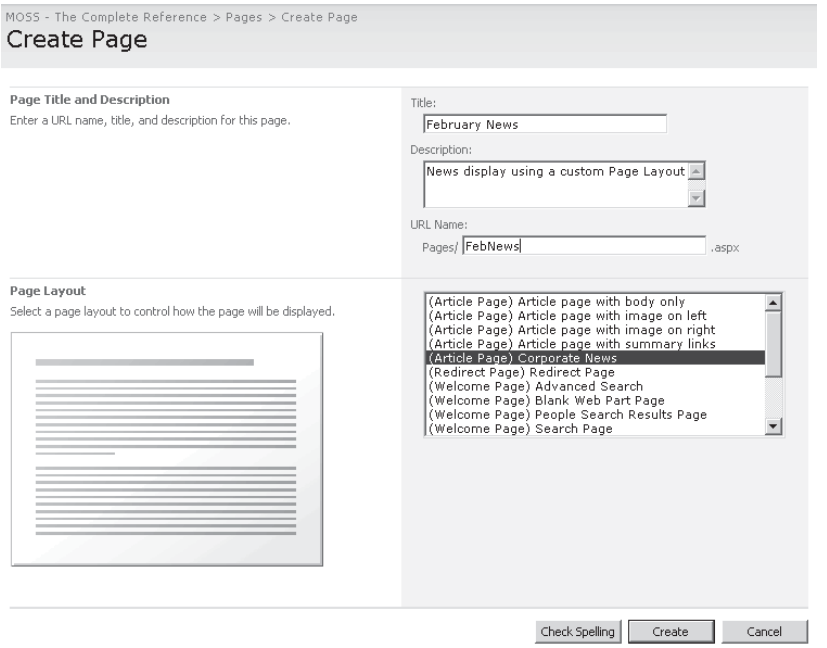
- **Summary Links** These quickly add author, style, and organize links on a page that uses XSL for styling.
- **Table of Contents** These are data-driven views of navigation.
- **Content Query** This is a data-driven view of content across a site.

When a new page is added, it automatically enters Editing Mode. This is evident by the fields showing in Entry Mode with controls (like date pickers) added by SharePoint. The Publishing toolbar shown on the page provides the publishing controls; from here, you can save drafts while working on the page and publish it to make it available on the site. At any time after a page has been published, simply return to the page and enter Edit Mode (via Site Actions | Edit Page) and the controls will become visible.

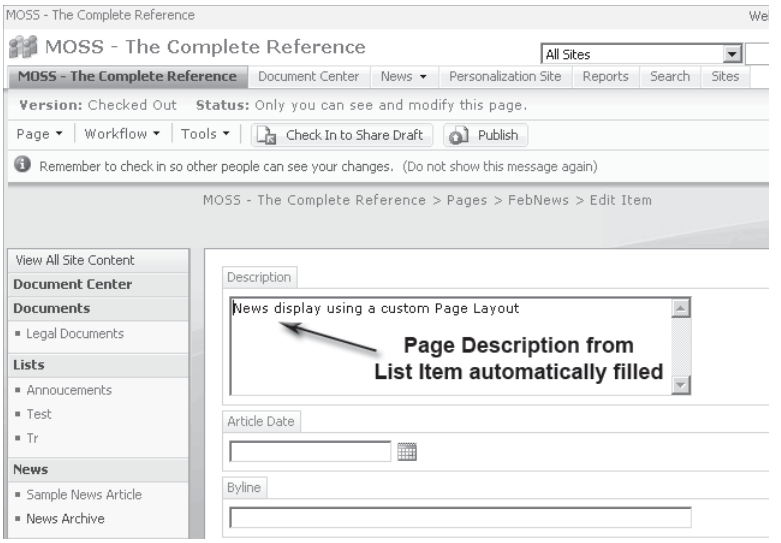
The process of creating and publishing a new page only takes a handful of steps:

1. Navigate to the home page of the site, click Site Actions, and then Create Page.
2. When the Create Page page opens, you'll see the list of templates that are available. For example, the new template that was created previously is now

available in the selection (categorized as an Article Page because that is the content type it uses):



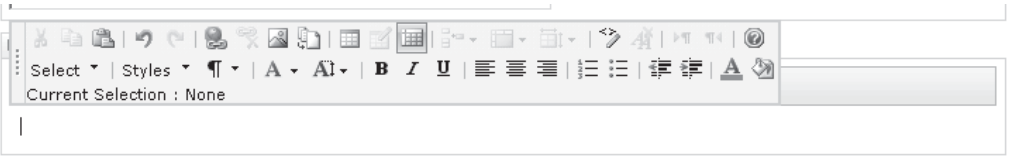
3. Enter a name, URL, select the template, and then click Create to create the new page. This will open the page in Edit Mode:



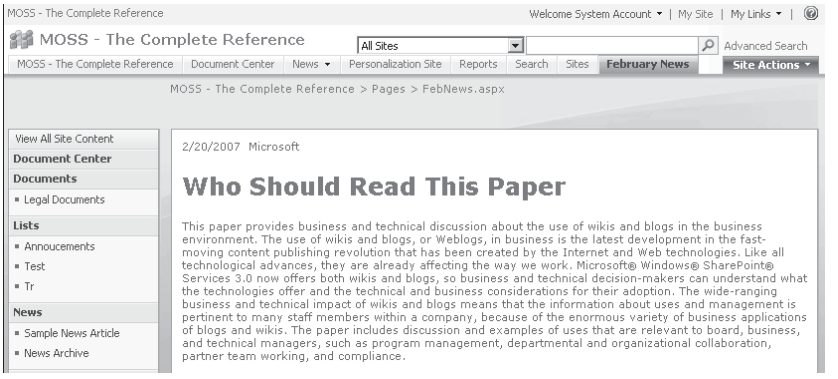
4. As you'll also notice, the controls added for the field editing are fully functional:



5. Also note that the Page Content control actually provides a full feature HTML editor:



6. One special button on the bar allows you to use and/or create reusable content in the site. A good example is a copyright notice; if set as reusable content, it can be updated yearly across the entire site. Once content has been entered, the page can be saved as a draft or published. The finished product might look something like this (notice that the Edit Mode Panel goes away):



Extending the Page Editing Toolbar

Toolbars in SharePoint can be modified to add additional commands for performing additional functions, auto-starting custom workflow, and so on. Adding a command simply requires a reference to the Assembly and the adding of a ConsoleNode element, which represents a command (the action, image to use, and so on). The XML file used to define this toolbar is called EditingMenu.xml located in the SharePoint Hive Template\Layouts\EditingMenu folder. The definition of the XML nodes and attributes used are as follows:

- **Console** This defines the toolbar itself.
- **Reference** This defines an assembly reference and type name.

- **ConsoleNode Sequence** This creates a top-level toolbar item and determines the order of appearance.
- **ConsoleNode Action** This defines a command for the toolbar menu drop-down list.
- **ConsoleNode Is Separator Image** This defines a break line in the toolbar menu drop-down list.
- **User Rights** This allows you to add a Rights Mask by the kind of roles that can see your particular option. EmptyMask, for example, indicates anyone (to set particular rights, you list them with a bar between each, such as BrowseDirectories | ManageLists).

Note that you can also modify most of the other toolbars that you use (such as the HTML toolbar and the Login/out menu and so on). You can locate these in C:\Program Files\Microsoft Office\Office12\1033\Toolbox. There is also a guide for creating a way to automate adding items to the toolbar. Refer to the Microsoft Office SharePoint Services 2007 Software Development Kit (SDK) article, "How to: Customize Page Editing Toolbar Components."

Smart Client Authoring

While page creation in the SharePoint UI is simple enough, I'm sure it is obvious that it can't accommodate all of your needs when it comes to adding content. In fact, the editors provided are not adequate for more than a few sentences. You can cut and paste between an application and the editor, but SharePoint has another way called Smart Client Authoring. Smart Client Authoring allows you to create content locally, and then create and publish it as a site page.

Smart Client Authoring is handled by way of a converter that converts the specific kind of document (like Word) into an HTML document that SharePoint can use to create a page. SharePoint runs this as a server-side service that even provides load balancing (see Chapter 6 for more on the services). A few converters are included out of the box, including docx (Word 2007), XML (with XSL), and InfoPath. Custom converters can be added easily; these are simply applications written to convert some unique file type into an HTML format that SharePoint's fields can use. Installed as a feature, these can be made available to users easily. For reference, conversion applications and the XML to identify them can be found in C:\Program Files\Microsoft Office Servers\12.0\TransformApps.

When a document is converted, it takes it from its native format, converts it to HTML, and then loads this information into SharePoint. The relationship between the two is maintained; changes to the original document will update the SharePoint data, and you will be warned if you try to modify the SharePoint item attached to a document. This avoids the pitfalls of having content on the site not match the actual document and vice versa. Conversions are handled by the Load Balancer and Launcher services within SharePoint. When a request to convert a document is made, this is issued to the Load Balancer, which then selects an available Launcher service (there can be any number of Launcher servers in the farm). As a timer job, this allows SharePoint to manage the conversion in the background, allowing the user to return to working (this is very handy when the conversion may take some time). The overview of the process is shown in Figure 14-2.

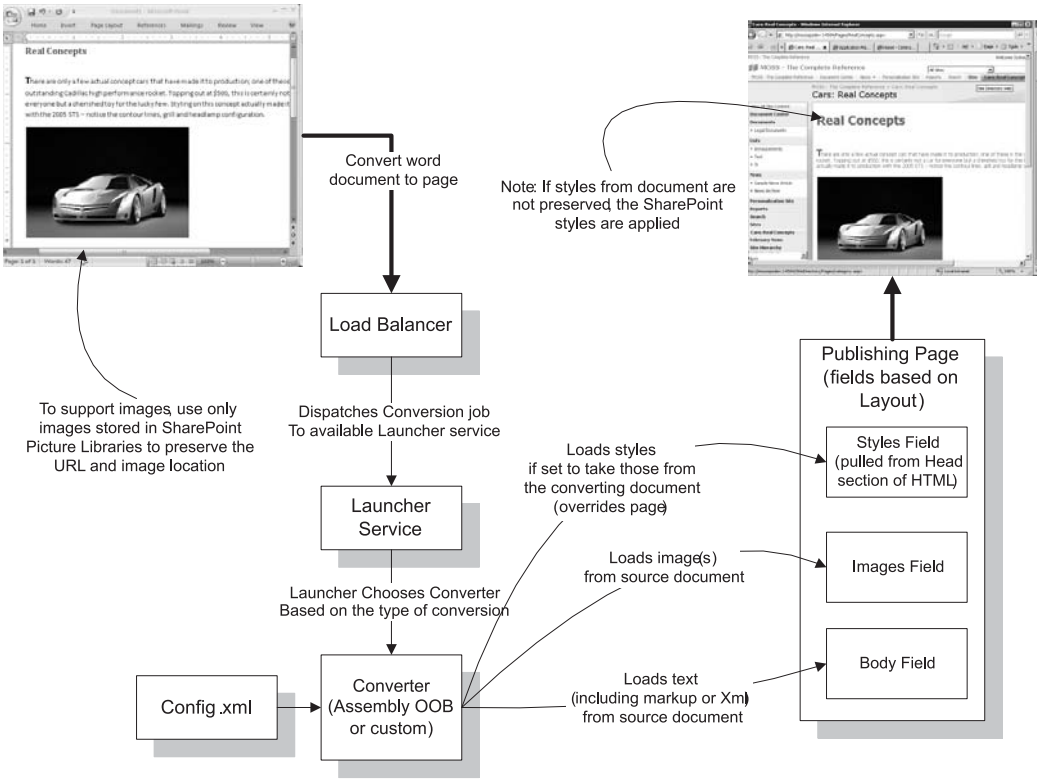


FIGURE 14-2 Overview of the document conversion process

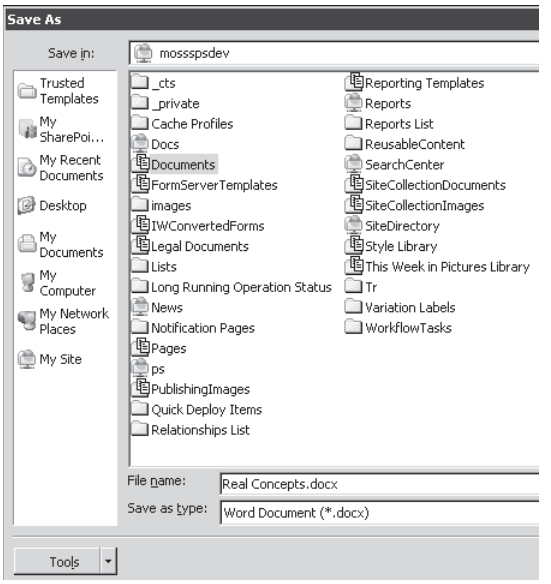
NOTE *Document conversions use a very low level (local) security account. Because of this, conversions cannot be run on a server running as a Domain Controller (it returns the generic error CE_OTHER if this is attempted).*

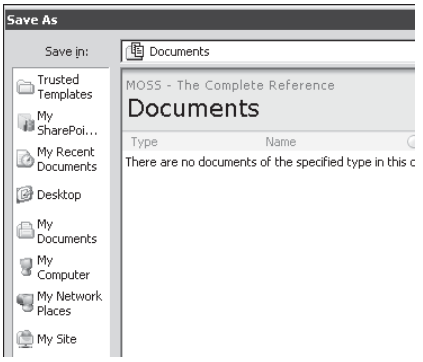
Converting a Word Document to a SharePoint Page

To perform a conversion, documents must be in a SharePoint library to begin with (you can't convert directly out of Word). The reason for this is that the original document will always be tied to the SharePoint page created. The document stored in the library becomes the source, and if changed, the difference is reflected in the pages created from it. As an example, here are the steps for creating a Word document and then producing a Publishing Page, using the Word converter:

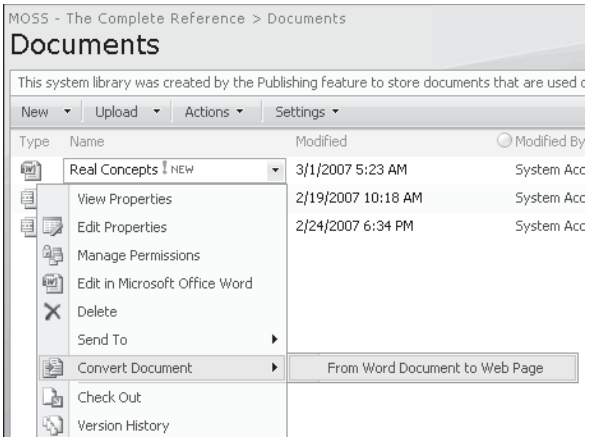
1. Create a document in Microsoft Office Word. If you decide to use images in the document, these must be images that are already loaded into a SharePoint Picture library, or you will receive a conversion error.

2. Save the document to a SharePoint library (if you've not done this before, from Word, select File | Save As… When the dialog appears, enter in the URL to the SharePoint website, and then select the library you want to store the document in).
3. Click the Open button to open the library.

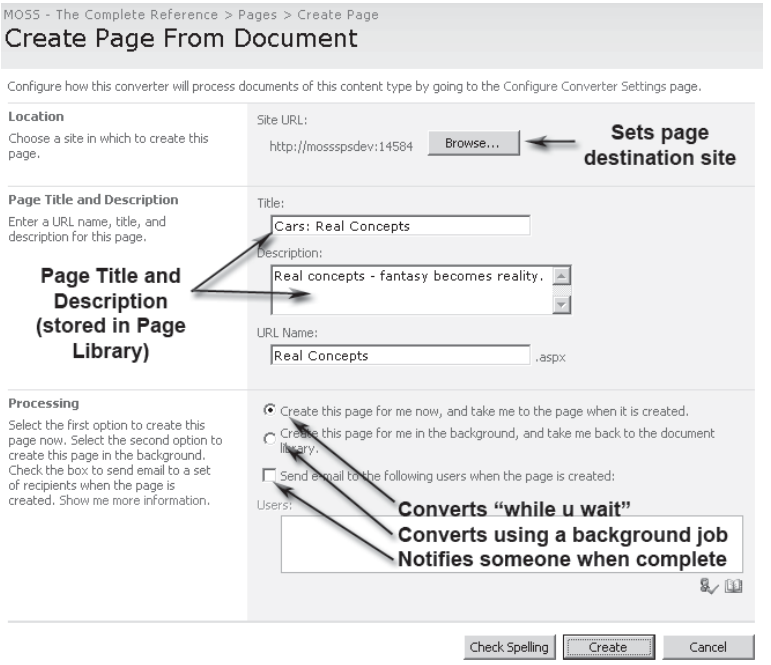




4. Click Save to save the document.
5. Navigate to the SharePoint site and then to the Document Library View (you must select a view that provides the Edit Item drop-down menu). Mouse over the name to expose the menu and select Convert Document | From Word Document to Web Page:



6. This opens the Create Page from Document page:

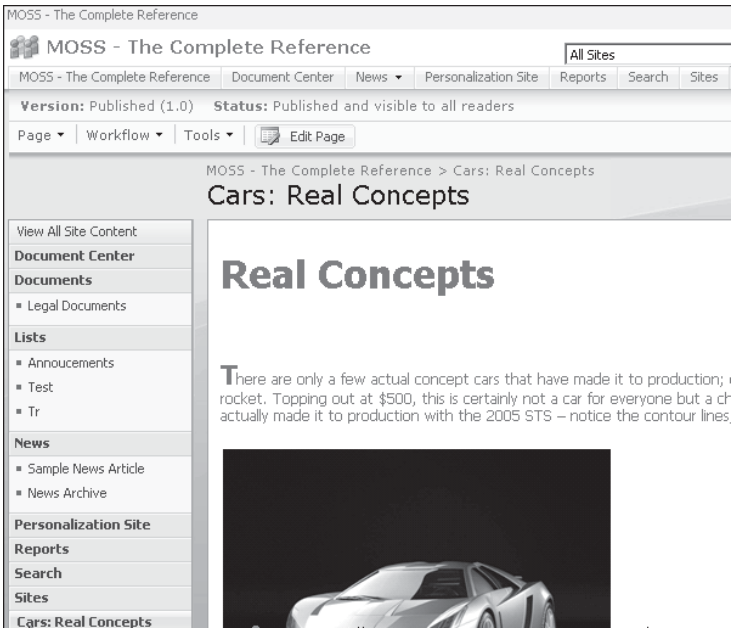


7. If you have sufficient rights, you'll notice a Configure Converter Settings link at the top of this conversion page. This option allows you to customize the settings for the conversion type; this is usually set in Application Management but can be modified here. Clicking on this link will show the following options:

   - **Inheritance** The Define Unique settings for this content type will already be selected and can't be changed.
   - **Page Layout options** These include the following:
     - **Page Layout** This allows choosing a page layout for conversions of this type.
     - **Field for Converted Document Contents** This allows selection of what field will be populated from the converted HTML content.
   - **Field for CSS Styles Options** These include the following:
     - **Remove CSS <styles> from the Converted Document** This strips any styles in the HEAD tag (this will assume SharePoint styles).
     - **Store CSS <styles> from the Converted HTML in Field** This allows selecting a field to store Styles in.
     - **Location** This enables/disables a default site for creating pages (this can be overridden).
     - **Do Not Allow Users to Change** This enables/disables users overriding the set location.

- **Processing Options** These include the following:
  - **Create Pages One at a Time** If enabled, creates a page, and then redirects the user there.
  - **Create Pages in the Background** If enabled, conversions are run in the background and the user is returned to the Document library when the conversion is complete.
  - **Do Allow Users to Change** This enables/disables user setting processing options.

8. The Create New Page options here set the new page settings:
   - **Title** The title shown in the page heading (stored in the Page Library Item Column).
   - **Description** The description of the page (stored in the Page Library Item Column).
   - **URL Name** The name to be used for the aspx page.
   - **Processing Options** These include the following:
     - **Create This Page For Me Now** This immediately converts the document and when complete, displays the page (page will be in Edit Mode).
     - **Create This Page for Me in the Background** This converts the document using a background timer job and returns to the Library View.
     - **Send Email To** If checked, a notification will be sent to all users specified that the conversion job has completed.
9. Enter a title and an optional description and then set the process method to use (in this example, I'm doing it immediately).
10. A few things can occur at this point:
    - If there is an error in the conversion and it cannot be completed, it will display an error page and then return to the document library. If attempted on a domain controller, CE_OTHER will be displayed.
    - If the document has an embedded image that is not in SharePoint, it will display a message similar to, "The document converter was able to create HTML from this document, but it had to modify the output: 1 embedded image(s) from the document were not preserved in the HTML." The page is then created and you can navigate to it, however, the image must be edited to point to an image in a SharePoint Library. To correct this, you must modify the original document to use images from SharePoint.
    - If the conversion is successful, you are brought to the page. When the page is displayed, you are in Editing Mode and the page is in Draft Mode.
11. Once published, the page will look similar to this:



Updating Pages from Converted Documents

Once a page has been created from a conversion process, it is loosely independent of the document; the original document can now be changed, edited, and so on, without the page changing. However, to update the page is quite simple. Open the page for editing and then select the Update Content from Source Document link; this will reload the page content automatically.

Creating Variations

Variations are the key method used for localization and customization of the SharePoint presentation. Variations in SharePoint represent variations in content; in other words, it's the same structure but different content. Variations are mostly used for localization (varying content by language), but are also used for any kind of variation (for example mobile, BlackBerry, and so on). The structure is fixed, but different styles and layout pages can be used on variations.

Variations are based on a master/peer relationship wherein one variant is master and other variants are below it. Variations are based on Variation Labels, which are explicitly named versions of supported content (for example, English, Spanish, Mobile, and so on). For each variant of the master, SharePoint automatically creates peer sites and pages based on the Variation Label. The relationships between the master and peers is loosely connected and is controlled by Variation Labels and a Relationship List. Updates and new pages created in the master are automatically created as drafts in the variation site(s). These drafts can then be separately reviewed and approved.

Be aware that just because you call a variation "Spanish" does not mean that SharePoint will create translated pages (now wouldn't that be cool?). Rather, it just duplicates the content so that a person (presumably Spanish speaking) can see the draft, translate it appropriately, update the content, and publish it. There are even translation workflow processes intended for this type of activity (in those, individual translators can be assigned to handle the conversion of specific languages). In the case of simply reformatting, such as with Mobile, there is obviously no translation, but pages must be reviewed to prevent content overflows, oversized images, and so on.

Variation settings are covered in Site Administration, however, as a review:

1. From a site page, use the Site Actions menu to select Site Settings | Modify All Site Settings.
2. On the Settings page, under the Site Collection Administration group, click Variations. On the Variation Settings page, you have the following options:
   - **Variation Home** This sets the starting location where variations of the site will be created.
   - **Automatic Creation** This enables/disables automatically creating site and page variations.
   - **Re-create Deleted Target Page** This enables/disables re-creating a page if it is deleted, but then needs to be republished later.
   - **Update Target Page Web Parts** This enables/disables carrying web part changes over to variations.
   - **Notification** This enables/disables sending an e-mail to owners when a new site or page has been published.
   - **Resources** This sets where to get resources for the variations (either used from the existing page (the source) or copied to the variation (duplicated).

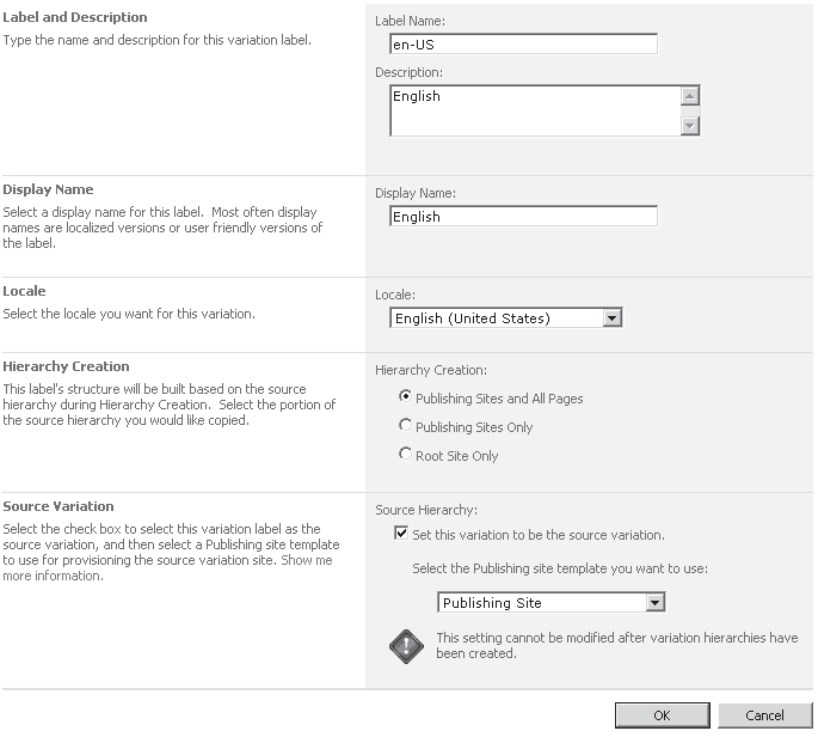
To use variations, you must create Variation Labels, with one of these designated as the Source (or master). This is also on the Settings page under the Variation Labels link. When you open it, this will list all of the current labels defined (though initially there will be none). To create new labels, click the New Label link. When the New Label page opens, you can specify the following options (be careful before saving, because if the label is set as the Source, the hierarchy cannot be changed):

- **Label Name** This is the name of the label.
- **Description** This is a detailed description of what the label is for.
- **Display Name** This is the display name (in other words, what the end user sees for the name, so it should be localized).
- **Locale** This is the locale for the variation label.
- **Hierarchy Creation** This sets the creation hierarchy, which is one of Sites and Pages, Only Publishing Sites, or Root Site Only.
- **Source Variation** This sets the source and the publishing template to use.

Enabling Variations

After a site is created in SharePoint, variations are not enabled, so the first step is to enable the variations using the Settings page mentioned previously. Once this has been set, a

Redirect.aspx page is added to the site (this is used to redirect users to the correct variation). Next, you need to establish the Source Variation Label, which is the master site that will push down to other sites. To set up a Variation Label with English as the Source, the setup would look something like this:



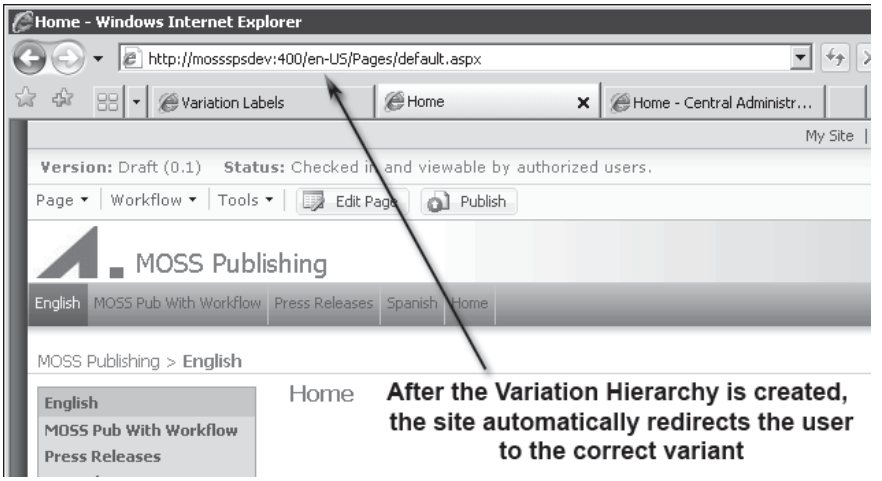
The language designation shown here sets en-US for English-US; to find other languages, you can use Internet Explorer (under settings, choose languages and then Add to view the complete list). Create as many Variation Labels as are needed. In this example, I've created an additional one for Spanish, as shown:



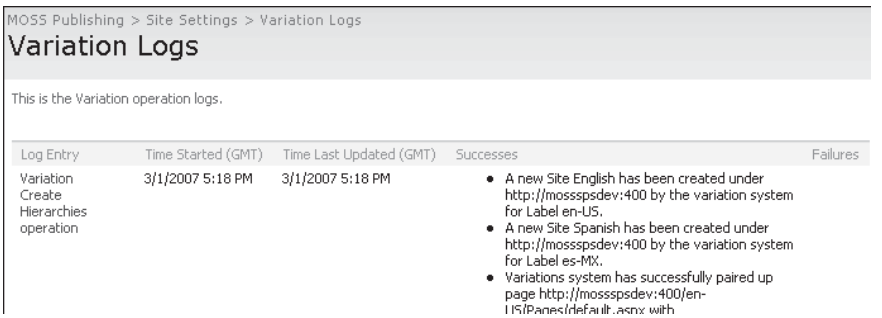
Once variations have been created, the last thing to do is to actually create the variation hierarchy by simply clicking the Create Hierarchy link. The ticker page will show a few times as each of the sites is created. When complete, the list will indicate that the hierarchy

has been created on each line item; at this point, the Site is now reachable from the Variant Labels:

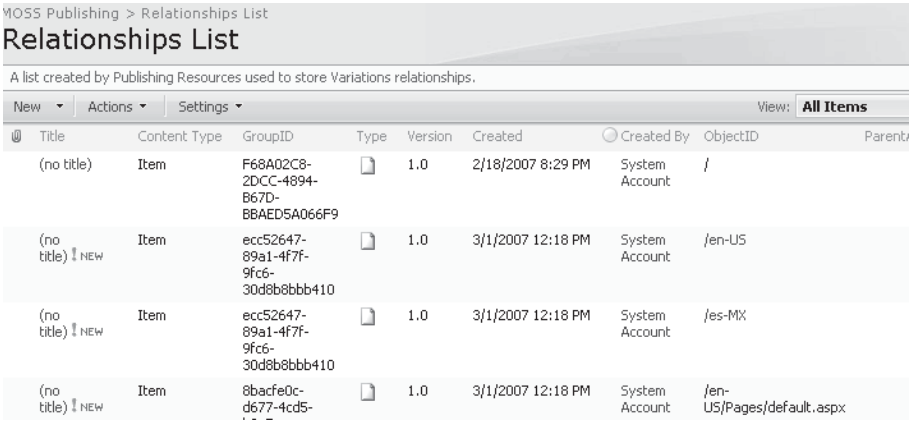
```
http://<site>/en-US/
http://<site><:port>/es-MX/
```



Be aware that if you add an additional Variant Label, you must generate the hierarchy again. Also, any time that variations have been created, the activities are tracked. To view this, go to Site Settings | Variation Logs:



Finally, all variations are managed by a simple SharePoint list that is created at the Root Site. There's not much to do with this list; however, it does list the page relationships to Source and so on. If, for example, you were not finding a page you thought should be there, you can check this list to see if the relationship was created. The URL for this is simply http://<site><:port>/Relationships List/AllItems.aspx (this is an internal list; modify the views all you like, but don't change anything else):


MOSS Publishing > Relationships List

Relationships List

A list created by Publishing Resources used to store Variations relationships.

New | Actions | Settings | View: All Items

Title	Content Type	GroupID	Type	Version	Created	Created By	ObjectID	Parent
(no title)	Item	F68A02C8-2DCC-4894-B67D-BBAED5A066F9		1.0	2/18/2007 8:29 PM	System Account	/	
(no title) NEW	Item	ecc52647-89a1-4f7f-9fc6-30d8b8bbb410		1.0	3/1/2007 12:18 PM	System Account	/en-US	
(no title) NEW	Item	ecc52647-89a1-4f7f-9fc6-30d8b8bbb410		1.0	3/1/2007 12:18 PM	System Account	/es-MX	
(no title) NEW	Item	8bacfe0c-d677-4cd5-		1.0	3/1/2007 12:18 PM	System Account	/en-US/Pages/default.aspx	

You should note that the default view settings show only the Title of the item (and often No Title for pages). I modified this view to include additional columns (as you will have to). Also, grouping by GUID will allow you to view the list grouped by the individual page groups (remember one is created for each Variant Label). If there is a source and two variants, three entries would appear in each GUID group.

SharePoint Content Deployment

In line with true content management, SharePoint provides the tools needed for publishing workflow: full-blown content authoring, approval, and publishing processing. This includes inline publishing capabilities (you've seen this with pages), but also the ability to push content from one site to the next. This is true content staging; a master (Source) site can publish content to one or more destinations, either en masse or incrementally by schedule or on demand.

To facilitate content deployment, SharePoint provides several tools and options to manage the process. The actual process is quite simple in that Deployment Paths define a specific Source (as the master) to one or more Site Collections. These paths are then used in Content Deployment Job definitions (set in Central Administration), which schedule when content is copied. For immediate/on-demand content, delegated users can run Quick Deploy jobs using the same definitions. Quick Deploy jobs run immediately regardless of the normal schedule. Content deployment is incremental by default (changes only) and handles most of the issues related to deployment, including content and link fix up as well as setting the appropriate security required on objects in the destination site. Developer hooks are also built in to extend the processes.

Here, I'll be covering the User Interface and the ways you can use it to manage content deployment. Be aware, however, that with just a few lines of code, all of the features can be managed directly from the object model. This includes setting up paths, adding or changing jobs, and so on. Two fine examples of this are in the Microsoft Office SharePoint Server 2007 Software Development Kit (SDK). Open the SDK and search for the "How to: Deploy Content Between Servers" article for simple deployment and "How to: Customize Deployment for Disconnected Scenarios."

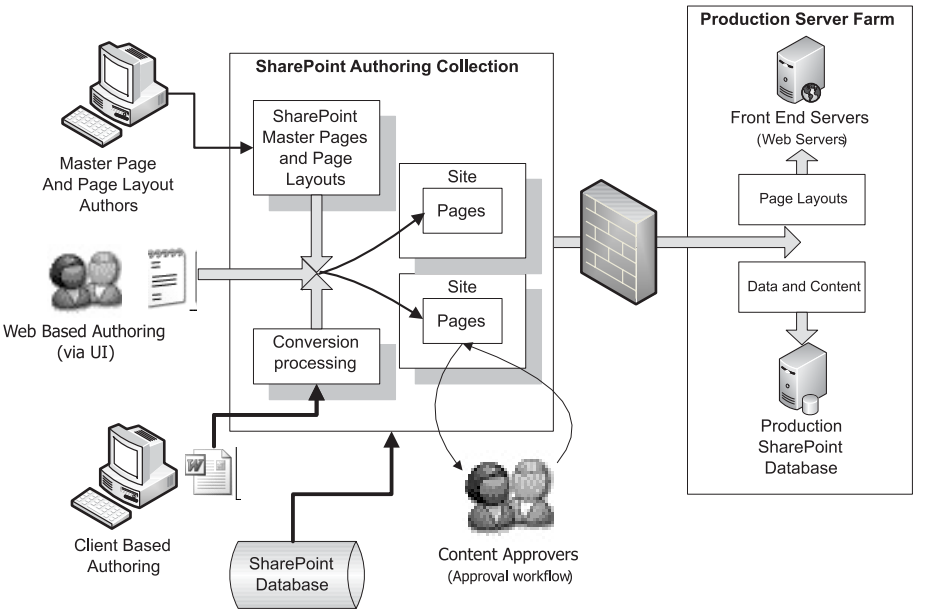


FIGURE 14-3 Overview of SharePoint Content Deployment

Content Deployment General Settings

Settings for content deployment are set up in Central Administration Operations. While this was covered in Chapter 6, it is worth revisiting in detail here. To check the settings, open up SharePoint 3.0 Central Administration, and then click the Operations tab. From the Operations page, the options available are shown (you must be a member of the Farm Administrators to do this) This provides the link for accessing content deployment paths and jobs, content deployment settings, and checking deployment of specific content. Deployment Paths must be created (before jobs) and can be from any site to any site; even different authentication can be used between source and destination (this is very handy on an extranet site!). However, the base settings must be verified first by clicking the Content Deployment Settings link. This sets the general options for all deployments:

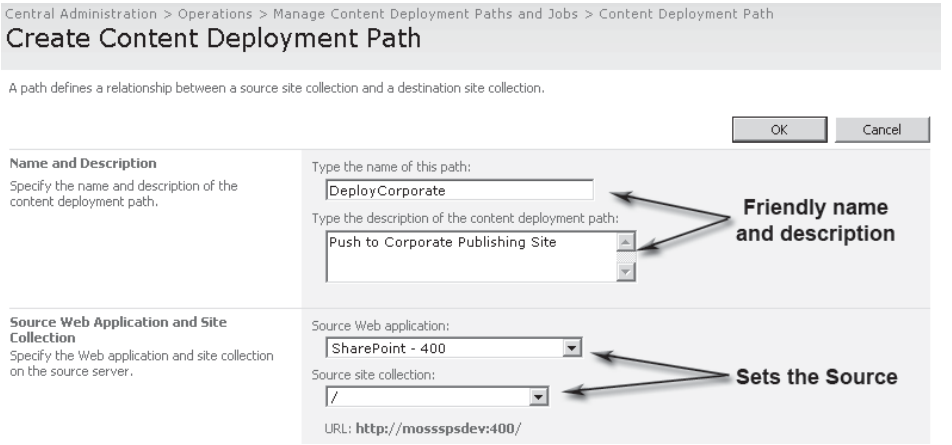
- **Deployment Jobs** This allows enabling or disabling of content deployments to this server.
- **Import/Export Servers** This sets the inbound and outbound servers to be used for deployments (any front-end or application server can be used).
- **Connection Security** This enables/disables using encryption when doing the deployment.
- **Temporary Files** Temporary storage for the deployment (this, by default, is the system drive where SharePoint is installed and should definitely be changed).
- **Reporting** This sets the number of reports to keep on deployments.

This gives you a lot of flexibility in managing the process because this is all dynamic (servers can be changed and so on). But these are usually set-and-forget settings; once set,

new deployment jobs can be created. Individual jobs can push only a part of a Site Collection (for example, pushing the News site hourly, the rest daily), and it is reasonable to run incremental deployments even during peak times.

Creating Deployment Paths

Access to the jobs and paths is through the same option. From the Operations page, click the Content Deployment Paths and Jobs link. Initially, this will open the empty list (side note: this list can be versioned, use workflow, and so on). Clicking New Path will open the Create Content Deployment Path page:



In the first sections, you will specify the Name and Description of the deployment path; this should be something that seems reasonable for the site owners who might use it. In the next section, the Source (master) Web Application and Site Collection are selected. This can be the top-level root (as shown), a Site Collection root site, or simply a sub site. Below that, you specify the destination's Central Administration Site URL (it uses this to contact the Database and Site Collections). During the deployment, SharePoint will use this site's features heavily to handle creating the application, the site collection, and the storing of the content (note that using HTTPS here is preferred). The next few sections complete the Connection Type and Security options:

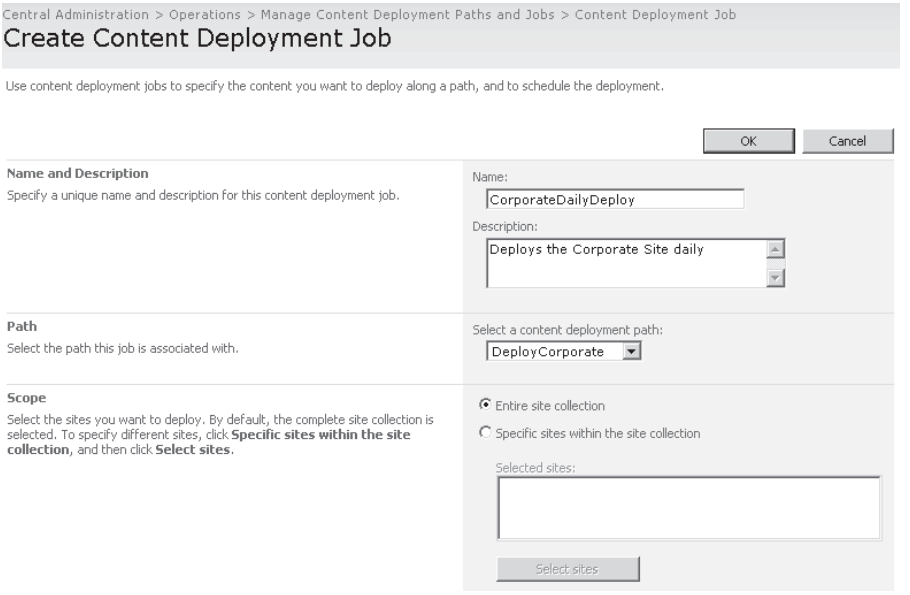
- **Authentication Information**
  - **Connect Using the Application Pool Account** This is normally used in a farm deployment.
  - **Connect Using a Different Account** This is used when connecting to a remote farm.
  - **Connect button** This is used to establish the connection (enables the Destination Web settings). Note that you may receive an error here; usually the problem occurs because you are using an incorrect account or you don't have the right access. You may also receive an error when the destination is blocking content import. If you do, the remote server is not enabled to accept incoming deployments (correct this in Central Administration on the destination).

- **Destination Web Application and Site Collection**
  - **Destination Web Application** This sets the destination application (virtual).
  - **Destination Site Collection** This sets the destination site within that application.
- **User Names**
  - **Deploy User Names** This enables/disables adding users from the source to the destination.
- **Security Information**
  - **Security Information in the Content Deployment** This sets the security objects to include with the deployment (All is the default).

Click OK to create the deployment path.

Creating Deployment Jobs

You gain access to jobs and paths through the same option. From the Operations page, click the Content Deployment Paths and Jobs link, which will open up the Deployment Paths and Jobs page. (Note that a deployment path must be defined before a job can be created.) Clicking New Job will open the Create Content Deployment Job page:



In the first section, you specify the Name and Description of the job (be descriptive so that users can quickly identify it). Under the Path, you can choose the deployment path for this job and then you can specify to copy either the whole Site Collection or one or more sites at one time (the Select Sites button allows you to browse for them). In the next section, you set the Job Schedule. From here, you can see that you have all kinds of options for frequency, starting dates, times, and so on. Below that you have the Deployment Options; as you can see, this is set to Change Only (incremental). This can also be set to Everything for a first time deployment or a complete site refresh. Finally, you can be notified when the job is complete or if there is a problem with it. Click OK to save the job; the job will automatically fire up on its own, based on the schedule you set.

Automating Publishing Deployment

There are many times when an organization might want to automatically deploy content pages where the content is from another source, comes from another item, or even comes from information found in Line of Business systems. MOSS provides the Publishing Namespace, a very easy way to work with documents and data sources for the automation of content creation. Using Publishing Namespace, there are several ways in which a little code here and there can completely automate the process:

- **Using Event Receivers** This is a list or library that can trigger an event (like an Add or Update) and then either create or even update an existing page using the list (or library) content.
- **Using Workflow** This hooks into workflow, and can easily automate the process; it works very well when dealing with external entities (translators, content providers, and so on) that communicate to SharePoint via e-mail. (This is another use for e-mail enabled libraries!)
- **Using Web Parts** A web part can easily be created to provide some interactivity with the user, perhaps by selecting the documents to publish and then handling the updating of pages automatically.
- **Using Applications** This is for connecting to a known SharePoint site from an application that enables the creation of timer jobs for managing publications, either on-demand or on schedule.

All of these methods use the same basic logic. Either data or a file is collected and then, using a document converter, converted to a page and published. While I don't want to jump too far into SharePoint development, it is worth it to take a quick look at programmatic publishing. While the User Interface is great, it certainly isn't the end all; if you had to publish more than just a few documents, it could take quite a while. The Publishing Namespace gets around issues like this. As an example, let's use a real scenario:

- **Problem** Your organization works with several outside contributors that provide articles that go into a yearly corporate report. Work goes on for the whole year and all of them must be reviewed. Because the editors sometimes need to send the whole document back to the contributor, it is easier to keep the articles in Word format until they are ready to be published. While a work in process, they are stored in a SharePoint Document Library in the site. The Document Library is e-mail-enabled, so that contributors simply mail the documents there. Alerts notify editors when a new document has been added. At year end, all articles must be published to the site on a certain date and hundreds of documents in the library must be converted to pages in the site.
- **One solution** Create a console application that can be run on demand, through Windows Scheduler or as a Timer job. Here is the sum total of the code you need to accomplish this:

NOTE *This requires adding a reference to Microsoft Content Management and Publishing to access the Publishing Namespace.*

```
// Open the site using the URL in a console app (Context in Features/Web Parts)
SPSite TopSite = new SPSite("http://<server><:port>/News");
SPWeb CreateInWeb = TopSite.OpenWeb();
```

```
// Open the Library and get the collection of items (the name
// in this case is 'ToBePublished'):
SPList TheDocLib = CreateInWeb.Lists["ToBePubished"];
SPListItemCollection ListItemsToProcess = TheDocLib.Items;
// Get a new reference to the web as a publishing web
// then get the Page collection of the web:
PublishingWeb PublishToWeb.GetPublishingWeb(CreateInWeb);
PublishingPageCollection ThePageCollection = PublishToWeb.GetPublishingPages();
// Loop through each item in the Collection:
foreach(SPListItem SPLI in ListItemsToProcess)
{
      // Get the title of the document to use as page name:
      string TitleOf = (string)SPLI["Title"];
      // Set the transformer - this GUID is the DocXtoPage converter:
      Guid DocXtoPage_TransformerGUID =
new Guid("6dfdc5b4-2a28-4a06-b0c6-ad3901e3a807");
      // Create a new page name automatically:
      string NewPageName = TitleOf + ".aspx";
      // Add it and do the conversion immediately:
      ThePageCollection.Add(NewPageName,
           SPLI.File,
      DocXtoPage_TransformerGUID, PageConversionPriority.Immediately);
}
```

With minor modifications, this code will work in any assembly, web part, feature, or similar. The major difference is the connection to the site (in a web part, for example, the web context is used to return the current site). In this example, you used the existing document converter (DocXtoPage). Custom document converters can be created and used on any Document Content Type. Notice that the PageConversionPriority is set to Immediately, meaning it will convert when started. This is but one of many additional options available here, including the starting/scheduling of conversion jobs. This is useful for handling large documents, those that take a long time to convert, and for the setting up of publication schedules. Also, expanding this is very easy; you could have easily scanned several libraries, compared publishing dates before publishing, checked list-item data, and so on.

Creating Custom Field Controls for Publishing

Right out of the box, SharePoint will translate pretty much any column type into the right kind of editing control for users to add content to the page. However, there are cases in which a custom field control is needed to deal with a special kind of content. One example is when displaying Flash or Windows Media content. This is a foreign idea to SharePoint (it's an object and can't be described by a data type), and a field definition does not come out of the box. Creating controls are based on a few specific elements:

- **Data Storage** The control has to have somewhere to store the content, so this is stored as a field in the list item (in the Page library) that goes with the page.
- **Base Field Type** A control has to have some base field to inherit from; SharePoint provides HTML, Text, and Links (there are a few others, but these three are the most common).
- **Rendering** A control has to be able to handle all of the aspects of content editing, as well as View, Edit, and Design Modes. This includes providing some way to preview for the editor.

Custom Field Controls are sort of like web parts, and in fact, they overlap in some of the functionality that web parts provide. Certainly the rendering aspects are similar; a web part has to know how to react to the Page Mode and so does a field control. But where web parts are "mini-applications," as I call them, field controls are much more suited to the content management side of things. For one, they are more concerned with content and storage of that content, and for another, they reside inline with a fixed Page Layout and can't be moved by the user. The process for creating a custom field is not very complicated. The steps are as follows:

1. Create a new assembly.
2. Strongly name the assembly.
3. Add the assembly as a SafeControl in the SharePoint web.config file.
4. Load the assembly into the Global Assembly Cache (using GACUtil or by dragging the assembly DLL into the c:\windows\assembly folder).
5. Run IISReset (this reloads the web.config and GAC).
6. Add the assembly reference to the Page Layout (using the <%@ Register tag).
7. Add the field reference to the page.

Creating custom fields obviously requires coding. Within the field logic, it must handle the various states, as well as the User Interface itself. While I originally set an example of creating a field here, for once Microsoft was ahead of me. In the Microsoft Office SharePoint Server Software Development Kit (also known as the SDK, freely downloadable from the Microsoft site), there happens to be a very complete example of creating a field for rendering Windows Media. For more information, search for the article "How to: Create a Custom Field Control" in the SDK.

Page Caching and Controls

A major consideration in enterprise content management is the ability to deliver content in a manner that is both efficient for users and scalable to meet demands. After all, what good is information if takes too long to get it? So, as you have probably guessed, there are several ways to speed up the process of delivering web content. The primary method for speeding up this process is called *caching*. Caching in a web is the practice of storing pages, images, and content information locally in memory and disk to provide faster access on requests.

Of course, everyone who has used a browser knows a little bit about this; information like cookies, images, and even code snippets are stored on the local hard drive. When returning to the page, assets found in the local cache are used instead of a trip to the server. This means a huge boost in performance. While this is great for the client, the server has a similar issue; every time something is requested from the site, whether it's a page, an image, a control, and so on, the server must retrieve this from the disk, convert it to HTML, and then send it to the client. Multiply that times 1000 users and that's a lot of hits on the disk.

To handle scalability in numbers, SharePoint leverages both disk and memory-based caching. When a user requests a page, the resulting page is stored in memory; if needed, assets (images and so on) are stored in disk cache (on the server). When the next user requests it, the server sends down the memory version of the page rather than having to retrieve it from the disk; the assets needed are pulled from the disk only if the client-side cache doesn't have them. Thus, while the first user sees a hit, subsequent users do not, because as long as

the page does not change, the cached version is reused. For pages that perform external functions, such as calls to databases and similar, this is quite a performance enhancer. Disk-based cache is usually used for stuff that can't (or shouldn't) be stored in memory, such as images, large files, or other assets like scripts. While displaying the item still requires disk access to handle a request, if that image or document came from an external source (like the database), it still saves a few operations. If a particular item is already in the client cache, the page uses the client asset, which is even faster.

While the caching of assets helps, SharePoint has many aspects: role-based security, navigation, audiences, and so on. This means that it actually uses several caches to do the following:

- Minimize the amount of work per user while still providing dynamic content
- Scale up (to number of users) and out (to number of servers) better
- Preserve per item and role security
- Reduce the number of roundtrips to the server, particularly to the SharePoint Database

The actual execution flow of processing a page request from the browser that SharePoint uses to render a page is very similar to ASP.NET, though several other things occur as well. For a first-time request (nothing cached):

1. SharePoint pulls the page details and controls from the database.
2. Next, it uses ASP.NET to compile the page for creating (on disk) the Page Class.
3. Then it pulls in the Page Layout (with controls) and compiles it to create the Page Layout Class.
4. Next, it uses SPContext (SharePoint Context) to pull the information about the Master Page for the site, and then compiles the Master Page and creates the Master Page Class.
5. Once all compilations are complete, the final HTML page is stored in the cache and is then sent back to the requestor.
6. On subsequent requests, the cached versions are used if possible.

I'm being very brief here—there are more steps in the process, but the previously mentioned are all that concern developers; the behind-the-scenes work SharePoint does isn't even accessible.

SharePoint Caches

To deal with the types of items SharePoint handles, including items, pages, documents, and so on, there are several kinds of caches set up to enhance performance:

- **The Root Web Cache** This cache is used internally by SharePoint and manages the behind-the-scenes internal controls, style sheets, and so on. In ASP.NET terms, this is like the IIS cache, but it is not intended for use by developers.
- **The Disk Based Cache** This cache is also used internally by SharePoint, but it is not used for page rendering. Rather, it is used to store images, documents, PDFs, and any otherwise "blob" objects stored locally on the front-end server, which prevents the necessity of getting these from the database. This is available for developers to take advantage of in coding.

- **The Content Query Cache** This cache manages the SPQuery Cache used for list items, based on queries to a list and the Cross Site Query Cache (stored in the database). The latter handles the cross-site item and document queries; when a query is executed, it is run twice initially, once as an Administrator, which retrieves all major and minor versions, and then again as a super reader that reads only the major versions. In both, the items in cache are stored with the ACLs (Access Control Lists or user rights) for all of the objects involved (items, images, and so on). This allows lookups to be trimmed, based on the security of the items in the cache without a roundtrip to the database. This cache can be configured so that it's either time- or change-based, and can also provide *row multipliers*. Row multipliers apply to the list data itself. Because all items are available, a request by a particular user filters the results by the rights to a set multiplier (for example, if the multiplier is the Last 10, the filter will examine all to try to filter down to 10 for that user without a return to the DB).
- **The Navigation Node Cache** This includes the WebSiteMapNode and the List Item (page) Caches, which store information about pages and webs (Title, URL, Icons) and also store ACLs for each (again allowing a security trimmed retrieval of pages).
- **The Page Item Cache** This is used to cache the actual pages in SharePoint (remember that pages are items) and is in effect as long as the page uses the SharePoint Template structure. This handles storing items when requested, as well as full-field data for the page row. Page items will always remain in cache with the exception of checked-out items, which are not cached.
- **The Page Output Cache** This cache is most beneficial for pages that don't require re-execution of the entire page, but that may provide some (minimal) personalization. This leverages the ASP.NET Post Cache Substitution features, where parts of a cached page can be personalized, but the majority of the page and controls cannot. This cache is also role-based and works on buckets of users. This means that a version is generated for each user bucket, which allows some personalization based on group. This cache is most beneficial to developers, because parts of the page can be designated to do static call backs for retrieving personalized information and can use the SPContext for user information.

Cache Policies and Profiles

The way SharePoint will handle a particular kind of item in a cache is based on its Cache Policies and Cache Profiles. Configured for the site and individual Page Layouts, Site Administrators have full granular control over how a particular site uses a cache (you should note that there is a hierarchy in profiles; when enabled, Page Layouts takes precedence over site).

- **Cache Policies**
  - Sets the default settings for the site
  - Sets the granularity of Cache Profiles and determines who can override them
  - Centrally controlled per site collection
  - Allows/disallows site owners and specific Page Layouts to choose a Cache Profile
- **Cache Profiles**
  - Provides the same granular settings as the OutputCache directive in ASP.NET

- Centrally defined; change in one place, takes effect on the Site Collection immediately
- Defined through a standard SharePoint List (provides standard list versioning and even workflow management if desired)
- Applied to the site/Page Layout through the User Interface
- Push down (like content types), which forces profile consistency in sub sites
- Are completely independent of one another

Configuring SharePoint for Caching

When SharePoint is installed, output caching is disabled by default, because for many site types it is not useful. In fact, it should really only be enabled for high-traffic sites such as Publishing and Internet sites (and with some planning!). Never turn this on in Shared Services or My Sites, because it will severely hamper performance. Accessing the settings is done at the Site Collection level (the root site) through the Site Settings (Site Actions | Site Settings | Manage All Site Settings). There are three selections involved: Site Collection Output Cache, Site Collection Cache Profiles, and Site Collection Object Cache. All of these are covered next.

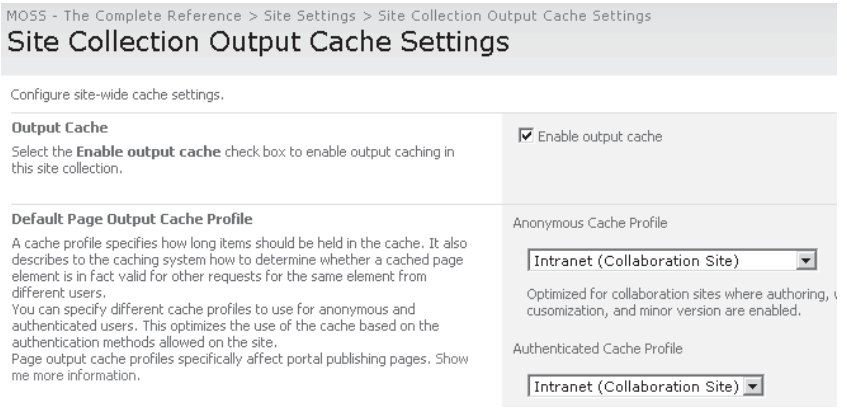
Enabling Disk-Based Cache Like caching in general, SharePoint's Disk-Based Cache is disabled by default. When disabled, items, such as images, are pulled from the SharePoint Database every time they are needed. When enabled, a local copy is put on the front-end web server. To enable this requires modifying the web.config file for the SharePoint Site, which is located in the root of the SharePoint Web Site (use IIS Manager to find the correct folder if need be). Navigate to this folder, make a backup copy of the web.config file, and then open it in an editor like Visual Studio or Notepad. Search for the word Blob until you find the line that looks like this:

```
<BlobCache location="C:\blobCache" path="\.(gif|jpg|png|css|js)$" maxSize="10"
enabled="false" />
```

The settings here will set up how the cache will operate, starting with the location. This is set by default to be C:\blobCache, which you'll need to change (using the system drive as a location is always a bad idea). Next, you'll need to review the list of file type extensions that will be stored; if you use files other than these (like PDFs), you can add the extension type(s) here. The maxSize sets the total number of megabytes to maintain in the cache. Finally, the enabled option must be set to true. Once you have made the desired changes, save this file. Next, you must run an IISReset to reset the site and then actually enable the cache. The folder you choose will be created automatically; however, there are a few things to watch out for here:

- If an invalid file location (like the wrong drive) is used, the site will still operate but performance will degrade quickly—very quickly!
- Don't use any drive that isn't physically connected to the system or is tight on space.
- The account used to manage the files here is the one that is running the application pool for the particular Site Collection; the folder selected (and any above it, if it's nested) must have ACLs that grant full privilege to this account (Modify Rights).

Site Collection Output Cache This option allows the enabling and disabling of caching as well as the general site settings and caching profiles. Clicking the Site Collection Output Cache link opens the Settings Page:



This page sets the base profiles for the Site Collection (including sub sites) as follows:

- **Output Cache** This enables/disables this feature.
- **Default Page Output Cache Profile** This sets the default profile for the entire site collection, while Anonymous Cache Profile sets the base setting for unknown users in the site. The default options here are
  - **Disabled** This disables all caching for anonymous users.
  - **Public Internet (Purely Anonymous)** This is a caching setting for typical single site use; ACL information and rights are not stored (highest level of caching).
  - **Extranet (Published Site)** This is a caching setting for a typical Publishing site using personalization features (medium caching).
  - **Intranet (Collaboration Site)** This is a caching setting for typical use in a corporate setting (less caching).

NOTE *These options are set by the Cache Profiles available.*

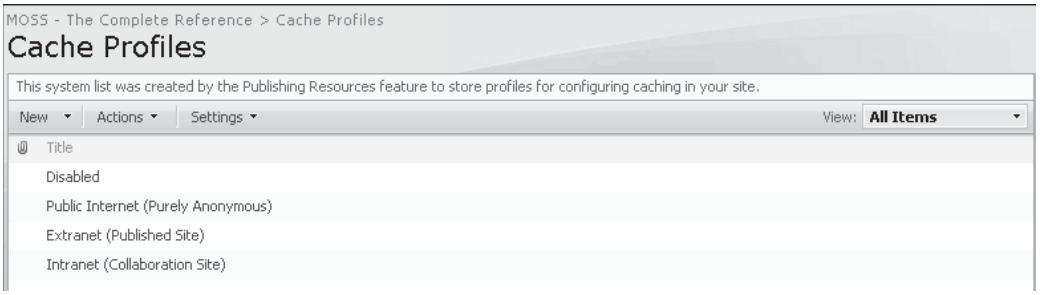
- **Authenticated Cache Profile** This sets the base for logged-in users; options here are Disabled, Extranet (Published Site), and Intranet (Collaboration Site), as described previously.

NOTE *In either of the previous cases, disabling caching is best for volatile sites with constantly changing content or lots of regular page replacement, as caching will actually hurt performance under those circumstances.*

- **Page Output Cache Policy** This sets what can be done in Publishing sites. If either of these is checked, Site Owners can override the default set previously.

- **Debug Cache Information** This enables/disables debug information (this includes date and time page last rendered and so on) being included in cache; this should always be disabled in a production environment as it adds quite a bit of overhead.

Site Collection Cache Profiles Once caching is enabled, the Cache Profiles come into effect. Clicking on the Site Collection Cache Profiles link will open the Cache Profile list:



This page lists all of the existing profiles that have been created, with the initial four default ones shown. The unique part of this is that it is, in fact, a SharePoint list. This means that the same capabilities of a list—such as versioning, approval, and workflow—all apply here as well. This allows for policies to use the same structure of the overall information architecture in the company. Profiles have the following options (simply click on any one of them to view):

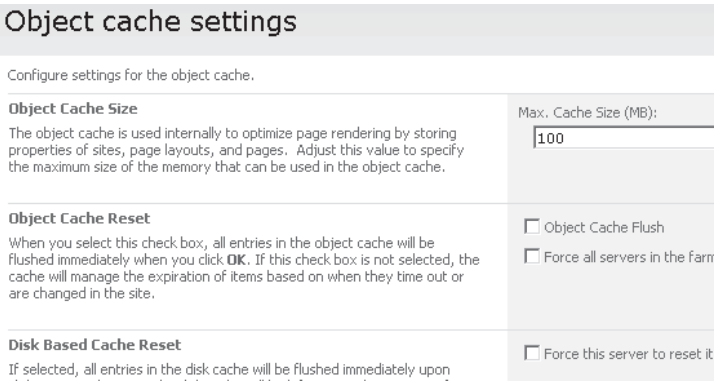
- **Title** This is the name of the profile (additional ones can be added).
- **Display Name** This is an arbitrary name you wish to assign.
- **Display Description** This is the description of the policy to explain what it is set up for.
- **Perform ACL Check** This enables/disables the ability to trim cached items by security (user roles).
- **Enabled** This enables/disables caching for this policy.
- **Duration** This is how long (in seconds) to keep items in cache (see note shown next).
- **Check for Changes** This enables/disables the cache to be driven by changes in the site, and it makes the cache change based. If this is disabled, the cache is flushed based on the duration time set (this makes it time based). The time-based option provides much better performance, but you may experience a lag in content.
- **Vary by Custom Parameter** This sets the parameters to vary the Cache Policy by, and it also controls how many cached versions of the page should be created based on name/value pairs sent through HTTP POST/GET. None means one version (the default), and the * means always pass the name/value pairs to create cached versions of the page (granularity can be controlled by naming parameters, with multiple parameters separated by semicolons). An example could be to vary the settings by current language or region. For more information, reference the ASP.NET documentation.

- **Vary by HTTP Header** The cached response will vary based on the major browser version number of the caller (IE 7, NS5, and so on). If set, different browser types will receive different cached versions of the page (the normal setting for this is Browser). For more information, reference the ASP.NET documentation.
- **Vary by Query String Parameters** This works the same as custom parameters except that name/value pairs are passed in the URL query. For more information, reference the ASP.NET documentation.
- **Vary by User Rights** This enables/disables checking individual item permissions. If enabled, it verifies that all of the permissions on objects are identical on all items before using a cached item.
- **Cacheability** For more information, you can reference the ASP.NET documentation. Some of the following explanations are borrowed from there:
  - **NoCache** This sets the Cache-Control to No-Cache Header. Without a field name, the directive applies to the entire request and a shared (proxy server) cache must force a successful revalidation with the origin web server before satisfying the request. With a field name, the directive applies only to the named field; the rest of the response may be supplied from a shared cache.
  - **Private** This sets Cache-Control: Private to specify that the response is cacheable only on the client and not by shared (proxy server) caches.
  - **Public** This sets Cache-Control: Public to specify that the response is cacheable by clients and shared (proxy) caches.
  - **Server** This specifies that the response is cached only at the origin server (similar to the NoCache option). Clients receive a Cache-Control: No-Cache directive, but the document is cached on the origin server (equivalent to ServerAndNoCache).
  - **ServerAndNoCache** This applies the settings of both Server and NoCache to indicate that the content is cached at the server, but that all others are explicitly denied the ability to cache the response.
  - **ServerAndPrivate** This indicates that the response is cached at the server and at the client, but nowhere else. Proxy servers are not allowed to cache the response.
  - **Safe for Authenticated Use** This enables/disables this policy from being selected by administrators and page layout designers.
  - **Allow Writers to View Cached Content** This enables/disables caching for items that are being edited (this should usually be disabled).

NOTE *When creating Cache Profiles, you should keep in mind a realistic view of how often the particular content involved really changes. For example, if you had a news site that worked on a 24-hour publishing schedule, you can take advantage of keeping pages in the cache longer. If you have a blog site where posts appear every few minutes, limiting the time in cache will provide better performance.*

Site Collection Object Cache The Object Cache settings apply to the Item Cache and Disk-Based Cache. They also provide the ability to forcibly clear one or both of the caches manually, which is useful after a content-restore operation. These are usually set once and

are used only when clearing the cache. Clicking on the Site Collection Object Cache link opens the Settings page:



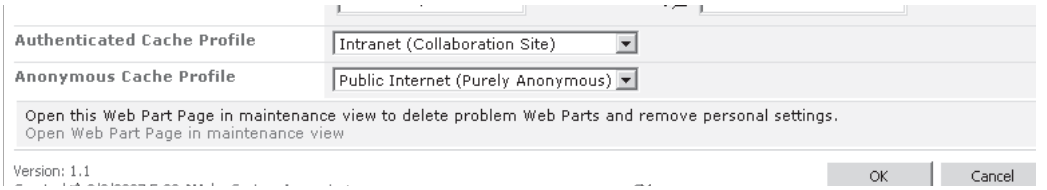
Options on this page are as follows:

- **Object Cache Size** This sets the amount of memory to set aside for the cache; the more the merrier, but you should never exceed 50 percent of the system memory.
- **Object Cache Reset** If checked, this will forcibly flush the local server cache and/or the entire server farm when OK is clicked.
- **Disk Based Cache Reset** If checked, SharePoint will force a reset of the disk-based cache when OK is clicked.
- **Cross List Query Changes** This sets either a change- or time-based method for when to clear items in cache.
- **Cross-List Results Multiplier** This sets the default number of items to obtain when using rights filtering. The number 3 indicates the message, "Try to find the top 3 items of all items based on who this user is and what rights they have."

Using Cache Polices on Page Layouts

Within SharePoint, each Page Layout can use its own caching profile. This allows granular tailoring of how a cache works on the individual page level. In the hierarchy, Page Layout settings will (if enabled) override the default ones set by the site. As an example, you might have different policies for different page types in the site; for example, those used in the legal department may not cache at all, whereas the human resources area might be heavily cached. This can be taken down to several more levels using Vary By options (this does require coding, however).

To modify the caching options for a particular Page Layout, from any page, use Site Actions | Site Settings | Manage All Site Settings to open the Settings page. Under the Galleries group, click the Master Pages and Page Layouts link. Mouse over the Page Layout desired, and from the menu, select Edit Item (you are prompted to check the item out before it will let you edit; click OK to allow this). In this example, I'm using the Article Links page, a common page used in both the Internet and intranets to provide links alongside an article. Scroll down to the bottom of the Edit page to find the Page Cache Settings:



From these drop-down menus, any Cache Profile can be selected. In this example, caching has been set to policies appropriate for an Internet site. One thing to be aware of is that any changes to a profile are immediate and affect all Page Layouts (and sites) that use them.

Developing Pages and Vary By Handlers

As you expand your use of SharePoint and begin to develop custom pages, you'll need to be sure to keep in line with the way SharePoint expects things to be with regards to caching. One of the biggest reasons that you'll want to stay in line with SharePoint's regular pages is that much of the work is done for you, and there is no reason to reinvent the wheel. By utilizing the structure of Master Pages and Page Layouts, users get the expected experience, regardless of the functionality your pages offer. To ensure this, any page you create must inherit from the TemplateRedirectionPage class from the Microsoft.SharePoint.Publishing namespace. If it doesn't, SharePoint won't know how to deal with the content and will not cache it (it may even throw an error). In addition, don't use cache settings for ASP.NET, because this will conflict with SharePoint (you will get a page error if you attempt this). All of the directives are covered by SharePoint.

Actual cache control can be handled at the page level if it's extended using an IVaryByCustomHandler interface. Much like an event handler, a custom handler enables you to make decisions about how a particular item is cached, based on environment, user, site, and so on. Using my previous example, wherein Page Layouts were used to set what is cached, you can see that custom handlers allow you to vary the cache in the same Page Layout, based on the settings you pass. A custom handler also enables you to modify cached web pages with personalization (such as a personal message with the user's name).

Creating a Vary-by-Custom Handler

Building a handler is not a very complicated process and it really involves only a few steps:

1. Use Visual Studio to create a VaryByCustom handler (an assembly class that provides a list of strings to the output cache for modifying the content on the returned page before sending the response.
2. Register the handler in the site Global.asax file.
3. Define the custom strings to vary in the Cache Profile settings; these are automatically passed to the handler when called.

Rather than repeat what is readily available, the SDK has a simple one page article that covers the exact code you need. To read more, search for the article "How to: Extend Caching by Using the VaryByCustom Event Handler" in the SDK.

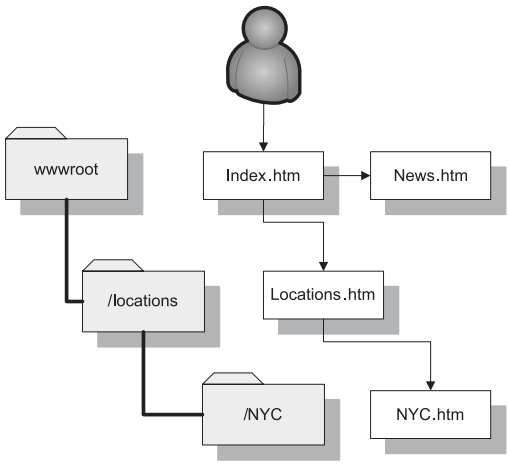
Summary

As you've seen in this chapter, SharePoint represents a true enterprise-level content management tool at all levels from basic Site Columns to Page Layouts. Combined with the built-in Publishing Workflow Tools, SharePoint accommodates any kind of content site and provides the extensibility needed to support a wide variety of publishing needs with custom converters. In addition, the SharePoint platform provides the kind of performance needed to support large and public-facing sites through multiple (and settable) caches. In the next chapter, you'll see much of this at work with the various Site Definitions provided.

CHAPTER 15

SharePoint Site Definitions

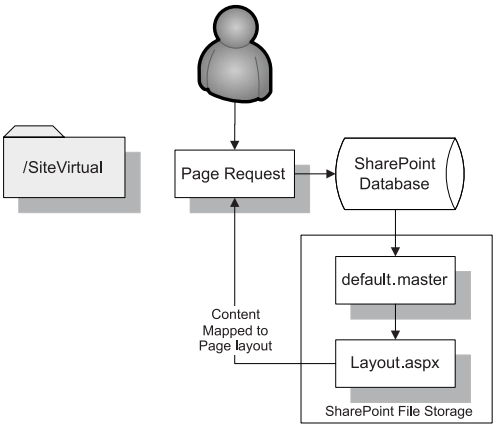
As you probably already know, SharePoint sites are not really physical sites and in fact do not use physical files. SharePoint sites do use aspx pages, but these are templates; actual content for a site is stored entirely in the database. For some, this might take a bit to sink in. To compare, consider a standard HTML website, which is based on a set of folders and then physical pages (HTML files) that are linked to each other using hyperlinks:



In this type of site, the physical layout sets the way the site will be navigated as well as its security. Links between pages are set by the actual name of the page and the folder it resides in (for example, /locations/Locations.htm). Security is managed by setting the Access Control List (ACL) permissions for specific accounts on the actual folders and files.

In contrast, SharePoint sites are completely virtual, with all content, navigation settings, and even layout stored in the database. The only physical part of the SharePoint site is the top-level directory, which is used as the anchor point for the site within Internet Information Services (IIS) and holds an XML file called web.config. This file also provides specific settings that SharePoint uses, including references to assemblies (DLLs) that it uses called SafeControls, trust levels, and so on.

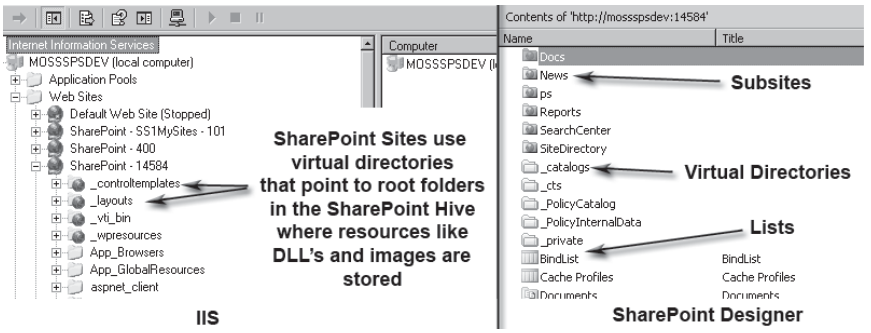
SharePoint does use some files to create pages, however. When a user makes a request, the content is retrieved from the database (not the file system). The Master and Layout Pages (also in the DB) are used to actually render (or draw) the page:



Master Pages and Layout Pages themselves are stored within SharePoint as documents in Galleries (a special site-level Document Library). While these are created physically using SharePoint Designer or Visual Studio, SharePoint stores them in the database. I'm often asked about where documents are actually stored in SharePoint. All content (list data) and documents (stored in byte format) are stored in the database. The only difference is search; all lists in SharePoint support attachments, but only the actual content of those attachments stored in libraries is actually searched and indexed. Galleries are just special kinds of lists; because the content they hold is for items such as templates, they are not included in the search.

In addition to pages as described, SharePoint sites are also based on Site Definitions (aka Application Templates). Site Definitions are basically Site Templates that provide a specific site layout including pages, lists, and libraries, and some automatically include web parts in the layout. Site Definitions are built using the Collaborative Application Markup Language (CAML), which is a combination of XML and HTML that describes the site to SharePoint. In addition to pages, Site Definitions also include list and library definitions that are created automatically along with the site. When created, the definition is stored in the database and is used to render the site and the pages in it. The definition can even be updated and pushed down to other sites that use it.

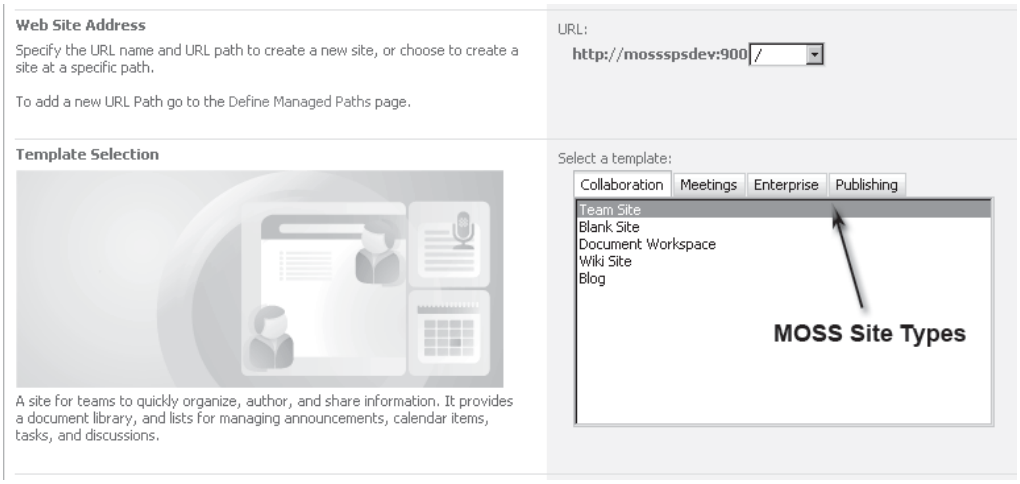
So you might be asking yourself how you go about working with sites, particularly in development, since there are no pages to work with. The simple answer is that aside from custom development of Master Pages, Page Layouts, and the raw Site Definitions built in Visual Studio, SharePoint Designer is the tool of choice. Since SharePoint Designer is aware of how SharePoint renders sites, it actually provides a view of the physical layout of the site, including simulating the folder structure. For example, if you were to open a SharePoint site in IIS, you'd see that there are only a few files there and that it was mainly just virtual directories (pointers). If you open this same site in SharePoint Designer, you'll see that the actual site, including folder structures, lists, and sub sites are visible:



Both SharePoint Designer and Visual Studio have their place in working with Site Definitions and pages, so I will touch on the use of both in this chapter.

There are different kinds of sites available, depending on what is installed. Windows SharePoint Services includes the collaboration-style sites (Team, Meeting, and Workspaces); Microsoft Office SharePoint Server includes those, plus a number of others including those that are used for the MOSS Search and Enterprise features. Selecting a specific Site Definition is done when the site is created. For top-level sites, this is done when creating the Site Collection via SharePoint 3.0 Central Administration. Sub sites (under a collection) can use any type with the exception of the MOSS Collaboration Site (this can only be used as a top-level site).

Here, I'll only cover the basic out-of-the-box site templates. There are additional application templates available for download from Microsoft (over 40 at last count, see http://www.microsoft.com/technet/windowsserver/sharepoint/wssapps/templates/default.aspx), and third-party offerings as well. Regardless of what the site is designed for, the operation and the way SharePoint renders them is the same. The actual site type (template definition) is selected when the site is created; for example, the default MOSS options look like this:



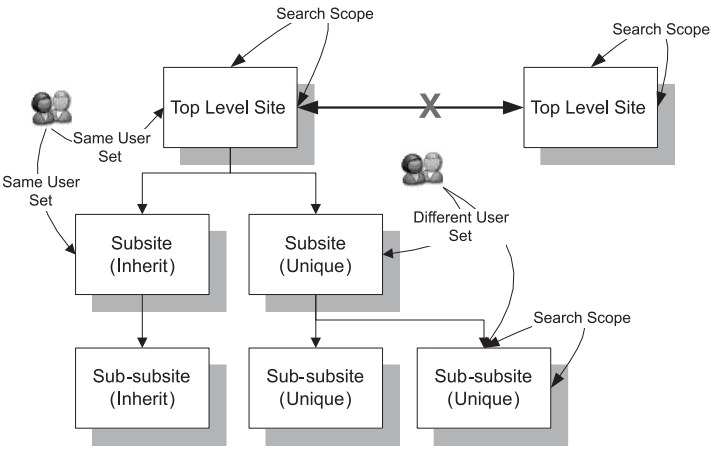
As you can see, the tab options shown allow selection of various template groups. In MOSS, this includes the Enterprise and Publishing Templates as shown. Additional templates (if installed) may also appear under an Other tab (you will see this demonstrated when saving a site at the end of this chapter). To avoid repeating a lot of content here, I won't cover the process of creating an application and top site. You should refer to Chapter 5 for details.

Windows SharePoint Services Sites

The best place to get acquainted with SharePoint sites is to start with the core sites that are the foundation of Windows SharePoint Services. These include the core definitions that have been around since the original Team Services. Now called collaboration-style sites, these represent the kinds of sites users use when working or collaborating, like the development of a new document (a Document Workspace) or a project site (a Team Site). Functionally, all WSS sites operate the same way; the only difference between types is the layout and which lists and libraries (if any) are predefined when created. Once created, any type of list, library, web part page, or sites (collaboration types) can be created within it.

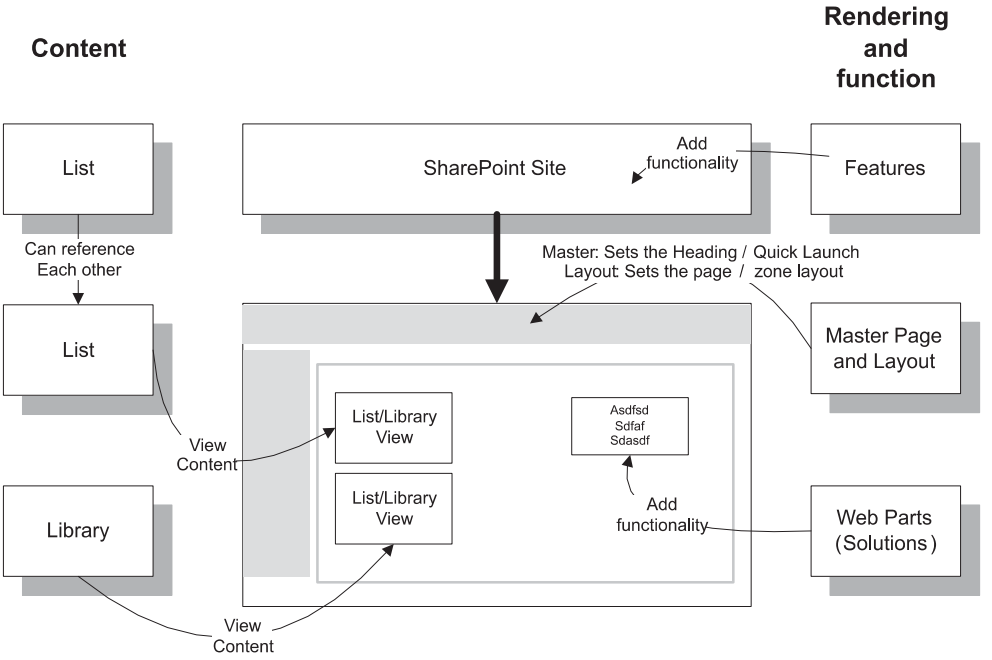
WSS sites are based on simple one-page sites created from the original design that was incorporated using FrontPage long ago. Though their construction has changed in Version 3 to use Master Pages with a single Page Layout (in prior versions, this was a single aspx page), they are still made up of a menu bar, a Quick Launch bar, and a Web Part Page Layout. Pages can be added and a site can have any number of sub sites in a hierarchy (sub sites can also have individual security). They do not support Publishing pages and the site hierarchy cannot be changed (even when created under a MOSS site).

WSS sites are distinct from each other in terms of content and search (one site cannot use or share content with another). Top-level sites have no relationship at all; sub sites (created under a top-level site) are connected only to their parent by navigation (not content). Security is also unique, because top-level sites cannot share the same security (users, settings, and so on); sub sites can either inherit the parent's security or have their own. Conceptually, this looks like so:



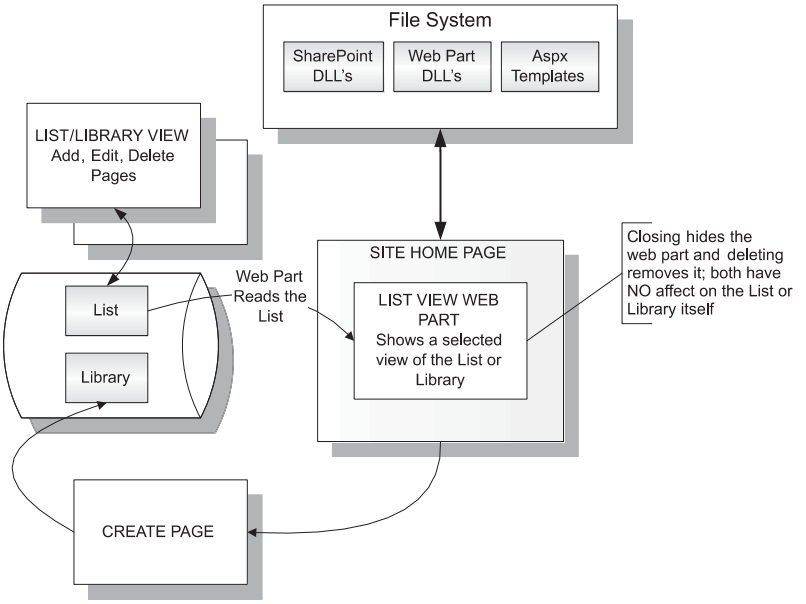
As you can see, there is no relationship between top-level sites. Sub sites are related only by the parent/child association, but can have different security settings (sub site security is based on its direct parent, regardless of where it is in the hierarchy). Search in all cases is site-specific.

The actual functionality of a site is made up of content, features, and web parts. These are rendered using the format of the Master and layout pages to actually draw the site the user sees:



Content Storage

A common area of confusion when dealing with sites is where the content goes when you add items. Where are the lists and libraries if they are not on the Quick Launch or viewed on the page? To explain this, it is best to consider the actual components that make up a site:



- SharePoint functionality is provided by SharePoint DLLs in the file system.
- Web parts that provide functionality (like Weather) are also DLLs that are stored in the File system and registered as Safe Controls within SharePoint (in other words, Safe Controls tell SharePoint which applications are okay to run).
- Aspx Page templates, also stored in the file system, are used to dynamically render the site page (Home Page).
- Lists and libraries are created using the Create Aspx Page; this creates them directly in the SharePoint Database.
- When adding, editing, or deleting content (items) from lists and libraries, each use their own set of editing pages (these are not part of the site).
- Lists and libraries are stored directly in the database and exist in a site regardless of whether they are visible on the site page or not. The Quick Launch is only a link to the List/Library View Page.
- List View Web Parts are the only views that read the list in the backend and provide a display of the data; closing or deleting the view has no affect on the list or the list content.

WSS Site Template Types

WSS provides several site templates, all based on the same core. The differences are in the layout and in which lists and libraries are created automatically. The actual construction of the site is the same. The template itself doesn't restrict what can be done on the site; any list, library, page, or site can be created, regardless of the original template (for example, a Blank Site can be made to look like a Team Site). The main difference in the Site Definitions is that each is intended for a specific purpose as follows:

- **Team Site** This is intended to be used as a project site, where multiple team members can share documents, calendars, and so on. It includes typical project-related lists like Announcements and Tasks.
- **Blank Site** This is simply a blank site; the page layout is the same as the Team Site, but no lists or libraries are created.
- **Document Workspace** Intended to be used for document workflow where a select group of users can collaborate on a document, store related assets, review and comment, and so on. This kind of site can be created directly from Microsoft Word or Excel.
- **Wiki Site** A Wiki Site allows the creation of pages that can hold free-form content (text, pictures, and so on) linked by the user.
- **Blog** A blog (short for web log) is a free-form comment thread in which a user as the author can post comments (akin to a diary) and optionally, allow other users to contribute comments.
- **Basic Meeting Workspace** Intended to facilitate meetings by providing lists and libraries to plan the meeting, set/store agendas, providing a contact list, and so on. This kind of site can be created directly from Microsoft Outlook.
- **Blank Meeting Workspace** A blank site with a page layout that is the same as the Basic Meeting Workspace, but no lists or libraries are created.

- **Decision Meeting Workspace** Intended for meetings focused on a specific decision or outcome; libraries include Meeting Minutes and Related Tracking.
- **Social Meeting Workspace** Intended for planning and setup of a social meeting like a party or corporate gathering and includes lists like What to Bring.
- **Multipage Meeting Workspace** This provides the same features as a meeting workspace; however, it provides a tabbed page view and is usually used for meetings that can be broken up into specific subjects.

As you might have noticed, primary uses for these types are temporary with the exception of Wiki and Blog Sites (indicated by the term *Workspaces*). The point is that with limited capabilities, WSS sites are driven by a specific purpose, not by content (they can't share) or navigationally (the hierarchy cannot be changed). Projects end and Team Sites end as well, and finalized production documents should be moved to the appropriate location and the workspace deleted and so on. Now, this not to say that you cannot use WSS sites for something permanent; they are excellent for use in creating protected or consolidated resources like a client site or a department's document storage. Be aware also that aside from those mentioned here, users, designers, and developers can create their own templates to suit their needs (an example is shown at the end of this chapter).

All of the WSS Site Definitions are pretty much the same with minor changes, such as creating a list by default and so on. All have the same set of lists, libraries, and web parts available to them once they are created. That said, we'll only break down the specifics by examining the different types: WSS (Sites and Workspaces), Wiki Sites, and Blog Sites.

Web Parts Included

In WSS Site Templates, there are only eight web parts that are included out of the box; MOSS has many others and depending on the installation, you may have more (third-party templates may have others too). The defaults are

- **Content Editor Web Part** This is used to hold/display free-form content, JavaScript, and HTML code. This part is generally the one used for all free-form content.
- **Form Web Part** This is used to create user-defined HTML forms.
- **Image Web Part** This is used to display a single image.
- **Page Viewer Web Part** This is used to provide a view of another website (in an IFrame).
- **Relevant Documents** This provides a list of relevant documents (has settings to choose last modified, created, or checked out by the current user).
- **Site Users** This provides a list of the site's current users.
- **User Tasks** This provides a drop-down list of any open tasks found for the current user.
- **XML Web Part** This displays XML either entered or from a file or a website (Note that this will not display RSS).

Lists and Libraries Included

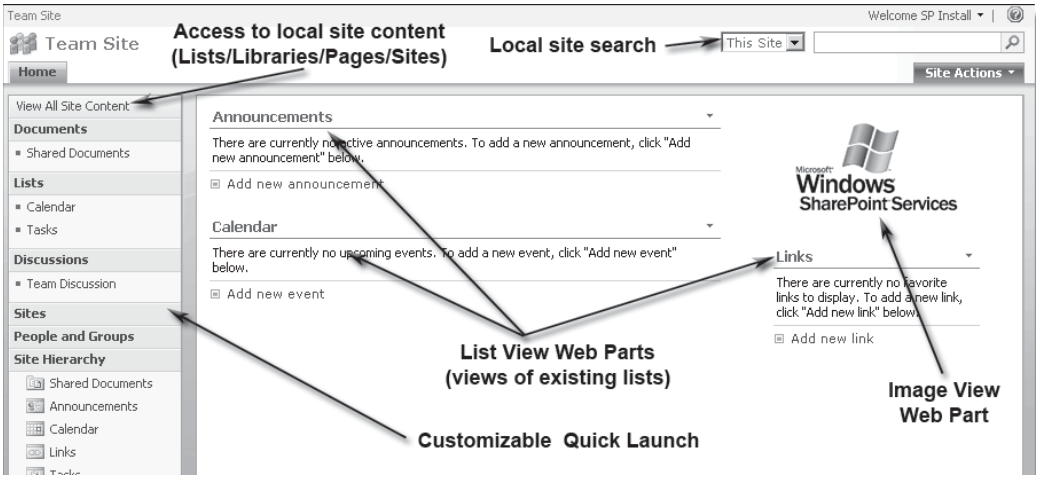
WSS Sites support several list and library types and, of course, includes the custom list type that enables any kind of list to be defined. In the different Site Templates, all of these are available and based on the template, some automatically created (the Team Site Template

creates a library called Shared Documents and several lists including Announcements, Calendar, Links, Tasks, and Team Discussion). The out-of-the-box list and library types include the following:

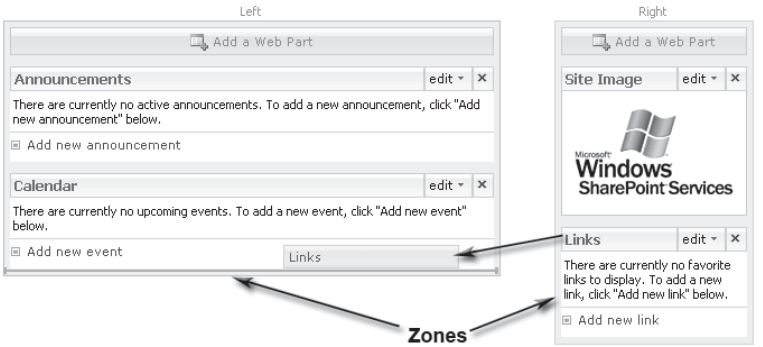
- **Libraries** Document Library, Form Library (for InfoPath), Wiki Page Library, Picture Library
- **Communications Lists** Announcements, Contacts, Discussion Board
- **Tracking Lists** Links, Calendar, Tasks, Project Tasks, Issue Tracking, Survey
- **Custom Lists** Custom List, Custom List in Datasheet View, Import Spreadsheet
- **Web Pages** Basic Page, Web Part Page, Sites, and Workspaces

WSS Site Definitions

All WSS Site Definitions have the same basic format and are built of similar components. In actual definitions, the only differences are the layout in the aspx page, the lists that are created and list views and web parts shown on the page. The heart of all of the definitions is based on the same components: an XML file that defines the site and an aspx page that dictates the layout. For an example, we'll look at the most common, a Team Site. When created, it has no content, but a few lists, a library, a few List View Web Parts, and an Image Web Part, and it looks like this:



As mentioned previously, the Team Site is based on a web part page, thus it has zones that allow web parts (either List Views or regular web parts) to be moved around on the page:



Site Definitions use an XML file called ONET.XML, which defines the site using CAML. The page layout and web part zones are from a partial aspx file. The whole thing is wrapped together with the Master Page for the site for display.

Wiki Site Definition

Wiki Sites (by default) have only a Wiki Page Library, which is a document library specifically set up to hold Wiki Pages holding the page metadata and content. The data from these items is combined with the page layout to emulate pages in the site itself. Notice that there are three options available on the Wiki display: Edit allows editing of the content (see next), History lists the history of this particular content, and Incoming Links lists all pages that are linked to this page.

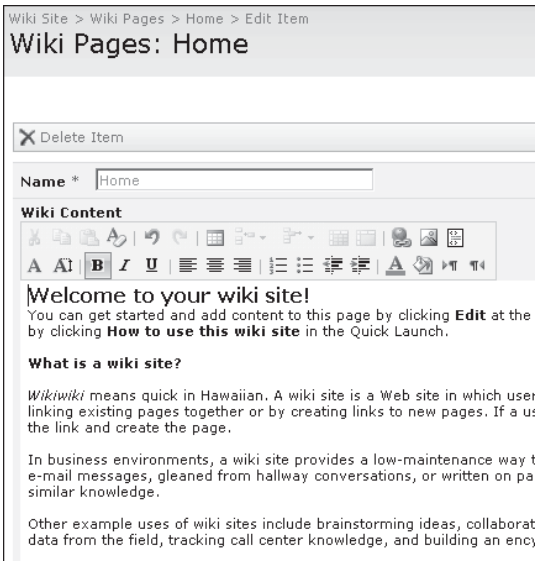
While in public use, Wiki Sites are used much the same way as blogs; users simply post random content in an ad-hoc manner. Information can be linked between pages by individual users creating a generic form of information-based navigation instead of a traditional top down. In corporate settings, Wiki Sites provide a quick way in which to record ongoing information and/or discussions. This can be useful in situations like large project meetings, where many subjects are discussed in detail but are following separate tracks. Be aware that there is no filtering of content here; users should be clearly advised on the policies regarding content, particularly in the areas of disclosure, security, and slander. In a corporate environment, information posted in Wiki Sites should be considered company property and not private to any employee.

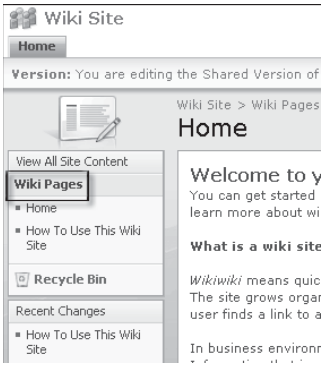
Editing and Creating Wiki Pages Wiki Pages are not the same as Basic or Web Part Pages and can only be created in a specific Wiki Page Library. One of these is automatically created with a Wiki Site (including the first page in the site that is shown when you open it; this is the one case where not only a list but predefined content is created). There are two modes to

editing a Wiki Page: to edit content, click the Edit tab, which will open the Rich Text Editor and allow editing (click to activate the control).

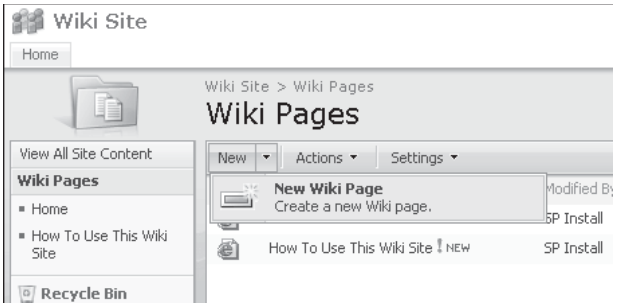
Clicking OK saves the changes. The page itself also has one web part zone on it, which is accessed by putting the page into Edit Mode. Do this by selecting Site Actions | Edit Page, and you'll see the web part zone appears at the bottom of the page.

To add a new Wiki Page, you need to go to the Wiki Page Library by clicking the link …to it on the Quick Launch (you can also get to it by clicking View All Site Content):





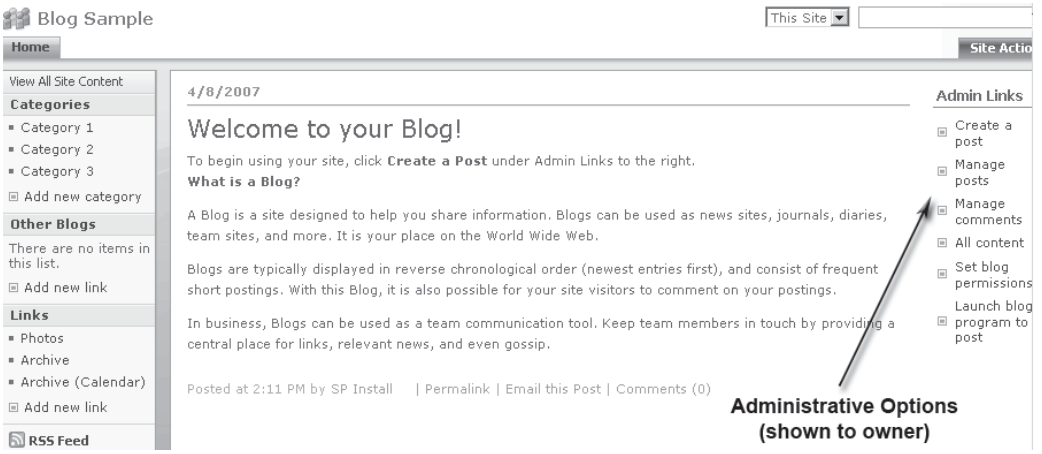
Once the library is open, you'll see all of the pages that have been created, including their content (note that item content can also be edited here instead of through the page itself). To create a page, simply click New | New Wiki Page:



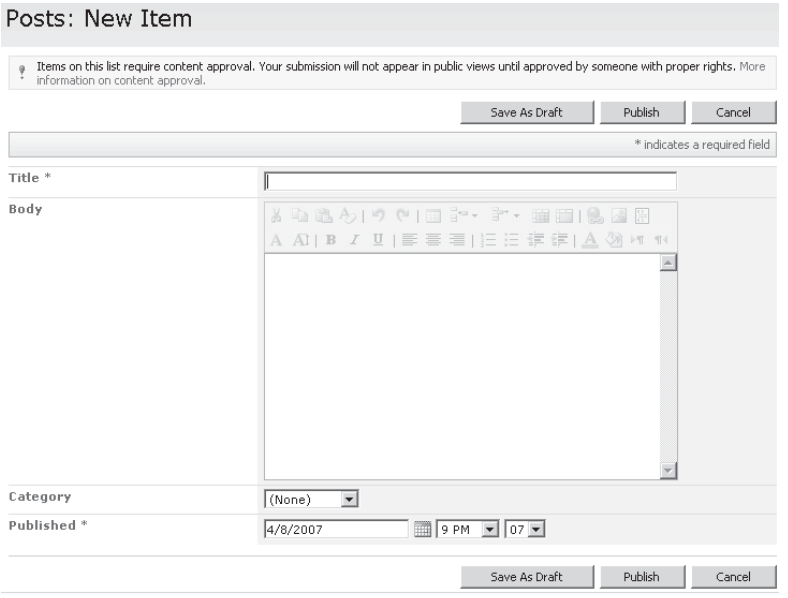
As before, the Rich Text Editor enables content to be added (click to activate). Click OK to save the page. While editing, you'll see a message showing how to link to other pages; by entering [[Page Name]] in the text (the page name must match), a link is created (for example, [[Home]] links to the Home Page). Note that this only works in the same Wiki Page Library; non-page links can be added using the Rich Text Link available on the RTE toolbar.

Blog Site Definition

Blog Sites are also a new addition to SharePoint site types. Just like those you might have seen on the web, the Blog Site provides the ability to log commentary in a diary-like format. The Blog Site has three main lists created with it that are specific to the blog: Categories, Comments, and Posts. Categories are used to categorize the posts to the blog (like a table of contents); Posts are the actual comments or thoughts of the author, whereas Comments are those added by other users about the original post. Additional lists created include Photos (a picture library), Links (a links list), and Other Blogs (also a links list). When first created, these look something like this:



Blogs are a little different from most site types; notice that the administrative options are shown on the home page (for the author of the site). It is from here that the user can manage all aspects of the blog. For example, adding a new post (a new subject) is done by clicking Create a Post:



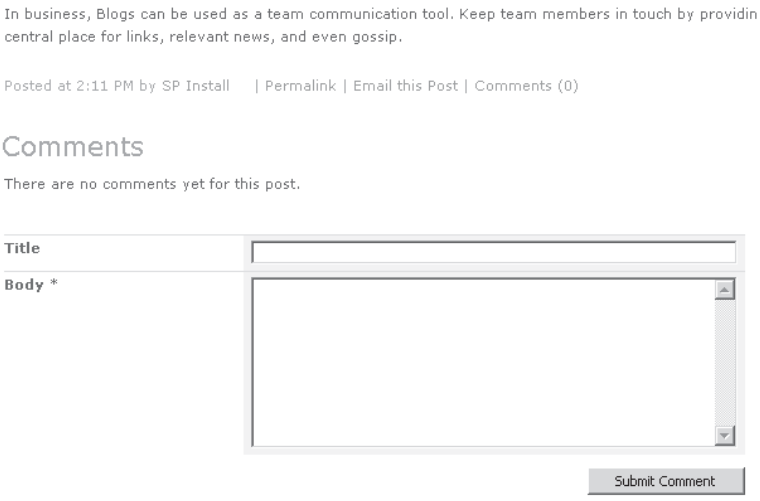
Notice that posts can be saved as drafts or published. This works similar to the way Publishing Pages do; a post can be worked on repeatedly and then published when completed. Options available (to the author) here include:

- **Create a Post** This creates a new posting as shown previously.
- **Manage Posts** This enables management (edit/delete) of existing posts.
- **Manage Comments** This enables management (edit/delete) of comments added to posts.

- **All Content** This provides access to all content in the site.
- **Set Blog Permissions** This is a link to the help for setting blog permissions, adding/removing users, and so on.
- **Launch Blog Program to Post** This opens a suitable program for creating posts, for example, Word (if no suitable program is found, the user is notified and returned to the site).

Each post has a few options as well:

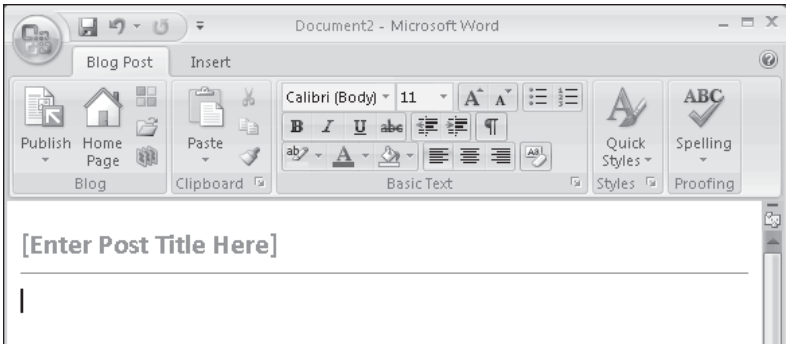
- **The By Line** This provides a link to the author's profile information.
- **Permalink** By definition, this is a URL that points to a specific blog entry, even after the entry has been archived; when only a single page, this opens a new comment entry.
- **Email This Post** This opens the local e-mail program (Outlook, Outlook Express) and automatically includes the link to the post (for mailing to others).
- **Comments ( )** This enables the user to add comments to the post:



Blogs also provide RSS feeds, just like all SharePoint lists. These feeds enable users to use RSS Readers to view the Blog entries on another site. Be aware, however, that by default, RSS Feed Readers cannot read RSS feeds on their own site, because access to RSS in SharePoint requires authentication. My company provides a solution for this at www .sterling-consulting.com.

Creating Posts from an Application Obviously, working with a blog within the Rich Text Editor is not really optimal when you are entering a lot of text or need formatting. A program like Word has much more capability because it includes spell check (pretty important sometimes) and other tools not available in the RTE. To simplify this, the Blog Site has the option to Launch Blog Program to Post. This option will invoke whatever client program is available for editing. The difference here is that you are presented with a document template for the posting; when saved, it is added to the blog and retains formatting.

For a first-time user, a blog account must be created. When posting for the first time, a dialog box appears that prompts you to create the account (the user can upload their picture as well). Once this is created, the user is also warned if the connection is not secured. Assuming the user clicks OK, the Posting Template opens directly in Word:



The format is already set up based on the Posting Document Template. You can format as you like, but stick with the basics for websites (use fonts such as Arial, Verdana, and so on). Note that to use Launch or publish from Word, you must have owner rights (contributor rights are not enough).

Microsoft Office SharePoint Server Sites

MOSS sites are the portal sites that are typically used for a high volume of users and are used for organization-wide collaboration. This includes closed corporate settings, such as intranets and extranets, as well as both closed and public publishing style sites for Internet use. While based on WSS sites, MOSS sites have additional enterprise features that include cross-site search, content sharing between sites, common users, community content, content targeting, and so on.

On the collaboration side, MOSS, of course, has all the functionality of WSS, though MOSS does have the added ability to target content by audiences, groups, and users. On the publishing side, MOSS includes Blank and Web Part Pages but for content management, has Publishing Pages which are defined page layouts that can be used anywhere. Combined with document conversion and workflow capabilities, this replaces all of the features that were available in Microsoft Content Management Server. Collaboration and publishing can be used side by side.

From a product point of view, MOSS can provide nearly any type of portal site used for corporate intranet sites, eCommerce sites, corporate presence sites, and so on. Collaboration tools such as document storage, lists, calendars, and so on, can be commingled with content publishing, page management, and content sharing.

MOSS sites are, of course, based on the WSS platform, but are much more flexible in terms of topology and site organization. MOSS does host WSS sites underneath it, which makes those sites searchable and able to use the same user base.

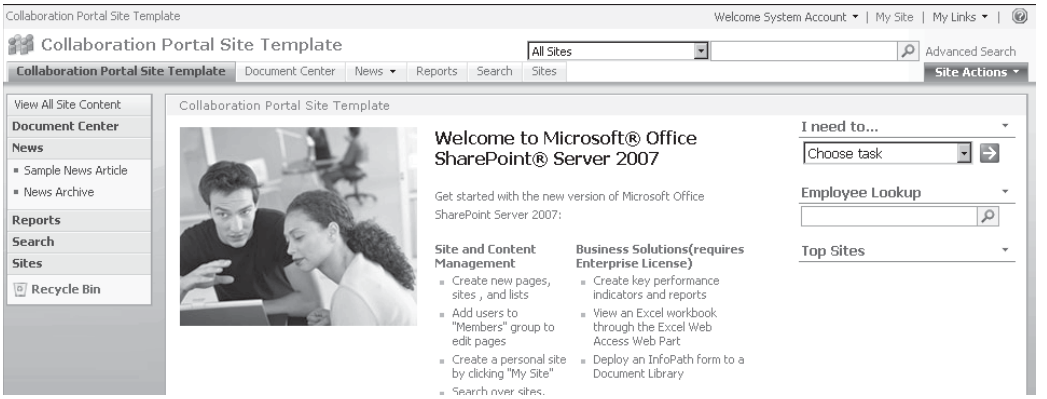
MOSS also features the My Site capability. This is a site specifically designed for individual users providing both a public view (as a business card/contact page) and a private view (much like a Team Site). This is augmented by a number of personalized web parts that are customized to the user (such as My Links) and others that provide access to Outlook Web Access (OWA), so that users can link their My Sites with their Email, Calendar, and Tasks.

Portal Site Definitions

There are only two portal site definitions that are intended to be top-level sites. The collaboration type, used for intranet-type sites, can only be created at the top level. Publishing sites can be either a root or a sub site, for example, a Publishing Site for knowledge management under a Corporate Collaboration Site.

Collaboration Portal

The Collaboration Portal Site is designed to fit the typical kind of intranet layout used in a corporate environment and provides a basic layout including a Home Page, News, Document Center, Search Center, and Site Directory. In addition, it includes some default content to help users get started. When created, this looks like this:



Notice that there is already some content included: the picture, some sample news items, and so on. It's more than just a one-page site and includes sub sites. While this all looks great, remember that it is simply a template and intended as a starting point. It is by no means fixed and everything can be changed, new items can be added, and with the exception of the home page, deleted as well. I mention this as it is often a contentious point; people see this and think it is fixed or worse, get hung up on the site names used, not realizing they just represent a suggested layout.

The Collaboration Portal provides the most presets in terms of collaboration features, including all of the publishing and workflow features. The enterprise sites (like Search Center) are used to set up a default topology and navigation (modifiable at any time). The top-level navigation can be based on sub sites or simply on pages. An example of this is the News Site; this in the top navigation but also includes articles (sample pages) in that site in the drop-down menu. Note that when sub sites are created, the top-level navigation can be carried down to the lower site.

Publishing Portal/Publishing Portal with Workflow

The Publishing Portal type is intended to provide the kind of sites that used to be built with Microsoft Content Management Server (MCMS) based on sites (Channels) and pages (Content Pages). Publishing is typically used for public facing sites, anonymous access sites, corporate sites, knowledge management, and other sites that do not necessarily use collaboration (interactivity). This site type is designated either a Publishing Portal or a Publishing Portal

with Workflow; the only difference is that *with workflow* means parallel workflows are already attached to all items and assigned to a generic group (Approvers). This simply speeds up the process since users are simply added to the group and the workflows do not have to be created. Workflow is usually disabled in static sites or content-push destinations (where publishing is done from a different or central system). When created, the Publishing Portal looks like this:



The look here is more web-like than the application feel of the Collaboration Portal. Publishing sites are intended to be used primarily for content (like an eCommerce site) and not collaboration (like an intranet site). The site still needs features offered for search and personalization (targeting), but the level of interactivity is limited; users generally are not contributing to the site.

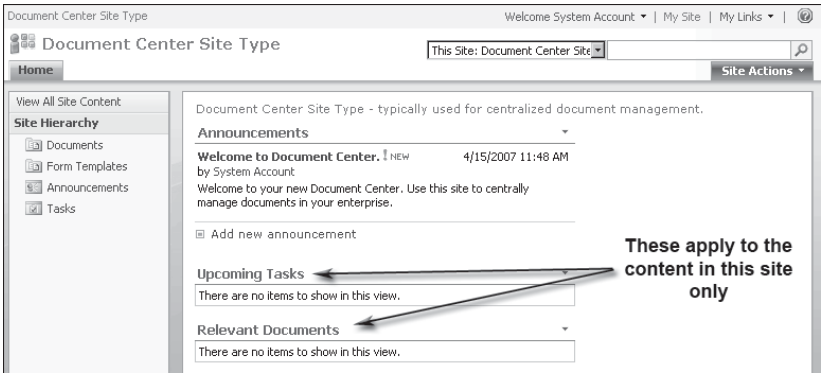
Enterprise Site Definitions

In addition to the portal sites, MOSS provides enterprise sites that are used for specific purposes. Most of these can be top-level sites, but are intended to complement the portal topology; it would be unusual to use them on their own (the exception is the My Site Host; although available here, it is only used in conjunction with Shared Service Providers).

Document Center

The Document Center Site is intended to be a centralized location for documents shared across the organization. While any site supports document storage, the purpose of this site is as a repository for long term storage, knowledge management, and high volumes. As a separate site, it can be backed up independently since it has its own content database and can even be expanded or moved if needed. This site type can be used either as a top-level site (typical when used to support a wide array of sites as a central store) or as a sub site in a portal environment, providing centralized (site-specific) storage. The only difference from actual use is search. As a stand-alone (top-level) site, search is specifically based on the content of that site only. To incorporate the content search with another site requires adding a search scope. In a portal environment, the content is also included with the All Sites search scope.

Like other sites, this site type also creates a few out-of-the-box lists, adds web parts to the display, and looks something like this (top-level site shown):



When created, this site type automatically defines the following:

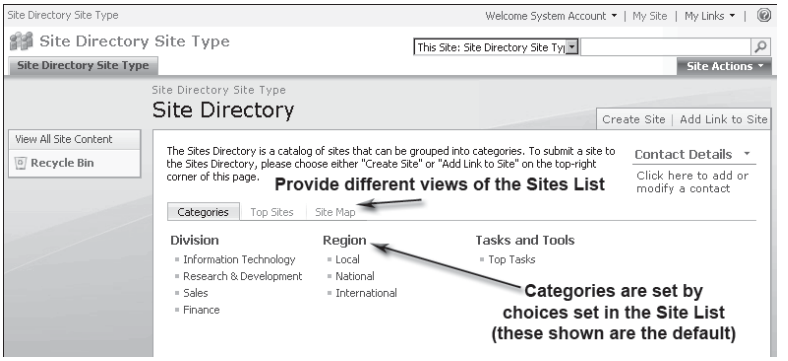
- **Documents** This is a document library to hold documents; only one is created by default and it is expected that several more will be created.
- **Form Templates** This is a form library to hold any InfoPath forms used in the site (a component in document management, forms can kick off various workflows in the site).
- **Announcements** This is for providing notices to users of the site and content.
- **Tasks** This is a Task List for tasks related to documents in the site.

This site template includes two web parts (in addition to the Announcements List View) that are displayed on the main page:

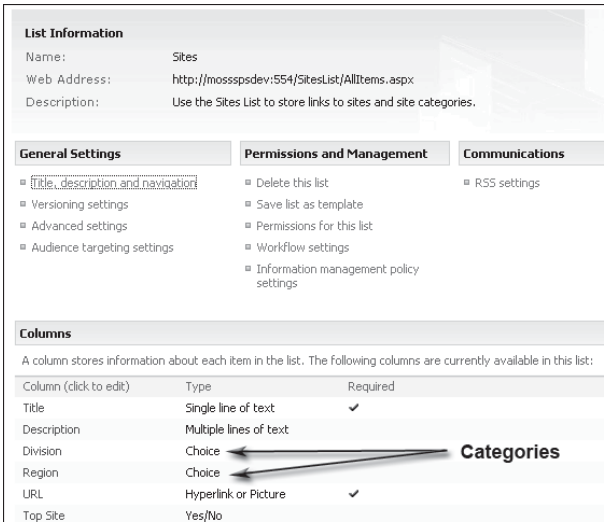
- **Upcoming Tasks** This displays a list of any tasks open for the user viewing the site (based on tasks in the Tasks List in this site); options for this part include:
  - **Display Item Link column** This enables/disables displaying a link to the task in the display.
  - **Maximum Number of Items Shown** This sets the maximum number of items that will be shown.
- **Relevant Documents** This displays a summary of relevant documents to the current (viewing) user in the site; options for this part include:
  - **Include Documents Last Modified by Me** If enabled, this includes any documents recently created by the viewing user.
  - **Include Documents Created By Me** If enabled, this includes any documents recently created by the viewing user.
  - **Included Documents Checked Out to Me** If enabled, this includes any documents checked out to the viewing user.
  - **Display a Link to the Containing Folder or List** This enables/disables displaying a link to the document or folder in the display.
  - **Maximum Number of Items Shown** This sets the maximum number of items that will be shown.

Site Directory

The Site Directory Template provides a few lists and views that are used to organize the sub sites created beneath it (and others added to the Sites List). This includes a Categorized List, Site Map, and Top Sites (highlighted sites) Views; all read from the Sites list. The Site Directory is usually a sub site of a Portal site, and there can be any number of them used. Any sites created from the Site Directory become sub sites of the directory. The Sites List (one per directory) is a list of site names and URLs that can optionally include entries for sites created beneath it. The tabbed views simply provide different views of the list. The Sites list is just a list; if a listed site is deleted, it is not removed; this must be done manually in the list in the site. When first created, this site looks like this (notice the tab selections):



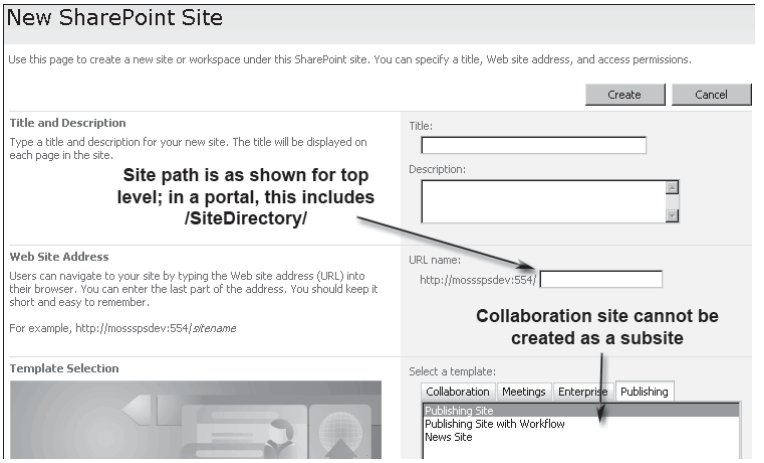
The first view shown is the Category View, which lists sites by category. Categories are simply choice fields defined in the Site List, and you can view this from the site by clicking View All Site Content. Next, click on the Sites List to open it. Then, select List Settings. Under the Columns group, you will see the categories (choice fields) that are set (the choice fields are simply placeholders and can be changed via the web part or directly in the list):



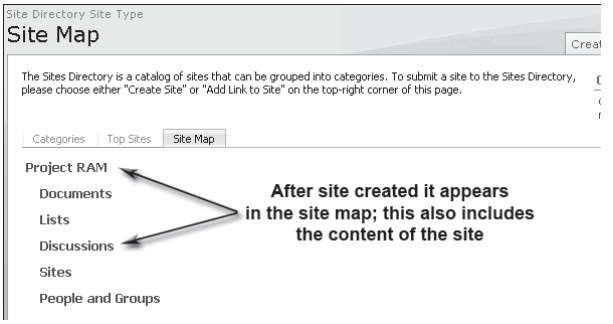
If you viewed the list, you may have noticed there is only one entry; this is a default link for users to use to create their My Site. If you go back to the Site Directory home page and click on either the Top Sites or Site Map, you will see they are blank since no sites have been added.

Creating Sites in the Site Directory The Site Directory provides two links: Add a Link to Site, which allows adding an entry to the list (for any site or URL) and Create Site, which starts the same site creation process as creating a top-level site. (Remember, you can do this from Central Administration or from Site Actions | Create Site; the difference is that you are given the option to add the site to the Sites List). All site types (except the Collaboration

Portal) are available here. To create a site from a Site Directory or to add a link to an existing site, simply click the appropriate link at the top of the page (note the link for site creation does not work when the Site Directory is a top-level site). On the Create New Site Page, any site type (except collaboration) is available:



Be aware that the same rules for creating a site apply, including the security settings of inheritance. If Use Same Permissions from The Parent Site is selected, the new site will assume the same security as the Site Directory site (its parent). On this same page, you also have the option to list the site and in what categories to list it (or not). Once the site is created and listed in the Sites List, it appears in the Directory Site Map:

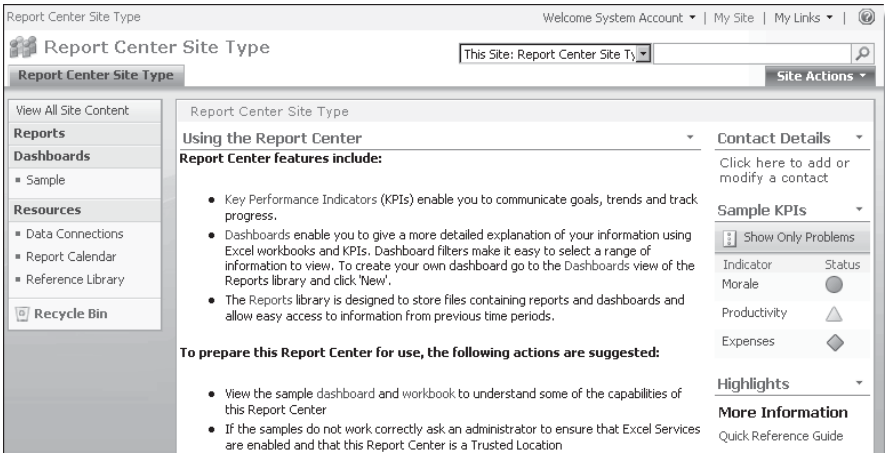


Report Center

The Report Center Site Template is designed to provide a dashboard-like site to be used for report storage, KPI Lists, and KPI Indicators. This site definition creates a custom Report Calendar List and special libraries including a Report Library, a Data Connection Library (used to store data connections used in KPI Lists), and a Reference Library (intended to provide documents and information about the Report Center and what information it provides). The purpose of the dashboard style is to provide a one-page summary of Reports, Indicators, and so on.

Personally, I don't think this is aptly named. Report Center should be where you would expect to find reports, not where you'd find KPIs. Perhaps a better name would be a Business Status and Reporting Center Dashboard, providing a central place to view enterprise information both internally and externally. Since information here can be filtered on a per-user basis, this makes it ideal for all kinds of applications. An executive snapshot can provide bottom-line summary information for management, an operations management site could provide a one-page summary of the operational statistics, and it can even be used for user applications, such as a dashboard for customer service representatives. As with all SharePoint

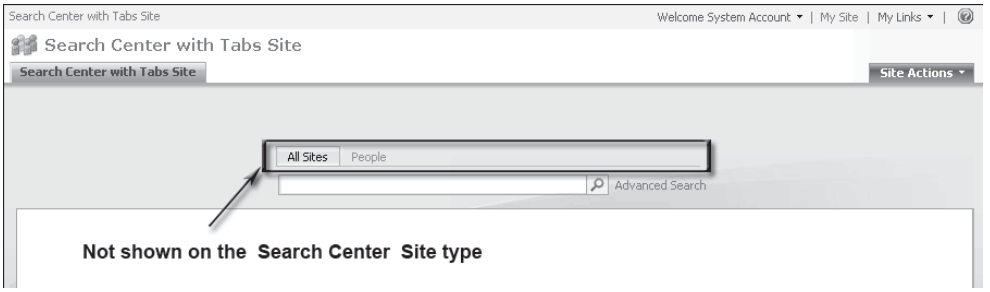
sites, this site type supports policies for storage, retention, and security. The Reports Center looks like this:



Notice that a lot of information appears on the page and even includes a KPI indicator; this displays some filler content included simply to demonstrate how it works (it does have to be removed when configuring the site). All of the settings for KPIs and data connections require use of the Business Data Catalog and Excel Web Services (EWS), covered in the online Web Chapter 1 on the BDC.

Search Center/Search Center with Tabs

The Search Center and Search Center with Tabs Site Definitions are basically the same; the only difference is that the Tabs Site provides a tab bar, based on the search scopes available. The Search Center with Tabs looks like this:



The only difference is if the tabs are shown. The main purpose for this site is to provide a central location to search, like your own Live Search or Google Search Center (without all the advertising!). In actual operation, this search works like the Advanced Search Page (without criteria selection) and is easily customized with the Search Web Parts that make up the page.

My Site Host

The My Site Host Site Type is only used when creating a new Shared Services Provider and is specifically for hosting My Sites, though in fact, it has no pages or site. While shown in the list of site types, this type is created in association with an SSP and is never shown as a Portal or as a sub site.

Records Center and Information Management Policies

One of SharePoint's key components in content management is the way documents are handled in terms of security and content types. This, combined with workflow controls, represents the core of SharePoint's document management (or, as I prefer to call it, Document Control). SharePoint provides two extensions for management: the Records Center and Information Management Policies. The Records Center is used to maintain permanent records, including items to be tracked, expired, periodically reviewed, and so on, in order to keep in line with compliance, tax, and legal issues. These come in the form of documents, e-mails, spreadsheets, and databases. Information Management Policies enable setting governance on records (and other) information in the Records Center, including security, compliance, access control, and auditing. SharePoint includes:

- **Information Management Policies (IMP)** These provide individual document or content- type governance and control; a handful of policy types are provided out of the box and can be extended through development.
- **Records Center Site Type** This provides a centralized Records Repository Site that is a catch-all for all kinds of lists, documents, e-mails, and similar that must be retained or otherwise tracked.
- **Auditing and Audit Reporting** Auditing in SharePoint can be done at the document and content level and can selectively record activities such as the check-in of a document, editing an item, approving an item, and so on. Since this is available at the Site Collection level, auditing applied by content type is tracked throughout all sites, but is easily managed in a central location. Out of the box, SharePoint provides several auditing reports and the information can be exported to other systems.

Information Management Policies

As it sounds, Information Management Policies (IMP) are policies that govern the way specific items are managed, including retention, security, and even workflow. This is vital to how documents and records are managed because it is often the document type that classifies its worth. In document management terms, this means the way items themselves are handled in terms of security, compliance, or other governance. This goes well beyond simple document control and brings SharePoint in line with guidelines and requirements set forth under Sarbanes-Oxley, DoD, HIPPA, NASD, and more. The overall purpose of the policy is to provide uniformity across documents to meet compliance while keeping the maintenance effort to a minimum. SharePoint IMPs include the most common types out of the box:

- **Labels** Labels can be designated for a particular document type that must be provided by the author and can be set as permanent or changeable; these labels are an add-on to the metadata of the item itself.
- **Auditing** This creates heavy audit tracking on specific items and includes all the actions taken (and by whom) on it. Auditing also features built-in reporting capabilities through Audit Log Reports (available through Site Actions | Site Settings | Manage All Site Settings).
- **Expiration** Removal of content automatically based on a date or number of days since modified or created; this can be automatic or can even trigger workflow processes to handle migration to archive, prior notification of removal, and so on.
- **Barcodes** Automatic document tagging either with user-defined labels or using automatic bar coding.

Since these may not fit everyone's needs, additional custom policy features can be built. An example I've mentioned is in the legal world; in case work, documents are assigned Bates numbers to track them, and a custom policy might be to assign this number automatically. Information Management Policies and Document Policy features are used throughout the publishing and collaboration processes in addition to records. Policies are created at the Site Collection (top-level site) and can be applied to any document library in the site or sub sites below it. IMP features within SharePoint include:

- **Document Conversion Processing** This is used to convert documents and records from one form to another (like Word to HTML); this is used in the publishing process when converting from a document type to a SharePoint Publishing Page and also when converting Records from their original form into one more suitable for storage.
- **Document Policies** This is used to manage documents by retention, for setting review dates and for setting expiration dates, and can be used to manage all document types. Policies also include many of the features required for true records management such as Sarbanes-Oxley compliance and other regulatory needs including auditing. Policies can be used to provide Document Labels, Document Bar Coding, and Use Auditing, which provides a comprehensive history of all actions performed on a given document.
- **The Central Records Management Site (mentioned previously)** This is used to create an enterprise-level records management site used to store records (documents); these can be routed based on type into the appropriate libraries for storage.
- **Line Item Security** All items, which includes documents, can be secured at the item level. This allows documents and records with differing security levels to be commingled in the same library (this is quite a jump from the previous version!).

Custom Information Management Policies While the out-of-the-box policies likely cover 90 percent, there are many cases where they are not enough or do not fit into a custom requirement. Like everything else in SharePoint, IMPs are extensible and can be created for any purpose. For example, if an organization deals with sensitive material and/or otherwise proprietary information that cannot be exposed, they could create a custom IMP that can open the document and scan the contents looking for specific phrases or words. If such words were found, it could either block or even alter the contents of the document.

Building custom IMPs is akin to the process of building a custom Document Converter. These are built as SharePoint Solutions (aka Features) that can be enabled or disabled once installed. Like a document converter, web part, and receiver, IMPs are based on a class, some controls, and a few XML files that define it for SharePoint:

- A class assembly that implements the IPolicyFeature interface (this must be in the Global Assembly Cache to work).
- Two (optional) user controls to provide users the ability to set options for the policy: one is the Global Configuration Page (goes into the template\admin folder), control (ascx), and the code behind, and the other is the Configuration Page (goes into the template\layouts folder), control, and code behind.
- A Manifest file that defines the assembly (and SafeControl entry).
- The Feature and Element Manifest files needed to register the Feature in SharePoint.

Like a document converter, most of the work is done for you. The actual assembly is invoked (as an Event Receiver) when the policy needs to be applied. In the code, the IPolicyFeature object model provides access to all of the attributes of the document and allows complete manipulation of the document, the item that holds it (and the metadata), and so on. For example, when a document is uploaded as a specific document or content type, any associated policies are applied. This means that the event is passed to the assembly, and inside there, the actual policy can be enforced and necessary business rules applied. IMPs are Solutions like other add-ons to SharePoint. You can get more information in Chapter 17.

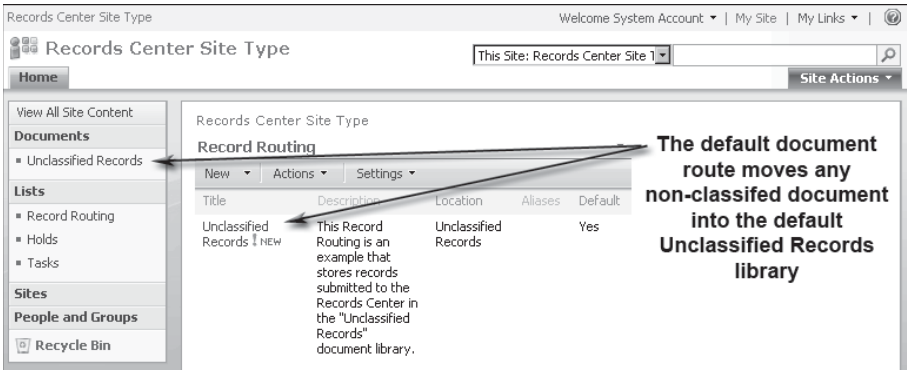
Records Center

The Records Center Site is a special site type that is designed to handle incoming records (documents) from various sources, including sites, the user, and through web services from external applications. In terms of the overall Information Architecture, Governance, and Records Storage, this site is intended to be the catch-all for the organization (or organizational level) as a central records repository. For items and documents that need to be kept based on policies, items are sent to the site, which in turn uses an internal routing list to automatically store the content in the correct list within the site.

The Records Center Site handles the work of storing documents (through routing), which is a real boon to users and external applications since posting can be done blindly. The Information Management Policies (IMP) routing dictates how records are handled in the site (and where they go). IMPs are created at the Site Collection level and can be used by any libraries within the site or sub sites. Since assigned at the library and library level, users again don't have to be concerned about what's going on behind the scenes.

Though it is an enterprise site type, I've singled this one out because it is a core component of Records Management and Information Management Policies used within SharePoint. In most cases, the Records Center Site will be the only place that real policies about documents have to be created, since day-to-day content usually has little or no restrictions. Because all of the components to set up a policy and other related settings are the same whether used in a Records Center Site or in any other site, I've consolidated the process here so as to avoid duplication.

The difference between using IMP in other site types and the Records Center is the routing feature. When items are sent or stored in a Records Center Site, the content type is matched against a routing list; this routing dictates where the document or item will be stored. When created, this Records Center Site looks like this:

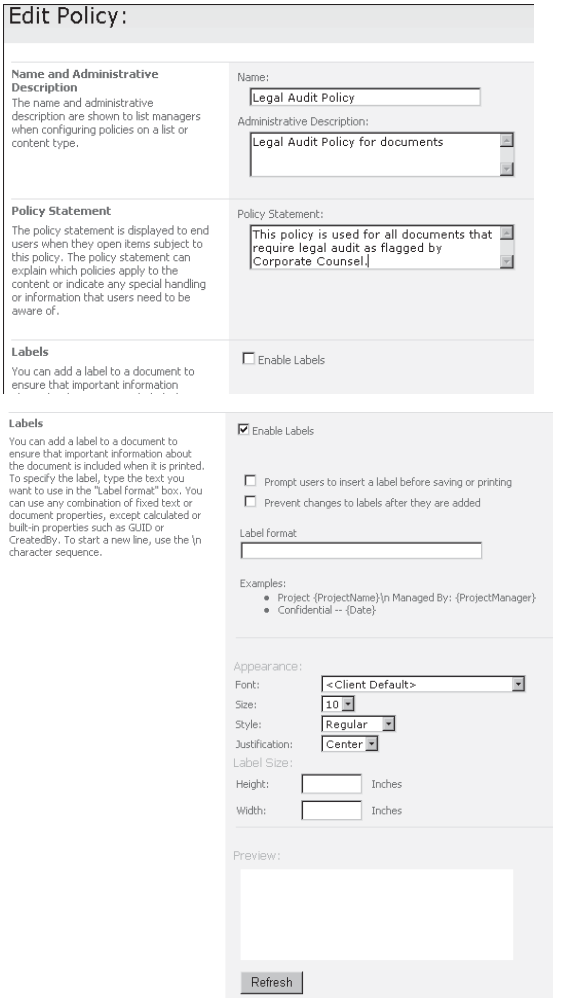


As you can see, there is a Record Routing List displayed that includes the default routing for unclassified documents (those that do not have a specific route or policy associated with them). This is set as the default catch-all and also includes a document library named appropriately for storage of these documents.

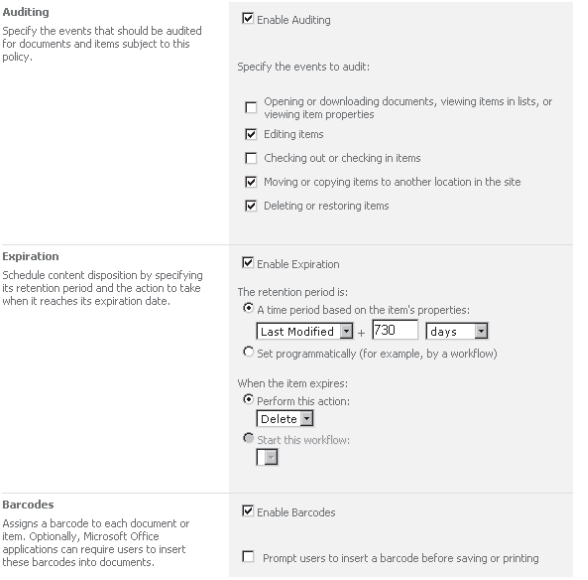
So how do records get to the site? There are several ways actually—documents and items can be simply uploaded to the site like any other; they can be saved directly to the site from Microsoft Office; they can be e-mailed to a library; and they can be sent there directly from a SharePoint site. Sending directly from a site is most common, because much like document conversions, users can hand-select an item to push to the Records Center (you'll see this in action next).

Creating a New Information Management Policy Routing of documents can be driven by storage factors such as size and type; however, they are mainly based on the policies that apply to the specific type of document. The most common factors are security, retention, and expiration of a record. Creating and viewing Site Collection Policies is done through the Sites Site Settings page; under the Site Collection Administration group, you'll find the links to Site Collection Audit Settings, Audit Log Reports, and Site Collection Policies. Click Site Collection Policies to view the policies defined for the site. From here, new policies can be created and imported. Since there are none to start with, the first step is to click Create. When the Edit Policy page appears, you define the name, title, and description (for users).

To view options, you have to click the Enable check box to expand each option. The first option here is for Labels. Labels are a way to tag a document with additional information beyond metadata (labels are specific to the document). Option settings available are as shown to the right.



Labels are used to tag documents related to something specific such as a law case, project file, and so on. Note that while I've shown it here, in this example, I will not specify a label (this was enabled to show the options). The following policies that can be enabled include Auditing and Expiration, which I will set, along with enabling Barcodes.



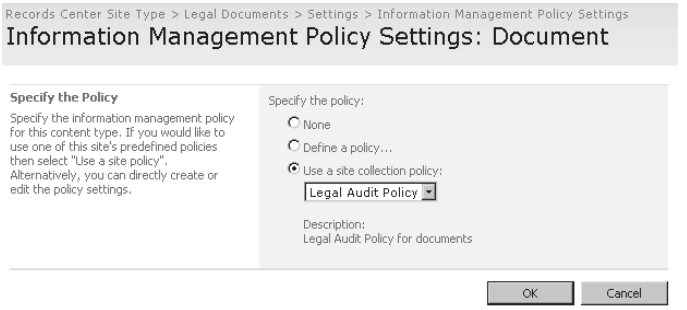
The policies shown are the defaults provided by SharePoint (when additional policies are custom developed, they will appear here as well). Once policies are created, they can be exported so they can be imported in other sites. When viewing a policy, there is an Export button that will extract the definition and save it as a simple XML file. These XML files can also be created by hand to perform custom actions not available with the default policies. For example, the policy above enabled the Auditing and Expiration policies and set some properties (conditions) that are applied. The exported XML file looks like this (note the ID is unique to this policy):

```
<p:Policy xmlns:p=office.server.policy local=false id=a45d5363-11df-45ed-9b43-da3b11bfe0a4>
<p:Name>Legal Audit Policy</p:Name>
<p:Description>Legal Audit Policy for documents</p:Description>
<p:Statement>This policy is used for all documents that require legal audit as
flagged by Corporate Counsel.</p:Statement>
<p:PolicyItems>
  <p:PolicyItem featureId=Microsoft.Office.RecordsManagement.PolicyFeatures.PolicyAudit>
    <p:Name>Auditing</p:Name>
    <p:Description>Audits user actions on documents and list items to the Audit
Log.</p:Description>
    <p:CustomData><Audit><Update /><MoveCopy /><DeleteRestore
/></Audit></p:CustomData>
  </p:PolicyItem>
  <p:PolicyItem featureId=Microsoft.Office.RecordsManagement.PolicyFeatures.Expiration>
    <p:Name>Expiration</p:Name>
    <p:Description>Automatic scheduling of content for processing, and expiry
of content that has reached its due date.</p:Description>
    <p:CustomData>
    <data>
      <formula
id=Microsoft.Office.RecordsManagement.PolicyFeatures.Expiration.Formula.BuiltIn>
        <number>730</number>
        <property>Modified</property>
        <period>days</period>
      </formula>
      <action type=action
```

```
id=Microsoft.Office.RecordsManagement.PolicyFeatures.Expiration.Action.MoveToRecycleBin />
    </data>
    </p:CustomData>
  </p:PolicyItem>
</p:PolicyItems>
</p:Policy>
```

In the XML form, this policy can easily be applied to other sites simply by importing. Since the ID of the feature is a randomly assigned Global Unique Identifier (GUID), it ensures that there will be no conflicts with other policies when loaded. Note also, policies and libraries must be created before routing can be added.

Applying a Policy to a Library Once a policy has been created, it can be used anywhere within the Site Collection, such as a document library. From a library itself, the Advanced Settings can be used to set the policies that are active for it. The first step is to create a new (or navigate to a) document library. To create one, navigate to the site and select View All Site Content and then click the Create link to open the Create page. Under the Libraries group, click Document Library; in this example, I'm creating a library called Legal Documents. Once the library is created and opened, select Document Library Settings from the menu to open the Library Settings Page. On this page, you'll see the Information Management Policy Settings link. Click on this link to open it. From here, polices can be disabled, created, or a Site Collection Policy can be applied like the one created previously (do not create a policy at this time, instead use the Site Content Type method explained below):



If you are creating this library in the Records Center, it will be routed by its library type. For example, Legal Documents in an organization would most likely be segregated into several libraries, with various policies like Legal Documents for Attorney Review and Legal Documents for Retention.

Applying a Policy to a Content Type While assigning a policy to an individual document library can be handy, the better place is to assign them to Site Content Types. This enables a policy to be set one time and used in any library across the site. If something changes in the policy or content type, it can be immediately applied to all the libraries that use it. This saves time and effort since policies do change and new compliance regulations are created every day. This also provides a side benefit; while good for policy, it also allows management of more than one type of content and policy in the same library. It's a much better approach to organize information by architecture instead of document type.

As an example, the policy created previously applied to legal documents; typically, these would be of several content types based on the kind of document (contract versus legal notice) and thus, have their own unique metadata. To keep this simple, you'll simply create a single content type called Legal Contracts. This process is covered under Content Management so I'll be brief:

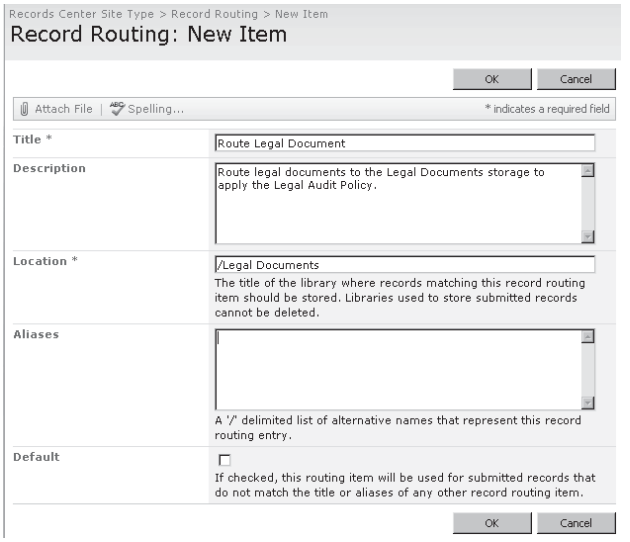
1. To start off, open the Site Collection top-level site, and then select Site Actions | Site Settings | Manage All Site Settings (in a MOSS portal site) or simply Site Actions | Site Settings in the Records Center Site.
2. On the Site Settings Page, under the Galleries Group, click Site Content Types.
3. When the Site Content Type Gallery Page appears, click the Create link.
4. On the New Site Content Type Page, enter the name as **Legal Contracts** and its description.
5. Set the Select Parent Content Type drop-down menu to Document Content Types.
6. Set the Parent Content Type to Document.
7. Select New Group In and enter a new name of **Legal Document Types**.
8. Click OK to save the new type.
9. On the Site Content Type: Legal Contracts Page, under the Settings Group, click the Information Management Policy Settings link.
10. Under Specify the Policy, click Use a Site Collection Policy.
11. Select Legal Audit Policy from the drop-down menu and click OK to save; the policy is now associated with the Site Content Type.

To add the content type to the library, return to the site and open it. When opened, select Document Library Settings from the menu. Then on the Settings Page, click Advanced Settings and then make sure Yes is checked for Allow Management of Content Types. Click OK to save.

When you return to the page, under the Content Types section (which should now be shown), click Add From Existing Site Content Types. When the Add Content Types: Legal Documents Page opens, in the Select Content Types drop-down menu, select Legal Document Types (the new group created previously). When the list is shown, click Legal Contracts on the left and then the Add button, followed by OK. Return to the list and verify that the Site Content Type has been added by selecting New from the menu. The drop-down menu should indicate the options: Document, Legal Contracts (the new Site Content Type), and Folder (folder only shows if enabled).

Creating a New Route The main purpose of the Records Center is provide a quick way to redirect records to where they need to go, based on the type of document and policy to be applied. In the Records Center, this is accomplished by a custom list called Record Routing. The entries in this list are used to match record types to the actual libraries that store them. Before a route can be created, a library must be created as the destination for the route. The actual policies that apply to that specific document type are set on the library itself, so the routing is not concerned about the actual policy, simply the routing name to use. In the previous section, a library was created and a policy assigned so a route can be created for it.

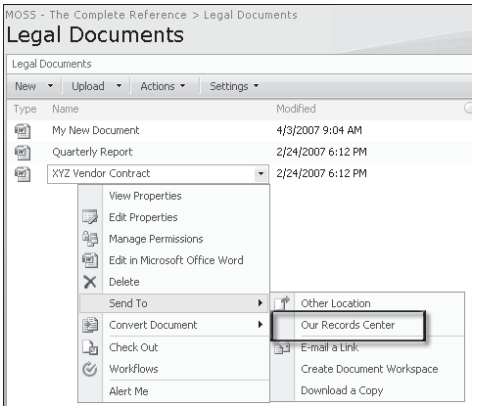
Creating a route is done by simply clicking New Item on the Record Routing menu bar. This opens the New Item Page (as shown, a new route has been defined to point to the Legal Documents library):



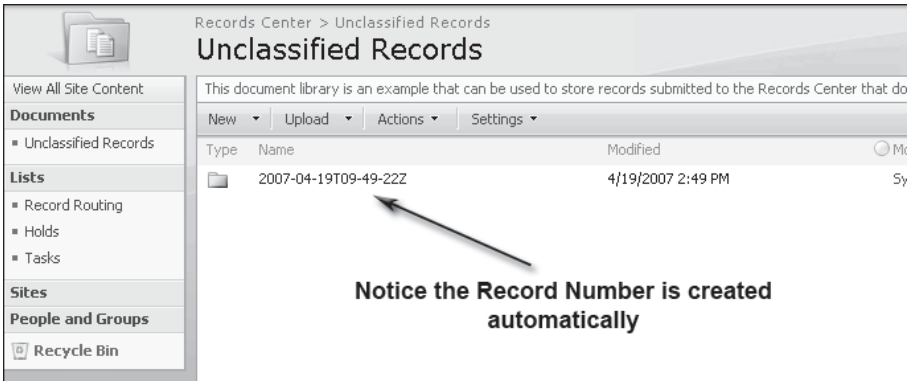
Configuring and Using the Records Center

As you have seen, the Records Center can be used in a stand-alone mode or as a sub site to a portal. You've also seen that you can apply Information Management Policies at the list/library level and assign them to content types. The Record Routing handles redirecting when you upload or add an item based on the IMP. So at this point, you might be wondering how a Records Center is used with other sites. Quite simply, this is done by identifying a Records Center Site for the entire SharePoint instance. There can only be one default for a farm.

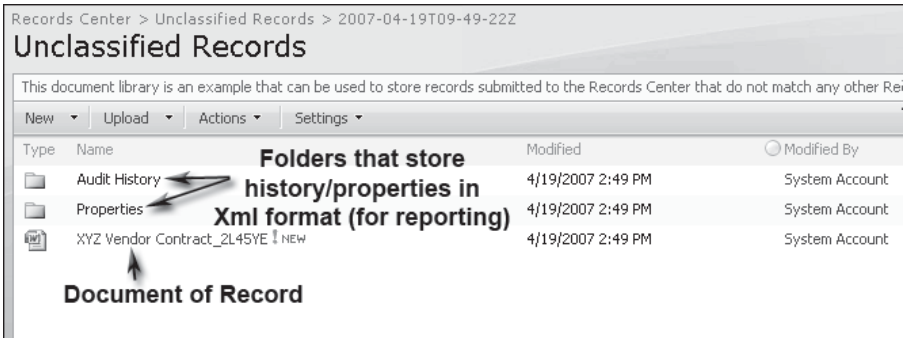
Setting the Records Center Site for a farm is done through SharePoint 3.0 Central Administrations Application Management Page. On this page, under the External Service Connections Group, is a link to Records Center. Clicking this link allows enabling/disabling and specifying the URL to the site. The URL to use here is to a SharePoint Web Control called officialfile.asmx. This is in a special folder in SharePoint Hive (in the ISAPI folder), defined as a virtual directory in IIS as _vti_bin. The format of the full URL to the Records Center connection is http://<server>:<port>/_vti_bin/officialfile.asmx. Once this has been set, this becomes available to all sites in the application. From inside a site, a user can send an item to the records center by selecting it from the item drop-down menu as shown (note the name shown is the one that was set in the Application Maintenance Page).



Once this has been selected, the item is sent to the Records Center. Assuming the connection is good, it returns a success screen (if there are any problems, the user is notified). On the Records Center side, it compares the document against content types set up in the Record Route list. If a matching type is found, the item is sent to the appropriate library; if not, it is sent to the Unclassified Library:



Viewing a Record All records are stored in the same way regardless of how they are classified. Opening the previous example shows the contents of the record:



As you can see, the document of record and the folders for holding XML files contain Audit and Properties information; these are specifically set up for reporting. For example, the Properties XML file in this example looks like so:

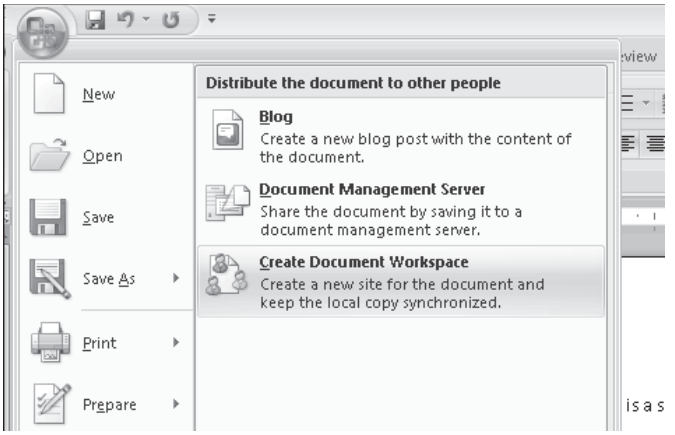
```
- <RecordsRepositorySubmission>
  - <Properties>
    - <Property>
        <Name>ContentTypeId</Name>
        <Value>0x010100575418FB71C14B49954D0B23B8037BE2</Value>
        <Type>ContentTypeId</Type>
      </Property>
    - <Property>
        <Name>ContentType</Name>
        <Value>Document</Value>
        <Type>Choice</Type>
      </Property>
    - <Property>
        <Name>_ModerationComments</Name>
        <Value />
        <Type>Note</Type>
      </Property>
    - <Property>
        <Name>FileLeafRef</Name>
        <Value>XYZ Vendor Contract.docx</Value>
        <Type>File</Type>
```

Creating Sites Using Microsoft Office

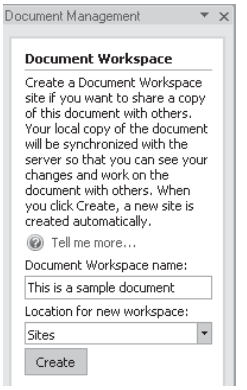
SharePoint supports a few site types that are specifically linked to Microsoft Office, called Document and Meeting Workspaces. While these site types can be created and used directly through SharePoint, these are also set up to work with Microsoft Outlook, Word, Excel, and so on. The purpose of these sites are as workspaces—temporary sites in which to collaborate on a document or meeting. Once the product of the effort is complete, the assets or product of the effort is saved to a permanent location and the site itself deleted.

The sites themselves, while part of the SharePoint site, are not directly linked; in fact, when created in a portal environment, they are not listed in the Site Directory and are not included in the search. The reason for this is that, as work in progress, there is no content there for general users to view. Creating these sites can be done by any authorized user directly from Microsoft Office, and virtually all of the general operations, including adding users, saving a document, and so on, can be handled from there. The user doesn't even have to actually go to the SharePoint site itself.

The site type created from Microsoft Office reflects the type of site to use in SharePoint. Word creates Document Workspaces whereas Outlook creates Meeting Workspaces. For example, creating a workspace from Microsoft Word is done by creating a document, saving it (it has to be saved locally once), and then selecting Create Document Workspace from the menu:

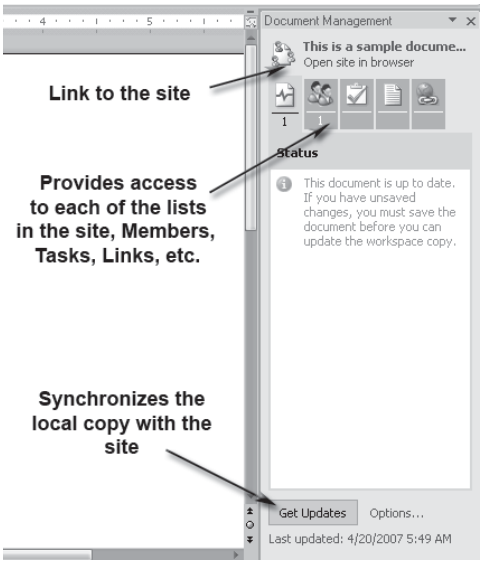


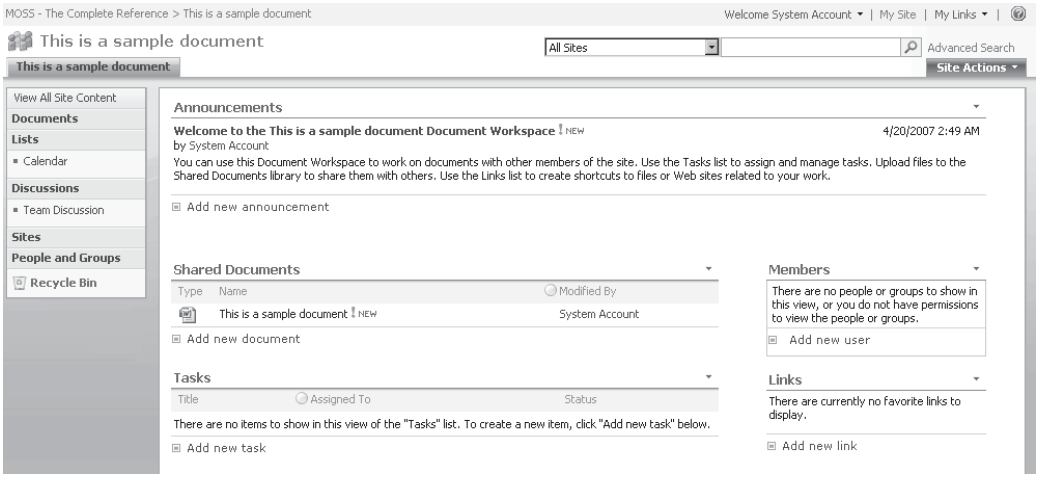
Once selected, the Workspace pane opens:



As you can see, this provides the opportunity to use the name of the document or to choose a name to be set for the workspace. You also have the option of selecting where it will actually be created, whether under the user My Site or a designated URL in a portal site. In a document workspace, when the Create button is clicked, a new site is created under the path indicated, which includes a document library in which the document is automatically stored. Once the creation process is complete, all of the site features can be accessed directly from Word.

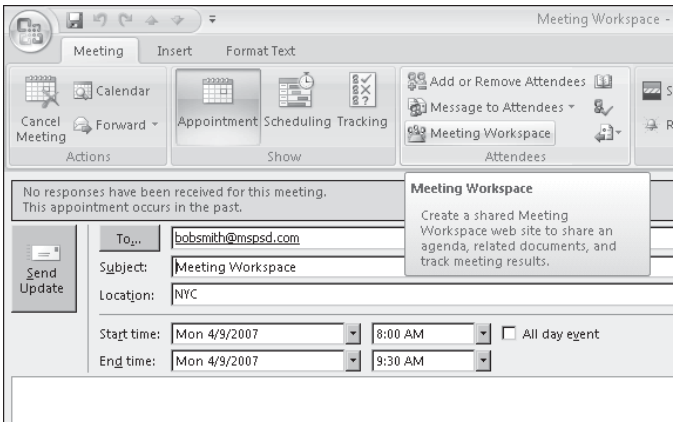
From these controls, additional users can be added to the site, tasks updated, links created, and so on. Other users can participate either through Office (by opening the document) or directly in SharePoint:



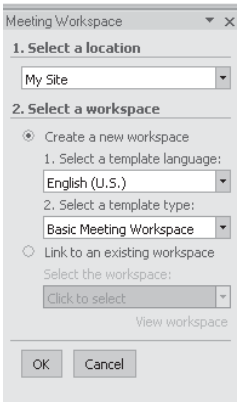


Virtually everything that can be done from the site can be done here, which is handy if users do not want to use the SharePoint Interface (or it is being purposely abstracted from them as is often the case with extranets). The interface here is the same for all members (based on what they are permitted to do), whether they open the document directly from SharePoint or they are sent a link to it. The Document Workspace site creation and functionality is the same throughout all of the Office publishing tools—Word, Excel, and PowerPoint.

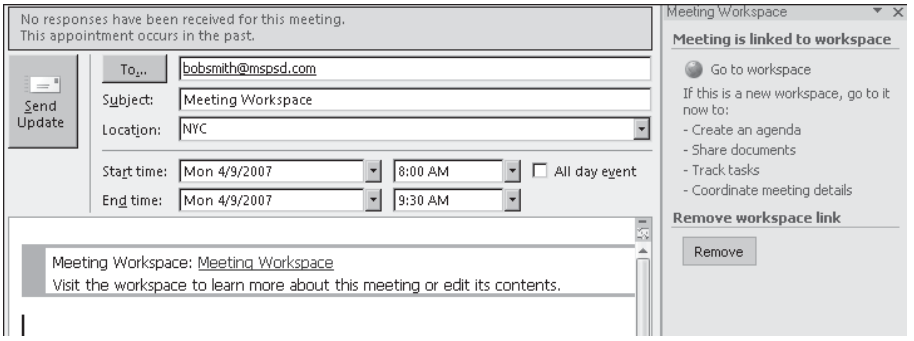
Only slightly different, Microsoft Outlook uses the Meeting Workspace site types that are created in association with a meeting event. Once a meeting has been created in an Outlook Calendar, a Meeting Workspace can be created along with it:



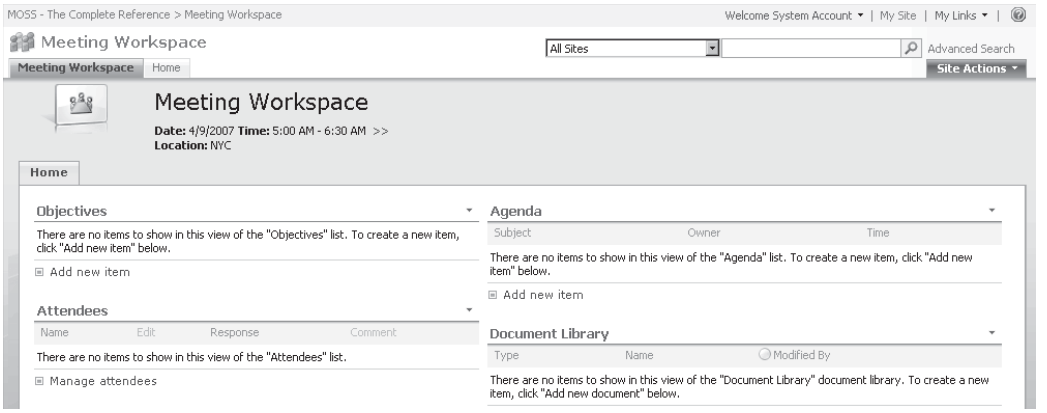
The Meeting Workspace provides the option of choosing the type of Meeting Workspace (from Basic to Social) and where it will be created, or you can even connect to an existing one:



All of the features of the site can be accessed using Outlook without having to go to the SharePoint site. Also, the Meeting Announcement includes a link to the Meeting Site:



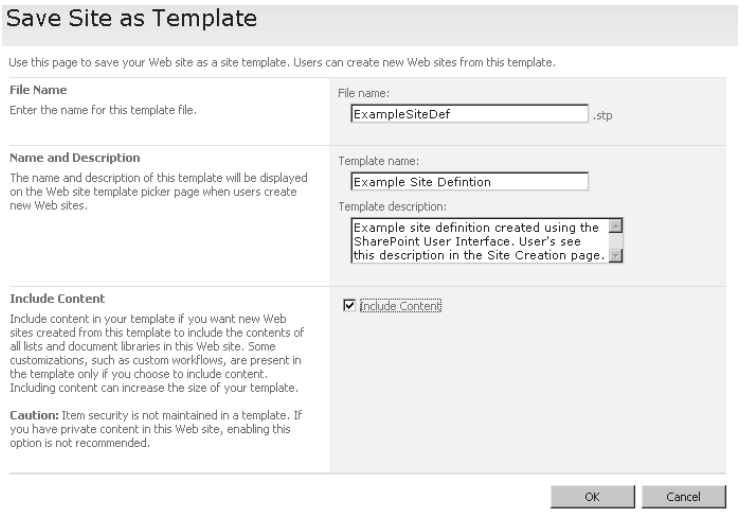
Using a Meeting Workspace Site is basically the same as using a Document Workspace. However, where a document is tied to a workspace, meetings can be linked or unlinked from a Meeting Site (even if created by it) at any time. Once created, other users can link to the site from Outlook or they can use the site directly:



Creating a New Site Definition with the User Interface

Aside from the out-of-the-box site definitions, users can create their own from within the SharePoint User Interface (UI). Formerly called Site Templates, this technique allows the end user to create a site, define lists, libraries, and even add content. The final design can then be saved as a list, library, or Site Template/Definition (including content if desired), and then made available to anyone else to use. Changes to the site theme, and other look-and-feel options, are also saved with the definition. The definition can also be exported and imported into other Site Collections and portals. (Note that portal and publishing site types don't support this feature.)

Creating a new Site Definition is quite simple since it is not done completely from scratch; the initial site must be created using an existing Site Definition (out-of-the-box and custom ones can also be used). For example, a project site type might be initially based on a Team Site, and then customized with additional libraries, lists, and so on. Once a site has been customized, it can be saved as a custom Site Definition. From the home page of the customized site, select Site Actions | Site Settings to open the Site Settings page. Under the Look-and-Feel Group, you will find the Save Site as template option. Clicking this will open the Save Site as Template Page, and it's here that you can specify the name of the STP file (Site Template file), name, and description of the site type. You can also decide whether or not to include content:



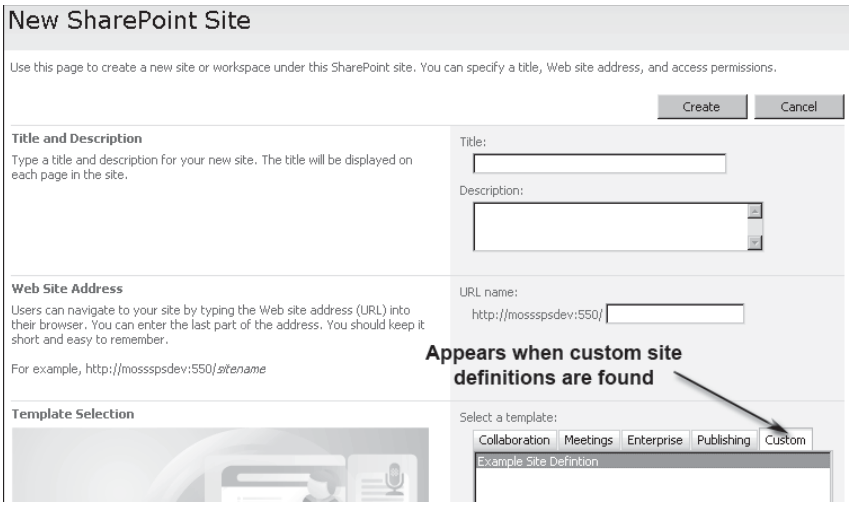
Clicking the OK button will save the template, and if successful (this only fails when trying to save a duplicate), a confirmation page with a link to the site template gallery is shown. Clicking OK here simply returns to the Site Settings Page. Clicking the Site Template Gallery link will open the gallery where the new definition has been saved:



There is only one gallery per site collection. This means that regardless of where you create a Site Definition in the hierarchy of sites, you'll find that all are stored in the same place. From this gallery, you can maintain all saved Site Definitions as well as import those from files. Clicking the name of the template will allow you to save it as a file to disk; clicking the Edit link allows you to change the title and description (note you cannot disassociate content). You can also access this gallery at any time from the Site Collection Site Settings Page (top-level site).

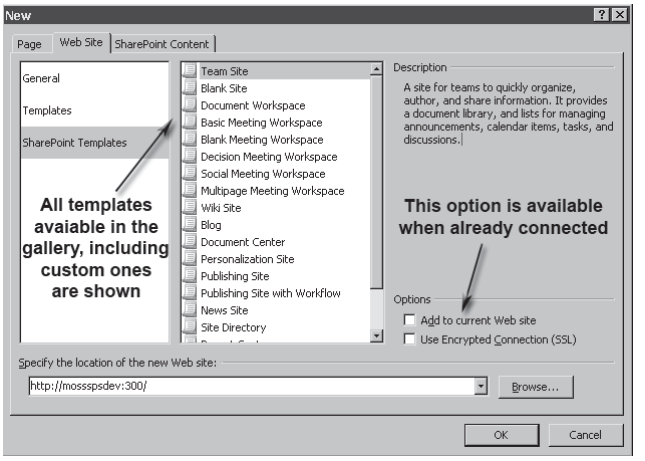
Using a Site Template Defined in the SharePoint UI

Once a Site Template has been saved to the Site Collection Site Gallery, it can be used anywhere in the collection to create new sites. The end user simply selects the Custom tab (shown only when custom templates exist) when selecting the Site Definition:



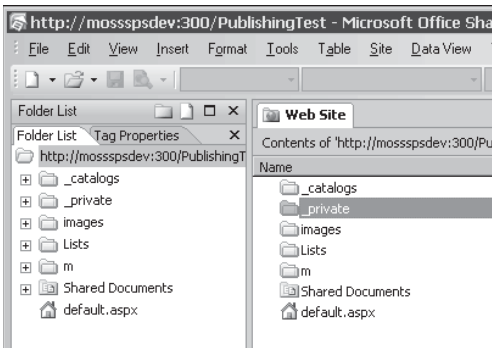
Creating a New Site Definition in SharePoint Designer

While templates can be created in the UI, these are limited to the options available out of the box. The next level up for creating Site Definitions is done with SharePoint Designer. The capabilities here are much more advanced and can utilize the drag-and-drop features, styles, fonts, as well as other things provided by Designer. Creating sites from within SharePoint Designer is always done after connecting to an existing site (you will get a warning if you attempt to create a site without one). You can create a site in two ways: either selecting File | New | Website | SharePoint Templates and then specifying the URL of an existing Site Collection or opening the existing site first. Like a template created from the SharePoint UI, a site created this way is also based on the existing templates in the Site Collection.



For example, to create a new site type based on the Publishing Site Definition, you would click Publishing in the list of templates, specify the URL to the new site, and then click OK to create. Designer will pop up a dialog box once the site creation process starts; if there are any problems, it will indicate them right away. When all is done, the site will be opened in Designer, as shown next.

At this point, you are free to modify the site at will. You can add new lists and libraries, change the default home page to add components like breadcrumbs, and so on. Note that during my own testing, I found that creating sites this way sometimes displayed errors, but the site was still created. In one case, a Publishing Site Type created a Team Site; when the Team Site was deleted, the Publishing Site appeared. Because of this, I suggest you verify the site after it is created.

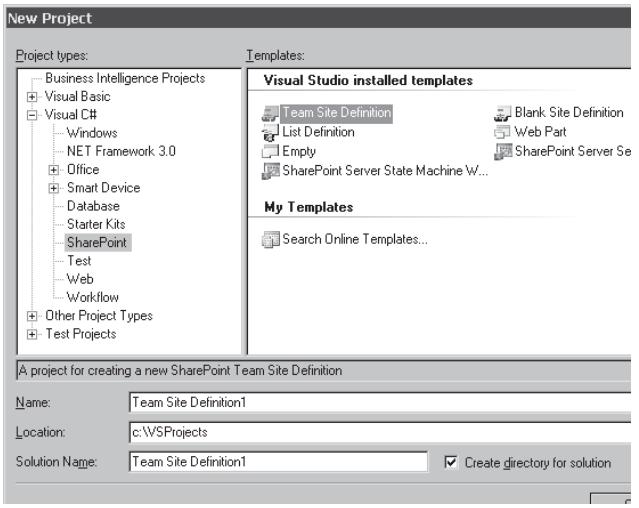


Creating a Site Template Using Designer

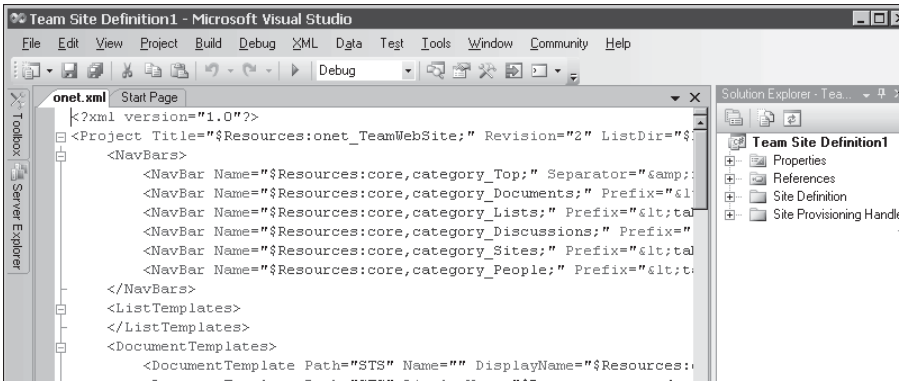
Once a site has been opened using Designer, it can be saved as a Site Template (whether or not actual changes were made to the site). This is done (when the site is open in Designer) by simply selecting File | Export | SharePoint Site Template. The site will be opened and the site definition can be saved to the Gallery in the same manner as the UI in the previous section.

Creating a new Site Definition in Visual Studio

In terms of customization, creating Site Definitions in Visual Studio provides the most advanced type of development for sites. To work with these, SharePoint Tools for Visual Studio must be installed. These provide the default shell projects in which to create SharePoint Sites, Lists, Libraries, and so on. Creating any SharePoint item is done by opening Visual Studio, selecting File | New… | Project, and then selecting the type of object you want to create:



As an example, creating a new site using the Team Site definition creates a new Visual Studio Project, which has all of the files necessary for a Site Definition—most notably the ONET.XML file that SharePoint uses:



The ONET.XML file is an XML file that contains Collaborative Application Markup Language (CAML), which defines the composition of the site. SharePoint uses this whenever the Site Definition is used to create a new site and can use it to revert a site to the original definition used to create it. As shown, the start of the ONET.XML file defines what the navigation is, followed by list templates, libraries, and so on. This also defines the views of what is shown on the home page. Each Site Definition can have multiple configurations, which can be used to dictate which lists or libraries are created by default, the features the configuration will use, and more. SharePoint itself uses this same method; the SharePoint Collaboration Site types are all defined using one ONET.XML file, and they use different configurations for each type. The Team Site, for example, is defined as Configuration 0; the Blank Site definition is Configuration 1, and so on.

In addition to the ONET.XML file, the Site Definition also includes a Default.aspx file, which allows you to either use the default settings or modify the page to change the layout, add zones, and so on. The Default.aspx page can be altered using straight aspx code or you can use the Designers WSIWYG design capabilities to modify it.

ONET.XML uses a schema found in C:\Program Files\Common Files\Microsoft Shared\web server extensions\12\TEMPLATE\XML\wss.xsd. As a language itself, it is quite beyond the scope of this book. As a starting point, you can refer to the SharePoint Software Development Kit (SDK), and there are also several other resources under Microsoft's MSDN and TechNet sites.

Deploying a Site Definition from Visual Studio

As you saw previously, creating a Site Template is simple using the UI and SharePoint Designer. When using Visual Studio, however, Site Definitions (and List Definitions) are deployed as Solutions. These include a feature.xml and an elementsmaster.xml used to install the Solution in SharePoint. The Site Definition (ONET.XML), default.aspx file (if included), and any other assets needed (such as images) are packaged into a folder structure and then deployed (copied) into the SharePoint file locations. New in VS 2005 is that the actual deployment works like web parts; simply setting the URL as the debug starting location and selecting Tools | Deploy will cause the definition to be added. See Chapters 17 and 20 for more information.

Customizing and Branding the SharePoint User Interface

There are several ways in which to modify the User Interface (UI), also known as the look and feel of the site. Often known as *branding*, the UI can be modified to suit taste as well as promote the brand of the organization including logo and colors. While the effort to brand SharePoint is a straightforward process, I do offer a few words of wisdom. When you are considering modifying how SharePoint looks, remember that it is a customizable application and not a website. For publishing-style sites, I say knock yourself out in design. In collaboration sites, however, take a deep breath, and be sure you want to invest time and money into the looks of something that should be an efficient and useful tool. Slight customizations are fine, but invest design work in publishing sites, not ones used for collaboration. For any site, I suggest you limit the design decision process; I've seen several projects extended for months just because a pixel was too far to the left.

Out-of-the-Box Modifications

SharePoint provides a number of look-and-feel options that are already provided within the UI. These include generic settings, such as the Site Title and Description, but it also supports using an alternate Style Sheet at the Site Collection/Site levels. The modifications are accessible through Site Actions | Site Settings. Under the Look-and-Feel Group, you have the following options:

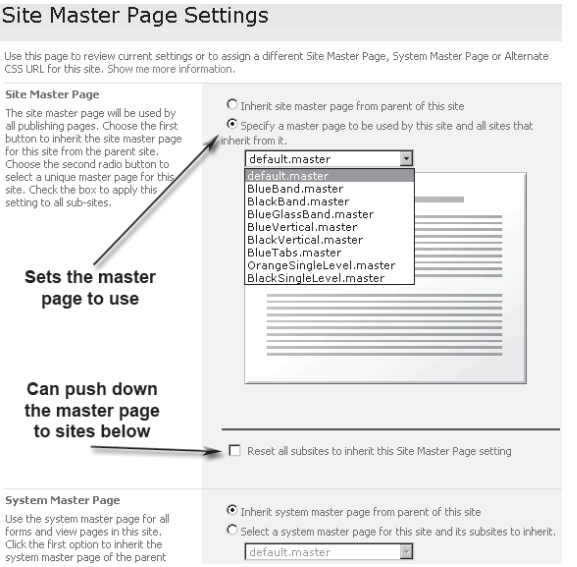
- **Master Page (MOSS)** This sets the Master Page to use for the current site (any Master Page in the Master Page Gallery can be used).
- **Title, Description, and Icon** This allows customization of the title and description and a site logo (images should be stored in the SharePoint file system or actual site).
- **Welcome Page (MOSS)** This sets the page you want to use for the home page of the site.
- **Tree View** This adds a Site Hierarchy Tree View to the Quick Launch (this has no effect in publishing sites that do not have left navigation defined by default).
- **Site Theme** This allows selection of several predefined color schemes.
- **Top Link Bar** This allows customization of the Top Link bar (top of page).
- **Quick Launch** This allows customization of the Quick Launch (left side of the page).
- **Save Site as Template** This saves this site as a reusable template (covered previously).
- **Reset to Site Definition** This resets the site's current definition to the one that created it.
- **Searchable Columns (MOSS)** This is used to set Site Columns as searchable and does not actually affect look and feel—only the search characteristics.

Custom Modifications

In all site types, there are two basic elements that dictate the actual look and feel of the site. In WSS sites, these are the default.master page (the heading and navigation) for the Site Collection and a single default.aspx page, which defines the home page layout including content and web part zones. In enterprise sites, there is one assigned Master Page for the Site Collection or for an individual site, which can be from a number of masters in the Site Collection Master Page Gallery. For pages, any number of Page Layouts can be used in the

site and the Welcome Page can be one of three different types out of the box (Welcome Pages are launch pages intended to be used as home pages, top pages in site collections, and so on) or are custom built.

In both WSS and MOSS site types, the Master Page set for the site is what handles the page structure, including the navigation and styles applied. The Page Layouts handle what is displayed within the Master Page including content and web part zones. There are two ways in which Master and Page Layout Pages can be modified and added to SharePoint. The first method is basically manual, and is done directly through the SharePoint User Interface. Via the Site Settings Page of the Site Collection, the Master Page and Page Layout Gallery can be accessed, and from the gallery, Master Pages can be uploaded and downloaded. In enterprise-type sites, the Site Settings Page also provides access to setting the Master Page and Welcome Page for the site. When changed, the settings can be used for the single site or pushed down to all sites below the current one, as well as shown from the Master Page setting.



The second method is to create a Master Page and/or any number of Page Layouts and install them using the SharePoint Feature method. While this is the preferred method, in truth, there is no difference in the actual mechanics (the master files are still loaded into the gallery and so on). The primary difference is for farm deployments and administration. The Feature method enables easy deployment to servers, and it can be enabled or disabled by the Site Administrator. This is a great feature when new changes have a problem and have to be rolled back. An important distinction between custom sites and custom Master and Layout Pages is that Site Definitions—that is, those defined by ONET.XML files and in the Site Template Gallery—are the real definition of the site; the Master and Layout Pages simply control presentation. When a user reverts to the original Site Definition, this means reverting to the Site Template; a change to the Master Page or Layout Page has no effect.

Creating and Customizing SharePoint Master Pages

The core look and feel of the SharePoint site is controlled by the Master Page set for it. This includes the heading, side navigation, page background, and styles. For the actual content area of the page, it includes a content placeholder. Pages created from Page Layouts are rendered inside of the placeholder and assume the styles set by the master.

Master Pages are .NET aspx pages that use SharePoint components, may include custom components, and can be created or modified using either SharePoint Designer (providing

WSIWYG design) or Visual Studio (using aspx coding). These are made available in SharePoint by simply loading them into the Master Pages and Page Layouts Gallery, which, in turn, makes them available in the Master Page Choice for sites.

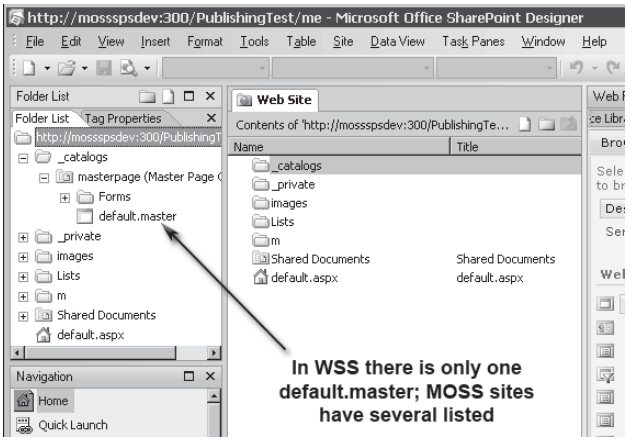
Similarly, Page Layouts are pieces of ASP.NET pages. They are trimmed of tags like <HTML> and <BODY> and include a single reference to the Master Page used. These partial pages define layouts using tables and web part zones and can be used as publishing pages linked to a Content Type (providing the layout for content fields). These are also made available in SharePoint by uploading into the Master Pages and Page Layouts Gallery and can be selected in any site using Site Actions | Create Page. Page Layouts can use Designer's WSIWYG editor, provided that the Master Page it references is loaded into the gallery.

Where you will apply changes to alter look and feel depends on the level of customization you want to impose on the site. Modifying a single Master Page in a single site as a one-off applies only to that site. Modifying a Master Page in the Site Collection can be applied to the site only or to the site and all sites below it. To make a global change, the core files in the SharePoint Hive can be modified to apply to all sites in a SharePoint Server farm.

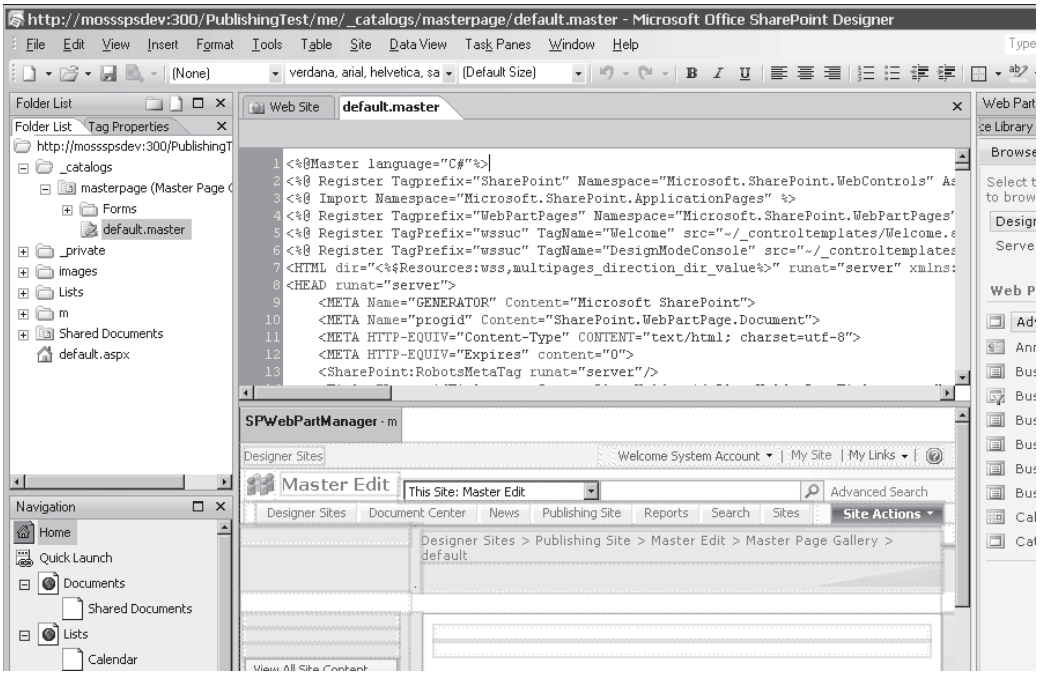
Making a One-off Change

A one-off change applies only to a single site. A case wherein you might use this is to add some feature that will only be used in one place, such as a dashboard site that has a secured data feed. This is also a great way to develop a new master and test the functionality without breaking other sites in the collection. Note that choosing Master Pages (and Layouts) can only be done through the UI; physically changing them requires Designer or Visual Studio.

The easiest method to modify a Master Page is through SharePoint Designer as you will do here. Using Visual Studio is also simple, but it requires downloading the page, editing in Visual Studio (without WSIWYG), and then uploading back to SharePoint. In SharePoint Designer, you can simply connect to the website and then open the page. To do this, open SharePoint Designer and then select File | Open Site; on the dialog page, specify the URL to the site you want to modify. When opened, the Master Page is accessible through the _catalogs\masterpages folder:



To edit the master, simply double-click the name under the Folders tab; depending on your settings, this will open the page in Design, Code, or Split Mode. Split Mode (showing both) looks like this:



Once open, the master can be edited and even component settings (like the navigation bar) can be changed. Use File | Save to save the change or simply press CTRL-S. When the file is saved, you can then see the actual changes in the site. As long as you have rights, the changes will be visible without having to approve the page in the Master Page and Page Layout Gallery.

Creating a New Master Page

There are a few tricks to creating a new Master Page for use in your SharePoint sites. The fastest method is to simply make a copy of an existing Master Page and then modify it using SharePoint Designer or Visual Studio. For those not familiar with .NET, I suggest you keep the changes limited to those you can do with Designer. Those of you familiar with .NET Web Components and ASP.NET pages will find they are easier to deal with in Visual Studio; working with the SharePoint Web Components is the same as any custom component.

Another option is to literally start from scratch. That is, starting with a blank canvas where you can design and integrate the various SharePoint components (like Search) at will. To help you with this, Microsoft also provides a minimal Master Page that has all of the components you need for a Master Page to run in SharePoint, but it has no layout. (This is available for download from http://msdn2.microsoft.com/en-us/library/aa660698.aspx.) Do note that there are a few minor corrections to the code on this page that are indicated in the comments below it.

Once your new Master Page has been created, you have two options to use it in SharePoint. The first is to manually upload it into the SharePoint Master Pages and Page Layout Gallery; once there, you can select to use it through the Sites Master Page Site Settings. The second is to create a new SharePoint Feature for it, which effectively does the same thing. As I've mentioned, the advantage to using a Feature is that manual steps, such as copying assets (images, and so on) can be automated, and the Feature can be enabled and disabled by a Site Administrator.

Modifying Core Pages

Outside of the Master Pages and Page Layouts explained next, there are times when a design or even compliance need requires changing the base definitions of the SharePoint sites. To change these quite simply use either Designer or Visual Studio. The real trick is finding the correct files to change.

For the primary Site Collection in an enterprise site, the default.master file is created on the fly. The only real copy is that which is stored in the site; change that and the change is universal. Other (WSS) site types are a bit mixed since they still straddle the previous version's way of doing things. For example, the basic layout pages for the Team and Blank Site Types have a basic layout page they use called default.aspx. The Document Workspace uses a file called defaultdws.aspx. These files are stored in <drive>:\Program Files\Common Files\Microsoft Shared\web server extensions\12\TEMPLATE\SiteTemplates\STS.

The Collaboration and Meeting Site Types share a single Master Page called default.master, which is common to all sites using the Team, Document Workspace, Blank, Wiki, Blog, and My Site types, and is stored in the SharePoint Hive under <drive>:\Program Files\Common Files\Microsoft Shared\web server extensions\12\TEMPLATE\GLOBAL. Meeting Workspaces use a file from this folder called mwsdefault.master. Personal Sites are another issue altogether and are covered in the next section.

Creating and Customizing SharePoint Page Layouts

Page Layout pages are different from the Blank and Web Part Page Types as they are specifically intended for publishing and are used only within the enterprise site types (not available in WSS). Based on a content type, Page Layouts organize and layout the fields for both Authors (used in create) and Users (used for presentation). MOSS includes several Page Layouts out of the box, all based on a few Site Content Types. These are grouped as Page Layout Types and Publish Page Types in the Gallery, and all inherit from a type called Page. For example, one is the Article Page Type that includes a byline, an image, body, and so on. Several pages use this same content type in different layouts; one includes the image on the left, another includes the image on the right, and another has no image shown at all. The Content Types are used both for defining the Page Layout and are attached to the Pages Library to support it.

There are two ways to create these pages: either in SharePoint Designer or in Visual Studio. The easiest method is to use Designer because it provides an outline view of the Master Page. In Visual Studio, the Master Page is not visible so the layout of the page is all done in code.

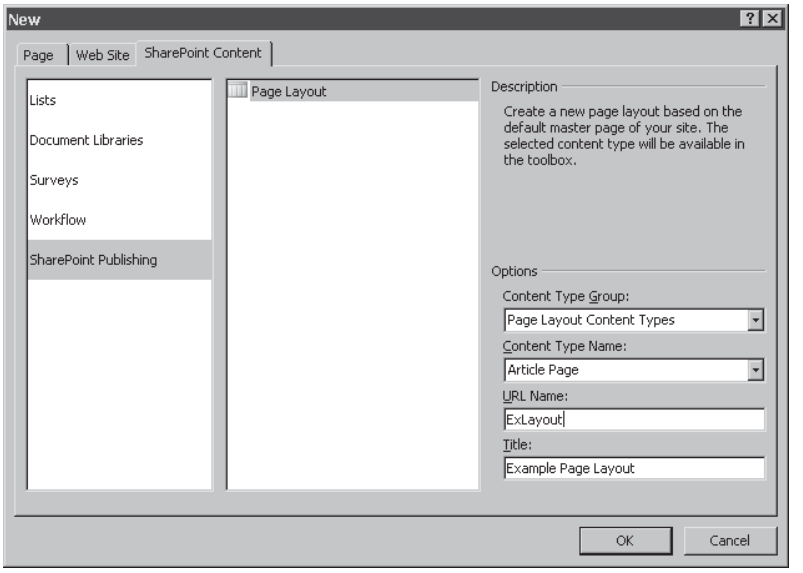
Creating a New Page Layout

As I mentioned, the Page Layout is based on a Site Content Type. The type defines the contents that will be available to (not necessarily on) the page. The Page Layout can use (or not use) any of the fields available; when a user selects the particular Page Layout, the fields

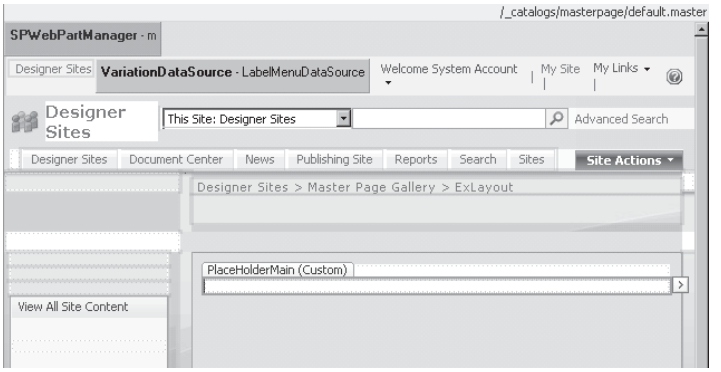
it exposes are where the user can enter content. Note that in addition to Site Content fields, custom fields can easily be created using Visual Studio, and then later used in Studio or Designer.

The first step in creating a Page Layout is determining the data it will contain. The typical article page, for example, has a byline (for the author), a content field, an image placeholder, and so on. If starting out from scratch, you will need to define the content type to match your content needs; alternately (as I will do here), you can simply use one of the existing types.

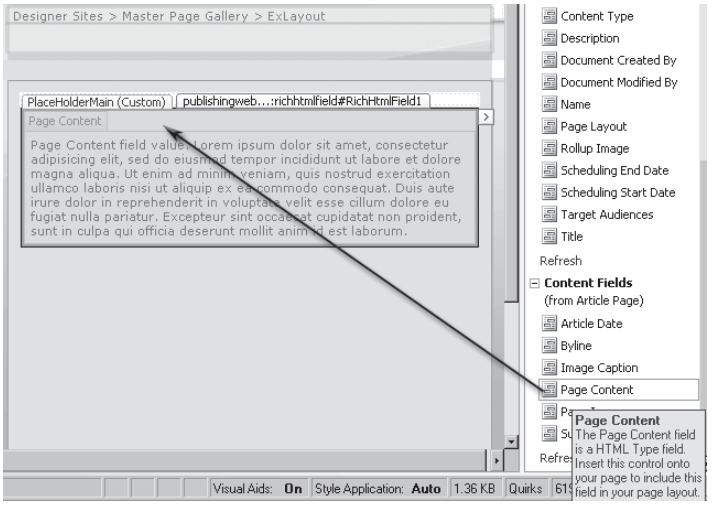
The next step is to actually create a page in SharePoint. This is done by connecting to a SharePoint MOSS site type (one that supports publishing pages). Once open, you can select File | New, and then from the dialog box, select the SharePoint Content tab and then SharePoint Publishing | Page Layout as shown:



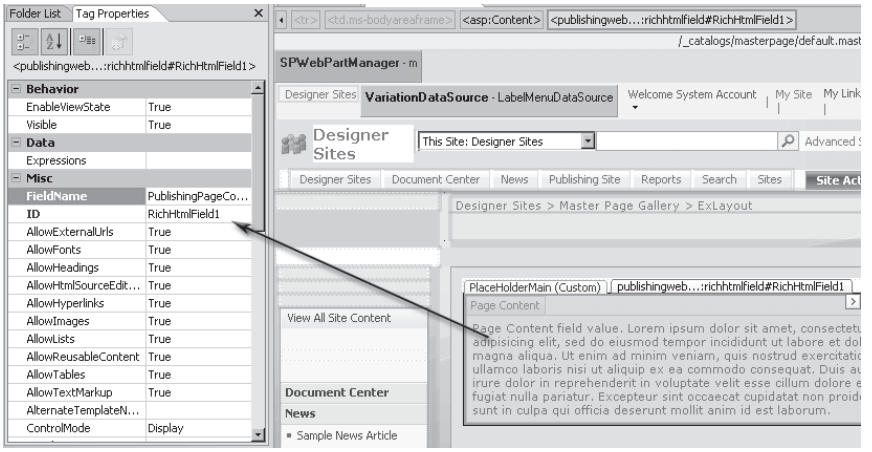
Notice that I've already selected a Content Type Group (in this case, Page Layout Content Types) and selected the specific Content Type Name (in this case, Article Page). I have also set a URL Name (this is the page name) and a Title for this page. Once you've clicked OK, the page is created. Designer will load up the site's Master Page and then display the new page within the design display (you cannot modify the Master Page here; it is view only):



As you can see, only one thing is defined for the Page Layout by default, and this is a tag called PlaceHolderMain, which is basically the design canvas for the page. By dragging and dropping fields and controls, the page can be painted out in format. For example, to add Page Content field to the page, simply drag it over from the list of field controls:

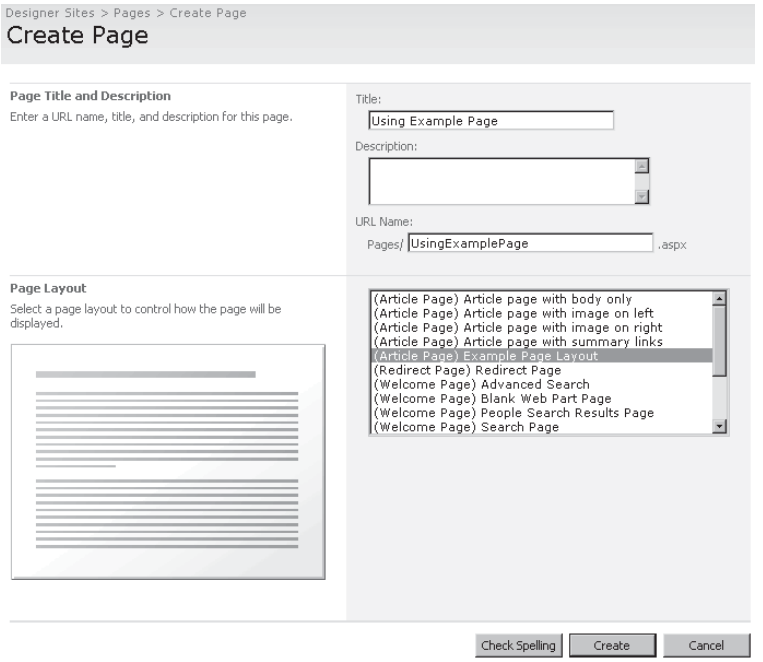


To adjust the properties of the field control, simply right-click on the field and select Properties; in the Properties window, any available settings can be modified (including the ID of the field, name, and so on, as shown):

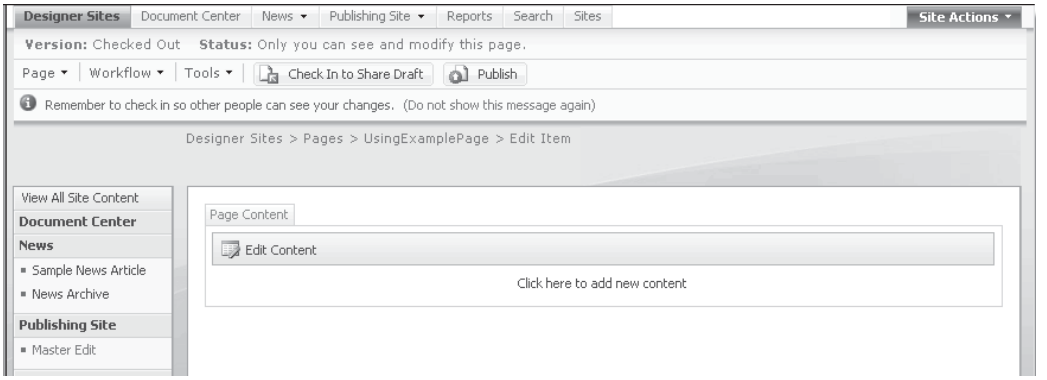


Saving the page automatically saves it to the Master Page and Page Layout Gallery. The initial status is set to Draft (not visible to user). To change this, go to the home page and select Site Actions | Site Settings | Modify All Site Settings. Click on the Master Page and Page Layout Gallery link and locate the file. Mouse over the name to view the drop-down menu and then select Check In. On the Check-In page, select Publish a Major Version, enter any optional comments, and then click OK to complete check-in. When you return to the gallery, mouse over the file again and select Approve/Reject; when the Approval Page

opens, click Approved, enter optional comments, and click OK to complete approval. When you return to the gallery, you'll see the page is now approved. To test out the new Page Layout, navigate back to the site and select Site Actions | Create Page. The new page type should appear in the list of choices:



Creating this page will provide the page with the layout, as set up in Designer (in this example, I added only a single content field as shown):



As you can see, once the Page Layout is saved (and approved), it can be used anywhere within an enterprise site. Since it is tied to a Site Content Type, the necessary fields and definitions are always available. Note that Page Layouts can also be used in other Site Collections (by uploading), but the collection must have the exact same Site Content Types the page uses.

Customizing the My Site Template

As you might have guessed, My Sites are simply another kind of Site Definition. There are a few additional pieces in this one, because there is more than one view of it. In addition, a level of control over the My Sites is handled by the System Administrators My Site (changes to the Administrators My Site Shared Web Parts are reflected in all users' My Sites). For a general background, you can refer to Chapter 13.

As a Site Template, the typical layout and use of the My Site is quite sufficient. However, there are cases when certain options need to be turned off or disabled. Much can be done with the out-of-the-box settings; certain profile values can be restricted from being changed and features can be disabled. Others require a bit of customization to actually alter the functionality; examples of this include removing the ability to create sites, not allowing personal blogs, and so on.

As with any site, the basis for the My Site is an ONET.XML file, but in this case, it's two aspx pages: one for the Personal View and the other for the Profile View. For style and Master Page control, the My Site Template uses the same default.master as the Collaboration and Meeting Sites (located in <drive>:\Program Files\Common Files\Microsoft Shared\web server extensions\12\TEMPLATE\GLOBAL). There is nothing too special about modifying the My Site templates, except for knowing what to modify, of course. The key to the My Site definition is the Master Page mentioned previously and in the SharePoint Site Templates (<drive>:\Program Files\Common files\Microsoft shared\web server extensions\12\template\SiteTemplates\SPSMSITEHOST). In this folder, you will find the definitions for the two page views: Person.aspx, which displays the Profile Page (Personal and Public Views) and Default.aspx, which handles the display of the actual My Site home page for the user. There are additional pages called Discussion.aspx and Blog.xsl, which are used for other aspects of the My Site as needed. Under this folder is a folder called XML, which holds the ONET.XML file that defines both the site and its layout.

Rather than delve into hundreds of possible changes that could be done, I'm going to cover a couple of the most common ones I've been asked about. Be aware that if you want to make some changes, the base definition modifies all My Sites, and unlike other site definitions, which can be duplicated, there is only one My Site Template Definition.

Using a Different Default Master for the My Site

In many cases, it can be desirable to use a different Master Page for My Sites. Consider having a slight color variation, just so users know they are on their own. It can also be functionally ideal: the My Site Master could be modified to include links to employee self-service systems and other such links only appropriate to personal use.

The default.master used by the My Sites is the same one used in the Collaboration and Meeting Sites. Certainly within the My Site itself, the master could be changed; however, that would be a one-off for a single user. To make a consistent change across all My Sites, the actual Site Template Pages must be modified, since this name is hard coded.

To set up the My Site Definition to use a different Master Page, do the following:

1. Navigate to the <drive>:\Program Files\Common Files\Microsoft Shared\web server extensions\12\TEMPLATE\GLOBAL folder.
2. Make a copy of the default.master file there and name it **mysite.master** (you can edit the new file using Designer or Visual Studio at will).

3. Navigate to the <drive>:\Program Files\Common files\Microsoft shared\web server extensions\12\template\SiteTemplates folder; make a copy of the SPSMSITEHOST folder and save it (this is for backup purposes).
4. Open the SPSMSITEHOST folder and edit each of the aspx pages. In each, change the following line:

   ```
   < %@ Page language=C# MasterPageFile=~masterurl/default.master
   ```

   to the following:

   ```
   < %@ Page language=C# MasterPageFile=~masterurl/mysite.master
   ```

5. After all files have been modified, run IISReset from the command line.
6. Navigate to a My Site to see the change.

Removing the Organization Part on the My Site Person Page

In many organizations, there is no real organizational source that provides a direct line of management up or down. If this information is not available, the Organization Display Web Part is of no use. While users can hide it, it can be removed permanently by editing the Person.aspx file (from the Template folder from above). In this file, locate the code:

```
<SPSWC:ProfileManages FrameType=None id=ProfileManages runat=server />
```

You can either delete the line or comment out like so (leave the <td> and </td> tags alone!):

```
<!-- <SPSWC:ProfileManages FrameType=None id=ProfileManages runat=server /> -->
```

Summary

This chapter explored the various kinds of sites that are provided within WSS and MOSS, as well as touched on creating custom templates using the tools available from the User Interface to Visual Studio. You should also know that this is by no means limited, as custom special-purpose Site Definitions are being created every day. If you're just starting out, you should have a good idea of what kind of site you might need, and for those new to 2007, a much better understanding of the new types provided.

16
CHAPTER

SharePoint Workflow Tools

Windows SharePoint Services Version 3 includes a completely integrated workflow feature, which enables users to initiate and participate in structured workflows for the purpose of emulating work processes (such as document approval). In SharePoint, workflow comes in three flavors: built-in workflow tools employed by the user, enhanced workflow created using SharePoint Designer, and custom workflows created using Visual Studio. I'm going to limit this discussion to using the built-in tools in both SharePoint and SharePoint Designer. Development in Visual Studio is heavily .NET-based and is a bit beyond the scope here.

The first set of workflow tools in SharePoint is at the user level. Users can assign (or attach) workflows built around lists, libraries, and site content types using out-of-the-box flows and user-defined workflows built by designers and developers. Using SharePoint Designer, designers can easily use wizards to create and add custom workflows that go beyond what is out of the box. Developers using Visual Studio can extend workflows for Designer and build workflows to use outside of the SharePoint environment.

Microsoft's vision for workflow boils down to three primary goals:

- **Goal 1** Facilitate human processes by attaching business logic to items and documents in WSS while providing context and tracking progress.

  Translation Provide common workflow activities, like document approval processing, and be able to see what state a workflow document or item is currently in.

- **Goal 2** Empower information workers using Office Server with out-of-the-box solutions and self-service tools for supporting a broad range of routing and tracking scenarios without IT involvement.

  Translation This provides several common workflow processes out-of-the-box that are easily customizable by users through the WSS User Interface so no development or administrative assistance is needed.

- **Goal 3** Empower organizations to build a broad range of sophisticated workflow solutions that take advantage of the Windows Workflow Foundation (WF) platform and integrate into the SharePoint and Office Experiences.

  Translation This offers a workflow that is seamlessly (read: familiar) integrated within SharePoint/Office.

In fulfilling this mission, SharePoint workflow is completely integrated into the WSS User Interface (UI) and is able to use all of the platform tools within it such as alerts, e-mail, and so on. Workflow operations can be invoked (and task status changed) in multiple ways; they can be initiated by a specific user action (such as editing), triggered by an event (like an upload), and they can be initiated through timer events. Integration is apparent in all of the Office products; for example, a user might receive an e-mail notification in Outlook that indicates that there is a newly assigned task to accomplish, such as approving a document (the link to both task and document are in the body). After you click the link, the appropriate program opens, and the user can review, make any changes, and then press Save. The task is marked complete, and the workflow continues to the next task. All of this is done in Office, so the user doesn't need to worry that SharePoint is in the background.

There are two things to consider: for one, SharePoint workflow runs within the context of WSS and is not a full-blown engine; it is merely an implementation of the WWF for WSS and not for extending into other systems or applications. That kind of functionality is meant for applications like BizTalk. For two, WSS workflow is an online process; that is, the user must be connected to a WSS site. Offline/External work can be done, but it must sync up to the site.

Who Does What?

There are several roles within the Workflow aspects of SharePoint; designers build workflows, end users invoke them, and so on. Even the Office products play a huge role. The beauty of it is that once in place, everything is under the covers as far as the users are concerned. The way they all fit together is as follows (though I wouldn't be surprised if all of these roles are done by one person):

- **Knowledge Workers (end users)** These people start workflows and complete tasks through the SharePoint interface and from Office publishing and database tools (Word, Access, and so on); users can manage tasks (or reassign, change, and so on) if enabled by the workflow author.
- **Administrators** Administrators set up workflows, change workflow settings, deploy/enable/disable workflows, reassign tasks for users, change approvers, and so on.
- **Site Designers** This group uses SharePoint Designer to assemble new workflows using existing components.
- **Developers** These people can build new components (and workflows) using the SharePoint Object Model, Windows Workflow Foundation, and SDKs.

Background on Windows Workflow Foundation

While SharePoint Workflow sounds impressive, what actually gives the workflow power is the .NET Framework. WWF is not an application or engine itself, but a set of components that are hosted by another application (in this case, SharePoint). SharePoint's use and interaction with WWF is abstracted away from the user and designers; the UI and Designer Wizards handle the majority of the work. Visual Studio developers can take advantage of the fact that WWF is SharePoint-aware and provides a small library of standard activities such as sending e-mail, connecting to SharePoint lists, and so on. This is useful when extending or creating custom workflow (note that using Visual Studio requires downloading

and installing Windows Workflow Foundation Extensions for Visual Studio). WWF key concepts include the following:

- **Workflow is just like it sounds** This is series of steps to get from point A to point B; workflows in WWF are created by linking a set of activities into steps that emulate a human process.
- **Activities are units of work that are called elements of the workflow** This concept is synonymous with a step in the process (a good example is the sending of an e-mail notification to let a user know that a document is read); steps can be multiconditional using if-then-else-if logic.
- **Workflows must run within a host process** In an application like WSS, it is not an engine and does not run on its own; this means that SharePoint is running WWF.
- **Workflow development includes building Custom Activity Libraries** The designer can use this to build workflow processes.

The components of Workflow include:

- **Base Activity Library** This encompasses out-of-the-box activities and the base object for custom activities.
- **Runtime Engine** This has to do with workflow execution and state management.
- **Runtime Services** This has to do with hosting flexibility and communication.
- **Visual Designer** This is a graphical and code-based workflow construction tool (part of Visual Studio).

These components do not run by themselves; the host process actually manages the Base Activity Library and Runtime services. In relation to WWF, you'll only focus on SharePoint's use of it. For more on WWF development and use, refer to MSDN or TechNet.

How SharePoint Implements Workflow Foundation

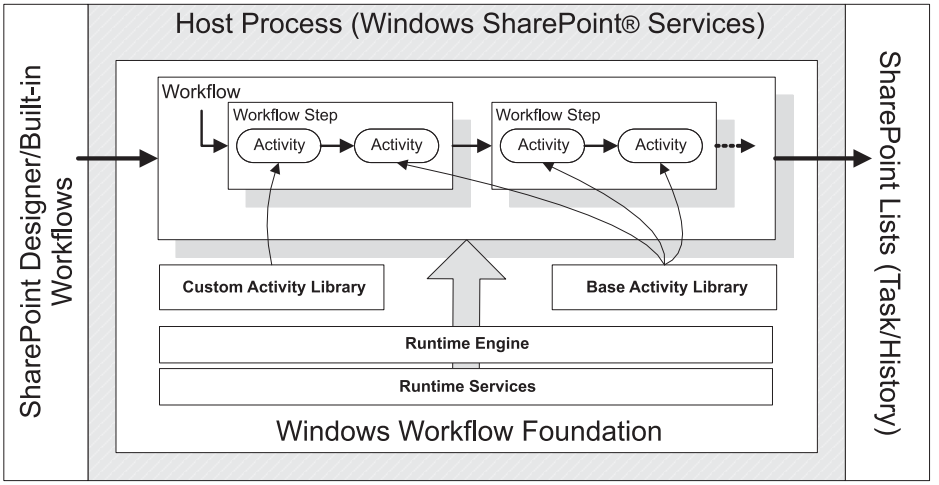
As the host, SharePoint handles each area needed to support WWF as follows:

- **Persistence** SharePoint stores workflow status info to the Status/History Lists in the DB.

NOTE *This does not use WWF persistence and uses SharePoint native lists (synonymous with a DB Table) as the storage instead.*

- **Event Delivery** This informs (triggers) the workflow when something has happened, such as an update or task change.
- **Timer Service** This enables running the Delay features of WWF (Reminders and so on).
- **History and Reporting** This maintains the History List, tasks, and so on.
- **WSS Application Integration** WWF is hosted in process with WSS on all front-end machines running WSS Web Services (this provides complete scalability across the farm).

How SharePoint tools, Windows Workflow Foundation, and SharePoint lists all fit together is shown here:



Note that workflow instances are aggressively dehydrated (saved) to the back-end database whenever possible (not running and so on) to ensure accuracy. This helps with human-intensive activities that require a longer time span (two days, one week, and so on); instances are swapped in and out to make those most in use available when requested.

SharePoint Workflow Basics

SharePoint's out-of-the-box workflows (available through the UI) cover typical kinds of one-step workflow processes needed for collaboration; these include routing and approval, signature collection, issue tracking, and Office integration features such as List Moderation, Publishing, and Document Expiration. Most of these include parameters that allow setting of run-time values; this gives them fairly structured processes that can be ad hoc since they are quickly implemented by the end user. Workflow fields in lists are also lightly customizable (for example, to set the states of a task), so although preset, they can be customized in several ways so offer quite a bit of flexibility.

SharePoint Designer can be used to create User Defined workflows that are more advanced than the UI. These are no-code workflows that combine actions with conditions with multiple branches and steps. Using a wizard, a nonprogrammer can quickly put together fairly complex workflows using actions (both built-in and custom, like creating a task). In Designer, workflows can only be created using the installed activities (they can't be extended with custom code, but can be through the aspx and InfoPath forms they use). Visual Studio (with WWF Extensions) is used for building custom workflows and also for custom activities that extend to normal Line of Business (LOB) integration and industry-specific processes. Typical implementations include order processing, hand off to BizTalk, and so on.

Combined with the tools are the parts of SharePoint that make it work, such as the Source List that initiates a workflow process, a Task List that stores tasks, and so on. A high-level overview is as shown in Figure 16-1.

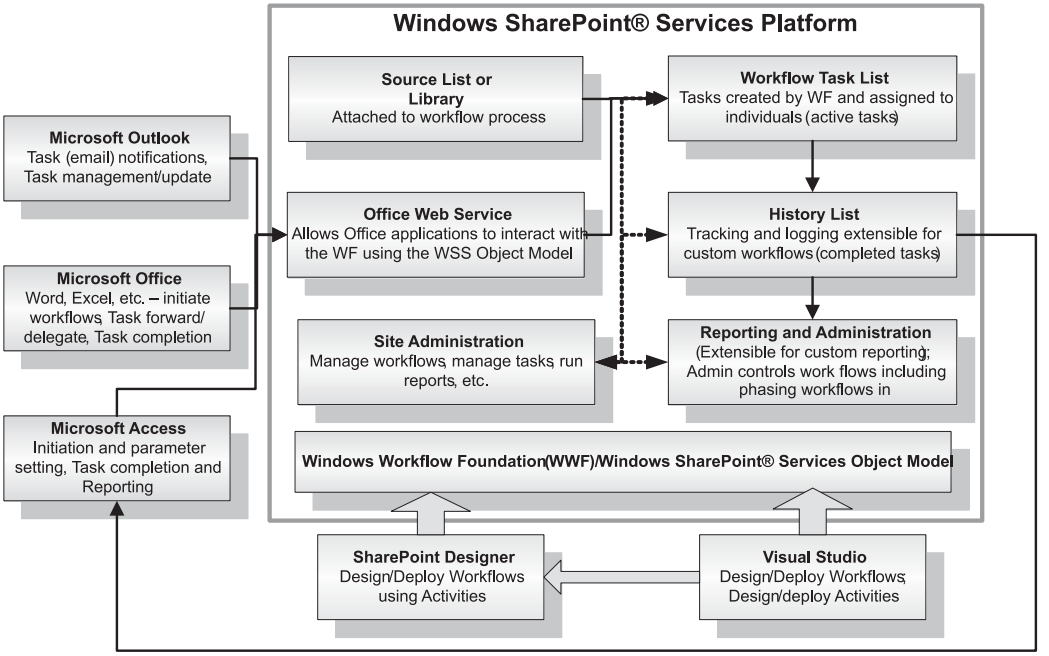


Figure 16-1 Component overview of SharePoint workflow

The heart of SharePoint workflow begins with the Source List or Library; this is where the actual workflow is started (for programmers, this is just like an event). This creates a new task item in the Workflow Task List and an entry in the History List, indicating that it started in the site. Next, either an event that affects a task from the User Interface (such as a document check-in), or from another Office application (Outlook, Word, Excel, Access, and so on), updates the task in the workflow.

If the task is complete, it is moved to the History List (later used for reporting). Since this is a list itself, it is, of course, easily extensible, particularly when using Microsoft Access. The built-in reporting provides reports on most of the common queries (tasks open, closed, and so on) using both the Workflow Task List and the History List. Administration includes the ability to phase in new workflows, disable workflows, and so on, all the way through Site Administration. Note that if a workflow is ever changed or deleted, any that are in progress will finish with the original flow; if there is a change, any new workflows start with the new flow.

The client interaction comes from within the Office system; Office tools are integrated with the workflow services and are able to check status, initiate new workflows, set operational parameters, and complete workflow tasks. Much like the workspace site integration, workflow activities can be handled from within Office applications without going to the SharePoint site. In the background, SharePoint provides the Office Web Service, which these applications use. In the true spirit of extensibility, these web services are readily available for use in creating custom actions. As a side note, Outlook integrates workflow tasks directly into Outlook Tasks and can handle them offline; tasks can be flagged as completed at any time, and it will update the site when a connection is made.

The development side includes both SharePoint Designer and Visual Studio. Designer provides an orchestration-like ability to create no-code workflows using defined activities. These can include parameter selection for enabling some customization at run-time. Visual Studio can be used to create activities that can be used within Designer (for example, a custom e-mail that sends a follow-up survey), and it is used to create custom workflows directly for SharePoint (or other applications).

Workflows in the WSS platform are Features and are enabled or disabled on a Site Collection or site basis. That also means they are isolated, because workflows can't cross sites. You can build one using Visual Studio, but as far as using out-of-the-box workflows or ones created through Designer, you are limited to the site you are in unless you're using Reusable Content (a library at the Site Collection) or using Site Content Types. Also, workflows are server-based processes, so connection to the site is required (even when using Office). Some key points:

- Lists and libraries can have any number of workflows created for the same list (caution here, as overlapping workflows might get confusing and may even cause some not to work).
- Users must have Edit Items permission to initiate a workflow.
- Status of the workflow is always available through both the site (and list) where it is operating and through Site Administration (all are visible).
- For workflow to operate, a Task List and a Workflow History Task List are required; these can be existing lists (if found) or you can have the process create them for you. (You can also centralize tasks into common task lists to help categorize them for users or to designate a list per workflow type.)
- Only Site Content Types and Reusable Content enable cross-list/library/site use.
- If disabled at the Application level, custom workflows cannot be created in Designer; this has no impact on out-of-the-box workflows because they are always available.

Workflow Limitations

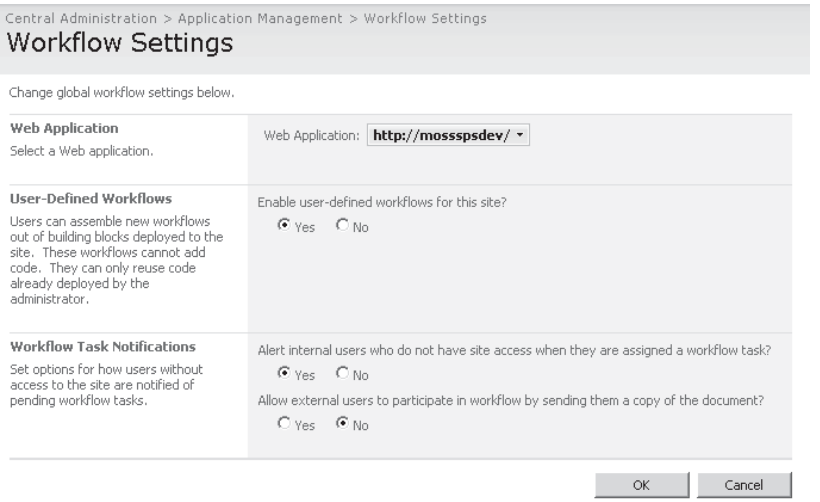
Workflow is a great tool in SharePoint, but it must be used judiciously in relation to the rest of the activities on the site. Too much workflow can choke the entire system, not to mention the users. A few things to consider:

- **Role Changes** During the operation of a workflow, individual tasks are assigned to users. Unfortunately, changes to user roles in the midst of a task are not caught by the workflow. My best suggestion here is that you use Group Task assignments whenever possible so that a member of the group can claim the task—allowing members to change anytime.
- **Scalability** With workflow, scalability can be an issue since it will be one of the most used resources on the site (and by the most number of users, as you'll see next). Dehydrating (saving back to the server as often as possible) helps, but it can become an issue. There are a few ways to deal with this if performance suffers too much. One option is to add more memory on the front-end servers to support the running operations and another is to add an additional front-end server. Finally, if memory usage is nominal, check the SQL connection latency.

- **Number of Users/Processes** The maximum number of people that can participate in a *single* workflow instance is 1000 (I sincerely hope you never try to even come close to this). One of the prices you pay for this is that during a workflow the data is held to track the serialization of the workflow data (variables, current steps, tasks, and so on), and the more tracking requirements, the more storage you need. Overly long workflows should be condensed, because it's better to have one workflow that kicks off 10 others. Remember that since a common history can be used, you can still produce reporting to cover an entire process, even if several workflows are involved.

Configuring SharePoint Workflow

The farm-wide workflow settings are configured within the SharePoint Central Administration. This is covered in Chapter 6, but as a quick review, the settings in Administration are shown next:



These settings control whether user-defined workflows (the kind built by Designer) are enabled, and they set the options for Notifications. Remember, Notifications include things like alerting users who are not members of the site and enabling external users to participate in workflows (via e-mail). These are set-and-forget settings and don't usually change.

Using User Based Workflow

User-based workflows within SharePoint are created and accessible through the actual lists and libraries that run them, as well as to Site Content Types that can be used in lists and libraries. When viewing a list or a library, workflows associated with it are accessible via the list drop-down menus and through the list/library settings. In both, the user-based workflows available are as follows (disabled workflow Features are not shown):

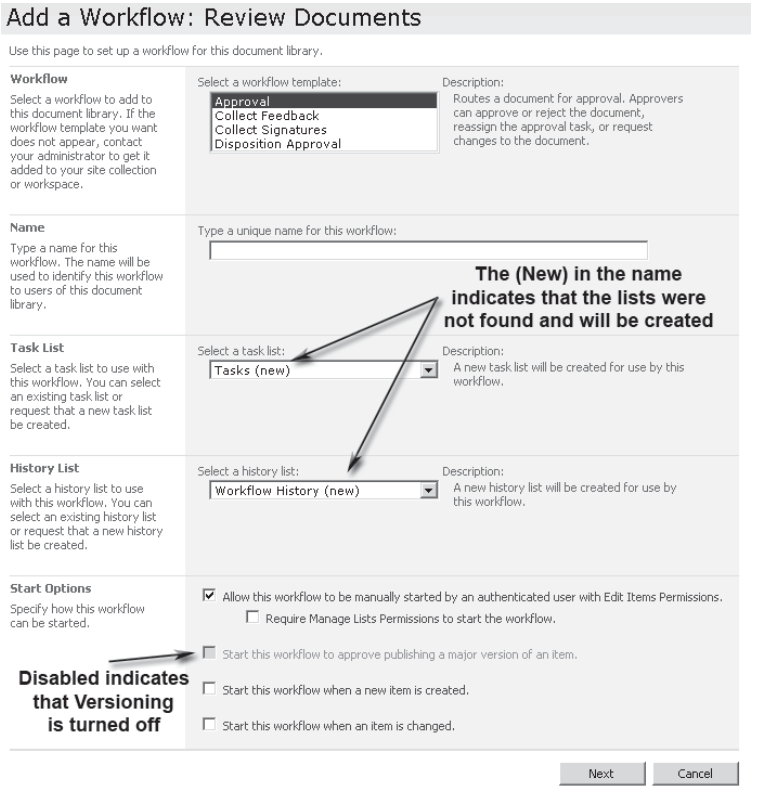
- **Approval** This routes a document through a standard approve/reject/reassign process.

- **Collect Feedback** This routes a document for review; feedback is compiled and sent back to the original author for consideration.
- **Collect Signature** This is for collecting multiple signatures needed to sign-off on a document; the workflow can only be started via an Office Client application (such as Word, or similar) on the client itself.
- **Disposition Approval** This sets document expiration and retention.
- **Three-State Approval** This provides a multistate approval process. This is a Feature and is disabled by default, so it must be enabled by an administrator to be available.

These cover the majority of collaboration-type workflows (even three-state is rare, which is why it is disabled by default), but it is certainly not the end-all to enterprise workflow. This is where Designer comes in. Custom workflows are created by Designer and then published to SharePoint, which enables users to access them the same way as they can the out-of-the-box methods. The same is true for Visual Studio.

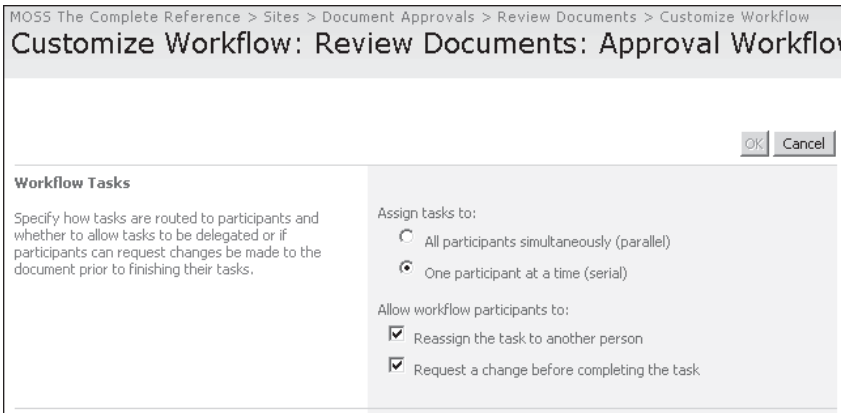
Workflow is best shown by a real demonstration, so you're going to set up a standard Approval workflow in a document library, start it, and then review the tasks. Bear in mind that when you create a workflow this way it's from a template of available workflows. When you save it to the list, it becomes a unique copy that is attached to the list (delete the list and the workflow goes with it); all modifications have no effect on the original template. For this example, I'll use a new site called Document Review and new document library that you'll then attach the workflow to. I'm assuming you know how to create a site and library, so I won't cover that here; you can use any site and library (or list) you like as the process is the same. To attach a new Approval workflow, do the following:

1. Open the SharePoint site, and then open the library you wish to use. Do this by navigating to the site, selecting View All Site Content, and then clicking the library to open it.
2. In the Library View, open the document library settings by clicking the Settings menu then selecting Document Library Settings.
3. When the Settings page opens, look under the Permissions and Management section, and click the Workflow Settings link. This will open the Workflow Settings page:



4. Here you'll specify settings for the new workflow (remember, this is a copy):
   a) Choose the type of workflow you want to use.
   b) Pick a name for this workflow; this keeps it unique within the library and for when you're viewing it from Administration. Be specific, such as Review Documents: Approval Workflow.
   c) Designate the Task and Workflow History Lists. The Tasks and Workflow History Task Lists indicate (New) in the name. In this example, I created a blank site that did not have any lists prebuilt. However, a Team Site, for example, already has a Task List, and at any rate, any Task List (within the site) can be used or a new one can be created.
   d) Set the Start Options, by user, by item created, item updated, or all three. You may also notice the disabled option of starting a workflow when a major version is published (you must turn on Versioning in the list or library to make this option available).

5. When you have set the options, click Next. This will display the Customize Workflow page:



6. For each of these sections:

   - **Workflow Tasks** (shown previously)
     - **Assign Tasks To: All Participants** This means that all assigned users get a task at the same time, which is similar to a group review sign-off. One participant means that the first assigned will get a task, and then the next, and so on. This is basically a typical (linear) approval process.
     - **Allow Workflow Options** This enables/disables Task Reassignment by a user responding to a task and Enables/disables Request a Change Before Completing the Task (if enabled, a change must be made before the workflow will complete).

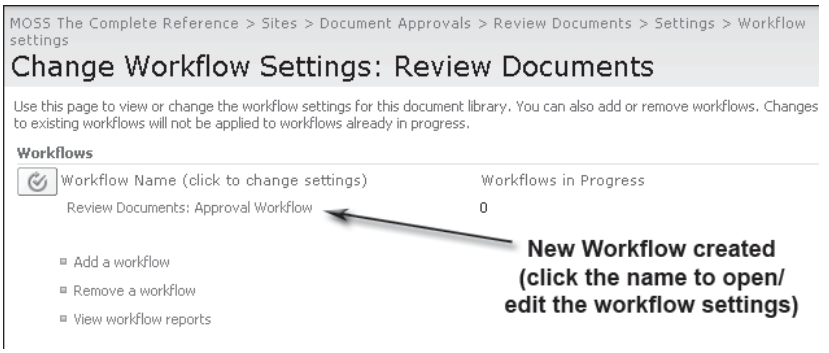
- **Default Workflow Start Values** These settings determine who will be involved in the workflow. Any number of approvers (users, groups, or both) can be assigned in this particular workflow (the Approval) type. You can also set a few options about task creation:
  - **Assign a Single Task to Each Group Entered** If this is enabled, you can assign a single task to a group, and a member of the group can later claim this as their task. If it's not enabled, groups are expanded and all users within a group are assigned a task.
  - **Allow Changes** If enabled, users can be swapped out while a workflow is in process. (This should usually be enabled!)

  Allow Changes provides you with a message that will be sent when the workflow is started, and it should be fairly descriptive of what the workflow is for. The Due Date section provides the ability to set a tickler reminder for unfinished tasks, but this only works in the parallel workflows (one end date). You can also set the

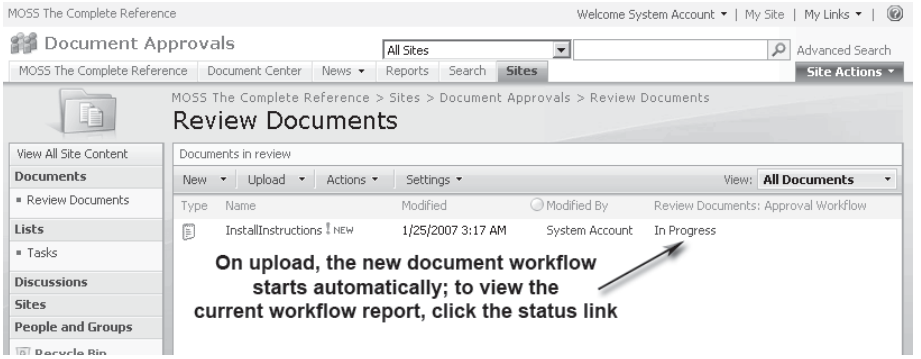
number of days to allow for each user to perform their task (for example, two days to review a document). Lastly, you can set up a Notification List for concerned parties to receive updates when the workflow status changes.

- **Complete the Workflow/Post-completion Workflow Activities**
  - **Completion** These options enable you to set some default ways of ending a workflow (other than simply completing all of the tasks). Completion includes stopping the workflow after a specified number of steps, as well as when the document is rejected or changed.
  - **Post** These activities include any options for when all of the workflow tasks have been completed. As shown for the approval workflow, you can enable the workflow to automatically approve a document as part of content management (if you recall, document approval is assigned, but this overrides that). When you are creating a workflow for a list, you will have some additional options here that include being able to set the value of a column if desired (for example, setting an Approved status).

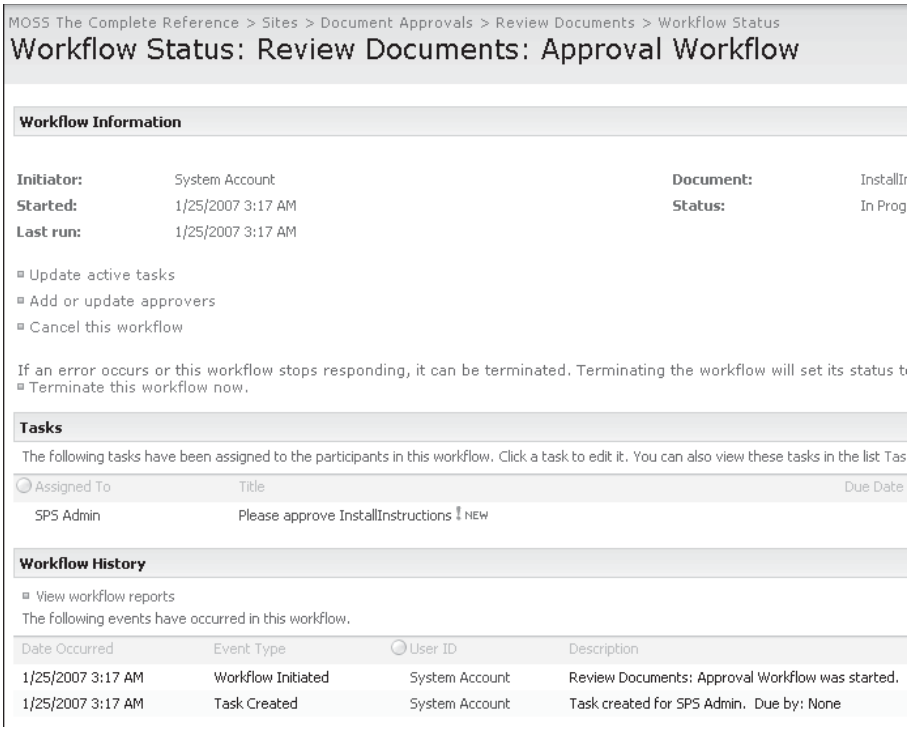
7. After you have made your selections, click OK to create this workflow. Once created, you are brought to the Workflow Status Page (note that once a workflow has been created, returning to the Workflow Settings for the library always displays this page):



Once a workflow has been created, it can be used immediately. To test out the approval workflow you've just created, simply upload a document (and check it in if needed) and the workflow kicks off automatically. This is indicated by a Status column in the Library View:

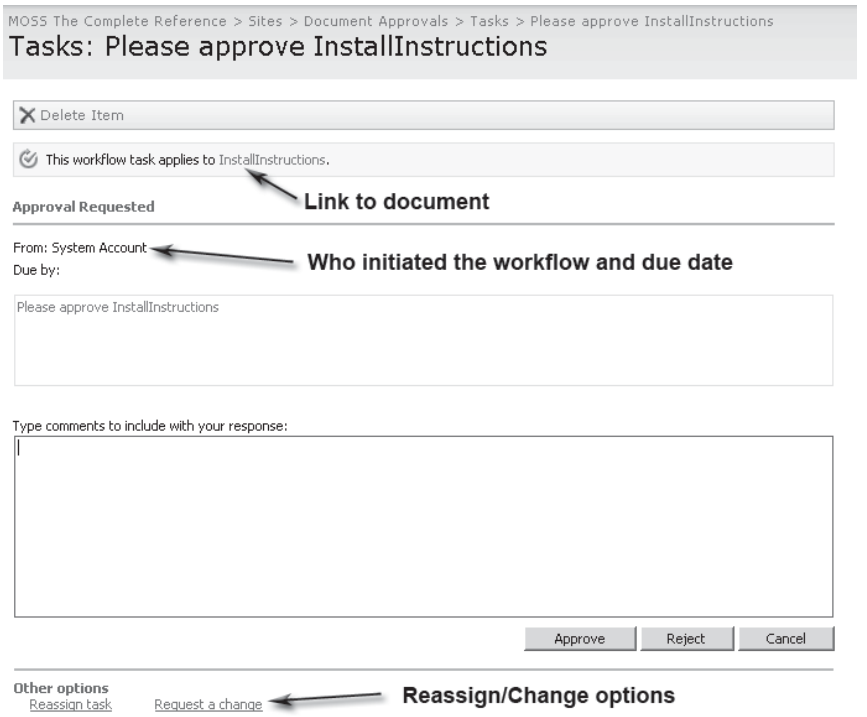


Clicking on the status (In Progress) will open the Workflow Status Report page:



From here, you can see that you have complete control over the workflow. You can see the current tasks, the history and, if necessary, stop the workflow, modify the settings, change approvers, and so on. Note that if you modify a workflow, this only affects the running instance (that stream); changes do not affect the workflow copy or the original template.

Workflows create tasks as needed in the assigned Workflow Task List. You can see this by going to the Task List and viewing the Approval Task that was created when the workflow started. The tasks themselves are based on the kind of request. For an Approval, this includes a Comments section and allows the approver to approve or reject the item. If enabled, this allows the approver to reassign the task. The task description includes a link to the item to be approved (in this case, a document):

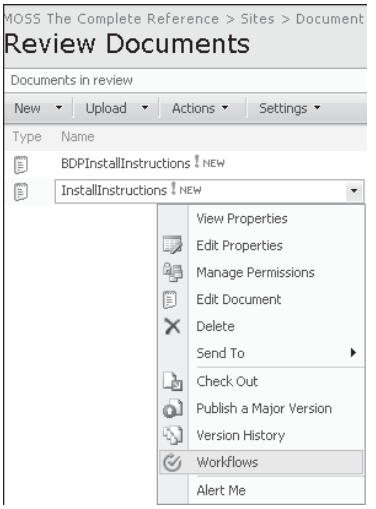


To wrap up the use of ad-hoc style workflows, remember that the key here is ad-hoc. Workflows built this way only apply to the particular list or library where they were created; workflows themselves cannot be copied, moved, or otherwise reused. They can, however, be reused if a list or library is saved as a template or by using Site Content Types and the Reusable Content library.

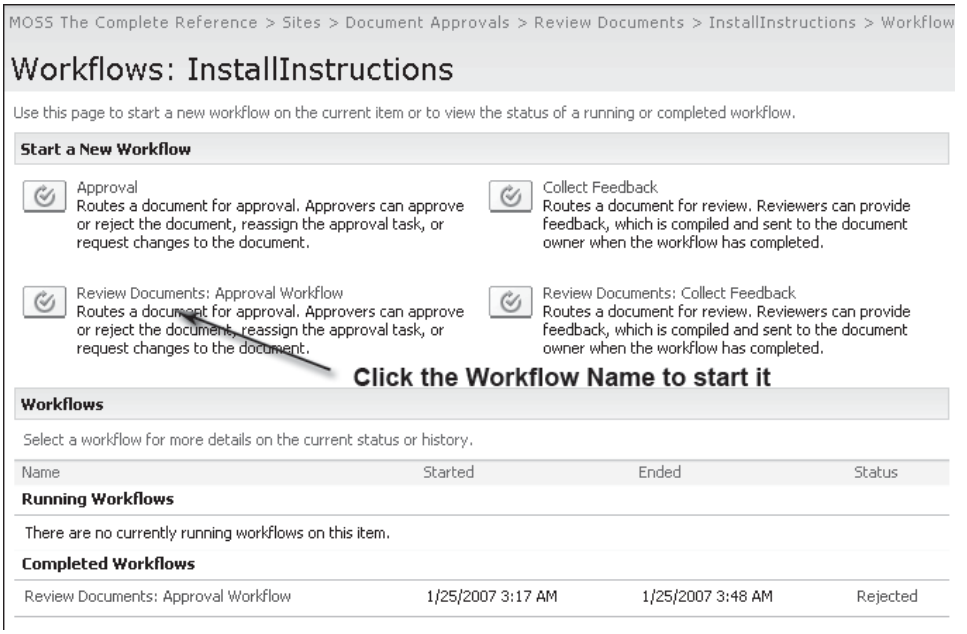
Starting a Workflow Manually

As you saw earlier, one of the options for the workflow is to be enabled for a manual start (you can limit this to users that have Edit Item rights). This enables better control around some kinds of workflows since the process is started only when it's needed. Starting the

workflow (providing this is enabled!) is as simple as using the drop-down menu provided through the list view:



Selecting Workflows will display the workflows available for this particular item. Simply click the name to start it:



Workflow and Microsoft Office

Once a workflow has been created for a library or list, it can be invoked directly from any of the publishing Microsoft Office products like Word. While the integration between SharePoint and Office has always been tight, the workflow options demonstrate how this integration

has been implemented in 2007. All of the out-of-the-box workflows are available (as well as any that have been added, as you will see later). The actual process of starting a workflow in the Office products is pretty much the same, so I won't cover the nuances of each. Using the previous example, we'll use the most common scenario: Word for the Approval and Outlook for the tasks.

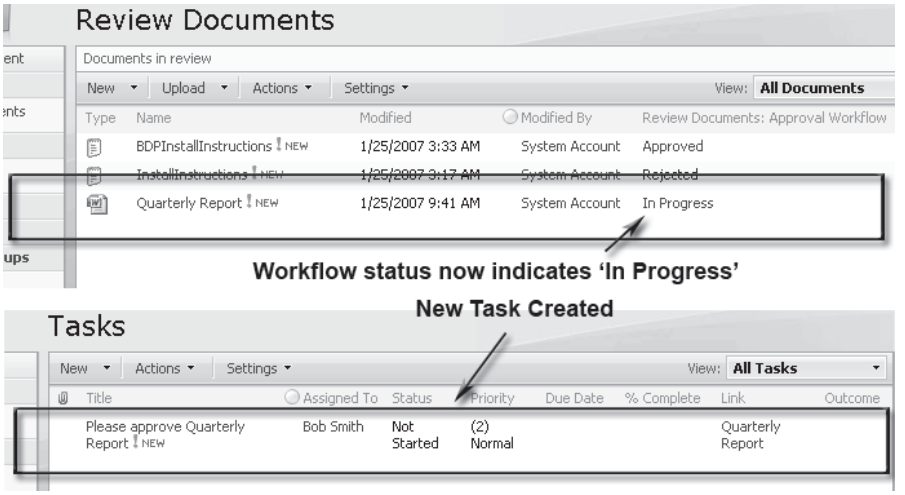
In this example, you'll go through the complete workflow of a document approval process without using SharePoint. The previously set workflow starts when a document is uploaded—though creating a new document in Word and then saving it to the document library does the same thing. For simplicity (and since it is covered elsewhere), you'll simply upload a document to start it and then handle the rest from Word and Outlook.

Before you do this, however, you want to set a specific user for this workflow (the original was assigned to the SPS Admin account). To make this change, do the following:

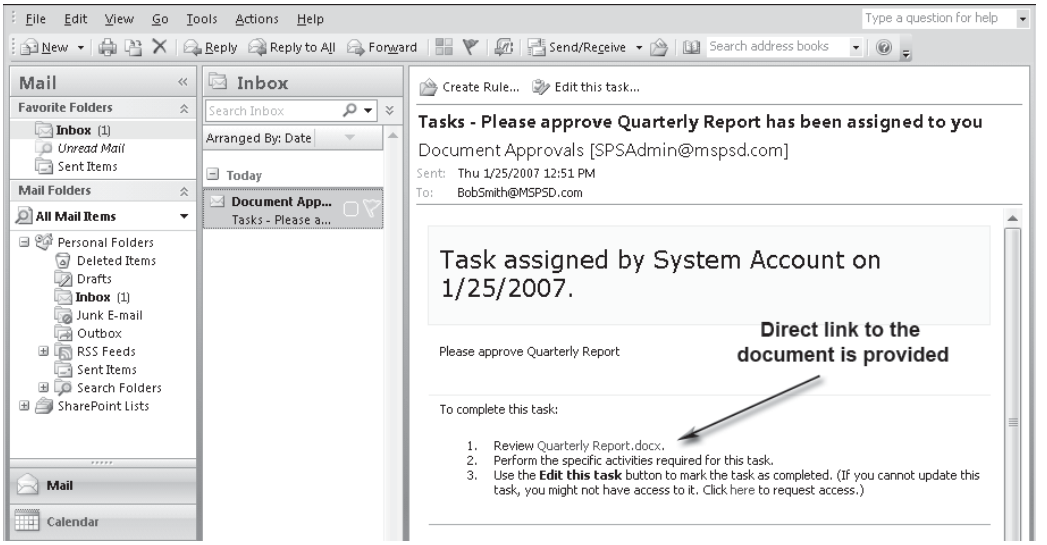
1. Navigate to the site where the library resides, open it via View All Site Content, and then click on the name of the library to open it. Then, from the toolbar, select Settings | Document Library Settings.
2. Under the Permissions and Management section, click the Workflow Settings link.
3. On the Change Workflow Settings page, click the title of the workflow to open it.
4. Click Next to get to the assignments page.
5. Modify the approver to the account desired.
6. Click OK to save your settings.

Starting the workflow:

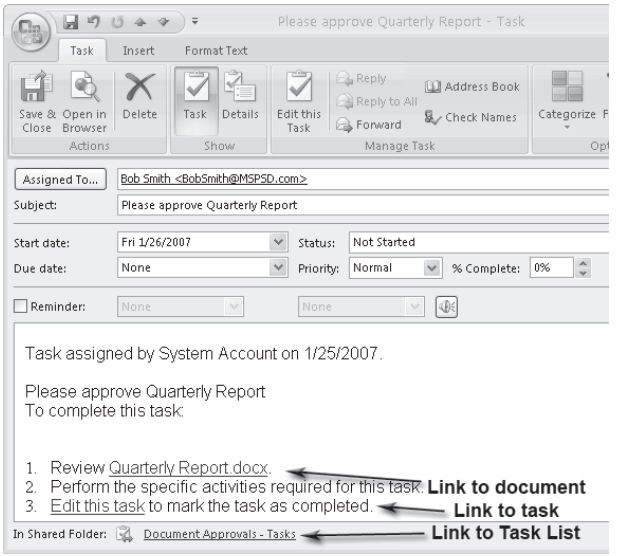
1. Navigate to the site and open the library; from the toolbar, click Upload and upload a document (the type is not important). In this example, I've uploaded a document called QuarterlyReport.docx. Because the workflow is set to start automatically, the status should show In Progress when you return to the Library View. At the same time, a task has been created for your approver:



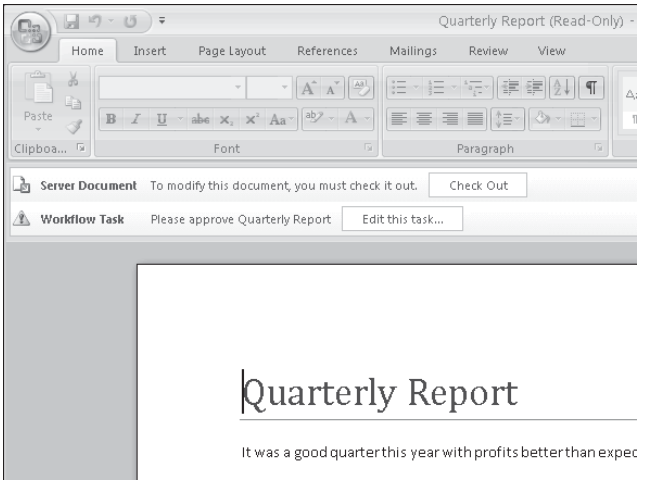
2. To notify Bob Smith that he's been assigned a task, an automatic e-mail is sent and a task (as shown) is added to the Task List. Bob can easily do everything from the UI; however, he can also do the same from Outlook and Word. Before this will completely work, Bob (at some point before the workflow starts) links the Task List to his Outlook by going to the Task List View. From the Actions drop-down menu, he selects Connect to Outlook (all of Bob's tasks in that list then become synchronized with his Outlook Task List; he can connect to any number of Task Lists this way if he wants to consolidate). This connection is required to respond to tasks outside of SharePoint.
3. So, after the workflow has started, let's see what Bob sees. First, in Outlook, he'll receive an e-mail message that indicates he's been assigned a task and that also gives him a link directly to the document he is supposed to review:



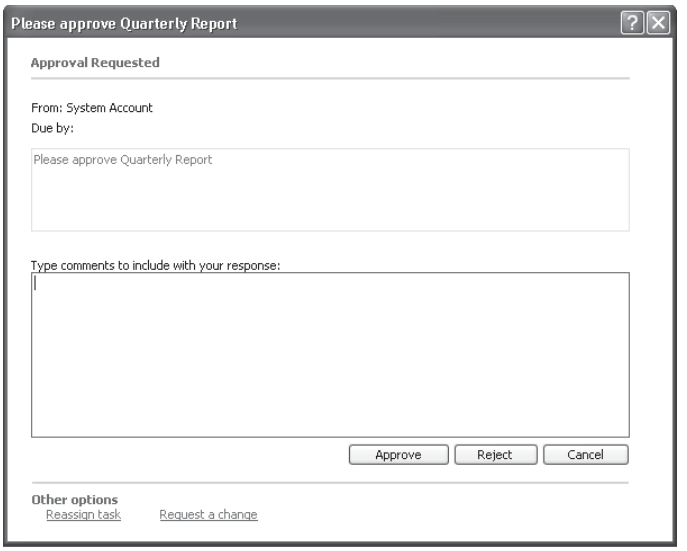
4. Also, providing Bob has linked the Task List to his Outlook, he can also see the task:



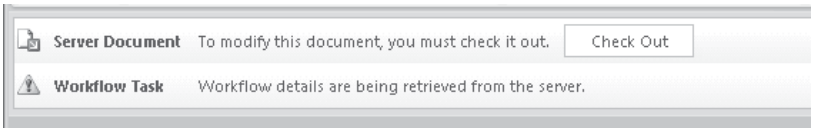
5. If Bob clicks on the link to the document in the e-mail message or from the task, it will automatically launch the application needed, which in this case is MS Word. The document will open as read-only and will include options to check out for editing and also to edit the task directly, as shown:



6. If Bob were to click on Edit Task, it would display basically the same task response page that he would see via the task in Outlook or the task in the list itself:

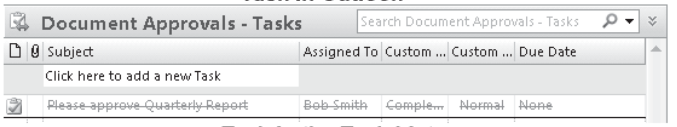


7. For this example, simply approve the document (comments optional). While the request is processed, it will indicate this as so:

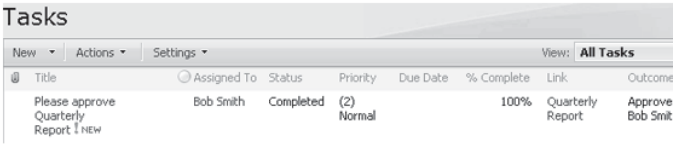


8. Once the task has been saved back to the server, everything is updated as shown:

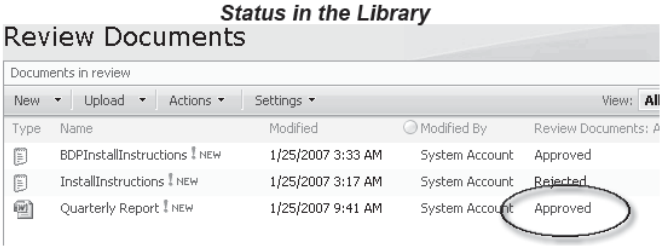
Task In Outlook



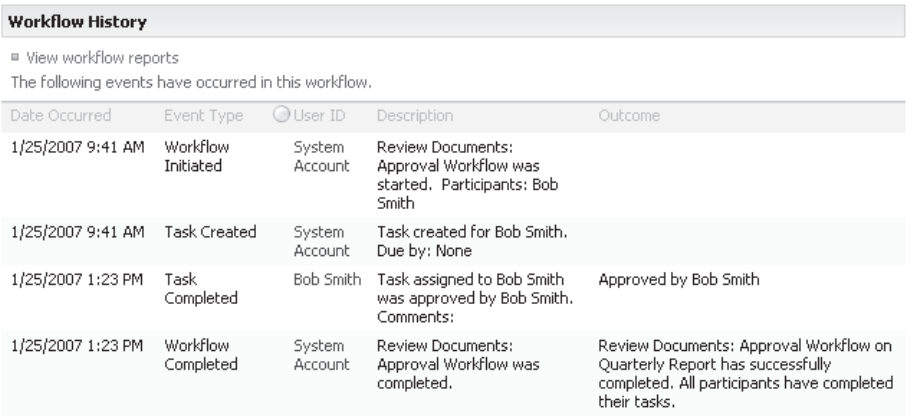
Task in the Task List



Status in the Library



9. Finally, you can check the workflow report by clicking the Completed link:



Workflow History

View workflow reports

The following events have occurred in this workflow.

Date Occurred	Event Type	User ID	Description	Outcome
1/25/2007 9:41 AM	Workflow Initiated	System Account	Review Documents: Approval Workflow was started. Participants: Bob Smith	
1/25/2007 9:41 AM	Task Created	System Account	Task created for Bob Smith. Due by: None	
1/25/2007 1:23 PM	Task Completed	Bob Smith	Task assigned to Bob Smith was approved by Bob Smith. Comments:	Approved by Bob Smith
1/25/2007 1:23 PM	Workflow Completed	System Account	Review Documents: Approval Workflow was completed.	Review Documents: Approval Workflow on Quarterly Report has successfully completed. All participants have completed their tasks.

Building Workflows Using SharePoint Designer

SharePoint Designer is likely the most powerful non-code tool to use with SharePoint for site management, modifying pages, and so on. In terms of workflow, SharePoint Designer enables site designers to create extensive and complex multistep workflows, using both logic and Activities (Actions). Designer can build the types of multistep workflows needed for enterprise-type activities, accomplishing well beyond what can be done in the UI.

There are a few limitations that should be known up front. First, Designer isn't for building components (that's Visual Studio); building workflow is based on using existing actions (such as sending an e-mail). Second, while Designer can create extensive workflows, they can only be created for a specific list or library. Reuse of a workflow can only be accomplished by using Designer to copy from one site to another (and may require a re-edit once there) or by creating templates from lists. Site Content Types don't support user-defined workflows built from Designer (they can be built very easily using the object model). Even with these limitations, most will greatly benefit from what Designer does out of the box.

Designers Workflow Wizards provide the same type of orchestration done in similar fashion for e-commerce pipelines and BizTalk transformations. Once a connection is established to a particular list or library, the wizard allows easy creation of steps with conditional actions. Designer also handles building any forms necessary to get information from the user (either aspx or InfoPath), and no coding is required for any of this. However, workflows can be modified using code once they are created, and the forms can be changed, styled, and so on. The workflow itself uses an XML file (more specifically, an XOML file for eXtensible Object Markup Language). When deploying, Designer actually compiles the XML and forms to create a DLL, which is then published to SharePoint as a Feature.

NOTE *Forms created via Designer do not support client-side validation (forms can be modified by hand if necessary).*

User-defined workflows created in Designer include the same options as the out-of-the-box flows in terms of what event triggers the start of the workflow (and whether it can be started manually). From that starting logic, the workflow is built on Steps, Conditions, and

Actions. A *Step* in the workflow process can have any number of Conditions that are set up in an If-Then-Else-If logical format. *Conditions* are true or false, and for each Condition, any number of *Actions* can be performed. Variable data (used for interacting with the user) includes:

- **Initiation Variables** These provide pre-workflow values that can be used to guide the workflow. Designer automatically creates a form that is used to prompt the user for these values when it is started; these can be used as default values, to update item columns, and so on.
- **Workflow Variables** These provide the in-process values that can be used to pass values between steps within the workflow. Like initiation variables, these can be used to update item columns, compare values, and so on.

Conditions include the following:

- **Compare <list> Field** This allows you to compare a variable or value against a field in the list.
- **Compare Any Data Source** This allows you to compare a variable or value against another list, database, or any other data source.
- **Title Field Contains Keywords** This allows you to select only specific items by title.
- **Modified in a Specific Date Span** This selects by Modified date.
- **Created in a Specific Date Span** This selects by Created date.
- **Created By a Specific Person** This selects by Author.
- **The File Type Is a Specific Type** This is true if the file type matches.
- **The File Size Is a Specific Range of Kilobytes** This is true if the file size within the range is specified.

Actions are what the workflow does when a particular condition is met. These fall into three groups, Core, List, and Task. Within a Condition, any number of actions can be performed in sequence. For example, updating a field in a list might require Check-out, Update, and Check-in as an action block.

Core Actions include:

- **Add Time to Date** This adds a specified time to a date/time field.
- **Build Dynamic String** This is used to compose a string.
- **Do Calculation** This calculates fields or variables.
- **Log to History List** This records a history entry.
- **Pause for Duration** This stops the workflow for a set period of time.
- **Pause Until Date** This stops the workflow until a specific date.
- **Send an E-mail** This sends an e-mail to someone.
- **Set Content Approval Status** This sets the item content approval status.
- **Set Field in Current Item** This sets a value in the current list item.
- **Set Time Portion of Date/Time Field** This sets a specific time for a date/time field.
- **Set Workflow Variable** This sets the value of a workflow variable.

- **Stop Workflow** This stops the workflow immediately (status shows Stopped, not Completed).
- **Wait for Field Change in Current Item** This pauses the workflow until a field has changed.

List Actions include:

- **Check In Item** This checks in an item with comments.
- **Check Out Item** This checks out an item.
- **Copy List Item** This duplicates an item.
- **Create List Item** This creates a new item.
- **Delete Item** This deletes an item.
- **Discard Check-Out Item** This discards a check-out item.
- **Update List Item** This updates a list item.

Task Actions include:

- **Assign a Form to a Group** This assigns a task form to a SharePoint group.
- **Assign a To-do Item** This assigns a to-do task item.
- **Collect Data from a User** This gets information from the user.

As you can see, you can do quite a bit out of the box. In addition, other actions can be created using Visual Studio for accommodating custom business logic, interaction with other systems, and so on. Designer is a bit of a one-off, since the reuse of workflows is limited. However, for development purposes, Designer is a great way to model otherwise complex workflows. Once you have used Designer to create some functionality, you can use the components (outside of Designer) to build additional functionality. There is even a compiler to build workflows called wfc.exe (see more about this on MSDN and WWF). Some other considerations about Designer:

- Workflows in Designer can only use out-of-the-box activities (both SharePoint and Office) and those built in Visual Studio that have been published to the application server.
- Workflow must be authored live in the server (in fact, it is stored on the server).
- Workflows are sequential and are not state machine workflows. This basically means that they are simple and step by step. While a logical if-then-else-if processing is available, branching is not supported.
- You cannot code in Designer; customizations must be done using Visual Studio.

So, with the basics covered, it's time for you to build a workflow. To begin, you need a reason to create a workflow. To keep this relatively simple, let's look at a simple problem statement:

> You have some documents that must be reviewed for legal content, but only when finalized. If the documents are contracts, you need to know the individual who has reviewed it from the large pool of attorneys.

To solve this problem in workflow, you'll do the following:

1. Create a document library specific for this workflow by adding a few columns.
2. Build the workflow to match the conditions and then set it to update the item automatically.
3. Publish the workflow to the library.

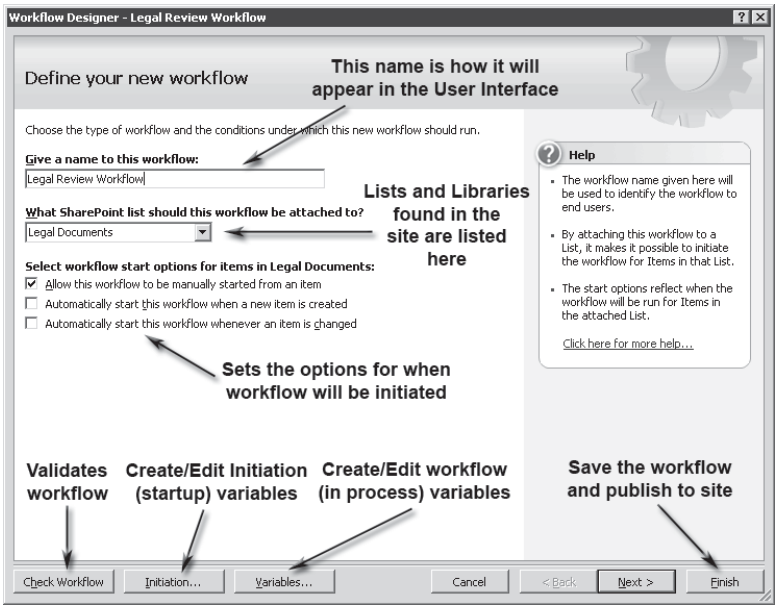
This is a very simple example, but it demonstrates common workflow; this will also give you a chance to see how workflow can manipulate the actual item that triggers an event.

Preparing to Build the Workflow

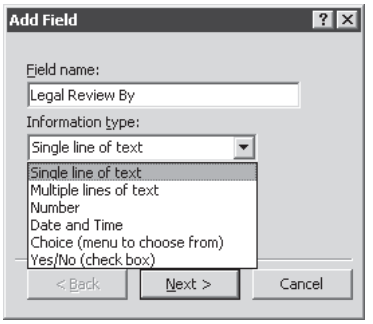
1. Navigate to the site you want to use.
2. Create a new document library called **Legal Documents**.
3. Modify the document library to add two columns: one is called **Legal Review By**, as a single line string, and another called **Review Status**. Review Status has a choice field with the options: Draft, Final, and Complete (you'll use this to control your workflow, as you will see). Make this a Required field.

Building the Workflow

1. Open SharePoint Designer.
2. From the File Menu, select Open Site. Specify the URL to the site (be sure to not include the filename; for example, http://server/SiteDirectory/TestSite/default.aspx must be entered as http://server/SiteDirectory/TestSite). You must be connected to a site to create a workflow.
3. Once the site is opened, select New to open the New Item dialog box, and then click the SharePoint Content tab. Select Workflow or use File | New | Workflow.
4. During the initial start up, Designer will load up the information for the site. Depending on the speed of your system, this can be very quick or may take a few moments; you may see a message box indicating that Designer is downloading information from SharePoint (at this time, it is loading all available activities and so on). Note that you may also receive a warning message that user-defined workflows have been disabled. This indicates this option is disabled in Central Administration (to correct it, go to the Central Administration site as a Farm Administrator). Under Applications | Workflows, enable them). Assuming all is well, you are presented with the Define Your New Workflow page:



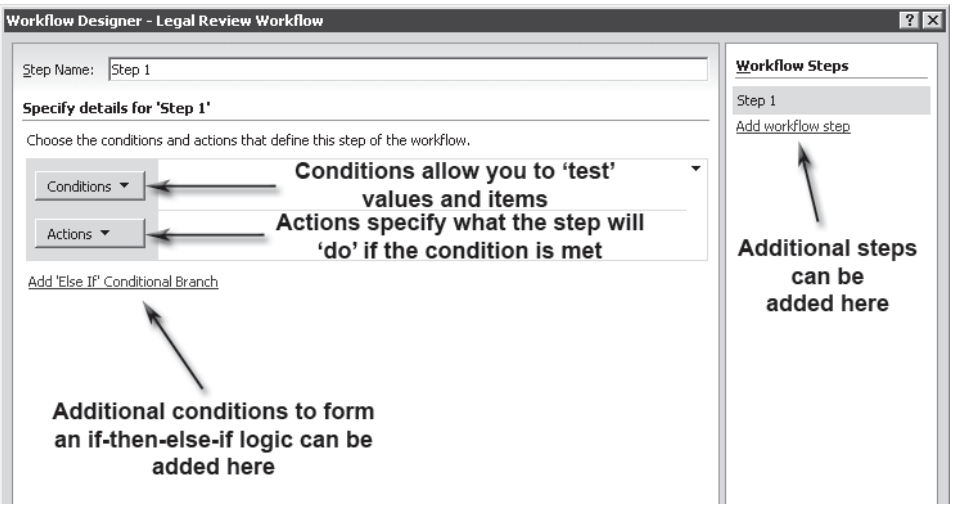
5. From this page, you can define all of the parameters used for and by the workflow. First, you will specify the name (as I have done here), and then you'll select a list from the drop-down menu. In this example, select Legal Documents as shown. You can also choose when this workflow will start up; since this workflow is sort of an end process, you don't want to get items when they are added or updated. This should only be available on demand.



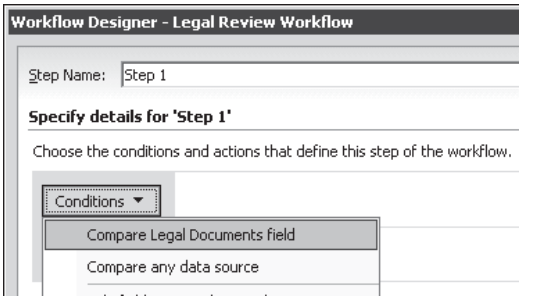
6. Next, you'll want to add the Initiation parameter for your workflow, which in this case is the name of the Legal Reviewer. Click the Initiation Variables button to open the variable definition window (it will be empty). Click the Add button to create a new variable. The variable in this example will be named **Legal Review By** and is a single line of text. Notice, as shown, that you have many options here.

7. Click Next, and then in the Default Value window, enter in a default value for this variable (optional). In this example, it will be set to Enter Name. This value will be shown in the form when the user opens the workflow.

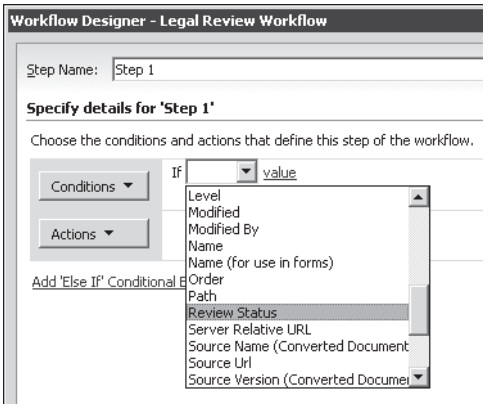
8. Click Finish to create the variable, and then click OK to return to the Workflow Definition. Then click Next, which will open up the Workflow Designer:



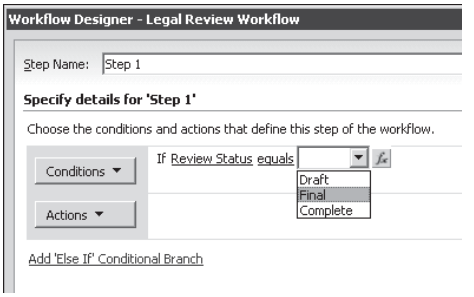
9. To begin this step in the workflow, you need to define a condition. Your condition is to determine whether the review status is set to Final. To set up this condition, click the Condition button, which reveals a drop-down list. Select the Compare <list> Field option (in this example, the list is called Legal Documents).



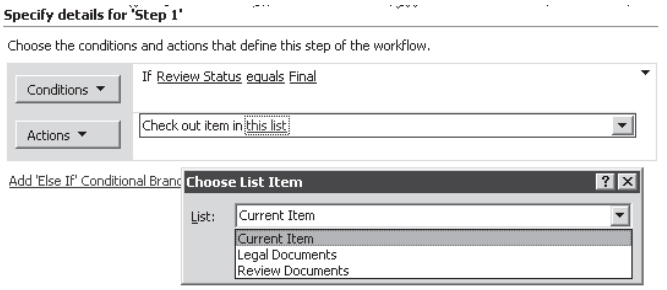
10. This returns a Condition statement that shows whether Field Equals Value. If you click on Field, you will see the list of fields from the list that can be selected. In this example, it should be the Review Status field:



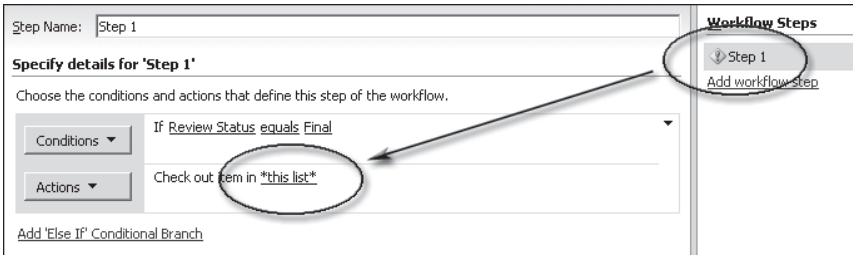
11. The condition in this case will be the default (Equals), so next, click the Value field. From there, you will see that the list was checked and that the values available for the Review Status field are shown.



12. In this example, select Final as the value to test. You may have noticed that there is another button that appears when you are selecting the value (*fx*); this button allows you to use a browser window to select the value you want (it does not have to come from the field itself). From this, you can choose any field within the list, any workflow or initiation variable, or any other list in the site.
13. Next, you need to select actions to perform if the condition is met (true); clicking on the Action button reveals the drop-down list of actions (not all are shown, and you must scroll down to select Show More to see them all). In this example, you are going to update the actual item so you must check it out first. To do this, select Check Out Item, which creates a new action as Check Out Item in This List. This List is a link that opens up a pop-up window allowing you to select which list you want to check out from:

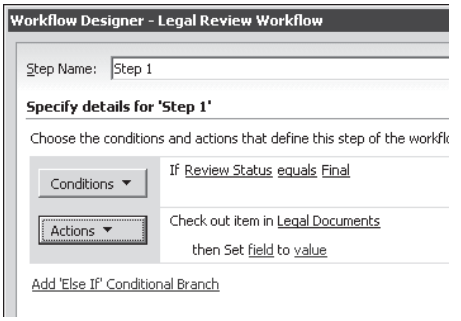


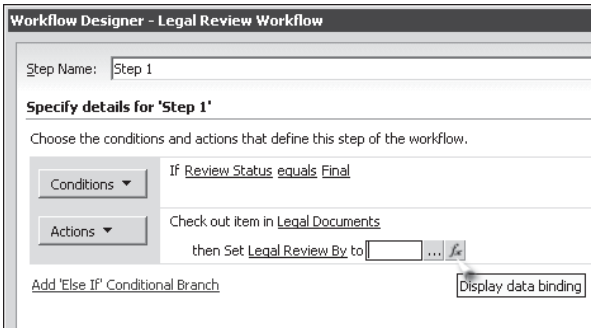
Just as an aside, if you were to click Check Workflow without a list selected, an error is shown:



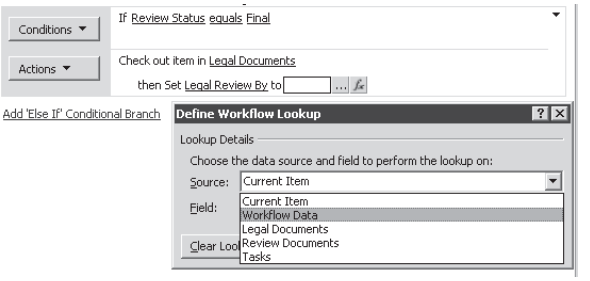
14. When you click This List and the pop-up window appears, you'll select Current Item, indicating that the item that triggered the event is what you want to check out. After you select this, the action will indicate Check Out Item in Legal Documents to indicate the current list.

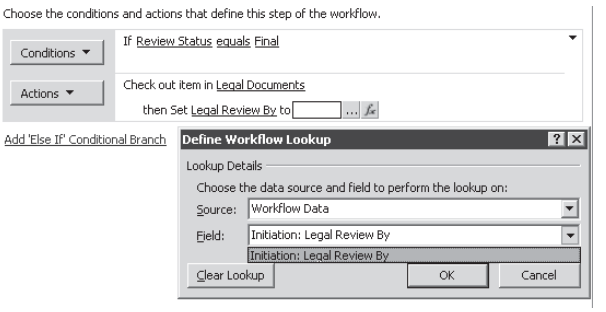
15. Click the Action button to select another action, and this time, select Set Field in Current Item. This will create a new action line preceded by a Then.
16. Click the Field link and select the Legal Review By field from the list. Then, click the value link. This will open a text box and shows an ellipsis button (…), which will open a text box (for easier editing of strings) as well as a Function button (fx).



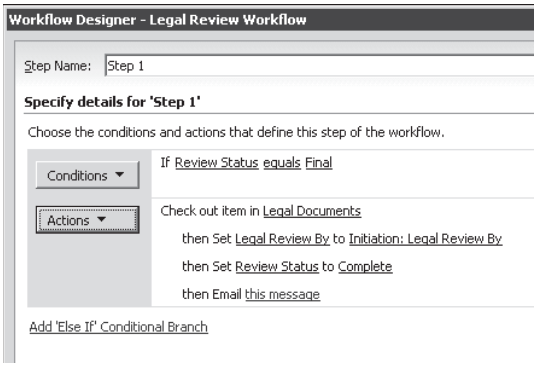


17. In this example, you'll want to update this field with the value the user enters when the workflow starts. Click the Function button, and the Define Workflow Lookup window will appear.
18. As you can see, you can select data from any of the lists in the site as well. In this example, you should select Workflow Data, and then from the Field Selection drop-down menu, select Initiation: Legal Review By (the only one shown).
19. Click OK to save the selection.
20. Click the Action button to select another action, and then select Set Field in Current Item again. Click the Field link, and then select the Review Status field from the list. Click the value link, and then select Complete from the drop-down menu.

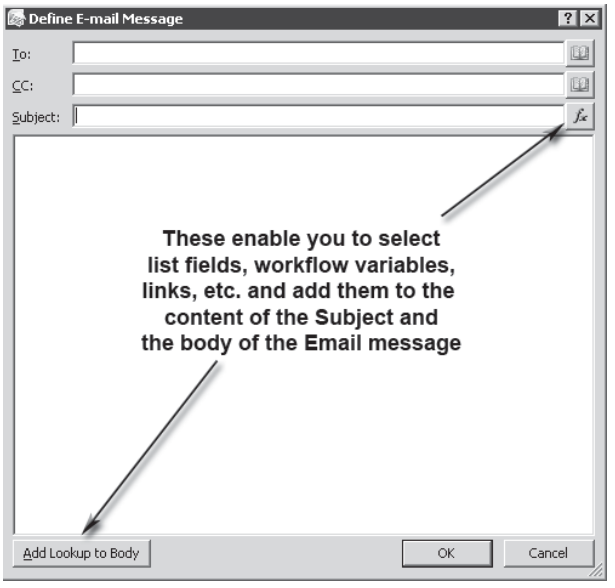




21. Click the Action button to select another action, and then select Check In Document. Next, click the This List link, and then select the Current Item from the list. Click OK to save. Next click on the Comment link and add in a comment stating that, **This has been reviewed.** Then, click OK to save it.
22. Finally, you'll want to notify someone when this workflow is completed, so you'll send an e-mail to your good friend Bob Smith. Click the Action button to select another action, and then select Send an E-mail, which adds another action line:

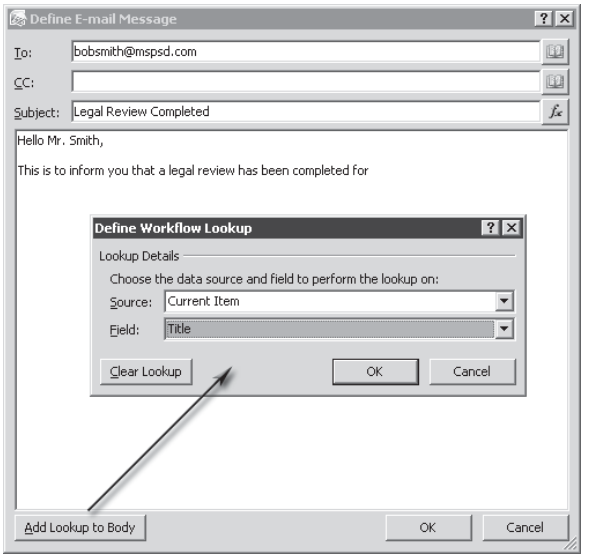


23. To specify the e-mail, click on the This Message link. This opens the e-mail composition window:



24. As indicated, you can use the Function Lookup to provide additional data within the subject and body of the e-mail message. This allows you to specify run-time values dynamically. For example, to add the title of the document within the body

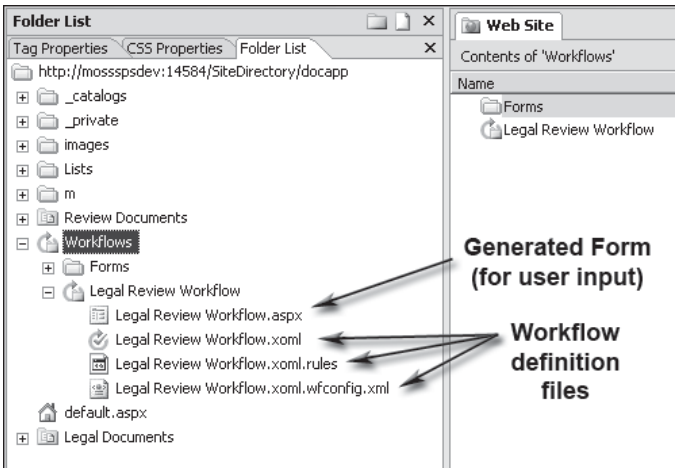
of the message, click the Add Lookup to text button, and then select the item from the pop-up window:



25. You can include additional information if you like; for example, I've specified that I want the URL to the item to be sent as well:

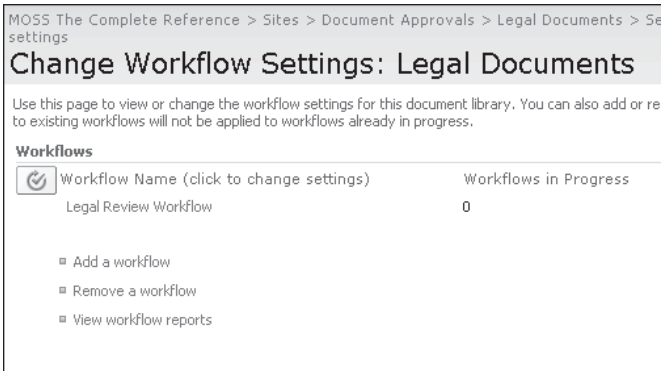


26. Click OK to save the message and return to the Workflow Designer.
27. Validate the workflow by clicking the Check Workflow button. You should receive a message indicating no errors (if not, go back and fix whatever the issue is).
28. To publish the workflow, click the Finish button. This will start the build (compile) and publishing process. You'll see a pop-up box during the upload; if there are any problems, you will be notified.
29. Once the workflow has been published (successfully), the Wizard will close, and you will return to the Designer. If you expand the workflow in the Folder list of the Designer, you'll see what was created, as shown next:



Verifying the Workflow

After the workflow has been published, it is immediately available in the library. If you navigate to the site and open Legal Documents, you can select Settings | Document Library Settings and then Workflow settings. You'll see the new workflow added:



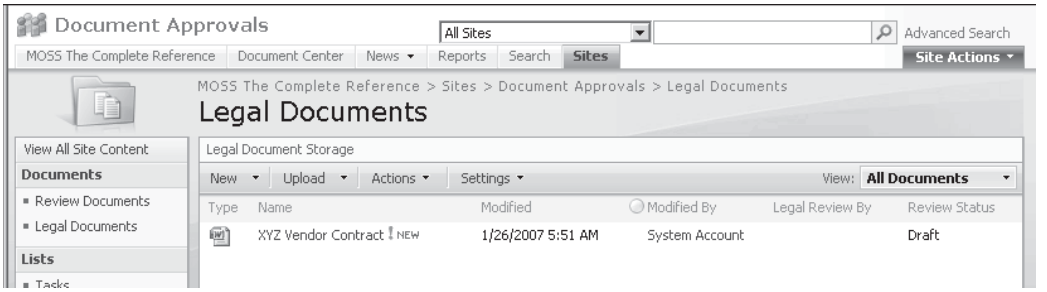
Now, of course, this looks like any other workflow would except that if a user clicks on the workflow to try to edit it, they receive the message that it cannot be customized from a browser and that a program like Designer is required.

Testing the Workflow

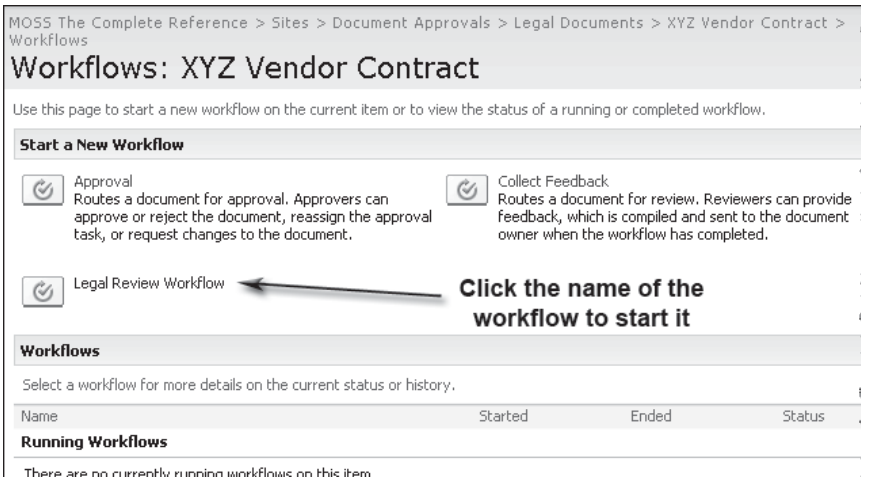
To test the workflow, you have to do something that will meet the conditions required. In this example, this means that a document must be loaded into the library and the status set to Final. Because of the options selected when the workflow is created, it intentionally does not fire when a new document is added (via New, Uploaded, or From Office), but only when it has been modified or is manually invoked. To test out the workflow, do the following:

1. Navigate to the site.
2. Open the list you used when creating the workflow (in the example, Legal Documents).

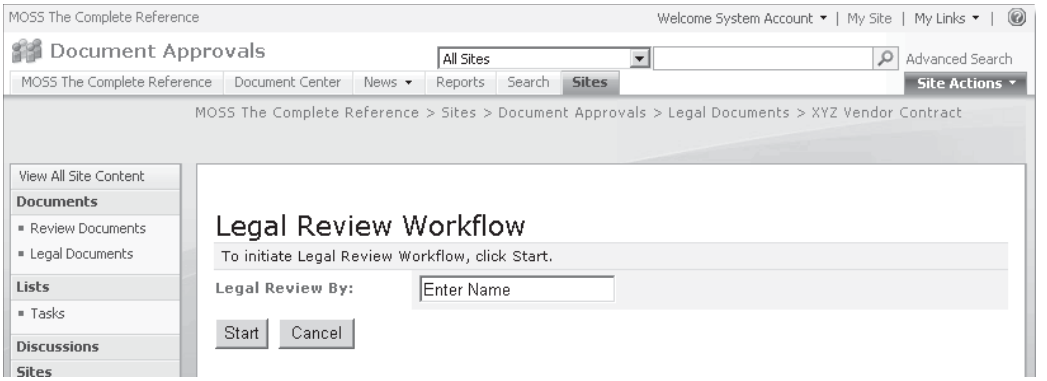
3. Upload or create a new document.
4. Make sure that the Review Status field is set to Draft when saving.
5. When complete, you will see the document in the library, similar to what's shown here:



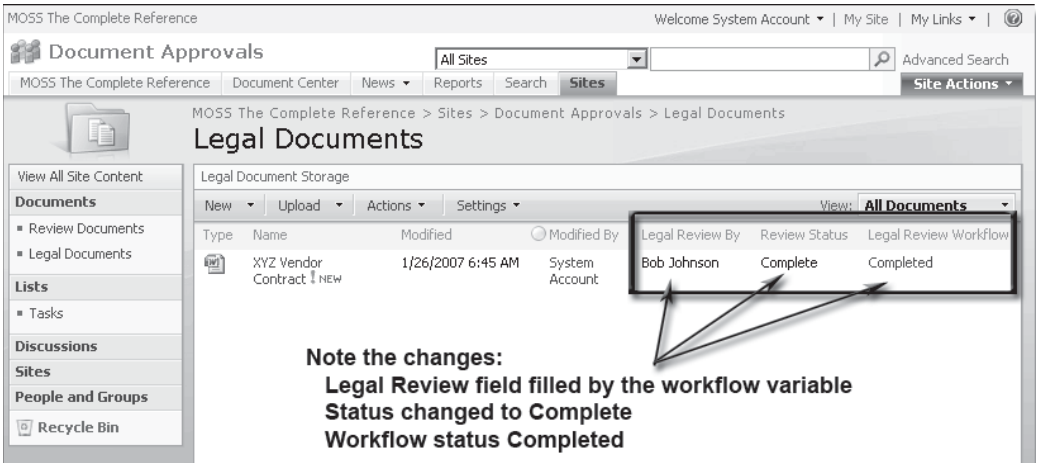
6. Notice that the workflow has not started. This indicates that it did not start the workflow with a new document (the expected behavior).
7. Using the drop-down document menu, select Edit Properties to open the document list item.
8. Change the Review Status to Final and click OK to save the change.
9. When you return to the Document Library View, the workflow will not have started.
10. Using the document drop-down menu, select Workflows. This will display the workflows available for the document:



11. Simply click on the workflow name to start it. When it starts, it will display the custom form it created based on the Initiation variable added, allowing you to get some information for the user (notice the default value set is shown):

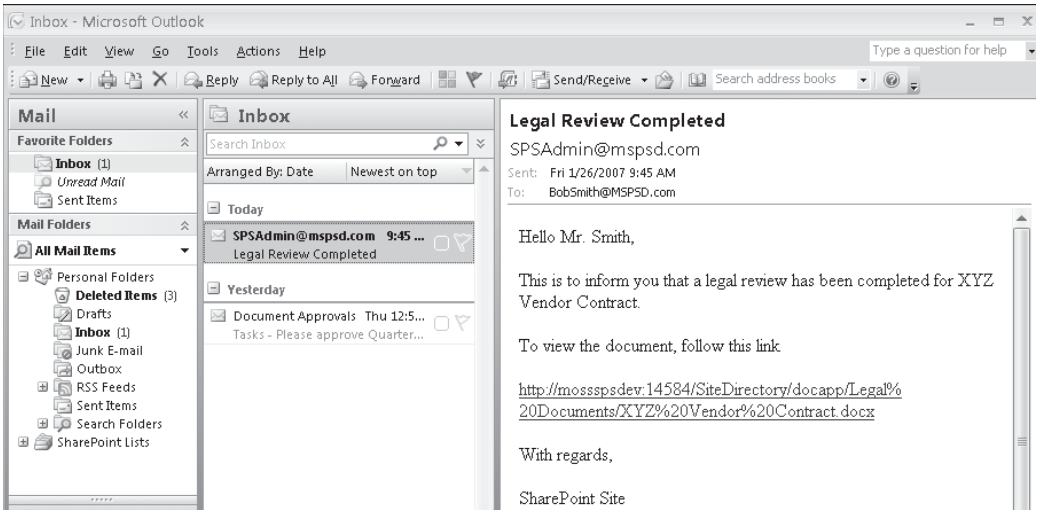


12. Enter a name and then click the Start button to start the workflow. When the workflow completes, you will return to the document library, and you should (if the workflow completed successfully) see the changes made:



13. The final test here is for Bob Smith. In checking his e-mail, you would see the message that was created as the last part of the workflow. Notice that not only

did it include the name of the document, but it also included the URL link to the actual file:



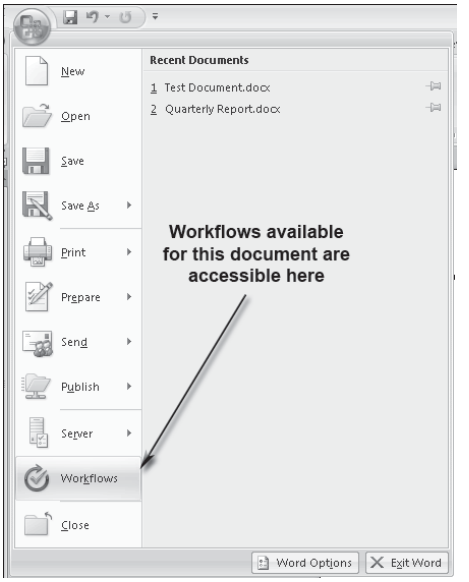
Editing an Existing Workflow

To edit a workflow within SharePoint Designer, simply open the folders to view the workflows, click the + next to the workflow to expand it, and then right-click on the <workflowname>.xoml file. Finally, select SharePoint Designer (Open as Workflow). Note you can use the other options here such as editing XML if desired. For developers, this is a great starting point for building custom workflows using Visual Studio.

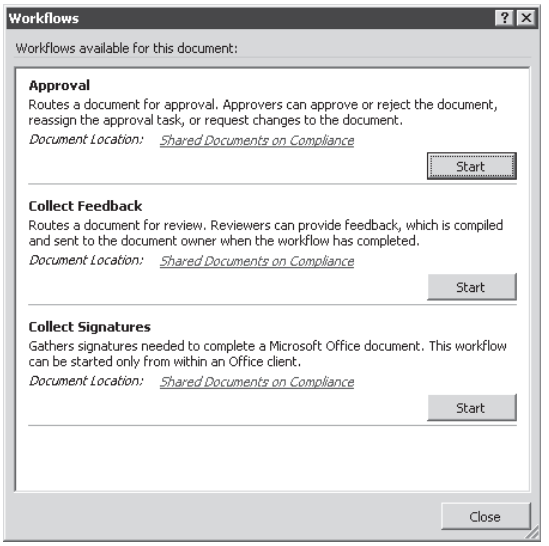
Starting Workflow from Microsoft Office

You have now walked through the Office integration between Outlook and Word as a part of a workflow initiated through SharePoint. Let's take a quick look on the flip side in how you go about starting workflow (both out of the box and those created with Designer) when using Office tools like Word, Excel, and PowerPoint. As a part of the new ribbon menu format for Office, access to the workflow is provided through the Microsoft Office menu. When you have a document open in Word, for example, you'll see the workflows available.

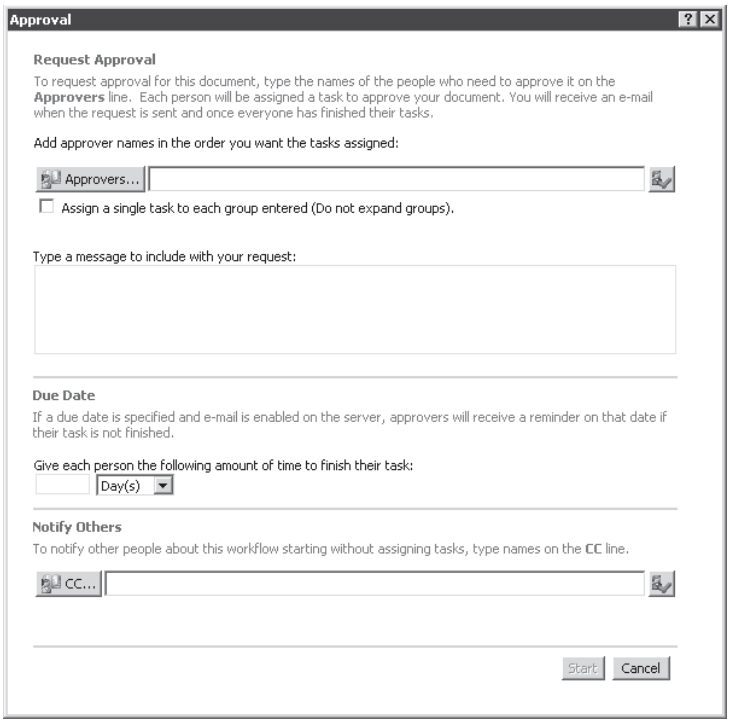
FYI, if you attempt to start a workflow before the document has been saved to SharePoint, you'll see a pop-up window that states the document must be saved before starting a workflow. Simply click Yes to accept this.



Workflows available are based on the site that the document is saved to, but only those that are activated are shown:

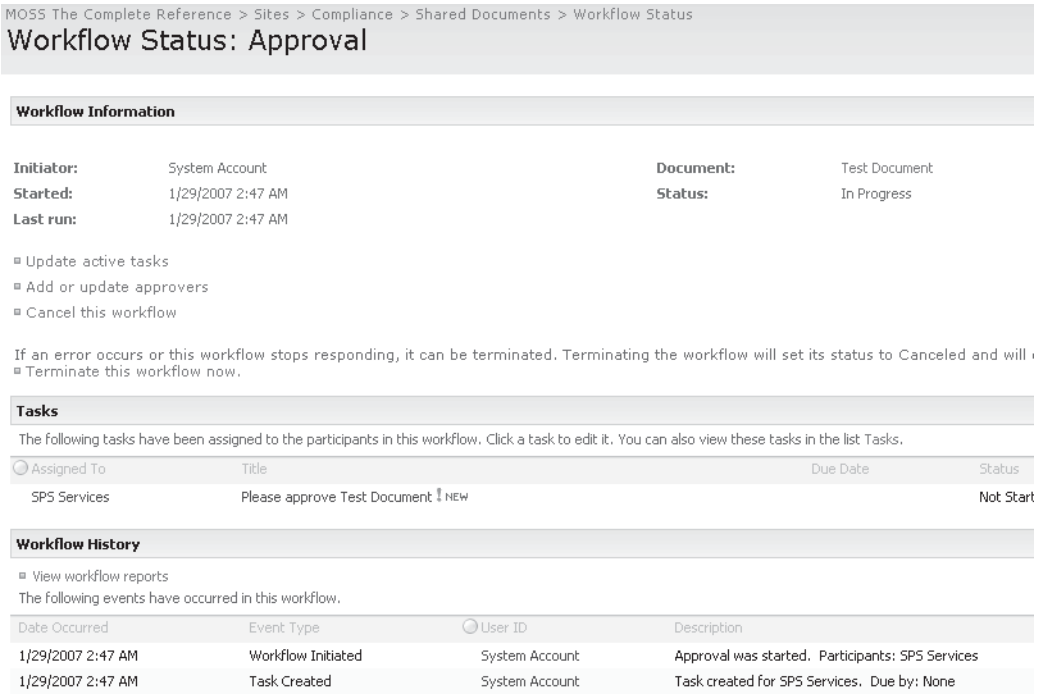


Simply click the Start button for the one you want. When it starts, you will be prompted for any preliminary information (aka Initiation values), the same as if you'd started it from SharePoint. For example, if you started the Approval workflow, you would see a similar form to what you see in SharePoint:



In addition to initiation, workflows started from Office also validate that any preliminary requirements are covered before you can start the flow. For example, if you start a Signature Collection workflow, you'll receive the message, "The current document must include signature lines to collection signatures. Please insert signature lines prior to collecting signatures." This is obviously telling you that you cannot start the workflow because the required Signature fields were not found in the document.

Once you have met the preliminary requirements and entered any required forms, the workflow is started on the server (a pop-up box will indicate that it is working). When the workflow has started, you are simply returned to the document. To view any workflow tasks assigned to you on the document, you can select this from the Microsoft Office menu. Select the Office button in Word, and then select Server | View Workflow Tasks. If you have tasks, they will be displayed in the pop-up window (these are clickable and will bring you to the task). The full status of the entire workflow is also accessible through the link shown. If you click this, you will be brought to the SharePoint site and the Workflow Status page displayed:



Summary

As you've seen in this chapter, the workflow capabilities provided out-of-the-box are pretty extensive and, in most cases, satisfy what most of us need on a day-to-day basis. You've also seen how these capabilities can be easily extended by power users with SharePoint Designer and even more complex ones created using Visual Studio (though I didn't do a walkthrough of that here as it would take another chapter itself!). While powerful, you must also use caution with workflows to prevent an excessive amount of work for the user; use them, but wisely.

CHAPTER 17

Features and Solutions

As mentioned throughout the book, a primary goal in the development of Windows SharePoint Services Version 3 was to make it easily extensible for developers, not only as an application, but also as a platform to build other applications. While you have always had the ability to add on functionality to SharePoint (mainly via web parts), the integration of content management in WSS and MOSS means there are many more items that can be added, in the form of site definitions, page layouts, custom lists, event receivers, and of course, web parts.

Understanding that to make a platform really extensible it must also be easy to deploy to, Microsoft introduced the new SharePoint Features and Solutions. Features and Solutions are the methods for provisioning or adding on extensions to SharePoint, from a complete site definition to a web part. There's not much difference between a Feature and a Solution. Features are used to deploy things like Master Pages and Page Layouts, whereas Solutions are used to deploy things like web parts and web components. Note that the two terms are used interchangeably; if deploying an assembly, it's a Solution; otherwise, it's a Feature.

From the developer's perspective, Features and Solutions are the methods for adding something to SharePoint; they package up the elements (images, assemblies, pages, and so on) and deploy the package to SharePoint. Once installed, they can be activated or deactivated using the STSAdm command tool or by the site administrator. This permits Features/Solutions to be enabled selectively (allowing phased deployment, which is particularly useful in content management), but it also provides a backout mechanism if something goes wrong. Features and Solutions can be deployed to any level: the Farm, Shared Services, Site Collection, or site. When activated, they become available to users.

Anatomy of a Feature

Features and Solutions are based on two primary XML files: a Feature Manifest file and an Elements Manifest file. The Feature Manifest file is used to identify a particular Feature in SharePoint, including giving it a name and a global unique identifier (or GUID) that becomes the Features ID in SharePoint. This also contains an element that specifies the actual file location of the Elements Manifest File(s) (technically there can be more than one Element Manifest for a single Feature). The Element Manifest file is used to identify the components (pages, script, images, and so on) that make up the Feature. You'll see this in the example that follows.

When they are ready to be installed, all of the files required to install the Feature are physically copied to the SharePoint Hive (the primary Feature files go into <drive>:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\TEMPLATE\Features in a folder named after the Feature). Once there, the Feature itself is first installed into SharePoint, and then activated. Finally, it's enabled from either the command line or from the User Interface. Removing a Feature is done in reverse; it is first disabled, and then deactivated by retracting it. Then, it can be uninstalled.

Features that are installed in a SharePoint site are always accessible from the Site Settings page, under either Site Features or Site Collection Features. Site Features are those that are available (installed) only on the current site, and Site Collection Features are those that are available within any site in the collection.

Solutions

Solutions are just like Features, and they are installed the same way, but they do have a few additional components. Unlike Features that are used for things like Site Templates, which is primarily a file copy, Solutions usually involve event receivers, web components, and web parts. Because these have a number of additional settings (such as loading a DLL into the Global Assembly Cache), there is an additional file called Manifest.xml, which is used to define the specifics of the Solution, and which is later used to create WSP files. WSP files are the equivalent of CAB files, which themselves are really a form of a compacted file or Zip file. Details on building the WSP file are covered in Chapter 20.

Feature Manifest File

The Manifest file defines the Feature or Solution for SharePoint, including the name, description, and ID. It also identifies the components (specifically to the Elements Manifest files, web parts, and so on). The Feature file includes the following elements:

- **Feature** This element is the outer wrapper of the XML that defines the Feature and specifies the XML Namespace (http://www.schemas.microsoft.com/sharepoint/). It includes the Feature Title, the GUID ID, a Description, the Version, the Scope (this sets the scope for the Feature, Web, Site, Web Application, or Farm), a flag to indicate whether it's hidden or not, and a pointer to a default resource file (if any).
- **ActivationDependencies** This element is used to identify the IDs of any Features that it's dependent upon, which prevents users from enabling Features in the wrong order. This is only used if the Feature or Solution is dependent on one or more being installed.
- **Elements Manifests** This lists any Elements Manifest files included in the Feature.
- **Properties** This defines Key-Value pairs as properties, which can be used as presettings for a Feature.

Elements Manifest File

The Elements Manifest file is used to identify the elements of the Feature. This includes modules (DLLs), receivers, and others that determine where elements of the Feature are to be placed or loaded. The entries in this file depend on what is being installed. For example, Receivers have a Receivers element. If deploying a web part, however, DLLs are referenced as Modules. The primary elements are somewhat self-describing. These include:

- **ContentType** This Feature defines a content type.
- **ContentTypeBinding** This Feature binds a list to an existing content type.
- **Control** This Feature is a web control.
- **CustomAction** This Feature is a custom action.
- **CustomActionGroup** This Feature is a custom action group.
- **DocumentConverter** This Solution is a document converter.
- **FeatureSiteTemplateAssociation** This Feature is associated to a specific Site Template.
- **Field** This Feature defines a field in a List Feature.
- **HideCustomAction** This Feature hides a custom action (for example, removing Delete Web).
- **ListInstance** This Feature creates a specific list instance.
- **ListTemplate** This Feature is a List Template.
- **Module** This Feature defines a single module included in the Feature.
- **Receivers** This Feature defines one or more receivers included in the Feature.
- **UserMigrator** This Feature is undocumented (usage unknown).
- **Workflow** This Feature is a workflow.

Each of the primary elements has attributes that define the specific items or directives for the type of Feature (for example, ContentType has an attribute called BaseType, which defines the kind of generic SharePoint type it is based on). All of these are documented in the SharePoint Software Development Kit (SDK), so I won't repeat them all here. However, for the example that follows and the following chapters on "Building Event Receivers" and "Building Web Parts," two key types are Modules and Receivers. These break down as follows:

- **Module Element** This is used to specify a file (specified using the File Element) that is part of the Feature or Solution and defines where the file will be placed within SharePoint; the Module Element attributes define the location:
  - **Name** This is the friendly name of the file (what users will see); for example, Our Custom Master.
  - **List** This is the actual ID number of the list where the file will be stored; for example, 116 indicates the Master Page and Page Layout Gallery.
  - **URL** This specifies the Relative URL to the list where the file will be stored. For example, _catalogs/masterpage indicates the Master Page library.

The Module File element attributes include:

- **IgnoreIfAlreadyExists** If this is set to True, the file will be overlaid if it already exists.
- **Name** This specifies an optional (but suggested) name for the file or module.
- **NavBarHome** If set to True, file will be included as a URL in the top navigation bar (this is typical for Site Definitions).

  - **Path** This optionally contains the URL for the file within SharePoint (in most cases, this is just the name of the file). This must be in URL format (no special characters or spaces).
  - **Type** If specified, this indicates that the file be cached in memory on the front-end web server. This can be set to either Ghostable or GhostableInLibrary; the latter is used if the file is stored in a Document Library.
  - **URL** This specifies the actual URL for the file (same as Path, but this is required). This must be in URL format (no special characters or spaces).
- **Receivers Element** This is used to group one or more Event Receivers (the Receiver child element) that attaches them to lists or libraries. Attributes for the Receiver element include:
  - **ListTemplateOwner** This is GUID of the List Template Owner (the GUID ID of the list).
  - **ListTemplateId** This is the List Type ID for the type.
  - **Name** This is the name of the Receiver.
  - **SequenceNumber** This specifies a firing order if there is more than one Receiver.
  - **Assembly** This sets the DLL's assembly information (fired when the event occurs).
  - **Class** This specifies the name of the Class (in the Receiver).
  - **Data** This is optional data that can be passed to the assembly.
  - **Filter** This specifies any filtering data to be passed to the assembly.

Solution Manifest File

Web parts and components also use a Manifest file, which is used when creating a WSP (CAB) file. This file is similar to the Elements Manifest, but it handles more specifics around DLLs. Elements include:

- **Solution** This element is the outer wrapper of the XML that defines the Solution and specifies the XML Namespace (http://www.schemas.microsoft.com/sharepoint/). This element includes three attributes: SolutionID (optional) set to the GUID of the Solution; DeploymentServerType set to either ApplicationServer or WebFrontEnd (the default) to set which kind of server it should run on; and ResetWebServer. If set to false (the default), it has no effect, but if set to true, it will perform an IISReset after installing the Feature.
- **Assemblies/SafeControls** This element defines the assemblies included in the Solution (for web parts, there is usually only one). Within the Assembly node is the SafeControls element; this defines the SafeControl entries for the web part, which will later be added to the SharePoint site web.config file when deployed (the SafeControl entry tells SharePoint that the assembly is valid to use). This element includes a parameter called DeploymentTarget. If this is set to WebApplication, the assembly is deployed to the Web Application bin directory (the BIN folder in the web's virtual directory called _app_bin), or if set to GlobalAssemblyCache, it will

deploy it to the Global Assembly Cache (GAC) in c:\Windows\Assembly. This is only used if DLLs are part of the Solution.

- **ApplicationResourceFiles** This element defines any application resource files used by the Solution, with the Location attribute specifying the relative location to the file. These are placed in the appropriate SharePoint site virtual directory in the Resources folder when deployed. This is used only in Features and Solutions that use Resources.
- **CodeAccessSecurity** This element contains the PolicyItem element, which contains the PermissionSet and Assemblies elements that specify the code access security for the Solution. When deployed, this is added to the SharePoint policy file. This is used only in Solutions that need code access security to be set.

NOTE *Code Access Security is beyond the scope of this book; for more information, as well as a tool to assist in scripting permissions, you can use the Code Access Security Policy Tool (Caspol.exe). This utility is installed with the .NET Framework SDK, or you can go to http://msdn2.microsoft.com/en-us/library/cb6t8dtz(VS.80).aspx.*

- **DwpFiles** This element contains the DwpFile entries needed to specify web part definition files that are part of the Solution, which get placed into the SharePoint site virtual directory under the WPCATALOG folder. This is used only in web part Solutions and for SharePoint web parts (now replaced with .NET web parts).
- **FeatureManifests** This element defines the Features included in the Solution, for example, MyFeature\Feature.XML. When deployed, the folder will be created in the SharePoint root TEMPLATE\FEATURES folder.
- **Resources** This element contains all the Resource elements that define files (images, script, CSS, and so on) that are to be included with the Solution. One example is MyFeature\Image1.bmp—like Features, which ends up in the SharePoint root TEMPLATE\FEATURES folder. This is only used if the Feature or Solution has a custom Resources file.
- **RootFiles** This element contains the RootFile elements that define files to be placed in the SharePoint root folder itself. Note that in terms of web parts, you should avoid using this to prevent clutter.
- **SiteDefinitionManifests** This element contains the SiteDefinitionManifest elements that describe site definition folders. The SiteDefinitionManifest includes the attribute Location and the WebTempFile (web template file) definition, as well as where it should be placed. By default, this is the SharePoint root template\sitetemplates folder.
- **TemplateFiles** This element defines the files, pages, CSS files, images, and so on that are used by the Solution. Files specified here are put into the SharePoint root folder called Template. These are only used in definition (site/list) Solutions.

WSP File

WSP files are the equivalent of Microsoft CAB files used to install Solutions with assemblies. The complete description and creation of a WSP file is covered in detail in Chapter 20.

Building a Feature

Having read the previous sections, you probably think that building a Feature or a Solution is pretty complicated. I understand this, because I had the same thoughts when I first reviewed how they worked. However, I am sure you will find, as I did, that once you've done it, they are really very easy to set up and maintain.

Creating a Feature or a Solution is done the same way, regardless of what you are creating (whether it's a site definition, page, event receiver, web part, and so on). The basic steps are to create a new project, create the XML files needed for the Feature, add any other components or assets, and then package it up to install into SharePoint. When you want to add something to the Feature, you can simply append it, repackage it, and away you go.

The benefit here is not so much on the setup side (those files still have to be created by hand), but on the deployment side when they are turned over for QA or production. To install or uninstall a Feature or Solution only requires a few clicks. And before you complain, the manual part of the setup replaces what you would have to do if you created an installation package yourself anyway—so there's no time lost in the process. In fact, the Solution deployment for definitions and web parts is automated in Visual Studio 2005 (with WSS Extensions), as you will later learn in Chapter 20.

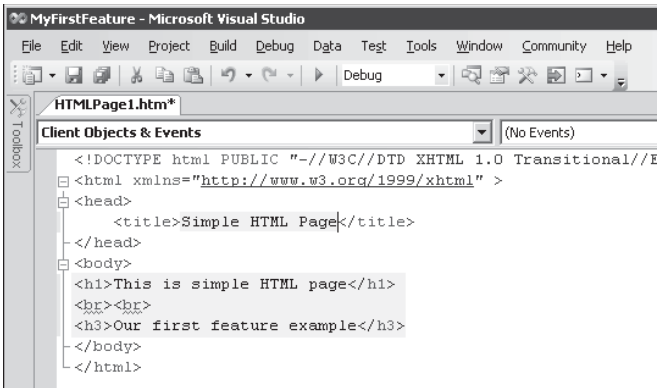
NOTE *Whenever you create an XML file in Visual Studio, it adds an XML declaration line at the top that starts with <?XML. In almost all cases when dealing with SharePoint XML Feature files, this line can be included or omitted. You will see examples of both ways.*

So, to take you through the entire development process: in this example, you are going to create a Feature that deploys a custom HTML page to a SharePoint Site. What you are deploying is immaterial; this could be a Master Page, a Page Layout, a site definition, or anything similar. Deploying multiple files is just as simple (see the example following this next one).

Assembling a Feature

To create a Feature, do the following:

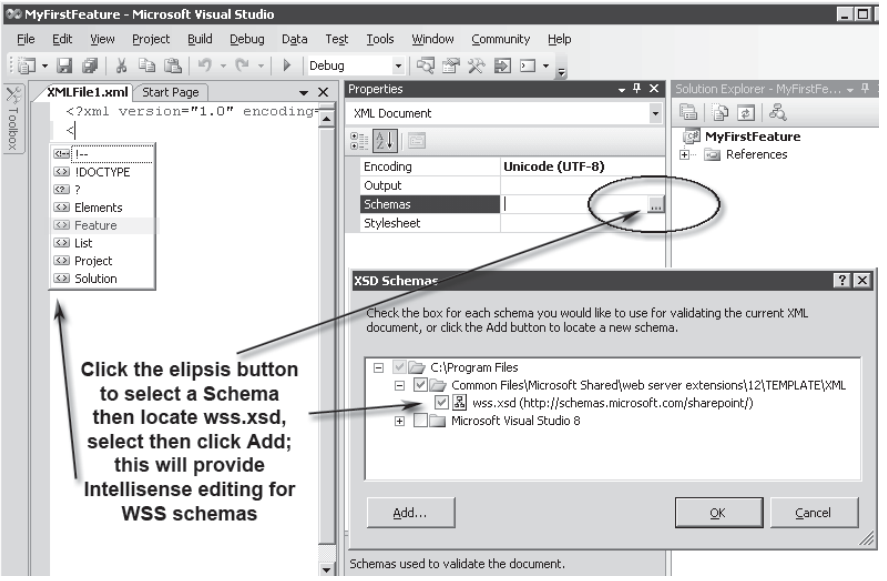
1. Open Visual Studio, select New | Project, and then select Empty. Name this project **MyFirstFeature**.
2. Next, create a new folder that will be used to hold the Feature and that will ultimately become the same folder name used by SharePoint when installed in the SharePoint Root. Call this **MyFirstFeature**. Next, you need to create the actual file that you want to install, which in this case is a simple HTML page that will be provisioned in SharePoint. Use File | New | File, and then, in the New File dialog box, click on General in the left-hand pane and select HTML Page. Change the title of the newly created one and then add some code to the HTML page to display something similar to this:



3. Use File | Save As to save this file as SamplePage.htm inside of the MyFirstFeature folder you just created.

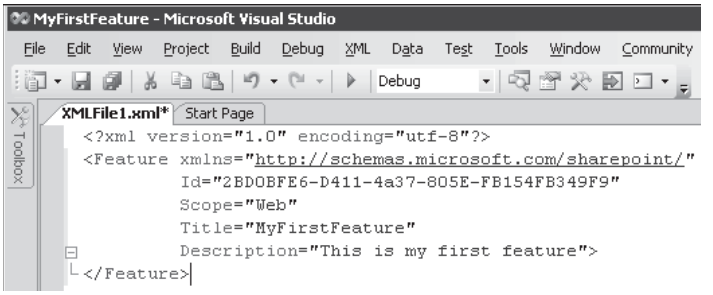
NOTE *After you save as, the file is not included in your project by default; you must click the Show All Files icon in the Solution Explorer, expand the MyFirstFeature folder, right-click on the filename, and then select Include in Project from the drop-down menu.*

4. Now, to create the Feature you will need to create a Feature.xml by selecting File | New | File. Then, in the New File dialog box, click on General in the left-hand pane, and then XML file in the right-hand pane. Click OK to create the file, which will open a new XML File in the editor. Now, to go with Features, SharePoint has provided you with built-in XSDs to assist in building the file. By referencing these within the editor, it will give you intellisense (or code-sense) when you are editing. With the XML file open, right-click in the editing window and select Properties; in the Properties window, locate the Schemas line. Next, click the ellipsis (…) button, which will allow you to select the wss.xsd schema:



Note that if you don't see the file listed, you may need to add it (use the Add button and specify the path as shown previously). Once this is done, you will see that as you type in XML element tags, the intellisense will help with both Elements and with Attributes of the elements.

5. Start out with adding the Feature tag and then add the XML Namespace for SharePoint (this is entered as **Xmlns=http://schemas.microsoft.com/sharepoint**). Next, you'll need a unique GUID to assign to this Feature. There are two ways to create one: in Visual Studio, use the Tools drop-down menu and then select Create GUID, or you can use the GUIDGEN.EXE utility, a program included with Visual Studio (the usual location is C:\Program Files\Microsoft Visual Studio 8\Common7\Tools). The easiest method to access this is using the Visual Studio command line because the path you need is already defined. To open it, select Start | All Programs | Microsoft Visual Studio 2005 | Visual Studio Tools | Visual Studio Command Prompt. When the command prompt appears, enter **GUIDGEN** and press ENTER.
6. Click New GUID, and then the Copy buttons, which will generate a new GUID and put it in your clipboard. Return to the Visual Studio editor and add it as the ID of the Feature (note that you should remove the { and } that is copied in with the GUID). Then you need to set the scope of this Feature, which in this example you'll set to Web, meaning that this Feature is only available at the Web level (not Farm or Site). You also need to add a title (call it **MyFirstFeature**) and add a description. When done, it should look something like this (note that lines should not be wrapped; they are just shown that way for readability):

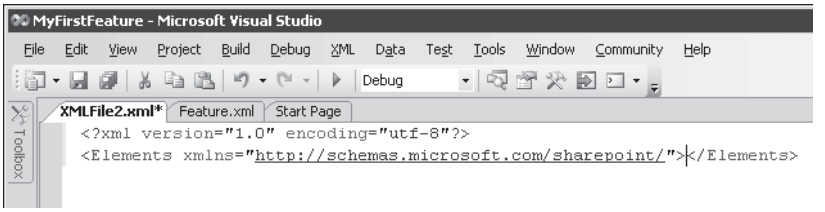


7. Now you need to specify the Element Manifest file that will go with this Feature (as mentioned, this defines all of the components that make up the Feature for SharePoint, so there can be more than one). For your simple example, however, you will just add the one you need. Between the Feature tags, enter in:

```
<ElementManifests>
<ElementManifest Location=ElementsManifest.Xml" />
</ElementManifests>
```

8. As you can see, there is only one attribute to the ElementManifest element called Location, which is the file path and name for the Elements Manifest file for this Feature. If no path is specified (as is typical), this means SharePoint will look for the same folder as the Features.xml file. When you're done, save this file to the MyFirstFeature folder in your project.

9. Next, you need to create the Elements Manifest file, so select File | New | File and then General | XML File again, as you did previously, and associate this with the wss.xsd schema. This time, you will define an Elements tag and specify the XML namespace (XMLns), also as you did previously:



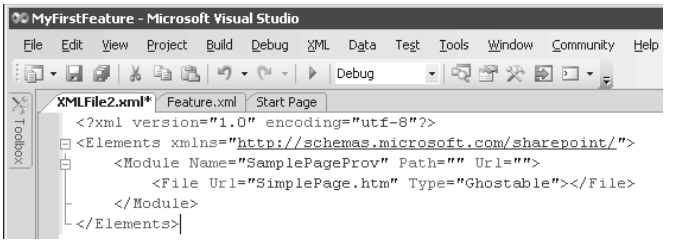
10. Next, add the Module section, which will contain the list of files that will be a part of this module. You can use any name, but you should call this example **SamplePageProvision** and then specify two attributes. These are a path set to blank, meaning the file(s) specified by this module will be in the same directory as the Element Manifest file, and a URL also set to blank, meaning the root of the web (as specified in the STSAdm command as you will see). It looks like the following:

```
<Module Name="SamplePageProv" Path="" URL=""></Module>
```

11. Finally for this file, you want to specify the file that is to be included in this Module, which in this example is SamplePage.htm. You need to add a file entry between the Module tags like so:

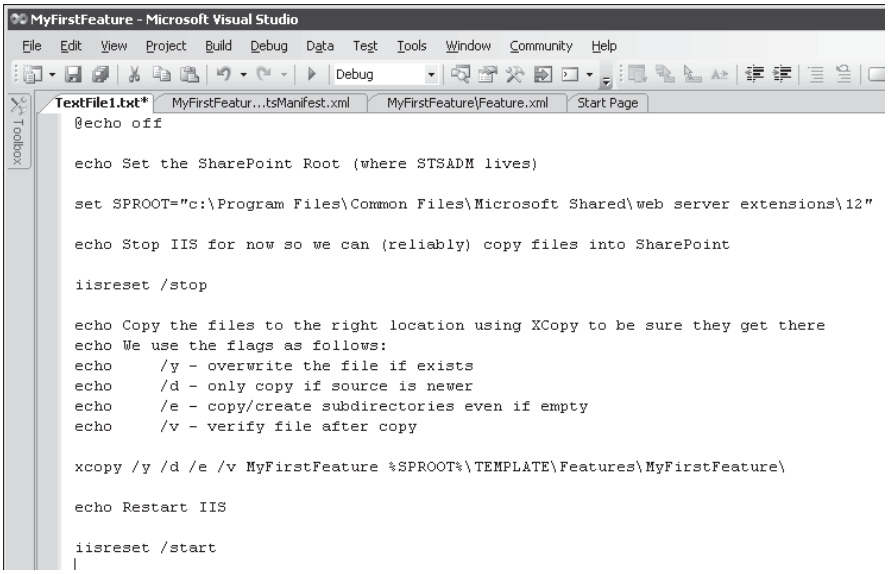
```
<File URL="SimplePage.htm" Type="Ghostable"></File>
```

12. Here, you are simply specifying the name with no path (this means the current directory) and it will be Ghostable (cached) in SharePoint. When you're done, it should look something like this:

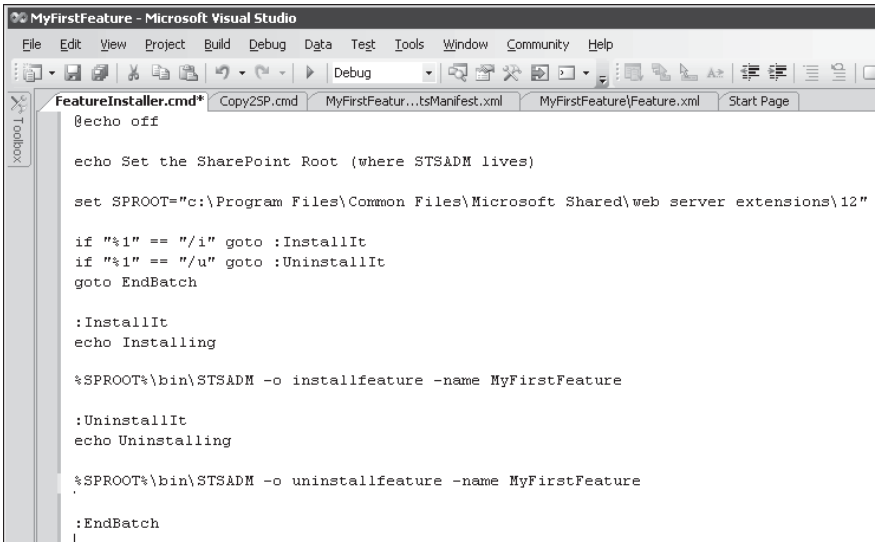


13. To save this file, click File | Save XMLFile2.XML (or whatever it is called). In the Save File dialog, save this file to MyFirstFeature folder as ElementManifest.xml.
14. At this point, the Feature itself is complete, so now it must be set up to install into SharePoint. This means copying the Feature to SharePoint, installing it (in SharePoint), and then activating it on the Web. This can all be done manually on the command line, but I suggest creating batch files to do each of the steps for you to save some typing. The first batch file you need will be used to copy the physical Feature files

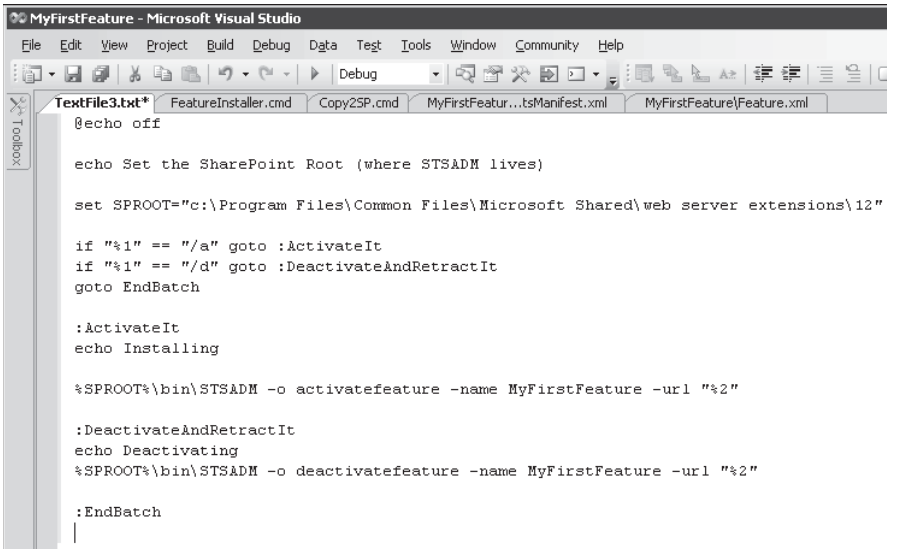
to the SharePoint Hive. In your project, select File | New | File and this time, select General | Text File. When the editor opens, enter in the text as shown:



15. As you can see, there are a few things going on here. First, you stop IIS to ensure that you can copy (reliably) into the SharePoint root folder. Next, copy the file(s) and restart IIS. When you are done editing, you want to save this file as **Copy2SP.cmd** in your project folder (the one holding the MyFirstFeature folder). Next, create the batch file that will install the Feature into SharePoint; this is a single command using the SharePoint Administration Command Line Tool STSAdm. Again, select File | New | File and then General | Text File, and when the editor opens, enter in the text as shown:



16. When done, save this file (again in the project folder) and name it **FeatureInstaller.cmd**. As you can see, this batch job gives you the option to use a switch on the batch file—/i for install and /u for uninstall—allowing you to use it both for installing and uninstalling. The last batch file you need will activate and deactivate the Feature using STSAdm. Then, you will again select File | New | File and then General | Text File. When the editor opens, enter in the text as shown:



17. Notice that in this batch file, you have added a second parameter, which will allow you to pass the URL of the site or Web you want to activate the Feature on. Save this file (again to your project folder) as **FeatureActivation.cmd**.

Deploying the Feature

Once the Feature has been completed and batch files created (these can be reused for your next Feature), they can be used to copy, install, and activate the Feature in SharePoint. To deploy this Feature, do the following:

1. Open a command window and navigate to your project folder where the batch files were saved. The first step is to copy files into the SharePoint Root folder, so enter the command:

   ```
   Copy2SP.cmd
   ```

   followed by return. IIS will be stopped, the files copied, and IIS restarted. Next, you want to install the Feature, so you will enter the command (again followed by return). This time you will add the /i switch that tells the batch job to install:

   ```
   FeatureInstaller.cmd /i
   ```

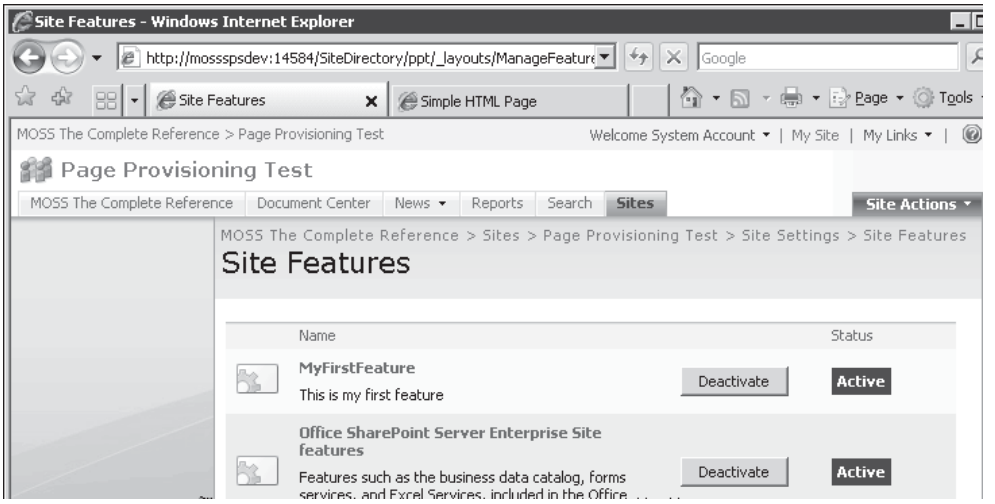
2. This /i switch causes the batch job to execute the InstallFeature command, and when this occurs, all of the files are validated (Feature and ElementsManifest.xml). If there are any errors, you will be notified. If all is well, you will receive an Operation Completed Successfully message.

3. Now you need to activate the Feature on a specific Web. For demonstration purposes, I'm using a Team Site for installing this Feature. Its URL is http://mossspsdev:14584/SiteDirectory/ppt/ (you can specify whichever is applicable for you). To activate it, enter the command:

```
FeatureActivation.cmd /a http://mossspsdev:14584/SiteDirectory/ppt/
```

4. Again, assuming all is well, you will receive an Operation Completed Successfully message. If there are any errors, they will be displayed and are usually quite descriptive to help you figure out what the problem is.

Now that the Feature has been deployed, installed, and activated, it is time to see if it actually works. First, you can verify that the Feature is indeed installed and activated, which is done through Site Settings. Select Site Actions | Site Settings (Site Actions | Site Settings | Modify All Site Settings in a MOSS site), and then under the Site Administration group, click Site Features. You will see that the Feature (with title and description from your Features.XML file) is installed and activated:



To test the actual page (added by the Feature), open up a browser and enter in the full URL to the page (this is the site path and page name). Following this example, the full URL path to the page would be http://mossspsdev:14584/SiteDirectory/ppt/SamplePage.htm. When this URL is entered, you should see the page displayed.



Problems Deploying Features

Yes, it does happen: sometimes Features don't deploy for one reason or another. Most often it has to do with folder privileges, but hey, anything can happen. Fortunately, you do have a few places you can look. The System Application Event Log (on the server itself) is the first place to look for errors. You can also check the Security Event Log; problems with accounts and passwords usually show up there.

Lastly, SharePoint maintains its own logs in the SharePoint Hive. Usually, provisioning errors (and their cause) can be found in the SharePoint Server log file (these logs are time stamped and created new each 1/2 hour and are located in <drive>:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\Logs).

Building Complex Features

The previous example was quite simple; you only installed a page and added it to a specific website. It would be rare (but possible) that you would create such a simple one, as typically a Feature involves more than one file or component. As an example, assume that you had a Branding Feature that included a Master Page and a Custom Picture library (that holds images used by the Master Page). Instead of the step by step, let's view the code directly:

Feature.xml File:

```
<?xml version="1.0" encoding="utf-8"?>
<Feature xmlns="http://schemas.microsoft.com/sharepoint/"
             Id="403B3574-D769-48cd-B2DB-662B799785E7"
             Scope="Site"
             Title="Branding Feature"
             Description="Branding feature for the Corporate sites.">
    <ElementManifests>
            <ElementManifest Location="BrandingElementManifest.xml" />
    </ElementManifests>
</Feature>
```

ElementManfiest.xml File:

```
<?xml version="1.0" encoding="utf-8"?>
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
      <Module Name="Corporate Master" List="116" URL="_catalogs/masterpage" >
            <File URL="Corporate.master"
                    Type="GhostableInLibrary"
                    IgnoreIfAlreadyExists="true"
                    Path="Corporate.master"
                    NavBarHome="False"
                    Name="Corporate.master" />
      </Module>
      <Module Name="Publishing Images" List="114" URL="_catalogs/lt" >
           <File URL="PublishingImages.stp"
                    Type="GhostableInLibrary"
                    IgnoreIfAlreadyExists="true"
                    Path="PublishingImages.stp"
                    Name="PublishingImages.stp" />
      </Module>
</Elements>
```

Notice that the STP file (a Library Template file) is saved to it. This is the List Template Gallery list (the ID of the list is 114).

Summary

As you've seen in this chapter, Features and Solutions are the primary methods for adding or extending SharePoint functionality. The primary purpose of these is to enable extensions to be added and removed easily from one or more servers (very helpful in a farm) and enabled or disabled via the User Interface (putting the control in the hands of the Site Administrator). As you've also seen, Features and Solutions are basically the same and use the same setup—the difference being that Solutions are typically used for components such as web parts and assemblies, as you will see in detail in Chapters 19 and 20.

CHAPTER 18

SharePoint Object Model and Web Services

One of the key elements in the Windows SharePoint Services platform is the extensibility of the product through the SharePoint Object Model (or Application Programming Interface/API if you prefer). The Object Model exposes all of SharePoint's functionality (the entire User Interface is written against it) and is the main way to extend it. There are two ways SharePoint makes it available: the first is a direct connection to a SharePoint site, while the other is via Web Services that use HTTP. Both methods (often combined) can be used to extend (or to access) SharePoint from external applications, web components, Event Receivers, and web parts.

SharePoint's object model is quite extensive and includes several namespaces. Unfortunately, many new items for V3 are undocumented; I'm hoping that updates to the Software Development Kit (SDK) will fill in the gaps. At any rate, because of the sheer size of the model, I wouldn't attempt to document it here even if I could! Instead, this chapter will focus on some of the most common elements that are used for nearly all purposes, as well as some of the ways in which to deal with SharePoint objects, security, and working with SharePoint Search.

Developing Against SharePoint

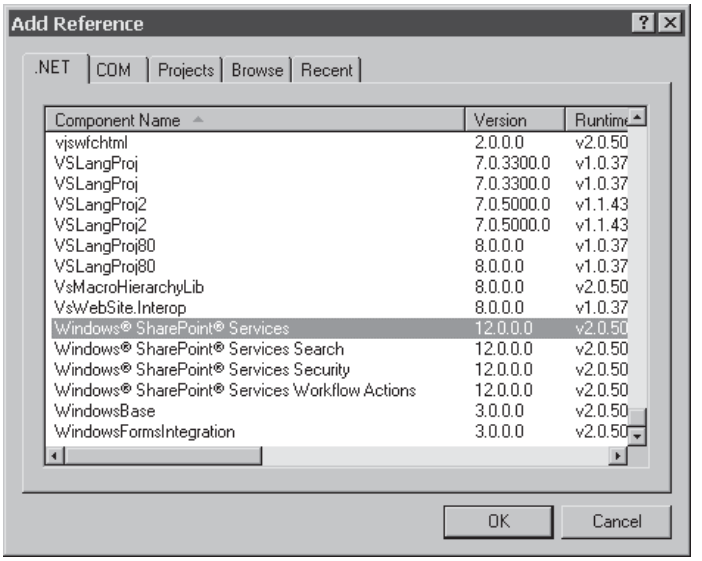
Over the time I've worked with SharePoint, it has always been a bit lax in documentation. While Visual Studio 2005 can now deploy web parts directly to SharePoint for debugging, other items you will build, such as Event Receivers and web components, for example, require debugging to be done by connecting to the IIS Process. If the assembly needs permissions that require loading it into the Global Assembly Cache, debugging can be a very time-consuming process.

While I cannot spare you from that, I can offer some advice. When you begin to work with SharePoint, start out with a standard Windows application. Using this, you have full access to the SharePoint Object Model and Web Services, and though you have to connect in a hard-coded manner (see the section "Connecting to a SharePoint Site"), the application overrides SharePoint Security. This allows you to develop and test without worrying about running into security issues (common in SharePoint), and it makes stepping through the code much easier.

Once you have developed a section of code and tested it, you can easily reimplement it into a web part or receiver. Of course, once you do that the normal security rules apply, which is a good thing; if the application code works, you know any problems will be more than likely security related (getting around this is a no-brainer; see the "SharePoint Security and Impersonation" section). I suggest you keep adding to this application. You'll soon find you've built up a complete cookbook of code and routines, and the more you develop new tools, the faster you'll become at it.

Referencing the SharePoint Object Model

Regardless of whether you are using an application or web part, you will always have to add a reference to the SharePoint Object Model within your project. In general, you'll only need to reference the core assembly for SharePoint, but there are additional assemblies available depending on what you need to access (like Search). Adding a reference in a project is done by opening the project, right-clicking on the References folder (in the Solution Explorer), and then selecting Add Reference. This opens up a Selection dialog, whereas adding the Windows SharePoint Services reference is done by scrolling down the .NET list, clicking to select it, and then clicking OK:



As you can see, there is more than one assembly: one for Search, another for Security, and so on. For MOSS, there are others; the Microsoft Office Server for working with some of the Shared Services (like Audiences), Microsoft Content Publishing and Management for working with content management, and another for SharePoint Single Sign-on. Once you have added a reference, you must also include it in the code, using a statement in C# or an Imports statement in Visual Basic. For example, in C# the code is as follows:

```
using Microsoft.SharePoint;              // Needed for SharePoint
using Microsoft.SharePoint.WebControls; // Needed for SP Controls
using Microsoft.Office.Server;           // Needed for SSP objects
```

The previous listing shows only the basic namespaces for SharePoint; you may need to add additional namespaces (like the one for Publishing or the one for Search) to access other parts of the object model. Remember that specifying particular namespaces will save you some typing. For example, to reference SPControl, the full format would be:

```
SPSite CurrAppLvlSite = Microsoft.SharePoint.WebControls.SPControl.GetContextSite(Context);
```

Adding the using statement shown previously shortens this to

```
SPSite CurrAppLvlSite = SPControl.GetContextSite(Context);
```

SharePoint Security and Impersonation

Something to be acutely aware of is that security in SharePoint is independent of the actual site/sub site architecture; any site in SharePoint can have a completely unique set of users and permissions irrespective of its parent or children. In development, security can be a very big issue. Whenever operating inside of SharePoint (as a web part or Event Receiver does), the code runs under the permissions and security of the current user. If you attempt to connect to a SharePoint site or list that the user doesn't have access to, you will get an access denied error, or worse, the end users get prompted to log in since you don't have much control over how SharePoint will deal with a permissions error.

In many types of web parts, you'll find that you can easily operate as the current user without any issues (only testing will prove this for sure!), but you will also find that there are some situations where you simply cannot get around it. A good example is when you want to know all of the sub sites of the current site, which is something a member user does not usually have access to do. Since advanced access is often necessary, you can skirt SharePoint's security by using .NET Impersonation. Impersonation allows you to temporarily log in as another user, perform some action, and then log out and resume the identity of the original user. Note that this is not the same as site impersonation in .NET; in-code impersonation uses the Windows Identity object.

While impersonation is simple enough, you do have an issue with what account to use. Impersonation requires the full account information (Domain, User Name, and Password) and you certainly don't want to hardcode such information (if changed, a recompile would be necessary) and/or expose it to the public either. To get around this, .NET provides a way to retrieve the information from the System Registry or the website web.config file (shown next).

Using Impersonation

Using impersonation is really quite simple. Prior to doing something that requires a higher level of security, you simply establish a new identity, do the work necessary, and then release the identity. A perfect example of this is accessing webs using the SPWeb object; the method AllWebs returns a collection of all sub sites in the site. For example:

```
using System.Security.Principal;    // Needed for Impersonation
using Microsoft.Win32;              // Needed for access to the Registry
using System.Diagnostics;           // Needed for Event Log
...
...
//-------------------------------------------------------------------
// START IMPERSONATION BEFORE WE ATTEMPT TO GET THE WEBS:
```

```
WindowsImpersonationContext GlobalWIC =
StartImpersonation("Domain","Account","Password");
try
{
SPWebCollection TheWebCol = MySPWeb.AllWebs();
}
catch (Exception ErrInAccess)
{
    string ErrIs = ErrInAccess.Message.ToString();
    if (ErrIs.IndexOf("401") > -1)
    {
        ErrIs = "ERROR: Account Access Denied";
    }
}
// Done - stop impersonation:
EndImpersonation(GlobalWIC);
```

Routines for Impersonation

```
//============================================================================
/// StartImpersonation - Start impersonating
//============================================================================
// Note: keep this on one line (broken here for readability):
private static WindowsImpersonationContext
StartImpersonation(string ADomain, string AName, string APwd)
{
    // START IMPERSONATION:
    //
    WindowsImpersonationContext WinImpContext = null;
    try
    {
        WinImpContext = CreateIdentity(AName, ADomain, APwd).Impersonate();
    }
    catch { }
    //
    return WinImpContext;
}
//============================================================================
/// EndImpersonation - End impersonating
//============================================================================
protected static void EndImpersonation(WindowsImpersonationContext WinImpContext)
{
    try
    {
        WinImpContext.Undo();
    }
    catch { }
}
//============================================================================
/// CreateIdentity Method (uses WriteEventLog routine!):
//============================================================================
/// <param name="User">User Account Name</param>
/// <param name="Domain">Domain Name</param>
/// <param name="Password">User Account Password</param>
/// <returns>Windows Identity</returns>
// Note: keep this on one line (broken here for readability):
protected static WindowsIdentity
```

```
CreateIdentity(string User, string Domain, string Password)
{
    //
    // Set up a user token:
    //
    IntPtr tokenHandle = new IntPtr(0);
    //
    const int LOGON32_PROVIDER_DEFAULT = 0;
    const int LOGON32_LOGON_NETWORK = 3;
    //
    // Init the TokenHandle:
    //
    tokenHandle = IntPtr.Zero;
    //
    // Call LogonUser to obtain a handle to an access token:
    //
    bool returnValue = LogonUser(User, Domain, Password,
         LOGON32_LOGON_NETWORK, LOGON32_PROVIDER_DEFAULT,
         ref tokenHandle);
    //
    // Check the return status:
    //
    if (false == returnValue)
    {
        int ret = Marshal.GetLastWin32Error();
        WriteEventLog("MyProgram","Login Failure!","ERROR",1,"Services");
    }
    // The WindowsIdentity class makes a new copy of the token - It also
    // handles calling CloseHandle for the copy:
    WindowsIdentity id = new WindowsIdentity(tokenHandle);
    CloseHandle(tokenHandle);
    // Return the ID:
    return id;
}
//
// DLL References (these are Required):
//
[DllImport("advapi32.dll", SetLastError = true)]
private static extern bool LogonUser(string lpszUsername,
       string lpszDomain, string lpszPassword, int dwLogonType,
       int dwLogonProvider, ref System.IntPtr phToken);
[DllImport("kernel32.dll", CharSet = CharSet.Auto)]
private extern static bool CloseHandle(System.IntPtr handle);
```

Retrieving Impersonation Account from Registry

Storing impersonation account information in the Registry is usually the most secure and it also provides a great mechanism for maintenance, because the password can be changed at any time with no impact on the code. To begin with, this routine requires that there be actual entries in the Registry that it can find the key values in. It looks something like this:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Sterling]

[HKEY_LOCAL_MACHINE\SOFTWARE\Sterling\ImpersonateAcct]
"AccountPwd"="M0SS123$"
"DomainName"="MOSS"
"AccountName"="MOSSAcct"
```

Obviously, you would use something other than Sterling, ImpersonateAcct, and the account I specified (you'll see where these are used later in the chapter). You can manually create these entries (using RegEdit) or create a text file named .reg using the previous text (altered, of course). Save the file, and then double-click it to automatically add it to the registry.

CAUTION *Accessing the system Registry can have disastrous results; make a backup and if unsure, seek assistance first!*

```
//=========================================================================
/// GetAccountInfoFromRegistry - Return account information from Registry
//=========================================================================
protected bool GetAccountInfoFromRegistry(out string AccountName, out string AccountPwd,
out string DomainName)
{
    // Init return values:
    AccountName = "";
    AccountPwd = "";
    DomainName = "";
    bool AcctInfoSet = true;
    try
    {
        // Get Impersonation account information from the Registry:
        string RtnStringValue = "";
        GetAppRegistryKey("DomainName", "Sterling", "ImpersonationAcct",
        out RtnStringValue);
        if (RtnStringValue.IndexOf("ERROR") != -1)
        {
            AcctInfoSet = false;
        }
        else
        {
            DomainName = RtnStringValue.ToString();
        }
        //
        GetAppRegistryKey("AccountName", "Sterling", "ImpersonationAcct",
        out RtnStringValue);
        if (RtnStringValue.IndexOf("ERROR") != -1)
        {
            AcctInfoSet = false;
        }
        else
        {
            AccountName = RtnStringValue.ToString();
        }
        //
        GetAppRegistryKey("AccountPwd", "Sterling", "ImpersonationAcct",
        out RtnStringValue);
        if (RtnStringValue.IndexOf("ERROR") != -1)
        {
            AcctInfoSet = false;
        }
        else
        {
            AccountPwd = RtnStringValue.ToString();
        }
```

```
    }
    catch { return false; }
    //
    return AcctInfoSet;
}
//=========================================================================
// GetAppRegistryKey - Return the selected key value
//=========================================================================
/// <param name="KeyValueToGet"></param>
/// <param name="PrimaryKey"></param>
/// <param name="SecondaryKey"></param>
/// <param name="RtnVal"></param>
public static void GetAppRegistryKey(string ItemValueToGet, string PrimaryKey, string
SecondaryKey, out string RtnVal)
{
    // Setup defaults:
    string PrimaryKeyDefault = "Sterling";
    string SecondaryKeyDefault = "ImpersonationAccount";
    //-------------------------------------------------------------------------
    // Get a reference to the Hive (TRAP ACCESS ERRORS):
    RegistryKey RegKeyRef;
    try
    {
        RegKeyRef = Registry.LocalMachine.OpenSubKey("SOFTWARE");
    }
    catch
    {
        // If we get an error here, we have no access; Return False:
        RtnVal = "ERROR - Unable to Access Registry!";
        WriteEventLog("MyProgram",RtnVal,"ERROR",1000,"System Event");
        return;
    }
    //-------------------------------------------------------------------------
    // Find the Primary Entry first:
    if (PrimaryKey != "") { PrimaryKeyDefault = PrimaryKey; }
    if (SecondaryKey != "") { SecondaryKeyDefault = SecondaryKey; }
    // Loop through to find it:
    bool RootWasFound = false;
    foreach (string RegSubKeyRef in RegKeyRef.GetSubKeyNames())
    {
        string RegSubKeyName = RegSubKeyRef.ToString();
        if (RegSubKeyName == PrimaryKeyDefault)
        {
            RootWasFound = true;
            break;
        }
    }
    if (RootWasFound == false)
    {
        // Return an error:
        RtnVal = "ERROR - MISSING PRIMARY KEY (" + PrimaryKey + ")";
        WriteEventLog("MyProgram",RtnVal,"ERROR",1001,"System Event");
        return;
    }
    else
    {
        // Now we want to check for the Secondary Key:
```

```
        bool WasFound = false;
        // Open the Root Entry:
        RegistryKey RootRegKeyRef = Registry.LocalMachine.OpenSubKey("SOFTWARE");
        RegKeyRef = RootRegKeyRef.OpenSubKey(PrimaryKeyDefault);
        // Loop through to see if we can find the application registry key:
        foreach (string RegSubKeyRef in RegKeyRef.GetSubKeyNames())
        {
            string RegSubKeyName = RegSubKeyRef.ToString();
            if (RegSubKeyName == SecondaryKeyDefault)
            {
                WasFound = true;
                break;
            }
        }
        // See if we found it and act as requested:
        if (WasFound == false)
        {
            RtnVal = "ERROR - MISSING SECONDARY KEY (" + SecondaryKey + ")";
            WriteEventLog("MyProgram",RtnVal,"ERROR",1002,"System Event");
            return;
        }
        else
        {
            // Get the key reference:
            RegistryKey AppRegKeyRef = RegKeyRef.OpenSubKey(SecondaryKeyDefault);
            // Loop through to find it:
            bool SetRtnVal = true;
            //
            foreach (string ValName in AppRegKeyRef.GetValueNames())
            {
                string TestName = ValName.ToString();
                if (ItemValueToGet == ValName.ToString())
                {
                    SetRtnVal = false;
                    break;
                }
            }
            // Set the return value:
            if (SetRtnVal)
            {
                RtnVal =
                "ERROR - Registry item " + ItemValueToGet + " was NOT found";
                WriteEventLog("MyProgram",RtnVal,"ERROR",1003,"System Event");
            }
            else
            {
                RtnVal = (string)AppRegKeyRef.GetValue(ItemValueToGet);
            }
        }
    }
}
```

Retrieving Impersonation Account from Web.Config

As an alternative to the registry, account information can also be stored in the site's web.config file. While still pretty secure, any changes to this file require that an IISReset be performed, which can disrupt site users, and that it must be added to the web.config file

of each site collection if there is more than one. Adding this requires adding a few XML statements to the web.config file just before the </configuration> tag:

```
<appSettings>
   <add key="ImpAcctDomainKey" value="MOSS" />
   <add key="ImpAcctNameKey" value="MOSSAcct" />
   <add key="ImpAcctPwdKey" value="M0SS123$" />
</appSettings>
</configuration>
```

If <appSettings> already exists (another app may have put it there), only add the Key entries. Once done (and an IISReset is performed), the information is easily retrieved in code:

```
// SETUP IMPERSONATION ACCOUNT INFO:
bool AcctInfoSet = true;
try
{
// Note: keep this on one line (broken here for readability):
    IAccountName =
       System.Configuration.ConfigurationSettings.
       AppSettings.Get("ImpAcctNameKey");
// Note: keep this on one line (broken here for readability):
    IAccountPwd =
        System.Configuration.ConfigurationSettings.
        AppSettings.Get("ImpAcctPwdKey");
// Note: keep this on one line (broken here for readability):
    IAccountDomain =
       System.Configuration.ConfigurationSettings.
       AppSettings.Get("ImpAcctDomainKey");
}
catch { }
```

Writing to the System Event Log

Over time, you will probably discover that simply in-code error handling is not enough; particularly in web parts. Most errors in SharePoint are trapped, so a problem can be very hard to determine. Since installation (and security) settings in SharePoint vary from one system to the next, something that works just fine on one might break in another. In a production environment, you are basically stuck. You cannot debug the part nor determine the actual error to reproduce it.

The best solution is to use the System Application Event Log to trap errors and provide tracing messages. Fortunately, .NET has got you covered here because writing to the event log is easy; simply include the following routine:

```
using System.Diagnostics;         // Needed for Event Log
...
//=========================================================================
/// WriteEventLog - Write an Event to the Event Log
//=========================================================================
/// <param name="ApplicationName">Name of the Application</param>
/// <param name="ErrMessage">Error, Warning or Info Message to post</param>
/// <param name="MsgType">Type of message (ERROR, and so on)</param>
```

```
/// <param name="ErrCode"> Error Code to use for the error</param>
/// <param name="SetCategory">Category of error (Blank, and so on) </param>
// Note: keep this on one line (broken here for readability):
public static void WriteEventLog(string ApplicationName,
string ErrMessage, string MsgType, int ErrCode, string SetCategory)
{
    // Write out the Event Log message - First, set the Type:
    EventLogEntryType SelectedType = EventLogEntryType.Error;
    if (MsgType == "ERROR") { SelectedType = EventLogEntryType.Error; }
    if (MsgType == "WARNING") { SelectedType = EventLogEntryType.Warning; }
    if (MsgType == "INFORMATION") { SelectedType = EventLogEntryType.Information; }
    if (MsgType == "SUCCESSAUDIT") { SelectedType =  EventLogEntryType.SuccessAudit; }
    if (MsgType == "FAILUREAUDIT") { SelectedType = EventLogEntryType.FailureAudit; }
    //
    // Now set the Event Category:
    short CategoryToSet = 0;
    if (SetCategory == "") { CategoryToSet = 0; }
    if (SetCategory == "Device") { CategoryToSet = 1; }
    if (SetCategory == "Disk") { CategoryToSet = 2; }
    if (SetCategory == "Printers") { CategoryToSet = 3; }
    if (SetCategory == "Services") { CategoryToSet = 4; }
    if (SetCategory == "Shell") { CategoryToSet = 5; }
    if (SetCategory == "System Event") { CategoryToSet = 6; }
    if (SetCategory == "Network") { CategoryToSet = 7; }
    //
    // Write the event:
    EventLog.WriteEntry(ApplicationName,
            ErrMessage, SelectedType, ErrCode, CategoryToSet);
    //
}
```

The SharePoint Site and Web Architecture

The best place to start with the object model is to quickly dispel the confusion as to what a SharePoint Site or SharePoint Web is and how it relates to security, users, and so on. Alas, the general documentation doesn't help much, because the two names are used interchangeably. The issue is naming; for a user, everything in SharePoint looks like a site, but the underlying object is actually a SharePoint Web. A good way to understand it is to compare SharePoint to IIS:

- A SharePoint site is provisioned (created) as an application through the SharePoint 3.0 Central Administration Site. This is an empty directory with no actual site (if you navigate to it, it will return Page Not Found). This, in IIS terms, is a new Virtual Directory; in SharePoint, this is represented by the SPSite object. The relationship is IIS Virtual Directory = SharePoint Application = SharePoint SPSite object.
- A SharePoint Site Collection (top-level site) is created within a SharePoint application. In IIS terms, this is a new website. SharePoint represents this as an SPWeb object, so the relationship is IIS Web = SharePoint Site = SharePoint SPWeb object.
- A sub site created underneath a Site Collection in IIS terms is a new directory under a website. In SharePoint terms, this is also an SPWeb object. The site above it is

considered the parent, while the sites below it are webs. This means the relationship is IIS Web Folder = SharePoint Sub Site = SharePoint SPWeb object.

One important concept to be sure you've got clear: all sites (the ones you see) in SharePoint are SPWebs. You will spend 98 percent of your time working with SPWeb, because it is the parent of all of a given site's content. Lists, libraries, and so on, are all accessible via the SPWeb object. The SPSite object is only used when you need to connect to the application-level settings or to simply open the top web there (the SPSite object has a method called OpenWeb).

Connecting to a SharePoint Site

Most anything you'll do in a web part or component will require connecting to a SharePoint site. To access content in a site, whether it's a list, a document, a page, or similar, a connection with the site itself must be established first. This is kind of a folder-like concept; the site is like a top-level folder, and the lists, libraries, and other content are subfolders. Then, the items (documents and so on) in lists and libraries are like files in a folder. Even a sub site is another folder. A SharePoint site is represented in the object model as a SPWeb object. Things like lists, libraries, and sub sites are available individually or as collections that are also objects (SPListCollection). Each of these, in turn, have their own collections of items (for example, SPListItemCollection).

Getting a reference to a SharePoint Site is easy in a web part or web component; in those cases, you are running inside of SharePoint and have the HTTP context, which allows you to use part of the object model to get the reference automatically. If you are outside of SharePoint, for example, when you're in an application, you don't have a context and must, therefore, use the actual site URL:

```
// IN WEB PARTS AND COMPONENTS THAT RUN ON A SHAREPOINT PAGE, THE HTTP
// CONTEXT CAN BE USED TO RETURN THE SITE/WEB:
// Open current Application Level Site (no UI):
SPSite CurrAppLvlSite = SPControl.GetContextSite(Context);
SPSite CurrAppLvlSite = new SPSite(ServerContext.Current);
// Open the current Site (web):
SPWeb CurrTopLvlSite = SPControl.GetContextWeb(Context);
//
// ACCESS VIA APPLICATION DOES NOT HAVE CONTEXT, MUST
// USE URL:
// Open the Application Site (no UI):
SPSite AppLvlSite = new SPSite("http://mossspsdev:14584");
// Open the top level web for the site:
SPWeb TopLvlSite = AppLvlSite.OpenWeb();
//
// SITES CAN BE ACCESSED FROM EITHER SPSite OR SPWeb objects:
//
// Open a site by URL:
SPWeb SubLvlSiteByUrl =
      AppLvlSite.OpenWeb("http://mossspsdev:14584/Sites/Subsite", false);
// Open a site by Name:
SPWeb SubLvlSiteByName = TopLvlSite.Webs["Sub site Name"];
// Open a site by GUID:
SPWeb SubLvlSiteByGUID = TopLvlSite.Webs["7707A2FD-9B72-45b2-A0A6-08EC087C1EAC"];
// Get the entire collection of sites:
SPWebCollection AllSiteCollection = TopLvlSite.Webs;
```

Connecting to a List

A common need within web parts is to get information from a list, library, or even documents stored there. While this can be for simple purposes, such as displaying a Summary View of the list or library (the RSS Feed Reader works like this), it is also one of the simple but very powerful ways to extend the use of SharePoint. This is a key point in development; using a list or a library provides the same features a database would, but it also provides all that is needed for the user to maintain the actual data and/or documents.

Consider this example: You want to display a small image map based on the value of a drop-down menu; the images will be maps and the drop-down menu will be a list of location. In a typical application, you would create an ASP.NET page that showed the drop-down menu and image placeholder. You would then either hardcode a list of selections or get it from a database. You would have to associate each choice with an image and then upload them to the server file system (I won't even go into security). To add a new location, you must re-edit if hardcoded or add it to the database, recompile, deploy the application to the server, and then upload an associated image. If users need to be able to do the maintenance, a new application must be created to provide add, update, and delete capabilities (with error checking), and also an upload routine for getting the images onto the server.

Building this in SharePoint is completely different. By using a SharePoint list that has a Location column and an Image attachment, a simple web part could read the list to provide the drop-down menu; when a location is selected, it could load and display the associated image. Since a list is already user maintained, no extra code is needed!

The primary point here is that a few lines of code in a web part that is leveraging SharePoint's existing features provides the same functionality in $1/10^{th}$ of the time (and provides a better end product!). Now, if you looked at the previous example of connecting to a site, you will be surprised at how amazingly simple this is. In this case, connecting to a list or library within a site takes only two lines of code:

```
// Line 1: Open the current Site (web):
SPWeb TopLvlSite = SPControl.GetContextWeb(Context);
// To open a list, line 2 of the code would be:
SPList AList = TopLvlSite.Lists["Links"];
// To open a library, line 2 of the code would be:
SPDocumentLibrary ADocLib = TopLvlSite.Lists["Shared Documents"];
```

Once you have connected to the list or library, you can access all of the items in it:

```
// Get all items:
SPListItemCollection ListItems = AList.Items;
// Get the first item:
SPListItem FirstItem = ListItems[0];
// Get the title field of the item;
// Notice the field must have a Cast
string ItemTitle = (string)FirstItem["Title"];
```

In SharePoint lists, remember that a library is still a list; the only difference is that attachments in a library are indexed for Search. Libraries are usually accessed as a list; however, to access attachments (SPFile objects), they must be recast as an SPDocumentLibrary object:

```
SPList ADocLibList = TopLvlSite.Lists["Shared Documents"];
// Recast:
SPDocumentLibrary ADocLib = (SPDocumentLibrary)ADocLibList;
```

Using List Columns

To save you from pulling your hair out, there are a few tricks to using SharePoint lists, due to the way it stores columns (fields). SharePoint stores list column names in a way that they can be added to the internal database, and it maintains a Display Name and an Internal Name for each field. The Internal Name is sometimes modified to remove invalid SQL characters, such as spaces, or is truncated if it's too long. When accessing a list item in code, most of the time you have to use SharePoint's internal name for a column, so you must know what that is.

Part of this problem is the UI; when a user creates a list and makes a mistake in a column name (or simply renames it), this only changes the display name; the internal name created with the column remains the same. This is common with the default column called Title that is created automatically with custom lists because it's often renamed. Fortunately, it is quite simple to determine what the column name is using the XML schema; this is accessible when you are connected to the list (or library):

```
// Open the current Site (web):
SPWeb TopLvlSite = SPControl.GetContextWeb(Context);
// Open a list:
SPList AList = TopLvlSite.Lists["Links"];
// Get the XML Schema:
string XMLSchema = AList.SchemaXml;
```

The text returned by the previous statement is the full XML that defines the list or library itself. Within this, you will find a <Fields> tag, which is, of course, the start of the field definitions in the list. As an example, the previous code returned the following:

```
<Fields>
<Field ID="{03e45e84-1992-4d42-9116-26f756012634}" RowOrdinal="0"
 Type="ContentTypeId" Sealed="TRUE" ReadOnly="TRUE" Hidden="TRUE"
 DisplayName="Content Type ID" Name="ContentTypeId" DisplaceOnUpgrade="TRUE"
 SourceID="http://schemas.microsoft.com/sharepoint/v3"
 StaticName="ContentTypeId"
 ColName="tp_ContentTypeId" FromBaseType="TRUE"/>
<Field ID="{39360f11-34cf-4356-9945-25c44e68dade}" ReadOnly="TRUE"
 Hidden="TRUE" Type="Text" Name="File_x0020_Type"
 DisplaceOnUpgrade="TRUE" DisplayName="File Type"
 SourceID="http://schemas.microsoft.com/sharepoint/v3"
 StaticName="File_x0020_Type" FromBaseType="TRUE" ColName="nvarchar2"/>
…
```

You'll notice the DisplayName attribute of File Type (this is what the end user sees), but you can also see that the Name attribute is actually File_x0020_Type (the real name of the field). To access this in code, the format would be (this statement returns the value; the (string) cast is required, which is true at all times when accessing a column value):

```
// Get the File Type field:
string FileType = (string)FirstItem["File_x0020_Type"];
```

Searching for a List Item

When using SharePoint lists and libraries, particularly for use as data, it can certainly be useful to loop through the items to find what you need. However, this can be time consuming as it is the equivalent of doing it in a database; in other words, the more items, the longer the loop. To eliminate this problem, SharePoint provides a common ability to search lists and libraries internally. Searching for items in a list is done using Collaborative Application Markup Language (CAML) Query syntax using an object called SPQuery. The basic syntax is much like XML queries, though there are some oddities (dates, for example, have to be in a specific format, as shown later in the chapter). The format is usually <Where> and then some comparison, <Leg> (Less than or Equal) or <Eq> (Equal), of the field to some value. As an example (note this should be on one line):

```
<Where><Eq><FieldRef Name=\"Title\" />
  <Value Type=\"String\">The Title</Value></Eq></Where>
```

The previous query says return all items where the Title is equal to The Title. In most cases, the literal value (string, number) is what you use to compare, but there is one gotcha in how it deals with dates; these must be the format of YYYYMMDDTHH:MM:SSZ. In the following example, this query will return all items in a list called Links that has had publishing date columns added to it (note that creating this list is shown in the next section):

```
// Open the current Site (web):
SPWeb TopLvlSite = SPControl.GetContextWeb(Context);
// Open a list:
SPList NavList = TopLvlSite.Lists["Navigation Page List"];
try
{
    SPQuery ListQuery = new SPQuery();
    System.DateTime CurDt = System.DateTime.Today;
    string Years = CurDt.Year.ToString();
    string Months = CurDt.Month.ToString();
    string Days = CurDt.Day.ToString();
    string FormattedDt = Years.ToString() + "-" +
          Months.ToString() + "-" + Days.ToString();
    // Never include the <Query> tags! The query will NOT execute:
    //WRONG:
    //ListQuery.Query = "<Query><Where><Leq><FieldRef...
    // RIGHT:
    // Note: keep this on one line (broken here for readability):
    ListQuery.Query = "<Where><Leq><FieldRef Name=\"Publish_x0020_Date\" />
       <Value Type=\"DateTime\">" + FormattedDt + "T00:00:00Z</Value>
       </Leq><Or><IsNull><FieldRef Name=\"Stop_x0020_Publish_x0020_Date\" />
       </IsNull><Geq><FieldRef Name=\"Stop_x0020_Publish_x0020_Date\" />
       <Value Type=\"DateTime\">" + FormattedDt + "T00:00:00Z</Value>
       </Geq></Or></Where><OrderBy><FieldRef Name=\"Appearance_x0020_Order\" />
       </OrderBy>";
    //
    SPListItemCollection AllItems = NavList.GetItems(ListQuery);
    int TheCountOf = AllItems.Count;
    foreach (SPListItem AnItem in AllItems)
    {
```

```
        string TitleOf = (string)AnItem["Title"];
        string UrlOf = (string)AnItem["Link"];
    }
}
catch (Exception ErrorLoadingFromList)
{
    // Some error handling
}
```

Notice that the actual query here equates to Return All Where the Publishing Date is Less Than or Equal to Today and the Stop Publishing Date is Greater than or Equal to Today or is Null. There is also an Order By clause that orders the results by field called Appearance_x0020_Order.

The full syntax for queries is in the SharePoint Software Development Kit (SDK). There is also a great (and free) tool available; download from Patrick Tisseghems CAML Query Builder V2's blog site: http://www.u2u.info/Blogs/Patrick/Lists/Posts/Post.aspx?ID=1315.

Creating a New List and Adding Items

Creating lists and libraries is, of course, an easy thing to do with the SharePoint User Interface (or SharePoint Designer). However, if you are using your own application or creating some custom functionality in SharePoint and need a list to store some information, you can do this easily in code. In many ways, you can use custom lists as a database; since access to a database can be very restrictive in some environments, you can use a SharePoint list the same way you use a data table and you get the added benefit that it can be both visible to and maintained by users.

For example, assume that users want to use a custom list of links to vendor sites and they want to include publishing dates to control whether a link shows or not. I don't want to get too complicated with this example, but the fundamentals are important in working with SharePoint. Because of this, rather than a bland example, the following is a real-life example that does the following:

1. The routine is passed an SPWeb object, which it will connect to and then create a new list.
2. The routine then adds several new columns (fields) to the new list; it also sets the field parameters (where it will appear, the type, and so on)
3. After adding the fields, it creates a new item and then adds that to the list.
4. The SPList object is returned to the original caller.

```
//====================================================================
// CreateNewList
//====================================================================
/// <summary>CreateNewList</summary>
/// <returns>True or false if list created</returns>
private bool CreateNewList(SPWeb WebToCreateListIn, out SPList CreatedList)
{
    System.Guid NewListGuid = new System.Guid();
    CreatedList = null;
    // THIS IS REQUIRED WHEN CREATING NEW LISTS!!
```

```
WebToCreateListIn.AllowUnsafeUpdates = true;
// Create the list in SharePoint:
try
{
    NewListGuid = WebToCreateListIn.Lists.Add("Navigation Page List",
       "List used to provide Custom Navigation",
       SPListTemplateType.GenericList);
}
catch (Exception UnableToCreateList) { return false; }
//
// Get an object reference to the new list:
try
{
    CreatedList = WebToCreateListIn.Lists[NewListGuid];
}
catch (Exception UnableToOpenList) { return false; }
//
// Add the Link and Appearance Order fields:
try
{
    // Rename the title field:
    SPField TitleField = CreatedList.Fields["Title"];
    TitleField.Description = "Title to appear in Navigation";
    TitleField.Update();
    // Add Link Field:
    CreatedList.Fields.Add("Link", SPFieldType.URL, true);
    SPField LinkField = CreatedList.Fields["Link"];
    LinkField.Description = "Relative Link and Tool Tip";
    LinkField.ShowInEditForm = true;
    LinkField.ShowInDisplayForm = true;
    LinkField.ShowInListSettings = true;
    LinkField.ShowInViewForms = true;
    LinkField.Update();
    // Add Appearance Order Field:
    CreatedList.Fields.Add("Appearance Order", SPFieldType.Number, true);
    SPField AppearanceOrderField = CreatedList.Fields["Appearance Order"];
    AppearanceOrderField.Description = "Order to display";
    AppearanceOrderField.ShowInEditForm = true;
    AppearanceOrderField.ShowInDisplayForm = true;
    AppearanceOrderField.ShowInListSettings = true;
    AppearanceOrderField.ShowInViewForms = true;
    AppearanceOrderField.Update();
    // Add Publish Date Field:
    CreatedList.Fields.Add("Publish Date", SPFieldType.DateTime, false);
    SPFieldDateTime PublishDateF =
      (SPFieldDateTime)CreatedList.Fields["Publish Date"];
    PublishDateF.Description = "Date to publish (show)";
    PublishDateF.DefaultValue = "[today]";
    PublishDateF.DisplayFormat = SPDateTimeFieldFormatType.DateOnly;
    PublishDateF.ShowInEditForm = true;
    PublishDateF.ShowInDisplayForm = true;
```

```
            PublishDateF.ShowInListSettings = true;
            PublishDateF.ShowInViewForms = true;
            // NOWCRAWL means don't search this field!
            PublishDateF.NoCrawl = true;
            PublishDateF.Update();
            // Add Stop Publish Date Field:
            CreatedList.Fields.Add("Stop Publish Date", SPFieldType.DateTime, false);
            SPFieldDateTime StopPublishDateF =
              (SPFieldDateTime)CreatedList.Fields["Stop Publish Date"];
            StopPublishDateF.Description = "Date to remove";
            StopPublishDateF.DisplayFormat = SPDateTimeFieldFormatType.DateOnly;
            StopPublishDateF.ShowInEditForm = true;
            StopPublishDateF.ShowInDisplayForm = true;
            StopPublishDateF.ShowInListSettings = true;
            StopPublishDateF.ShowInViewForms = true;
            // NOWCRAWL means don't search this field!
            StopPublishDateF.NoCrawl = true;
            StopPublishDateF.Update();
            //
            // Add the Columns (fields) to the default view:
            SPView DefView = CreatedList.DefaultView;
            DefView.ViewFields.Add(LinkField);
            DefView.ViewFields.Add(AppearanceOrderField);
            DefView.ViewFields.Add(PublishDateF);
            DefView.ViewFields.Add(StopPublishDateF);
            DefView.Update();
            //
            // Now add an item:
            SPListItem NewItem = CreatedList.Items.Add();
            //
            NewItem["Title"] = "Home";
            NewItem["Link"] =
               WebToCreateListIn.ServerRelativeUrl.ToString() +
               ", " + WebToCreateListIn.Title;
            NewItem["Appearance Order"] = 0;
            NewItem["Publish Date"] = System.DateTime.Today.AddDays(-1);
            NewItem["Stop Publish Date"] = null;
            NewItem.Update();
        }
        catch (Exception UnableToCreateFields) { return false; }
        return true;
    }
```

Extracting a Document from a Library

There are many occasions where exporting a document from SharePoint is necessary without using a client-side program like Word. A good example of when you might do this is during an external workflow; to send a user the actual file requires creating a disk-based file that can be sent via Outlook. Using the SharePoint object model, you can easily access a library

and export a file to disk. You simply connect to the library and then extract the binary form of the file as shown. The following example does the following:

- Attaches to a site at URL http://mossspsdev:14584.
- Opens a document library called Legal Documents.
- Gets the first item in the library (note, this should have error handling as there may not be an item).
- Saves the actual name of the document.
- Retrieves the file in binary format.
- Writes the file out (using a file stream) to a physical disk file.

```
SPSite AppLvlSite = new SPSite("http://mossspsdev:14584");
SPWeb TopLvlSite = AppLvlSite.OpenWeb();
// Open the Library:
SPDocumentLibrary LegalDocLib = TopLvlSite.Lists["Legal Documents"];
// Get the first item (by index):
SPListItem FirstItem = LegalDocLib.Items[0];
// Get the file attached to the item:
SPFile TheFile = FirstItem.File;
// Save the filename:
string OutName = TheFile.Name;
// Get the file in bytes:
byte[] binFile = null;
binFile = TheFile.OpenBinary();
// Save the file to disk:
try
{
    // Create a new File Stream and write out the file to C: drive:
    FileStream OurFileStream =
        new FileStream(@"C:\" + OutName, FileMode.CreateNew);
    // Create the writer for data:
    BinaryWriter BinWriter = new BinaryWriter(OurFileStream);
    // Write it out:
    BinWriter.Write(binFile);
    // Close up:
    BinWriter.Close();
    OurFileStream.Close();
}
catch (Exception FileStreamOpenExcep)
{
    // Handle error
}
```

Using SharePoint Web Services

Like the object model, SharePoint's Web Services provide access to many standard features in SharePoint in terms of reading data, checking alerts, and so on, and they are very useful when working between two SharePoint sites or SharePoint and an external application. Most Web Services (shown in Table 18-1) are available in both WSS and MOSS (MOSS includes services for enterprise features).

Service Name (.asmx)	Used For
Alerts	Enables retrieving and deleting user alerts.
Authentication	Enables remote login by mode: None, Windows, Passport, and Forms.
AreaService (MOSS)	A holdover from V2 (don't use).
BDCFieldResolver (MOSS)	Resolves a field reference against the Business Data Catalog.
BusinessDataCatalog (MOSS)	Queries against LOB systems information, including instances and methods.
ContentAreaToolboxService (MOSS)	Returns controls and panels (for CMS); not intended for direct use.
Copy	Used to copy a document from one SharePoint Server to another.
DSPSts	A query method to query data in SharePoint.
DWS	Methods for working with Document Workspaces.
ExcelService (MOSS)	Methods for working with Excel Spreadsheets.
Forms	Methods to retrieve a form or a form collection.
FormsServiceProxy (MOSS)	A method to forward soap requests (not used directly).
FormsServices (MOSS)	Methods to browser-enable a user form and check form templates.
Imaging	Methods for working with image list items.
Lists	Methods for working with lists and content types.
Meetings	Methods for working with Meeting Workspaces.
OfficialFile (MOSS)	Methods for routing/submitting official files to a Records Center Site.
People	Methods for resolving and searching for principals (by None, User, DistributionList, SecurityGroup, SharePointGroup, or All).
Permissions	Methods for working with permissions (Add, Remove, Update).
PublishedLinksServices (MOSS)	A method to retrieve published links.
PublishingService (MOSS)	Methods for working with Publishing: Page Layouts, adding/removing objects, page connections, and so on.
Search	Methods for searching content within SharePoint.
SharePointEmailWS (MOSS)	Methods for working with distribution groups and memberships.
SiteData	Methods for retrieving information from a site (folders, content, items, and so on).

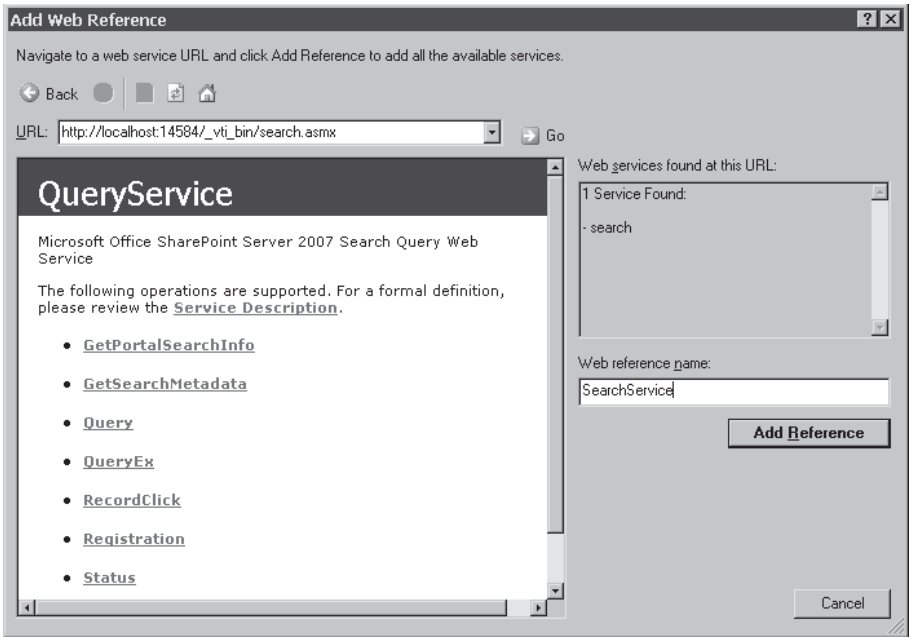
Table 18-1 SharePoint Web Services

Service Name (.asmx)	Used For
Sites	Methods that work with Site Templates (including export/import).
SlideLibrary (MOSS)	Methods for returning Slide Library information.
SpellCheck (MOSS)	Provides a single method, will check the spelling of a passed value.
SPSCrawl (MOSS)	Methods to get item, site, and bucket information from the index crawl.
SPSearch	Methods to query the WSS Search Engine (not Office Search).
UserGroup	Methods for working with User Groups and Roles.
UserProfileChangeService (MOSS)	Methods to return changes in User Profiles.
UserProfileService (MOSS)	Methods for working with User Profiles.
Versions	Methods for working with item and document versions.
Views	Methods for working with list/library views.
WebPartPages	Methods for working with web part pages.
Webs	Methods for working with webs (sites) in SharePoint.
Workflow (MOSS)	Methods for working with workflows (including Starting, Task Management, and so on).

Table 18-1 SharePoint Web Services *(continued)*

Referencing Web Services

Just like connecting to the object model, it is necessary to connect to a Web Service with a reference before you can use it in code—the only difference is that you connect via HTTP. For the SharePoint Web Services, the URL to be specified is the actual URL to the SharePoint site. From within Visual Studio, right-click on the References (or Web References) folder and then select Add Web Reference. The path for all SharePoint Web Services is always the URL to the site, plus both the virtual directory, _vti_bin, and the ASMX name. For example, to access the Alerts Web Service, the path would be http://<server>:port/_vti_bin/alerts.asmx, and to access the Workflow Web Service, the path would be http://<server>:port/_vti_bin/workflow.asmx. When you specify the service, you also assign a Service Name to it that you can use in the code (as you will see in the following example). Note that in code, you must set the HTTP path (the reference just gives you access to the namespace locally; on the destination system, it will be different). For example, adding a reference to the Search Service would look similar to this:



Example: Using the SharePoint Search Web Service

All of the SharePoint Web Services are called and used in the same way. Once a reference has been added to the service itself (and named), using a service requires authenticating, and then connecting to it. This code connects to the Search Service and returns the Search Scopes enabled:

```
try
{
    // Return Search Scopes using the Search Query Service:
    SearchService.QueryService SearchService = new SearchService.QueryService();
    SearchService.Credentials = System.Net.CredentialCache.DefaultCredentials;
    // Notice the setting of the http path:
    SearchService.Url = "http://mossspsdev:14584/_vti_bin/search.asmx";
    DataSet OurData = SearchService.GetSearchMetadata();
    foreach (DataRow ADR in OurData.Tables["Scopes"].Rows)
    {
        string NameOf = (string)ADR["Name"];
        string Description = (string)ADR["Description"];
    }
}
catch (Exception ErrInRetry)
{
    return;
}
```

In the previous example, the connection is made using Default Credentials (also known as the current user). In some cases, you may be accessing something that requires more privilege than the current user has, so just like impersonation, in code you can use different credentials:

```
NetworkCredential credential =
   new NetworkCredential("Username","Password","DomainName");
CredentialCache cc = new CredentialCache();
// You can use Windows Integrated to login:
cc.Add(new Uri(DLPWP_InstalledOnServerUrl), "NTLM", credential);
// OR (Alternately), you can use Basic (clear text):
cc.Add(new Uri(DLPWP_InstalledOnServerUrl), "Basic", credential);
// This sets the Credentials for the connection:
searchService.Credentials = cc;
```

Microsoft Office Search

The Office Server Search service provided with MOSS is a server-side service that provides the crawl, indexing, and query services for SharePoint content and other sources defined as Search Scopes. Like all of SharePoint, the Search Service is intended to be completely extensible, making it ideal for custom controls and web parts. Queries themselves are based on SQL-like syntax and feature full-text search, scope, and keyword search. All search results are conveniently returned in formats that can be easily used or displayed using Xsl or a data grid.

As the Enterprise Search, Office Search is not part of the SharePoint Search used in local sites, but is, in fact, completely separate and uses different storage. WSS Search uses the database, whereas Office Search uses its own database and uses actual files for indexes (files are used so it can copy index data from the index server to servers that are running the Search Query Service). Intended as a global site search, the Search Service runs within a Shared Services Provider. All of the sites and Site Collections that have been assigned to that SSP get indexed by the search. There can be any number of SSPs within a farm and each can have its own Search (as a Feature, Search can also be disabled).

The actual Search Service is actually configured in two different places: First, it has to be configured from the SharePoint 3.0 Central Administration Site, where the service must be enabled and service account assigned. Second, the running aspects of search are handled in SSP administration. There the service can be enabled/disabled, the service account set, along with other search properties including Search Scopes, crawl schedules, and so on. Individual site collections can control how search results are displayed using the default or custom search scopes and can define their own search scopes (subsets of the SSP scopes) as well as Keywords and Best Bets.

What Is Office Search

The Microsoft Office Search Service has two primary components:

- **The Index Engine** This handles the actual search of content, crawling of links, and storing (based on properties you can control) of information into the Content Indexes. The Index Engine uses:
  - **Content Sources** These specify what to search; this can be a website, Business Data, a file system, Exchange Folders, and even Lotus Notes folders.

- **Scopes** This sets scoping around indexed content. This can be high level, like an entire portal, or granular, such as limiting to a single type of document or site.
- **Crawl Log** This is used to record what's going on while the Index Engine is crawling the content on the Content Sources.

NOTE *The Index Engine uses IFilters, which tell it how to read a non-Office document and break it up into text that can be indexed. For example, Microsoft includes the Visio IFilter, which is used for Visio documents. Depending on the custom content, IFilters are usually available third-party (and must be purchased), though the most common, such as PDFs, are freely available. It also uses Protocol Handlers that enable content sources to be opened in their native format (such as Lotus Notes).*

- **The Query Engine** This handles all inbound queries against the Content Indexes, based on the properties (again, user definable) provided by the web part or search web service.
  - **Schema** This maps the properties that are searched and stored in the content indexes.
  - **Best Bets** This creates a subtext into the indexed content based on Best Bets, which can be individually searched and which rank higher in the search results.
  - **Keywords** This creates a subtext into the indexed content based on defined keywords, which rank higher in search results.

Search also has a few options that effect the way search works:

- **Stemming** This can be enabled or disabled on queries. If enabled, this applies a stemming algorithm, which reduces words in text to their base form (for example, *rider, riding,* or *ridden* to *ride*) and usually broadens the free-style text search results. This is disabled by default, which provides a more exact search. Note that if using the FullTextSqlQuery class, this only applies to terms from the FREETEXT SQL syntax.
- **URL Smashing** This can be enabled or disabled like stemming; if enabled, URLs are smashed to lowercase. The default is True (enabled).

Search also uses several configuration objects to perform both collection and query:

- **Content Sources** Each URL, file path, and Exchange Folder to be searched is created as a Content Type; these define the starting URL addresses, file paths, Exchange Folder address, and so on, that the Index Engine uses when starting a crawl. In addition to SharePoint content (naturally), Search can crawl any standard website (regardless of back end), file folders (both local and network shares), Exchange Folders (e-mail and group folders), and Lotus Notes folders (if installed). Also, if using MOSS, it can search Business Data Content, which allows actual indexing data in databases and web services. There are a few predefined types (the SharePointContentSource and the BusinessDataCatalogContentSource that are presets to match the type of content. Note that multiple starting addresses can be used in a single content source.
- **Shared Search Scopes** Search scopes are based on specific Content Sources and are just what they sound like. They scope the results returned in a search by some

criteria—in other words—limiting or filtering the results. For example, one of the default scopes is for People; this scope searches only User Profile information and uses specific fields to look for matches. These can be customizable in the UI for any purpose, such as limiting all returned results to English, for example. Scopes use scope rules that define what the criteria is based on, whether it's the address or some the value of property (like Division = Metro); for Outlook users, this is pretty much the same as adding Rules for your Inbox. Note that these are called *Shared*, not because they happen to be in the SSP, but because these are the scopes shared to the sites.

- **Document Property Mappings** These are used to map a specific column in a document to one returned in the search results. They are also used to provide crawled properties (columns stored in index) to columns in SharePoint (with meaningful names) for use by users.
- **Server Mappings** These are used to map URLs and paths returned in the Search Results into more friendly names—both for looks and to mask the actual location in the browser URL.
- **Relevance Inclusions** These are a series of settings that allow tweaking of the search results for the purposes of ranking and relevance to the users' search request. When enabled, these settings will push the ranking (relevance) up a notch over other results. This includes:
  - **Click Distance** This is the number of clicks away from the current location.
  - **Hyperlink Anchor Text** This pushes the title that goes with a URL field.
  - **URL Surf Depth** This is the number of pages down to crawl.
  - **URL Text Matching** This uses the actual URL as part of the ranking criteria.
  - **Automated Metadata Extraction** This extracts item metadata in the search automatically.
  - **Automatic Language Detection** This ranks based on the specific language of the content.
  - **File Type Relevancy Biasing** This leans toward a particular kind of file (for example, Word documents).
  - **Enhanced Text Analysis** This provides enhanced word breaking and stemming.
  - **File Type Inclusions** This is simply a list of file types that search should include or ignore when crawling content. This is useful for filtering out gunk or a particular file type that has no useful or searchable content (like a CAD file).
- **Logging** The Logging Feature in search is actually a semi-dashboard for reviewing content that has been crawled and looking at what users are doing with search (what they are searching for, how often, and so on). Split into a query log and a crawl log, these can also be used in other statistical programs for in-depth analysis on usage, user habits, and so on.
- **Site Search Scopes** These are the same as Shared Search Scopes with two exceptions. First, Site Scopes only apply to the specific site and the sub sites in the collection, and second, Display Groups can be defined to group results in a more relevant way specific to the site (in other words, in the terms that make sense in the site). Note that most portals will suffice with using only Site Search Scopes, since they can be managed by the Site Administrator.

- **Site Keywords and Best Bets** Keywords are basically some common words and/or phrases that have special significance in the context of the sites used. For example, a common keyword in CRM would *Client Account.* Best Bets are a way in which to artificially increase the ranking of a specific item (you see this in every e-commerce site running today). Note again that these are set at the site level and apply only to the Site Collection/sub sites.

How Search Works

Office Server Search is based on a crawling method in which it begins with a site and then parses the page content and related links. The crawl will follow links to a specified depth (meaning how many clicks deep). How this actually works is covered in the following sections.

Crawling

When the Index Engine is started (either on demand or via schedule), it puts in a request to a memory process called the Filter Daemon, which handles the reading of the content. Starting from the address (URL, file system, and so on), the Filter Daemon reads the site content (much like a browser), and then attempts to apply Protocol Handlers to parse the content. (This is a fancy name for a translator; for example, there is a Protocol Handler for HTML that knows how to separate HTML tags from the content.) Using the Protocol Handler, the Filter Daemon knows how to extract details from various sources (sites, documents, files, Exchange Folders, and so on). This also includes the Business Data Catalog, if you're using MOSS with the Enterprise License.

Once the content is found and extracted into chunks, IFilters (mentioned previously) are used to filter that content with whatever mapping the filter provides (such as finding text in a PDF). After this filter process, the actual content and metadata (information about the content) is sent back to the Index Engine. From there, the engine uses wordbreakers and stemmers (if enabled) to convert the returned content into known words and phrases. It then stores it into the content indexes (content indexes are both tables stored in the database and files used when propagating search in a server farm; these simply hold the metadata needed to look up an item).

Search Queries

Searches queries are based on parameters passed to the Query Engine through the User Interface or Web Service, or programmatically through the Object Model. The actual entered (or supplied) search string is run through a specific language wordbreaker (just like content; if the language is not available, the Query Engine uses a default that handles things like tabs, spaces, and so on). It is then run through a stemmer to create phrases and words that help match it against exact- and near-match indexed content. The actual search combines the string with Search Scopes, which add-on specific criteria to help widen or narrow the search results.

Using Microsoft's Full Text SQL syntax, the Query Engine searches the Content Indexes for matches to the search string and—for any found—the properties about those items that are stored in the Property Store. After loading, the store is compared to verify that all items are indeed a match and are ordered by relevance (weighted and using Keywords and Best Bets). Then, the entire result set is sent back to the Query Engine. The Query Engine sifts through the returned items and verifies the user has access to each, trimming out those the user doesn't have access to. When this security check, is complete, the output is returned to the caller. In the UI, this is the Search Results page; the Web Service returns a data set and the object model returns a ResultTableCollection.

Propagation

Office Search uses propagation (duplication) of index data by synchronizing files between search (query) providers in the server farm; this provides a localized search, which is much faster than reaching into a database or across the file system. Placement of files is set when SharePoint is installed so there is no configuration required by the user.

Tweaking Search

When it comes to search, most people are simply interested in getting the right information quickly. At the enterprise level, there shouldn't be any mystery to searching, and certainly having to endure endless searching is unacceptable. Putting aside the configuration and web parts, before you set your search up you should be considering what the content is, who will be using it, and the kind of things they will be searching for (typical terms here are very important). Sure, you'll tweak this as time goes by, but you need to get pretty close to the mark right out of the gate to be successful. To tweak search, you have to really plan it out. There are a lot of factors that affect how search runs from disk space to metadata, and if you have a lot of content, rebuilding indexes and things like that are not necessarily fun activities.

Step 1 To begin with, you need to plan out your server configuration. This isn't complicated but has to be done during the configuration of the server farm. A server can take on a few roles with regard to search; it can be an indexer, initiating crawls and storing results; or it can be a query handler that accepts search queries and returns the results or both. For best performance, indexing runs on a dedicated server and query will run on all web server front-ends.

Step 2 During the configuration process, you are given the opportunity to choose where the index files will be stored on the server. By default, this is the SharePoint Hive in the Logs folder, which should immediately be changed to a non-system drive, since the default for SharePoint usually is. The disk is a critical factor in performance: the slower the access time, the more fragmented, and so on, and it all affects how search will perform overall (this goes for all index file locations). Also, a tip to prevent problems: the SharePoint Search, Content, and all Application Pool accounts used should have full ACL access to the file system where the files are stored.

Step 3 Once you have the basic service installed, you have to plan out the configuration of the Content Sources and Search Scopes that will be available to users. Here's where you need to put some thought into what the users need: giving them scopes that are too broad or indexing irrelevant content sources will produce undesirable results. Spend some time matching the kind of content that the sites will contain and find the horizontal common threads across them (names, terms, phrases). For example, if a company builds software products, it would be typical for all users to want to search by product; thus, a scope By Product enables them to know that what they are searching for will be specifically related to products. If you are using multiple sites, be sure to consider the content in all of them, as this may produce broader scopes than anticipated. You can use additional Site Scopes to narrow this down.

Step 4 Next, you need to look at the search settings based on relevancy, which allows real tweaking of how the search will rank results it finds. Each of these has their own effect, so here's a few tips for each:

- **Click Distance** Use this only if the sites have reasonably related content that is hierarchy-based; in my experience, this is best used for Publishing Sites.
- **Hyperlink Anchor Text** This is most useful for Portal Sites with sub sites since site directories readily have this information.
- **URL Surf Depth** In Collaboration Sites, this should be disabled (you want to be sure you search the entire site tree) and should be set as high as possible when crawling external sites, Exchange Folders, and so on, to prevent digging too deep into a particular content source.
- **URL Text Matching** This is only really useful for SharePoint sites, since the content in most cases (documents and so on) is a part of the URL.
- **Automated Metadata Extraction** This option is only useful if the metadata is worth extracting. On external (non-SharePoint) sites likely to provide foreign (unknown) content, this should be disabled. If this is enabled and you are seeing performance problems, you may have insufficient memory on the query server(s) or be low on disk space.
- **Automatic Language Detection** This is useful only in multi-lingual sites (Variations).
- **File Type Relevancy Biasing** This is useful when there are diverse types of files, but certain ones are more commonly searched for. For example, if a specific file type is used by Sales to identify clients, this would push client documents to the top of the search.
- **Enhanced Text Analysis** This refines search results for better relevancy, and although it actually hurts performance on the index server, it will help on the query side; if you have the time to run the indexing and meet user requirements for seeing newly posted items, go for it—but be sure you have plenty of disk space where the index files are stored.
- **File Type Inclusions** This may sound fairly insignificant, but it isn't. Users have bad habits of uploading all kinds of things that are not important in the scheme of things. Filtering out file types that may represent junk will save space and resources overall.

Step 5 You'll also need to know how search is being used so you must configure logging. While it is a drag on performance, it is a necessary evil to know how (and if) a site is being used. While logging into something like an e-commerce site is a natural candidate for use in figuring out what a customer does, it plays just as important a role in collaboration. Knowing the kind of content that is being accessed, the areas of the site that are stale, and so on, is just as important. Fortunately the search logging is quite useful, since it will be your guide to developing Keywords later on.

In a new installation, it is ideal to use heavy logging when first starting out in order to understand user trends and potential bottlenecks in the system and to give you an opportunity to measure performance. After the initial trends have been established and you can estimate the click time (usual time and paths users take in the site), you can reduce logging to snapshots and use the trends and time estimates to generate accurate statistics using much smaller samples. If you are not too concerned with actual statistics, reduce logging to errors only for performance.

Step 6 Next, users need to do a few things on the content side to really make search work. One of the most overlooked is the metadata (list items) that SharePoint provides. If you look at a Document Library, it is a list with documents attached; if the content of the document cannot be indexed, it can mean someone gets a nasty phone call because a VP can't find a presentation. The key to this is the List Item Columns (the metadata) as it tags the document for search. If the list has columns that describe the document, the document content isn't as important and also, users don't have to open a document to know what it is. By providing set metadata rules for all types of content, search consistency and much better results can be achieved.

You also have to set expectations, because it takes time to search and index data. When a document is uploaded, it won't be available in search until the Index Engine can crawl it. If running by schedule, such as on the hour, this means up to 60 minutes before it will appear. If users are aware of this, they won't get ticked off and start calling you because you will have managed their expectations. Also note that as a best practice, I suggest you make heavy use of Site Content Types to eliminate the need for custom metadata. You should also enforce governance rules by providing default templates for lists and libraries to ensure metadata consistency in the site (and search).

Step 7 Return to step 1 and do it all over again! I suggest repeating this step three or four times.

Building Custom Search Tools

The Search Object Model provides both the administrative side and the query side. Using the Administration Object Model covers most of the things that can be done with the service, like creating a new content source, setting crawl schedules, and so on. The Runtime Object Model provides the same search capabilities as the UI and then some; since the data is in its original form, search results can be manipulated very easily and used to automate otherwise time-consuming tasks.

The Administrative Object Model allows for most of the administrative tasks. It is likely most useful in creating Windows applications or web parts that can micromanage the search for non-administrative users and utilities such as synchronizing search schedules on multiple SSPs. You could also use it to simply manage the schedules of things from a central point without having to navigate to different sites.

The Runtime Object Model is where you can really get creative since there are all kinds of things you can do with searches behind the scenes. Using searches as a data source to perform other actions is one way; another is to provide supersets of search that are not out of the box (for example, a well-known client of mine always needed a way to search documents that were discoverable but couldn't be opened and wanted to try to merge those with results they could open). From an extensibility point of view, the possibilities are quite impressive. Note that you should be aware that the Runtime Object Model must always be used within the context of a site.

Using the Search URL Parameters

Building many kinds of custom search tools can be done using query parameters as part of the URL. As does the Advanced Search Control does itself, various Keywords and Scopes can be specified by adding a Query String to the URL. The format for the search string is

based on the search destination page, Results.aspx. You can see this for yourself: simply enter something to search and then look at the URL used. The format of the base URL is always:

```
http://<server:port>/<search center>/results.aspx?<keywords, scopes, and so on>
```

For example:

```
http://mossspsdev:14584/SearchCenter/Pages/results.aspx?k=MyForm
```

Search parameters can also be combined. For example, to search by keyword and a scope, the format would be:

```
http://mossspsdev:14584/SearchCenter/Pages/results.aspx?k=region&s=All Sites
```

As a quick tip you can copy this URL from your server to be sure it is correct. From the base URL, you can add several parameters that will customize the search. In code, you can customize this with input from the user to create many kinds of custom searches.

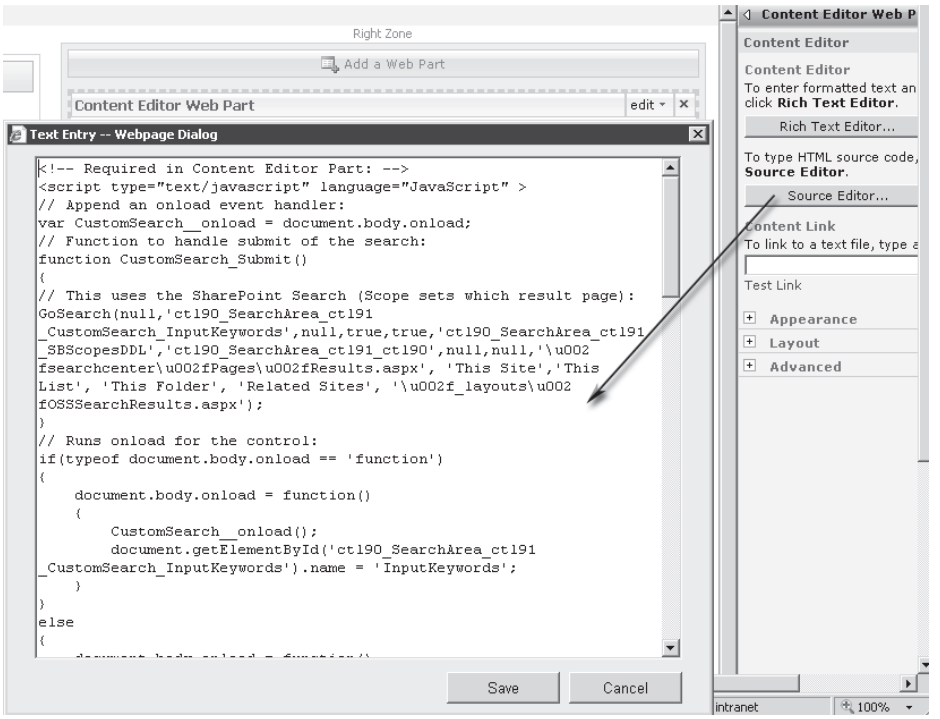
Parameter	Description	Example
k	Specifies Keywords to include in the search; you can include + and – here as inclusion/exclusion (for example: +this+but–not–that). If used, this must be the first parameter specified.	results.aspx?k=sharepoint%20search
s	Specifies the Search Scope(s) to be used; scopes must be separated by %2c.	results.aspx?s=All Sites%2CLocal Sites
v	Sets the Result View that will be used for the query; this is either Relevance or Date.	results.aspx?v=relevanceresults.aspx?v=date
start	Sets which Results page number to show.	results.aspx?start=2

Using Script to Create a Custom Search Part

There are many occasions where you might want to provide search in a format outside of those enabled by the search parts; perhaps you want to set a custom scope selection or add search to a page that doesn't have it already. You can create a Custom Search Web Part, but with a bit of JavaScript knowledge, the same thing can be accomplished with the Content Editor Web Part.

Creating this is easy; navigate to the page you want to add the search to, edit the page, and then add a new Content Editor Web Part. When the part appears on the page, click the Open the Tool Pane link. Click the Source Editor button to open the Source Editor window and then enter in the code that follows. Note that I highly advise that you build this code in an

appropriate editor, such as SharePoint Designer or Visual Studio. The Source Editor window is a very basic text editor and provides no colorization of code, error checking, and so on:



The first script line includes the SharePoint Search.js file into the page (this is the actual JavaScript used by SharePoint Search). If the page already has a search part on it (in the heading or page), you should *not* include this line as it will already be included:

```
<!-- This line should only be included if search is NOT on the page! -->
<!-- Note the rev= value is a random number to make it unique -->
<script type="text/javascript" language="javascript"
  src="/_layouts/1033/search.js?rev=zqBjfue%2Foi3KG5XVf%2FStmA%3D%3D"
 defer></script>
```

Following this is the script for actually adding the controls and functions necessary:

```
<!-- Required in Content Editor Part: -->
<script type="text/javascript" language="JavaScript" >
// Append an onload event handler:
var CustomSearch__onload = document.body.onload;
// Function to handle submit of the search:
function CustomSearch_Submit()
{
// This uses the SharePoint Search (Scope sets which result page):
// Note: keep this on one line (broken here for readability):
GoSearch(null,'ctl90_SearchArea_ctl91_CustomSearch_InputKeywords',
  null,true,true,'ctl90_SearchArea_ctl91_SBScopesDDL',
```

```
  'ctl90_SearchArea_ctl91_ctl90',null,null,
  '\u002fsearchcenter\u002fPages\u002fResults.aspx', 'This Site',
  'This List', 'This Folder', 'Related Sites',
   '\u002f_layouts\u002fOSSSearchResults.aspx');
}
// Runs onload for the control:
if(typeof document.body.onload == 'function')
{
    document.body.onload = function()
    {
        CustomSearch__onload();
// Note: keep this on one line (broken here for readability):
document.getElementById(
   'ctl90_SearchArea_ctl91_CustomSearch_InputKeywords').name =
   'InputKeywords';
    }
}
else
{
    document.body.onload = function()
    {
        eval(CustomSearch__onload);
// Note: keep this on one line (broken here for readability):
document.getElementById(
    'ctl90_SearchArea_ctl91_CustomSearch_InputKeywords').name =
    'InputKeywords';
    }
}
// Captures the Enter Key being pressed:
function CustomSearch_CaptureEnter(event1) {
    var kCode = String.fromCharCode(event1.keyCode);
    if(kCode == "\n" || kCode == "\r")
    {
        CustomSearch_Submit();
    }
}
</script>
<!-- HTML Code Begins Here: -->
<table id="CustSearchTable" border="0" cellpadding="0" cellspacing="0" width="100%">
  <tr>
    <td valign="top"><div WebPartID="00000000-0000-0000-0000-000000000090"
        HasPers="true" id="WebPart90" width="100%" OnlyForMePart="true"
        allowDelete="false" style="" >
      <div id="Div1">
        <div>
  <!-- This value of this must point to the URL of the top site: -->
          <input name="ctl90$SearchArea$ctl91$ctl90" type="hidden" id="ctl90_SearchArea_
ctl91_ctl90" value="http://mossspsdev:14584" />
          <table class="ms-sbtable ms-sbtable-ex" border="0">
            <tr class="ms-sbrow">
<!-- Scopes: These can be omitted if not needed: -->
              <td class="ms-sbscopes ms-sbcell">
<!-- You can set your own scopes: -->
                <select name="ctl90$SearchArea$ctl91$SBScopesDDL"
                  id="ctl90_SearchArea_ctl91_SBScopesDDL"
                  title="Search Scope" class="ms-sbscopes">
```

```
                  <option value="This Site">This Site: MOSS -
                         The Complete Reference</option>
                  <option selected="selected" value="">All Sites</option>
<!-- Custom Result Pages can be set here: -->
                  <option value="/searchcenter/Pages/peopleresults.aspx">People</option>
                </select>
              </td>
              <td class="ms-sbcell">
                <input name="ctl90$SearchArea$ctl91$CustomSearch_InputKeywords"
                   type="text" maxlength="200"
                   id="ctl90_SearchArea_ctl91_CustomSearch_InputKeywords"
                  accesskey="S" title="Enter search words" class="ms-sbplain"
                  alt="Enter search words"
                  onkeypress="javascript:return CustomSearch_CaptureEnter(event);"
                  style="width:170px;" />
              </td>
              <td class="ms-sbgo ms-sbcell">
<!-- Custom Images can be set here: -->
                <a id="ctl90_SearchArea_ctl91_CustomSearch_go" title="Go Search"
                 href="javascript:CustomSearch_Submit()"><img title="Go Search"
                 onmouseover="this.src='/_layouts/images/gosearch.gif'"
                 onmouseout="this.src='/_layouts/images/gosearch.gif'" alt="Go Search"
                 src="/_layouts/images/gosearch.gif" style="border-width:0px;" /></a>
              </td>
<!-- Advanced Search is also optional: -->
              <td class="ms-sbcell ms-sblink">
                <a id="ctl90_SearchArea_ctl91_CustomSearch_AdvSearchLink"
                title="Advanced Search"
               href="/searchcenter/Pages/Advanced.aspx">Advanced Search</a></td>
              <td class="ms-sbLastcell"></td>
            </tr>
          </table>
          </div>
        </div>
      </div>
    </td>
  </tr>
</table>
```

Note that there are some optional items in the code; for example, you can omit the Advanced Search link, set your own Results Pages, and so on. Clicking the Save button saves the code and returns you to the page. Since the default web part title is Content Editor Web Part, I've changed it to My Custom Search. Click the OK button in the tool pane to exit and then return to the Page Editing Mode; exit Editing Mode or publish the page to see the final result:

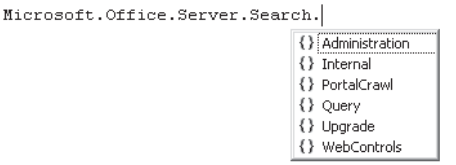


Using the Search Object Model

The object model for Search is well documented in the Microsoft Office SharePoint Server Software Development Kit (SDK), and it includes several samples that show various ways to use it. However, some lack demonstrations of all of the pieces so the following samples cover some of the more typical uses. To use the object model, you need to have a reference to Microsoft Office Search as well as include the namespaces in the code, for example:

```
// Need for Query:
using Microsoft.Office.Server;
using Microsoft.Office.Server.Search;
using Microsoft.Office.Server.Search.Query;
using Microsoft.Office.Server.Search.Administration;
```

Once these references are in, you can easily access the object model directly in the code:



Windows Application: Creating a new Content Source Here's an example of setting a start schedule, starting a crawl, and then waiting until it is done:

```
System.Uri SiteUri = new Uri("http://sharepointsite");
SharePointContentSource SPCS = null;
SPCS.FullCrawlSchedule.BeginDay = 1;
SPCS.FullCrawlSchedule.BeginMonth = 1;
SPCS.FullCrawlSchedule.BeginYear = 2007;
SPCS.SharePointCrawlBehavior = SharePointCrawlBehavior.CrawlSites;
SPCS.Update();
SPCS.StartFullCrawl();
bool IsFinished = false;
do
{
   if (SPCS.CrawlStatus == CrawlStatus.Idle)
   {
      IsFinished = true;
   }
} while (!IsFinished);
```

FreeText and Keyword Search The Free Text Search Object allows search using straightforward Full Text SQL syntax and allows selection of Scope:

```
// Set the Query:
// Note: keep this on one line (broken here for readability):
string QueryTextValue1 =
 "SELECT Title, Path, Description, Write, Rank, Size FROM Scope()
  WHERE \"scope\"='All Sites'";
// Setup the Full Text Query using context (i.e. in SharePoint):
FullTextSqlQuery FTSQ = new FullTextSqlQuery(ServerContext.Current);
// Setup the Full Text Query using site connection:
```

```
SPSite SearchSite = new SPSite("http://sharepointsite");
FullTextSqlQuery FTSqlQueryObject = new FullTextSqlQuery(SearchSite);
// Set the type of results and the query:
FTSqlQueryObject.ResultTypes = ResultType.RelevantResults;
FTSqlQueryObject.QueryText = QueryTextValue1;
// Execute the search (this returns a ResultTableCollection):
ResultTableCollection SearchResultsTableCol = FTSqlQueryObject.Execute();
ResultTable RelResults = SearchResultsTableCol[ResultType.RelevantResults];
// Note that you reference the different 'types' returned:
ResultTable HCResults = SearchResultsTableCol [ResultType.HighConfidenceResults];
```

Using the Keyword query is exactly the same except the query is limited to the Keyword only:

```
// Set the Query:
string QueryKeyWord = "Region";
// Set the Site to search:
SPSite SearchSite = new SPSite("http://sharepointsite");
KeywordQuery KeywordQuery = new KeywordQuery(SearchSite);
KeywordQuery.QueryText = QueryKeyWord;
```

Web Service Search Like the object model, the Query Web Service can be used to do a Search Query; to use it you must have a web reference to the search.asmx service in the project. The format of the query structure is different as it has to be wrapped for SOAP. Also note that the credentials are the default. In the case of the Web Service, you may have to impersonate (see the start of this chapter for more information on how to do this):

```
// Setup the query:
string XMLQueryString = "<QueryPacket xmlns='urn:Microsoft.Search.Query'>";
XMLQueryString += "<Query><SupportedFormats><Format revision='1'>";
XMLQueryString += "urn:Microsoft.Search.Response.Document:Document</Format>";
XMLQueryString += "</SupportedFormats><Context>";
XMLQueryString += "<QueryText language='en-US' type='STRING'>";
XMLQueryString += "Region"; // Keyword to search for
XMLQueryString += "</QueryText></Context></Query></QueryPacket>";
//
QueryWebServiceProxy.QueryService SearchService = new QueryWebServiceProxy.QueryService();
SearchService.Credentials =
System.Net.CredentialCache.DefaultCredentials;
System.Data.DataSet SearchResultDS =
SearchService.QueryEx(XMLQueryString);
```

Summary

As covered in this chapter, the object model provides a very powerful way to use and extend the SharePoint platform, not only employing things like web parts but also using applications to do background work while running outside of the SharePoint environment. In addition, the functionality provided by Web Services extends this power even across server farms. Although it can seem daunting at first (after all, there is quite a bit to it), you will find the object model features to be your best friend when developing in and on SharePoint.

CHAPTER 19

Building Event Receivers

Event Receivers (formerly known as event handlers) are yet another tool in the arsenal for working with and extending SharePoint. Event Receivers are code assemblies that you can attach to a given list or library type and that allow you to respond to a given event, such as an item being changed. Receivers provide a way to extend SharePoint both internally and externally, because receivers are not bound to SharePoint. They are started and run by SharePoint and have complete access to the SharePoint object model (and subsequently any lists, libraries, sites, and so on, within it), but as a .NET assembly, they can call web services, access databases, and they can even send e-mail.

Event Receivers apply to all items within SharePoint including lists and libraries. When an event occurs—such as a file move, a document upload, or similar—the receiver is notified about the event that occurred, as well as about what the item (and columns) looked like before and after the event. This also includes some trapping; for example, a file delete can be prevented before it occurs.

The mechanics of an Event Receiver are the same as for Features or web parts; Feature definition files are used to define the assembly and also to define the type of list/library (and the events in that list) that the receiver will listen for. Multiple receivers can also work on the same list type (like a workflow). Like a Feature, they can be enabled or disabled by site.

What Receivers Are For

Event Receivers give you the ability to react to a list or library event within SharePoint, which includes adding, updating, and deleting any item. For example, a Receiver might trigger when a Task Item is created and in turn, create a Calendar Item on another site. One receiver can react to any number of events.

You might be asking yourself why you would use Event Receivers when you have alerts and built-in workflows. The answer is simple: there are times when a workflow is too much or not enough; for example, if you want to send an e-mail message to a third party (like an auditor) when a certain document type is published. If you're not a member of the site, you can't use alerts. You could create a very simple workflow, but you would have to add it everywhere. An Event Receiver is tied to a list type, can run on all sites, and can easily handle such one-off or otherwise unique activities.

Event Receivers are really good for handling integrity, one-off situations, and for extending SharePoint beyond what workflows can do and into other systems like Microsoft BizTalk and CRM. On the integrity side, Event Receivers can be used to ensure that items are

correctly entered or that certain conditions are met before permitting a change. Event Receivers provide before and after methods, which permit you to catch what's changing before it changes. To extend SharePoint, Event Receivers can call web services, update databases, provide input to backend systems, and so on

While it's tempting, receivers shouldn't be used to create user workflows. They are only tied to a list type, and any processing between different receivers, lists, and sites must be managed by the developer. As code, they cannot be changed easily and are not accessible to the users except to turn them on and off. As well, they have some quirks (as you'll read later in the chapter) that require double-checking statuses and other tasks that make them unusable for some situations.

Anatomy of Event Receivers

Event Receivers are simply .NET classes that inherit from the SPItemEventReceiver class in the same way that web parts inherit the WebPart class (though SPItemEventReciever is a SharePoint namespace, not .NET). This class gives you all of the methods that could be invoked, both in response to an event and within the properties of the event, which includes the event type, the site it is in, the list, the list item that triggered the event, and so on. It also provides functions such as Cancel, which enables you to stop an event.

You can consider an Event Receiver a broad brush when working with events. Unlike a workflow attached to a specific list, receivers are installed at the site level and are based on listening to a specific type of list. The type of list the receiver looks for is found by the template ID of the list (for example, an Events List Type is 106). Any list of the specified type will cause the receiver to fire (regardless of name/page).

The methods are fairly self explanatory. ItemAdding, for example, indicates that an item is about to be added. For most event types, there are two parts: ItemAdding, followed by ItemAdded, which gives you a two-state view of the event, before it happens and after the event has completed. However, you should note that this is threaded and that you can manage both states, but you cannot pass between them. For example, if you were to process something during the ItemUpdating event, you can't carry it over to the ItemUpdated event, because it runs as a separate instance. In short, each individual event step runs in its own instance of the Event Receiver.

Any particular method(s) you want to use within an Event Receiver are overridden using the standard override method, and you only need to specify the methods you want to use. You can access ItemUpdated only, for example, if that is the only event you are concerned with. The override statement in the code also includes the SharePoint Object Model SPItemEventProperties object, which provides you access to the event details (who fired it, data within it, and so on). The statement looks like this:

```
public override void ItemUpdated(SPItemEventProperties properties)
{
    base.ItemUpdated(properties);
}
```

Notice that I've also included the base.ItemUpdated line; this ensures that the SPItemEvent inherited code is run before your custom code is. This isn't mandatory, but I do it as a best practice. The complete list of methods is as follows:

- **ContextEvent** This fires at the start of any event but is not accessible.
- **ItemAdded** This means that an item has been added.

- **ItemAdding** This means that an item is about to be added.
- **ItemAttachmentAdding** This means that a file attachment is being added.
- **ItemAttachmentAdded** This means that a file attachment has been added.
- **ItemAttachmentDeleting** This means that a file attachment is about to be deleted.
- **ItemAttachmentDeleted** This means that a file attachment has been deleted.
- **ItemCheckingIn** This means that an item is about to be checked in.
- **ItemCheckedIn** This means that an item is checked in.
- **ItemCheckingOut** This means that an item is about to be checked out.
- **ItemCheckedOut** This means that an item is checked out.
- **ItemDeleting** This means that an item is about to be deleted.
- **ItemDeleted** This means that an item has been deleted.
- **ItemFileMoving** This means that a file is about to move.
- **ItemFileMoved** This means that a file move is complete.
- **ItemFileTransformed** This means that a file exists but cannot be overridden; actual use of this is not documented.
- **ItemUncheckingOut** This means that an item is about to be unchecked out.
- **ItemUncheckedOut** This means that an item has been unchecked out.
- **ItemUpdating** This means that the item is about to be updated.
- **ItemUpdated** This means that an item has been updated.

Within an Event Receiver, you can expose any or all of the events you need, based on the way they fire and, of course, what you want to do. Since you have complete access (in most cases) to what caused the event, you also can make decisions like "Only react to the event if the list name is X" or "Only update the status if the status in another list is set." The basic event firing pattern is shown in Table 19-1.

As you can see, the naming of the methods matches the way that they fire, although you have to watch out for those that fire additional events (such as an Add with an Attachment). One thing to be aware of is how Event Receivers are executed within SharePoint. While invoked by an event, you are running alongside the SharePoint process, and you're not part of it like a web part is. This means that your logic is based on background functionality, and you cannot interact with the User Interface, prompt users for information, and so on. Also, you don't have a web context like you do in a web part; you must use the Properties object in order to determine things such as the site you are running in. Basically, it's like a stand-alone application that simply uses SharePoint to start it up.

There is a gotcha in using Event Receivers, in that events can fire more than once even though only one action occurred. In fact, I've seen the ItemAdded event fire ten (10) times in a row! The ItemUpdated event also fires twice regularly. This is not documented at Microsoft, so I can't tell you if this looping is the intended behavior or not; even with the updates available, this still remains undocumented. Regardless, it means you have to accommodate for this in your logic. For example, to update a field from Approved to Completed within the ItemUpdate method, you would need to add a check of the status first to ensure that you do not do a double update.

List Event	Event Receiver Method Firing Order
Item Add	ItemAdding ItemAdded
Item Added with Attachment(s)	ItemAdding For each attachment: ItemAttachmentAdding ItemAttachmentAdded ItemAdded
Item Added That Has Metadata	ItemAdding ItemAdded ItemUpdating ItemUpdated
Creating a Folder	ItemAdding Item Added
Item Update	ItemUpdating ItemUpdated
Update Item Adding Attachment(s)	ItemUpdating For each attachment: ItemAttachmentAdding ItemAttachmentAdded ItemUpdated
Update Item Removing Attachment(s)	ItemUpdating For each attachment: ItemAttachmentDeleting ItemAttachmentDeleted ItemUpdated
Deleting an Item	ItemDeleting ItemDeleted
Updating/Checking in a Document (SharePoint or Word)	ItemUpdating ItemUpdated ItemCheckingIn ItemUpdated ItemCheckedIn
Moving a File	ItemAdding ItemUpdating ItemAdded ItemUpdating ItemUpdated ItemDeleting ItemDeleted

Table 19-1 List Events and Methods Fired

This looping effect can be pretty tricky. Instead of thinking in terms of "Do this and I'm done," you have to think in terms of "Do this, and then check whatever I need to so as to prevent me from doing it again." This is particularly tricky in terms of trapping for Items Added. You might also have another problem; if there's no status to check for, you might

have to add a field to the list so you can prevent a double update. Note that there are methods available called DisableEventFiring and EnableEventFiring, which are supposed to prevent this looping process for an update. While they do apparently prevent a loop within a single event, they do not stop the looping process described previously. However, you can use them as a best practice (I'll demonstrate this later on). Information and several examples on Event Receivers can also be found in the WSS SDK, though none address the looping issue—sorry! When using the SDK, be aware that some of the how-to examples are incorrect (one shows setting values on AfterProperties, which is read only).

Event Properties

When an Event Receiver receives an event, the properties tied to the event tell you what happened, where, and to whom, depending on the context. If you are in an "ing" state, such as Adding, Updating, and so on, you have event information, but not the item. In the "ed" state, such as Added, all properties are filled. Some of the properties are useful, others may be later (but they don't function now). The complete list of properties is shown in Table 19-2.

Property	Purpose	Useful?
AfterProperties	An SPItemEventDataCollection object	No, never returns any value regardless of event.
AfterUrl	The URL of the item after the change/add	Yes, this is the relative URL to the item.
BeforeProperties	An SPItemEventDataCollection object	No, never returns any value regardless of event.
BeforeUrl	The URL of the item after the change/add	Yes, only useful for a moved item.
Cancel	Cancel the current event (roll back)	Yes, best for pre-events (ItemAdding, ItemDeleting).
Context	Context of the event	No.
CurrentUserID	ID of the user	Yes, ID of the current user.
ErrorMessage	Sets Error message to display on the return of the event	Yes, allows you to notify the user (item blocked, error occurred, and so on).
EventType	Type of event	Yes, event that actually fired.
ListID	List ID (GUID) of the list	Yes, only valid for after events (ItemAdded, ItemUpdated).
ListItemID	List Item ID (Integer) of the list item in the list	Yes, only valid for after events (ItemAdded, ItemUpdated).
ListItem	Entire list item (all field values are accessible) on which the event occurred	Yes, only valid for after events (ItemAdded, ItemUpdated).
ListTitle	Title of the list or library where the event occurred	Yes, only valid for after events (ItemAdded, ItemUpdated).

Table 19-2 Event Properties, Purpose, and Use

Property	Purpose	Useful?
RelativeWebUrl	Relative URL to the parent site of the list	Yes, provides access to the parent list of the item.
SiteID	Site ID of the root site of the collection	Yes, returns the GUID.
Status	Current process status (possible values: Continue or Stop)	Yes, indicates whether a Cancel has been issued.
Versionless	Unknown	Undocumented use at this time.
WebUrl	Full URL to the parent site of the list	Yes, provides access to the parent.
Zone	URL zone such as Internet, extranet, and so on	Yes, indicates the security zone you are in; can be used to control functionality.

Table 19-2 Event Properties, Purpose, and Use *(continued)*

Creating an Event Receiver

Creating a new Event Receiver is a fairly straightforward process. Once the shell has been created, you simply add the necessary logic to the methods you choose. Packaging it up for SharePoint is done the same way as for any Feature, although the Elements Manifest definitions are unique for Event Receivers (they are used to tie a receiver to a list type). Of course, before you build one, you should balance it against the other tools available, since they don't offer the kind of flexibility that a web part (with properties) or a workflow does. I like to use them for background-type activities like dealing with events that don't really need a workflow and don't need an interface like a web part. They can also be used to handle things that might otherwise need a timer job and can prevent things from happening (such as a delete).

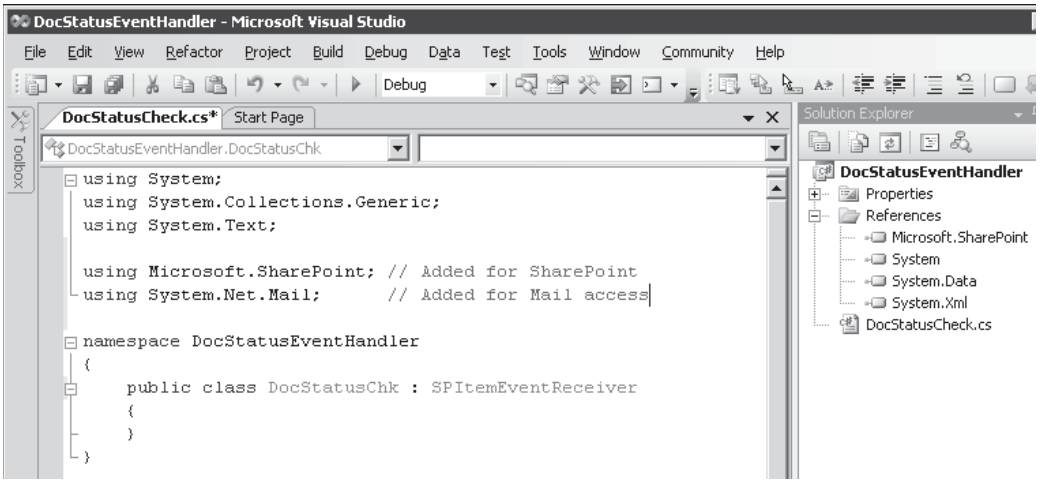
To demonstrate this, I'm going to set out a simple example based on handling two aspects of a document library (this demonstrates something that you could both do and not do in a workflow):

- Whenever an item in a document library has its status set to Final, send an e-mail to the creator of the document (this can be done in a workflow).
- Prevent items from being deleted in the library (this can't be done in a workflow).

Building the Project

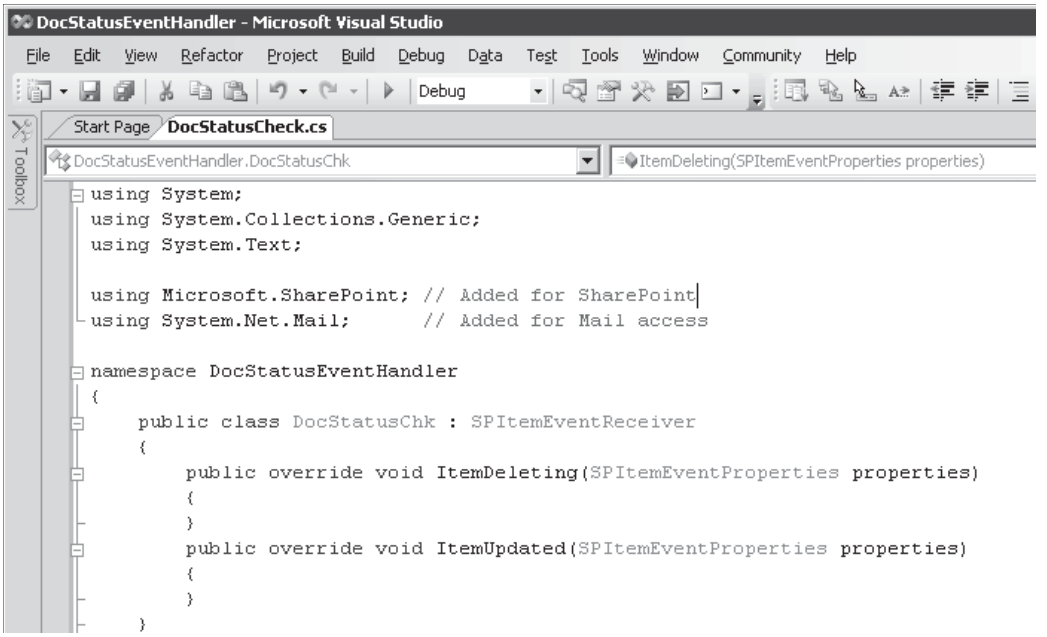
The initial shell of the Event Receiver is based on setting up just the override methods and the XML files needed to install it. By building and testing this first, it simplifies adding your own logic (and you can easily reuse the project by simply renaming it). To build the project, you must have a development system with SharePoint and Visual Studio 2005 installed (see Chapter 5 for requirements). Log in to the system and do the following:

1. Open Visual Studio and select File | New | Project.
2. From the project types, select Visual C# (or VB) and then Class Library.
3. Name the project as you like, but for this example, I'm calling it D**ocStatusEventHandler** (note that there are no VS Project templates for receivers because they are coded as classes).
4. In the Solution Explorer, right-click on References and select Add Reference.
5. In the Add Reference pop-up window, select the .NET tab.
6. Scroll down to find Windows SharePoint Services, click to select it, and then click OK (you should see the reference added).
7. Right-click on the Class1.cs file in the Solution Explorer, and then rename it to **DocStatusCheck.cs**.
8. Open the class code (if it's not open already) by double-clicking on the file in the Solution Explorer.
9. Above the namespace, add a Using or Imports statement to include the Microsoft.SharePoint namespace (if you created this in VB, add a namespace around the code now).
10. Above the namespace, add another Using or Imports statement to include the System.Net.Mail namespace (needed for e-mail).
11. Change the class name to be **DocStatusCheck** and inherit from the SPItemEventReceiver class; at this point, your code should look like this:



At this point, this Event Receiver does nothing, but it can be installed. Once you have modified the code as you saw previously, compile the project to be sure all is well.

12. Now add the override methods you will be using. In this case, it will be Item Deleting where you will block deletion and Item Updated where you will send a message. You might think you should use Item Added, but the assumption is that new items would be works in process, not the final versions (though you could add the logic there as well). The code for this is as shown:



13. First, you want to code the Item Deleting event. Our goal here is to stop the item from being deleted and send an error message back to the user. The code for this is:

```
// Do not permit deletions:
properties.Cancel = true;
// Return an Error message that will be shown to the user:
string ErrMsg = "Deleting Items from ";
ErrMsg += properties.ListTitle.ToString();
ErrMsg += " is not permitted.";
properties.ErrorMessage = ErrMsg;
```

Notice that the Properties object includes the ListTitle; this is just one of the properties available to you in a receiver.

14. The code for the ItemUpdated event is a little more complex, because you have the looping issue to deal with as this method always fires (at least) twice. You'll accommodate this by adding a field to the document library, which you can use to prevent sending it more than once. The complete code for this is as follows:

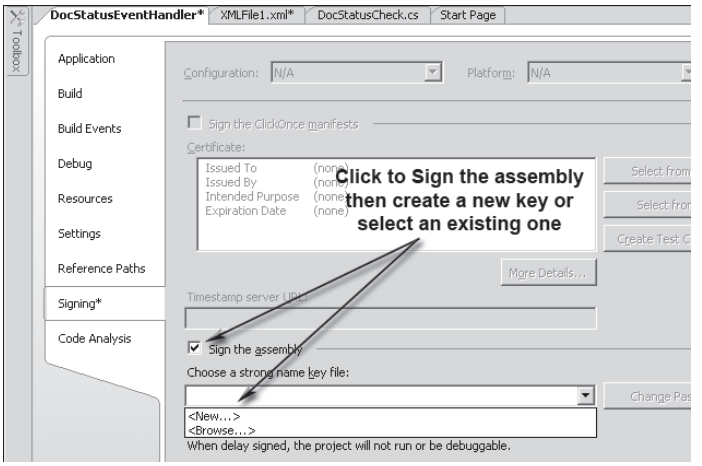
```
// Open the item to check the status of the MailSent field;
// this prevents us from updating more than once:
bool WasMailSent = false;
// Try catch just in case field not found:
```

```
try
{
    WasMailSent = (bool)properties.ListItem["MailSent"];
}
catch (Exception FldNotFound)
{
    return;
}
if (WasMailSent)
{
    // Sent - do nothing
    return;
}
// Check status of the item to determine if final:
string DocumentStatus = "";
// Try catch just in case field not found:
try
{
    DocumentStatus = (string)properties.ListItem["Status"];
}
catch (Exception FldNotFound)
{
    return;
}
if (DocumentStatus != "Final")
{
    // Not a status we are concerned with
    return;
}
// Status is final, send email to the author:
SPWeb TheCurWeb = properties.ListItem.ParentList.ParentWeb;
// Get the author (this field is in all List types):
string Author = (string)properties.ListItem["Author"];
// Get the ID:
int IDOf = Convert.ToInt32(Author.Substring(0, Author.IndexOf(";")));
SPUser CreatedBy = TheCurWeb.SiteUsers.GetByID(IDOf);
// Set the To address:
string ToWhom = CreatedBy.Email;
string SmtpServerName = "127.0.0.1"; // or server name
string FromAddress = "spadmin@mosswss.com";
SmtpClient client = new SmtpClient(SmtpServerName);
// Set From address:
MailAddress from =
 new MailAddress(FromAddress, "SharePoint Admin", System.Text.Encoding.UTF8);
// Set recipient for the e-mail message:
MailAddress to = new MailAddress(ToWhom);
// Specify the message content:
MailMessage message = new MailMessage(from, to);
message.IsBodyHtml = true;
message.BodyEncoding = System.Text.Encoding.UTF8;
message.SubjectEncoding = System.Text.Encoding.UTF8;
message.Subject = "Document has been finalized";
```

```
// Start the message:
string MessageBody = "Your document: <br>";
// Use the web url to construct a link to the document:
string UrlOf = properties.WebUrl;
if (UrlOf.Substring(UrlOf.Length - 1) != "/")
{
    UrlOf = UrlOf + "/";
}
string ItemUrl = UrlOf + "Lists/"
    + properties.ListTitle
    + "/DispForm.aspx?ID="
    + properties.ListItem.ID.ToString();
MessageBody += "<a href=\""
    + ItemUrl + "\">"
    + properties.ListItem.Name
    + "</a><br><br>";
MessageBody += "was finalized at " + System.DateTime.Now.ToString();
message.Body = MessageBody;
// Send the message:
try
{
    client.Send(message);
}
catch (Exception ErrInSend)
{
    string WhyErr = ErrInSend.Message.ToString();
}
// Now since the message has been sent, set the flag to
// prevent us from doing this again.
// Disable event firing while we update:
base.DisableEventFiring();
properties.ListItem["MailSent"] = true;
properties.ListItem.Update();
base.EnableEventFiring();
```

Notice that you employ a few things here that might need to be adjusted before this will work. For one, you are assuming that there are two columns (fields) in the list called Status and MailSent (these must be added below). Also, you are using the .NET mail object with an SMTP server of 127.0.0.1. It is likely you will use the server name instead. Also, notice in the code that the User Email is obtained using the User ID. Since you will be the one uploading a document, you will be the Author—and thus, the e-mail will be sent to you, but the account *must* have an e-mail account associated with it or this will fail!

15. Compile the project to be sure there are no errors.
16. Now you must sign the assembly to ensure that it is strongly named. In the Solution Explorer, right-click on the name of the project and select Open. When the Properties page opens, select the Signing tab:



17. Save the project to save your settings and compile the project again. Now with the Event Receiver built, you need to build the solution files needed to deploy it for testing. The first file you need to build is the Feature.xml file (same as any other feature). To do this, you first need a unique GUID; in Visual Studio 2005, this should be available in the Tools menu or you can use the GUIDGEN.exe utility. From the computer desktop, select Start | All Programs | Microsoft Visual Studio 2005 | Visual Studio Tools | Visual Studio 2005 Command Prompt, and when GUIDGEN opens, there is already a unique GUID shown. Simply click Copy to select the one shown or click New GUID and then Copy.
18. In Visual Studio, create a new XML file via File | New File | File. Then, from the dialog box, select General in the left pane, XML File in the right pane, and click OK to create.
19. Enter the Feature definition code as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<Feature Scope="Web"
   Title="Document Status Author Notification"
   Description="Notifies author when documents are approved"
   Id="FA70EB99-0321-446c-9226-B9006DADD237"
   xmlns="http://schemas.microsoft.com/sharepoint/">
      <ElementManifests>
          <ElementManifest Location="Elements.xml"/>
       </ElementManifests>
</Feature>
```

20. As you can see, you've set the Scope to Web, set a title (this displays in the Features selection in Site Administration), and you have a description that will tell the user what this Feature is—remember the GUID (Id=) should be yours, not the one shown. Click OK and save this file into your Visual Studio project folder named Features.xml.

21. Next is where it gets interesting. Unlike other features where you identify what is available in the Elements file, for receivers this file is used to indicate which type of list and what kind of events to fire on (remember this means every list of that type in the site!). For each list type (the List Template ID from the ONET.XML file definitions; the SDK has the complete list), there is a section for each type of event to be triggered, which includes the name of the assembly (in this example, DocStatusEventHandler), the event type, a sequence number (receivers can run in a sequential format), and the assembly and data information that would be included in a normal Elements file. For this example, you need to define a receiver for a Document Library Type (type 102) and two events: ItemUpdated and ItemDeleting. To create the Elements file, you'll also need the PublicKeyToken of the class you've just built (after it has been signed). To get this key, select Start | All Programs | Microsoft Visual Studio 2005 | Visual Studio Tools | Visual Studio 2005 Command Prompt (using the VS command prompt includes the path needed to the sn.exe file used here). From the command prompt, change your directory to the debug directory in your project.
22. Enter the command **SN -T DocStatusEventHandler.dll** (the -T must be capitalized) and record the 16-character string returned. This is the public key token for the assembly.
23. Back in Visual Studio, create another XML file via File | New File | File, and then from the dialog box, select General in the left pane, XML File in the right pane, and then click OK. When the file opens, enter the following code:

```
<?xml version="1.0" encoding="utf-8"?>
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
      <Receivers ListTemplateId="101">
            <Receiver>
                  <Name>DocStatusEventHandler</Name>
                  <Type>ItemDeleting</Type>
                  <SequenceNumber>10000</SequenceNumber>
                  <Assembly>DocStatusEventHandler,
Version=1.0.0.0,
                        Culture=neutral,
PublicKeyToken=a6c35025d6298441</Assembly>
                  <Class>DocStatusEventHandler.DocStatusChk</Class>
                  <Data></Data>
                  <Filter></Filter>
            </Receiver>
            <Receiver>
                  <Name>DocStatusEventHandler</Name>
                  <Type>ItemUpdated</Type>
                  <SequenceNumber>10000</SequenceNumber>
                  <Assembly>DocStatusEventHandler,
Version=1.0.0.0,
                       Culture=neutral,
PublicKeyToken=a6c35025d6298441</Assembly>
                  <Class>DocStatusEventHandler.DocStatusChk</Class>
                  <Data></Data>
                  <Filter></Filter>
          </Receiver>
       </Receivers>
</Elements>
```

NOTE *Don't wrap lines as I've done here for readability; also, the PublicKeyToken you will use will be your own and not the one shown.*

24. Click OK to save this file back to your Visual Studio project folder. Name the file Elements.xml, which must match the name in the previous Features.xml file.

NOTE *Events defined within the Elements file dictate the events that will trigger the Event Receiver; even if you code for the event, it will not fire unless it appears here.*

Deploying the Event Receiver

Deploying Event Receivers is a little easier than deploying a web part because you only need the DLL and the Feature Definition files to deploy it. You should note, however, that Event Receivers *must* be installed in the Global Assembly Cache (GAC) to operate. Typically, you would set up a package to deploy it like a regular feature; however, you can manually deploy it as well. For example purposes, you are going to manually deploy this one so that you know the mechanics of where everything goes. Refer to Chapter 20 for the details on building a Solution package to install.

Before deploying, you should either pick a site to use or create a brand-new blank site where you can install. You will be creating a document library in that site later, but the site must be created before the Feature can be installed. To deploy it, do the following:

1. Use Windows Explorer to navigate to the SharePoint Hive, and then locate the Features folder (under TEMPLATES).
2. Create a new folder (in Features) called **DocStatusEventHandler**.
3. Copy the Feature.xml and Elements.xml files from your project folder to the new folder.
4. Navigate to the debug directory in your project to locate the DocStatusEventHandler.dll file. Copy this file to the SharePoint site bin directory located (by default) under <drive>:\Inetpub\wwwroot\wss\virtualdirectories\<virtual>\bin. To locate the correct folder, you can use Internet Information Services (IIS) Manager in Administrative tools. Right-click on the web name, select Properties, and then select the Home Directory tab. Also, if a bin folder does not exist, you must create one, using IIS Manager to set the permissions on the folder to be Read/Executed. Note that you can copy the DLL to the correct place or simply set it as your build destination in your project (you still must add the DLL to the GAC in the next step).
5. To install the DLL into the GAC, you can open the c:\windows\assembly folder and drag the DLL into it (it cannot be copied; the GAC simply holds a reference to where the file is), or you can use the GACUTIL.exe utility. Select Start | All Programs | Microsoft Visual Studio 2005 | Visual Studio Tools | Visual Studio 2005 Command Prompt; when the command window opens, navigate to the bin directory and enter the command:

   ```
   gacutil /i <assembly name>
   ```

6. Finally, you need to install and activate the Feature using the SharePoint Command Line Administration Tool (STSAdm), which must be done from a command prompt.

Open a command window, navigate to the SharePoint Hive bin directory (12\bin), and then enter the following command to install the feature:

```
stsadm -o installfeature -filename DocStatusEventHandler\feature.xml
```

7. Next, enter the command to activate the feature on the site:

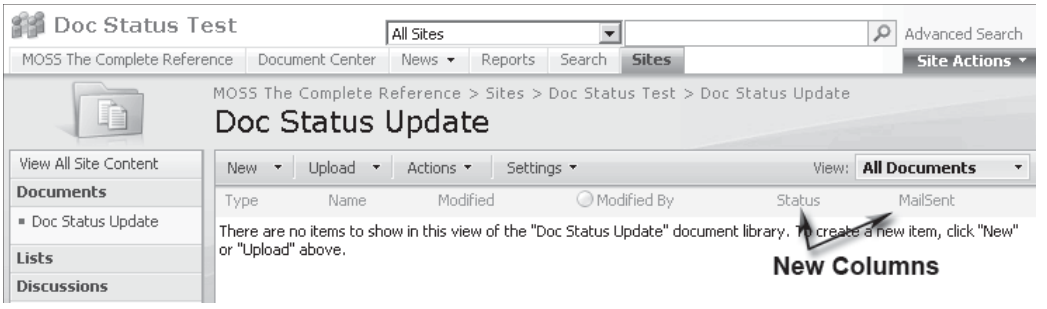
```
stsadm -o activatefeature -filename DocStatusEventHandler\feature.xml
       -url <url to site>
```

8. Finally, you must run IISReset to refresh SharePoint (use Start | Run, type in **IISReset**, and then click OK).

Pre-setup for Testing the Event Receiver

Once an Event Receiver has been installed and activated, it starts working immediately. Of course, if you started out with a blank site, there's nothing there to trigger it. It is running, however. To set up an Event Receiver for testing, you need to create a new document library:

1. Navigate to the site and select View All Site Content from the left-hand navigation.
2. When the Site Content Page opens, click the Create link.
3. On the Create Page, click Document Library under the Libraries section.
4. Name the library whatever you like (names mean nothing to receivers); leave the rest of the options as they are, and then click OK to create it.
5. When the library opens, select Settings | Document Library Settings.
6. On the Customize page, click the Create Column link in the Columns section. On the Create Column Page, enter in the name as **Status**, make it a Choice field, and then add three options: Pending, In Review, and Final. Make sure the default value is set to Pending, Add to Default View is checked, and then click OK to create the column (note that you would usually use a better name than Status).
7. Click the Create Column link again.
8. On the Create Column Page, enter in the name as **MailSent**, make it a Yes/No (checkbox) field, set the default value to No, and make sure Add to Default view is checked. Then, click OK to create the column.
9. Click the link at the top of the page to return to the library. Your view should look something like this:

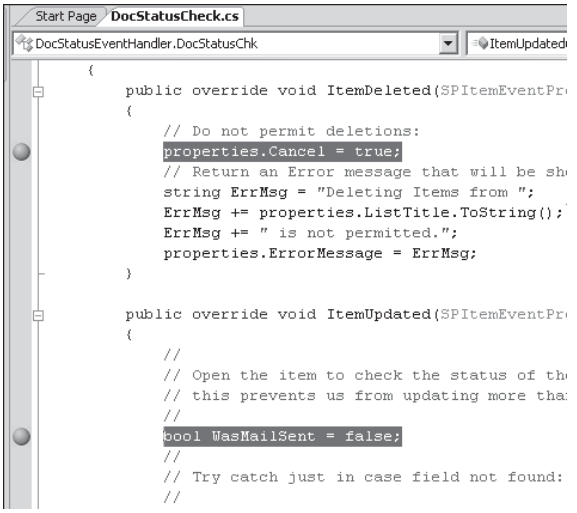


Testing and Debugging the Event Receiver

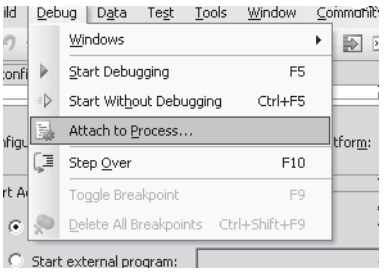
If you feel lucky, you can go ahead and test the Event Receiver using the interface. You can attempt to upload a file, change its status, and see if the MailSent flag is changed and you get an e-mail. At that point, you can try to delete the file and it should prevent you from doing it.

However, the best way to test an Event Receiver is to set it up for debugging; this will allow you to set up break points at each event to see what's going on. The normal method for debugging an Event Receiver is by attaching it to the web process of a site that is already running (starting from Visual Studio is dicey and doesn't always work). When you have compiled a DLL and put it in the GAC, a recompile can cause the debugger to think it is a different version. To use the Attach method, do the following:

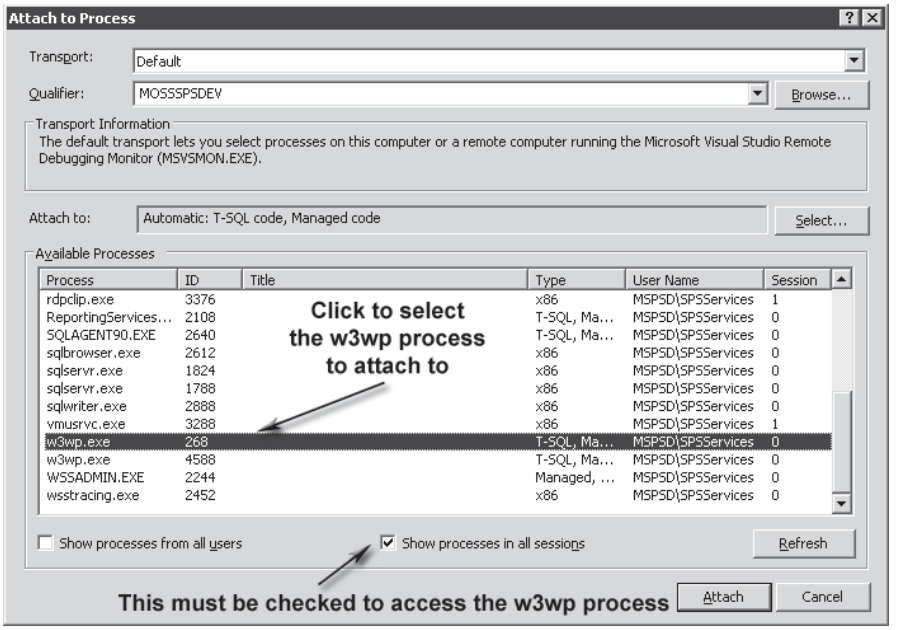
1. Open the SharePoint site where you have installed the Event Receiver and open up the document library you created previously.
2. In the Visual Studio project, set a break point at the start of the ItemDeleting and ItemUpdated events:



3. From the top menu, select Debug | Attach to Process.

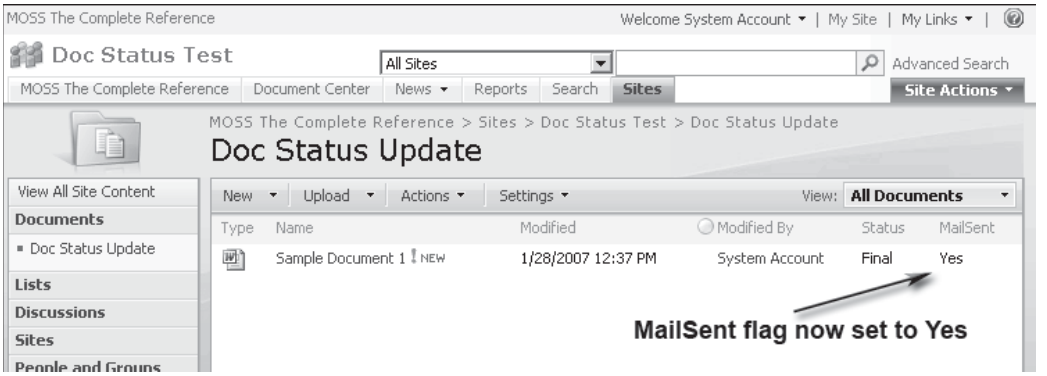


4. When the Attach to Process window appears, make sure that Show Processes in All Sessions is checked, and then scroll down the list to locate the w3wp process as shown:

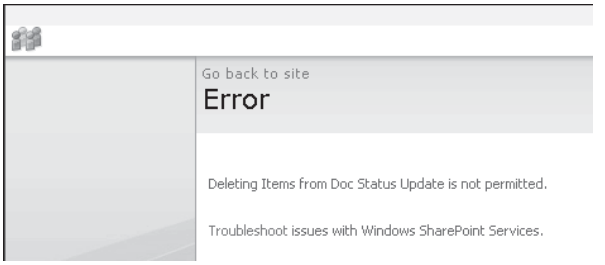


5. Click to select the process name and then click the Attach button (there's a catch here; IIS 6.0 runs in Isolation Mode—a process for every site—and since you don't know which one is for your site, simply click to select them all). When you are attached to the right process, break points in your code will appear as a solid ball; if you see an empty circle, you are not attached to the correct process. Stop debugging and try attaching to a different one.
6. With this process running, return to the browser.
7. Upload a new document and make sure that the two columns are set. Status should be set to Pending and MailSent should be set to No (unchecked). You should see the upload events fire, but because the metadata fields are not what you want, you do nothing. As you step through the code, you'll see that the first check of the MailSent passes, but since the Status field is set to Pending and not Final, you simply exit.
8. Return the browser to the view of the document library.
9. Use the drop-down menu from the document name to select Edit Properties.
10. When the Properties Page appears, change the Status field value to Final.
11. Click OK to save the change.
12. Debugging should switch you back to the Visual Studio project, and you can begin stepping through the code again. This time, however, you should drop down through the checks, create and send the e-mail message, update the status, and then exit.

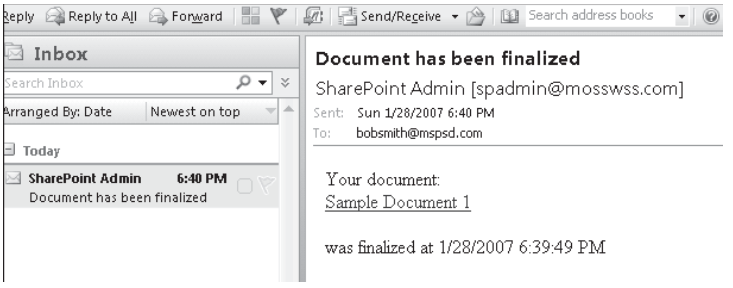
13. When you return to the browser, refresh the Document Library View and you should see that the MailSent value has changed from No to Yes:



14. The last piece to test is the receiver preventing deletion of an item. While still in debug mode, in the browser, use the drop-down menu on the document name and select Delete.
15. Click OK to confirm that you want to send the document to the Recycle Bin.
16. After you step through the code and the debugger exits, return to the browser, and you should see something like this:



17. Last but not least, if your SMTP server is operational, you should receive an e-mail message that looks something like this (be sure to enable HTML viewing):



Deleting the Event Receiver

1. Removing an Event Receiver is just like the installation you did previously, but in the reverse order, using the SharePoint Command Line Administration tool (STSAdm) from a command prompt. While you can activate and deactivate through

the User Interface, the command line removes it from the farm. To do this, open a command window and navigate to the SharePoint Hive bin directory (12\bin).

2. Enter the command to deactivate the feature on the site:

```
stsadm -o deactivatefeature -filename DocStatusEventHandler\feature.xml
       -url <url to site>
```

3. Enter the command to uninstall the feature:

```
stsadm -o uninstallfeature -filename DocStatusEventHandler\feature.xml
```

4. Finally, you must run IISReset to refresh SharePoint (use Start | Run, type in **IISReset**, and click OK).

Summary

As you've seen in this chapter, Event Receivers are a great way to extend the functionality of SharePoint internally and externally. As a Solution, they can easily be deployed to any site and enabled or disabled at will. It's also important to see the extensibility of what Receivers can do because they provide the ability to react to and use SharePoint data. While they do have some limitations, you'll find these are an excellent way to extend SharePoint into the enterprise.

CHAPTER 20

Building Web Parts

What are web parts anyway? Simply put, they have become the evolution of what every development methodology has tried to achieve in the past decade: fast ways to build applications, efficient use of tools, true reuse of code, structured development, you name it—web parts are it. Web parts can be thought of as mini-applications that provide some sort of application functionality on a SharePoint page (SharePoint lists, libraries, surveys, and so on, are all web parts). Pretty much most anything that can be done within a .NET web application can be done in a web part. Web parts are a key player in SharePoint as a platform; with them you can build entire applications that work on top of SharePoint.

So how could web parts make such an impact? Compared to a web application where you have to build a complete framework, SharePoint and web parts eliminate all of this because most of the tough stuff is done. Users are already managed, search is built in, site provisioning is done on the fly, and so on. What's left to build is simply the functionality you want.

Web parts are just a custom web control, and in fact, are built pretty much the same way. The real difference is that web parts have XML definitions around them, which enable the SharePoint User Interface (or custom .NET interface) to provide the zone-to-zone drag-and-drop functionality. In addition, web parts running within the context of SharePoint have complete access to the SharePoint Object Model, including security, user profiles, and so on. This means that web parts can use SharePoint to move items around, display business data, utilize content targeting, and so on. Web parts can also communicate with each other to provide a number of application like features, such as master-detail relationships or multi-selection displays.

Web parts are deployed to SharePoint via Solution packages just like Features. They can be selectively deployed to a farm, Site Collection, or site, and enabled/disabled by a Site Administrator. This enables granular control over where web parts are installed and available.

Anatomy of a Web Part

A web part (like an application) is made up of several components that are compiled and copied into SharePoint. In addition to the base class file (the assembly), it also includes Resources (images, CSS, script, and so on), a manifest file (a list of what's in the web part), a .webpart file (a declaration file for SharePoint), a .DDF (Diamond Directive File) used to

create a WSP file (similar to a CAB or ZIP file that stores the assembly and resources), a Features XML file (identifies the Solution in SharePoint), and an Elements Manifest XML file (identifies the elements in the Solution). I'll get into each of these later on in the chapter.

A web part is a control that sits on top of an aspx page type called a Web Part Page. A Web Part Page uses several SharePoint controls including Web Part Zones. Web Part Zones are used to manage web parts dropped into them and control the layout on the page. Within a zone, parts can be added, removed, and dragged between zones giving the end user the impression of designing the page; the zone means nothing to the web part; it controls what it outputs to the user regardless. Web parts have some standard properties that control them in SharePoint (display height, width, and so on), and they can also have custom properties that provide runtime settings that can alter the way a part works.

Probably the oddest thing about working with web parts is that it is technically not part of the page it is on, so it has no control over it. This means that anything might cause the page to refresh, so if you don't keep track of the state that your web part is in, it can be a problem. For example, if displaying data, a refresh could clear the view (I'll cover that in examples shown later on in the chapter).

The process of building a web part is pretty simple. A Class (code) is used to make a DLL, and the Resources store images and assets. These are combined with the Manifest (part components), Features, and Element Manifest XML files (for SharePoint) into a WSP file, using a DDF file. Installing in SharePoint is done using the STSAdm tool, which creates a Feature folder that holds the XML files (and potentially any assets), creates a reference to the assembly in the Global Assembly Cache (GAC) if necessary, and then loads the Web Part Definition (.webpart) into the SharePoint Web Part Gallery. If/when the Solution is enabled, users can begin using the part. A high-level view of how all this fits together is shown in Figure 20-1.

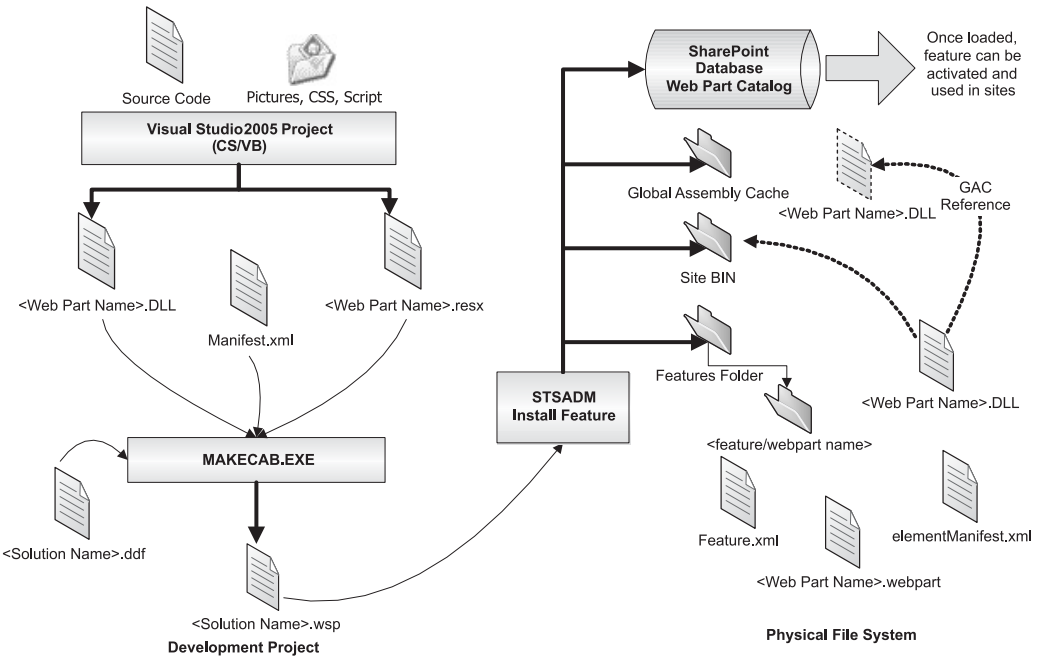


Figure 20-1 Break down of the web part components

- **The Class (source code)** Web parts themselves are class assemblies that inherit the .NET WebPart namespace. This is inherited from System.Web.UI.WebControls.WebControl, the same class used to create a custom control in .NET. The class(es) can do anything a .NET web application can do. The only exception is in Visual Studio; because it is a custom control, VSS Form Designer (allowing drag-and-drop of controls) is not available. If the web part uses controls (like a text box), you have to define it and tell it to render. This sounds like a pain, but the granular control settings make it worth it.
- **Resources** A web part can use the standard .NET resources file (resx) to reference any resources it needs including strings (titles, localization, and so on) and assets (Script, images, and CSS). In deployment, these are usually placed in the SharePoint Hive Resources folder.
- **Manifest file** The Manifest XML file defines the components needed to package up all of the necessary files into a WSP file—a WSP file is the same as a Windows Cabinet file (CAB)—and is essentially a list of the actual resource files and an entry for the web.config file (the <SafeControl entry). It's the web.config file that enables SharePoint to find (and trust) the assembly.
- **Webpart file** The webpart file is an XML file that defines the web part for SharePoint, including the assembly information, a message for errors (when the part has a problem), and any properties including the default title and description. This is loaded into the SharePoint Web Part Gallery, and the name, description in the file, and grouping within the Gallery are what users see in the Add Web Part dialog box when adding a part to a page.
- **DDF File** The Diamond Directive File (or DDF, named after the original program DIAMOND.exe) is simply a text file of instructions used by the Windows MAKECAB .exe to build a CAB or WSP file. The DDF file is the same as a .NET INF file; it contains the instructions for MAKECAB to package up the items defined in a Manifest file to create a WSP (or CAB) file. It also tells where to deploy the file(s) when it is installed (just like any installation program).
- **Features.XML file** This file defines the solution/feature for SharePoint exactly the same way that features work. This includes the title, ID, description, version, scope, and so on.
- **ElementsManifest.XML File** This file defines the elements that make up the Feature and the Module elements that define where the Feature will be installed in the SharePoint site (by default this is _catalogs/wp).
- **WSP file** The WSP file is a Windows Cabinet (CAB) file renamed as WSP for SharePoint's sake. Just like a CAB file, this contains all of the files that are included in the Solution (DLL, XML files, and so on). As mentioned previously, this file is created using the MAKECAB.exe utility; using the DDF file as its instructions, it assembles the WSP file for loading into SharePoint.

How Web Parts Work

The biggest obstacle that most people get hung up on is the way that web parts actually execute. If you are familiar with web controls, then this is not new, though you do have to accommodate for SharePoint controlling the page. If you've not built custom web controls

before, knowing what to code in which method can be a bit hard to understand. I'm hoping to be able to help get you over this hurdle.

As you can see in Figure 20-2, the life of a web part begins with the request for the SharePoint page by the user. When the page opens, the page controls are created, page events fire, and then any web parts on the page are executed. Within each web part, the execution order of the methods is OnInit, OnLoad, CreateChildControls, PreRender, Render, and Dispose. When a Postback event occurs (like a button click) in the web part, the page is reloaded, whereas on the return (Postback) trip, two additional methods fire: LoadViewState loads the current view state, and the Control Event (like OnClick) fires before the PreRender event.

The Postback state is where most developers stumble; getting everything to work on first display is usually easy, but the handling of events on the return can be tricky. The issue is that all of the controls in a web part are created in either the OnLoad or CreateChildControls events. This means that things like drop-down menus and so on are created and loaded at that time, which is fine for the first pass. When in Postback, however, the Events fire just after CreateChildControls and just before rendering, and if the controls are not in the right state, your events will not work correctly.

For example, assume you have a drop-down menu and it loads items from the database. When the user selects a new item, you want it to Postback to set some other items on the page. When OnLoad or CreateChildControls executes, you must check the Controls View State to avoid reloading it; otherwise, the control will be initialized, and you could lose what was selected. Also, the control itself does not exist until after OnLoad or CreateChildControls executes. If you are not quite grasping this yet idea, not to worry; the example coming up builds a web part that demonstrates all of this.

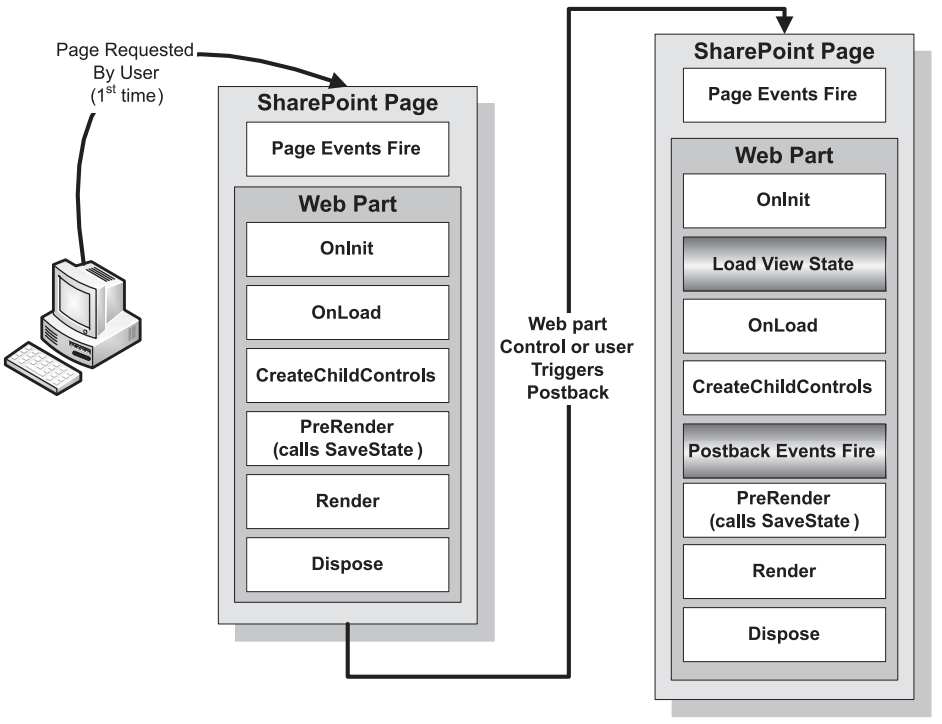


Figure 20-2 Execution cycle of a SharePoint page and web part

Building Web Part Solutions

With the release of Windows SharePoint Services 3.0 Visual Studio Extensions, building web parts and solutions (and debugging) is almost completely automatic. All that you need to do to create and deploy a solution is open a project, write some code, press the F5 key to build it, and then automatically deploy the solution to a SharePoint site (including running an IISReset!). While the automation is very cool, it only works in development and only covers installing; removing is a manual process and the Web Part Definition (.webpart) must be deleted through Site Administration. The manual way that I'll be covering this may seem time consuming, but this method is the only way to deploy to server farms. I will, however, cover a shortcut for creating deployment packages that you will be able to reuse. Note that again, I'm using C#; for VB, this example should be generic enough to convert to your situation.

Note a few assumptions are made here for the examples in this chapter. You should have a complete development environment that includes Visual Studio 2005 and SharePoint (on the same system); see Chapter 5 for the requirements. For using Visual Studio, it is assumed that you know how to select a project type, create a new project, add references to the project, compile, and debug a project. You must also have Windows SharePoint Services 3.0 Visual Studio Extensions installed (download the latest from the Microsoft site).

Methods for Building Web Parts

There are two methods for building SharePoint web parts based on what you need them to do. The first method is to directly add a web part to a specific site. This is useful for one-offs, where it is not likely that the part will be used in other sites. The other (and preferred) method is to deploy it as a Solution, which is much easier. Fortunately, the automated process in Visual Studio deploys in development, but leaves an install package that can be reused for deployment. During development, you will use all of these methods.

The simple one-off deployment method requires the least amount of steps. The best aspect about using this particular method is that it is quick; most often installing does not require an IISReset and debugging is simply a matter of recompiling the class.

The steps are

1. Create a new Web Part Project in Visual Studio.
2. Add references and modify the class to inherit from the WebPart class.
3. Set the Debug/Release destination for the DLL as the Site Bin directory.
4. Add code to the class and compile.
5. Modify the site web.config file to add a Safe Control entry.
6. Add the web part through Site Administration and then test the part.

The Solution method is the preferred (or long-term) way to build a reusable package that can be easily deployed to production or a farm.

From start to finish, the steps are

1. Create a new Web Part Project in Visual Studio (using WSS Extensions).
2. Add code to the class and then compile.
3. Define/Add Resources.

4. Sign the assembly.*
5. Set the SharePoint Solution parameters.**
6. Create a webpart file.
7. Create an Element Manifest file (referencing the webpart file and assembly).
8. Create a Feature file (referencing the Element Manifest).
9. Create a Manifest file (referencing the Feature and Assembly).
10. Create a DDF file (MAKECAB instructions and list of files to include).
11. Use MAKECAB.exe to create a WSP file.***
12. Use STSADM to add, deploy, and activate the solution.
13. Add the part to a page in the site.

Building a One-off Web Part

Building a one-off web part is a straightforward process and a quick method to get up and running. This is perfect for testing out pieces of functionality or building the foundation of a web part, and it doesn't require the WSS Extensions to be installed. In this initial example, you'll keep it simple and build a part that displays the username and the time they loaded the page.

Step 1: Create the Project

1. Open Visual Studio 2005 and select New | Project.
2. Select either C# or VB.NET projects and select Class. Set the title of the project to be WebPartMethod1, and then click OK to create the project.

Step 2: Add References and Modify the Class

By default, only a few system references are included in a class project. To use the user interface tools for a web part, you need to add a reference in the project to the .NET System Web namespace and another to the Windows SharePoint Services namespace, in order to enable access to the SharePoint Object Model. In the Solution Explorer, right-click on the References folder and then select Add Reference. When the pop-up window appears, make sure the .NET tab is selected and then use the scroll bar to locate System.Web. Click to select it. Scroll down to the end of the list, hold down the CTRL key, and click Windows SharePoint Services as well. To use the namespaces (objects) in the code, they must be included in the class with using (or Imports) statements. In addition, for web parts, the SharePoint WebControls namespace is also needed, and because of this, I have added System.Web.UI.WebControls.WebParts (the namespace for .NET web parts as shown next). Rename the

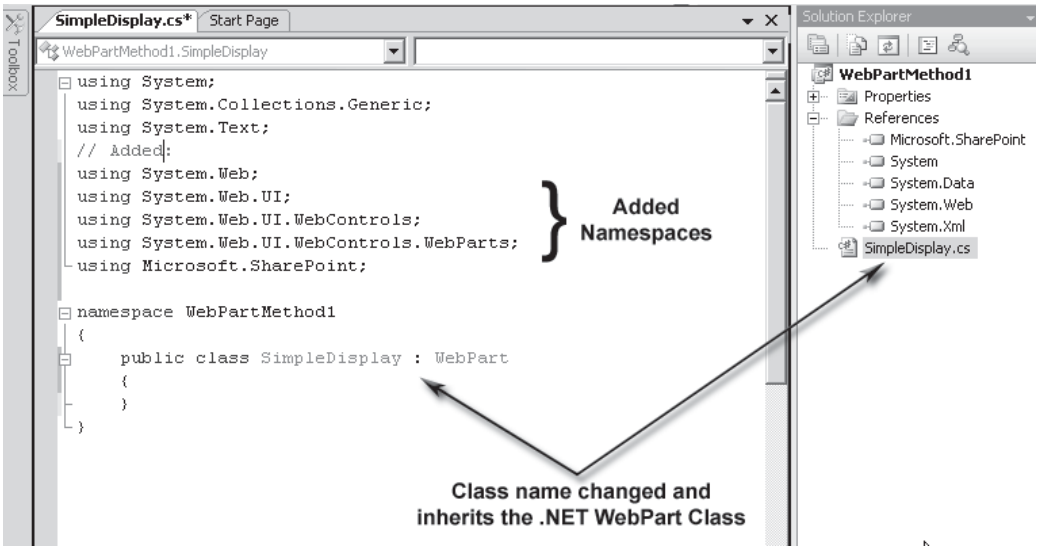
---

*** Web part assemblies, Event Receivers, and any other extensions to SharePoint should be signed (strongly named); a default key is set by Visual Studio, which may need to be changed.**

**** Setting the SharePoint Solution parameters does not create the necessary files for deployment until actually deployed; if setting up the files manually, this step can be skipped.**

***** The MAKECAB.exe utility is installed with the .NET Framework SDK or it can be downloaded from the Microsoft website.**

Class file (Class1.cs) and the Class Name (Class1) in the code to be **SimpleDisplay** and change the Class Declaration to inherit from the WebPart class:



NOTE *Be sure to inherit from the .NET WebPart and not the SharePoint WebPart (the SharePoint one was for version 2 and is no longer used).*

Step 3: Set the Debug/Release Destination

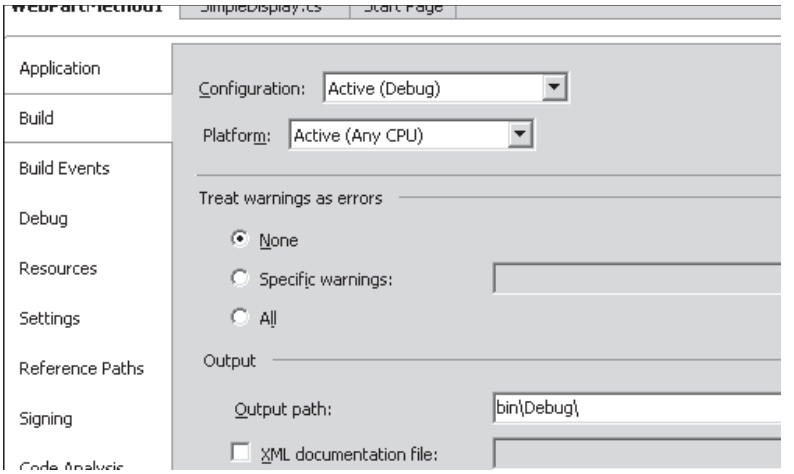
For SharePoint to use assemblies (DLLs and so on), it must be in a place where SharePoint can access them. In most cases, the usual destination for them is in the Global Assembly Cache (GAC), which makes them accessible from any site on the server. However, in a one-off environment, you can use the bin directory of the actual site. Sites that have been provisioned through SharePoint include two directories, one called bin and the other _app_bin; the bin directory should be used for web parts and otherwise external assemblies, whereas the _app_bin directory should be reserved for SharePoint and Custom User Controls. Note that the Application Pool account(s) must have file system execute privilege on these folders (verify this in IIS Manager).

If the SharePoint site was installed on the IIS default website, then the default physical location of the sites directory is c:\inetpub\wwwroot. If SharePoint provisioned the site, the physical directory is c:\inetpub\wwwroot\wss\VirtualDirectories\<Port number of the site>. Note that you may not have used the c:\ drive. In this example, you are using a site created at port 14584 and the default file locations for IIS. For this site, the full path to the bin directory is c:\inetpub\wwwroot\wss\VirtualDirectories\14584\bin.

To deploy the DLL directly to the site on a build, do the following:

1. In the Solution Explorer, right-click on the project name and select Properties.
2. On the Properties page, click the Build tab.

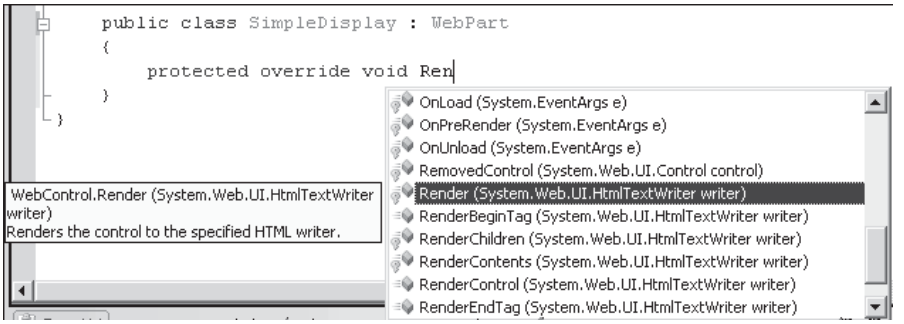
3. Locate the Output line as shown:



4. You can either enter the correct directory path or click the Browse… button to locate it; in this example, the directory is c:\inetpub\wwwroot\wss\VirtualDirectories\14584\bin.
5. Save your changes by clicking the Save icon or by simply pressing CTRL-S, and then closing the properties page.

Step 4: Add Code to the Class and Compile

At this stage, you have a part, but it doesn't do anything. While there are several methods within a web part, you'll keep this one simple so you are going to override only one method called Render, which is the method that outputs to the actual page. To do this, you will override this method by entering in some new code in the class. Visual Studio 2005 provides type-ahead; as you type in the code, it will fill in for you as shown:



Double-click the Render method, and it will create the method code for you:

```
protected override void Render(HtmlTextWriter writer)
{
base.Render(writer);
}
```

You must keep the base.Render(writer) line and add code after it; this ensures that the base class method is executed before your code. As a technical note, there are three methods that are used for rendering a web part: Render (as you saw previously), RenderContents, and RenderControl. There is a bit of confusion around which of these to use, but I'm hoping my description will help:

- **Render** This should be used for all multicontrol web parts (and for most, this is the only one they will use). Examples include a data grid data display, a data entry form, and so on.
- **RenderControl** This should be used for a single control or a composite control; examples include a graphical display, simple data display, and so on.
- **RenderContents** This is preferable when rendering a single control; examples include a custom button, simple data display (without child controls), and so on.

With the Render method defined, you can now add code to perform some work. In this example, you want to display the current username and the time the user viewed the page. Getting the user name via the .NET namespace and the time from the System.DateTime namespace, the Render method provides the HTML Writer (writer), which writes to the page. All of this requires just one line of code (added just after the base.Render line):

```
writer.Write("Current User is: " +
       this.Context.User.Identity.Name +
       " Viewed at: " + System.DateTime.Now);
```

To build the project, select Build | Build WebPartMethod1. Assuming there are no errors, you should be able to navigate to the bin directory and find the WebPartMethod1.dll.

Step 5: Add SafeControl Entry to the Web.config File

Before SharePoint can use an assembly, it must be listed as a SafeControl within the site web.config file. These entries identify safe assemblies that can run within SharePoint and include the assembly name, the namespace, class name, and so on. It also indicates the Public Key Token (for strongly named assemblies, which you will use in the next example). Any assembly not listed will be blocked by SharePoint security. For this example, you won't sign the assembly, but a SafeControl entry is still needed; these are found in XML format with the following attributes:

- **Assembly** This is the name (without the extension) or the fully qualified name (including the PublicKeyToken).
- **Namespace** This is the namespace for the assembly.
- **TypeName** This is the class name or * meaning all classes in this assembly.
- **Safe** The options for this are True or False; this should always be set to True.

For this simple non-signed assembly, the Safe Control entry will be (keep this on one line):

```
<SafeControl Assembly="WebPartMethod1"
   Namespace="WebPartMethod1" TypeName="*" Safe="True" />
```

In this example, you used a type of * meaning all classes within this assembly; this could also be set as **TypeName="SmpleDisplay"** to allow just that class. This allows you to have

classes in the assembly not exposed to SharePoint. Adding this entry is only done once (and if you remove the part, you must manually remove the entry as well). To add the SafeControl to the web.config file, follow these steps:

1. Using Visual Studio, use File | Open | File… (or press CTRL-O), navigate to the root directory of the site, and then open the web.config file (in this example, the file would be located in c:\inetpub\wwwroot\wss\VirtualDirectories\14584).
2. Locate the <SafeControls> tag (this is the start of the section).
3. Add the SafeControl entry from previously (below the <SafeControls> tag); be sure you keep the entry on one line. Don't alter any other lines and then save the file (but keep it open).
4. Next, you need to enable the site for debugging and reduce the security in SharePoint from WSS_Minimal to WSS_Medium. In the web.config file, locate this line:

   ```
   <compilation batch="false" debug ="false" />
   ```

   And change it to:

   ```
   <compilation batch="false" debug ="true" />
   ```

5. Next, locate the line (be *sure* you have the correct one!):

   ```
   <trust level="WSS_Minimal" originUrl="" />
   ```

   And change this to:

   ```
   <trust level="WSS_Medium" originUrl="" />
   ```

6. Save the web.config file again and close it.
7. From the OS, select Start | Run, enter **IISReset**, and click OK (this reloads web.config).

NOTE *Modifying the trust level as done previously is not always necessary; when a web part DLL is in the bin directory of the site, it is actually given partial trust, which is sufficient for what this part does. However, when you begin using the SharePoint Object Model, it must be raised as shown previously or even to Full. Other alternatives are to use GAC, which gives a part Full privilege but must be strongly named, or to create a trust policy for the part (preferred for production installs).*

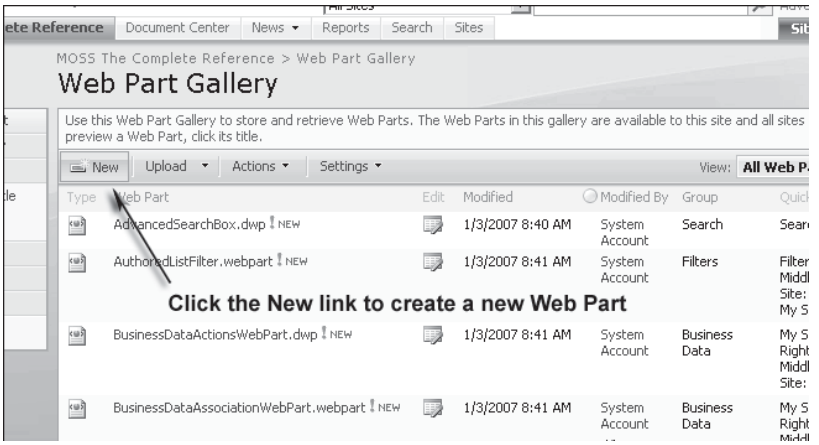
Step 6: Add the Web Part to SharePoint and Test

Although adding the assembly and SafeControl to the website tells SharePoint an assembly is okay to run, the Web Part Definition itself must also be added to the site. This is done with the .webpart file, which tells SharePoint about the part, the assembly that goes with it, and includes values for the properties like name, description, and so on. Creating this file can be done using the Web Part Gallery as you'll do here or simply by hand editing (as shown in the next section).

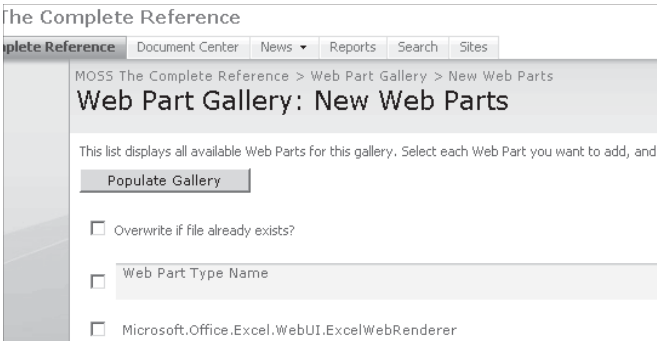
NOTE *Creating the file using the Gallery is a quick way to create a webpart file to be used for packaging in solutions, as covered in the next example.*

To add this part to the site, you must do the following:

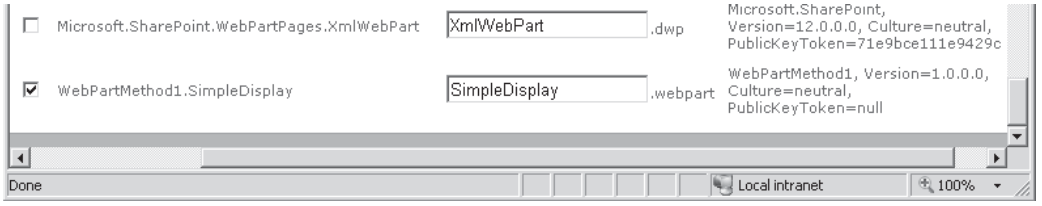
1. Log in to the site and then select Site Actions | Site Settings (or Site Actions | Site Settings | Modify all Site Settings in MOSS).
2. Under the Gallery group, locate and click the Web Parts link.
3. When the Web Part Gallery page opens, you'll see the list of web parts that are already loaded, which includes both DWP files (holdovers from the previous version) and webpart files. Click the New link to create a new Web Part Definition File:



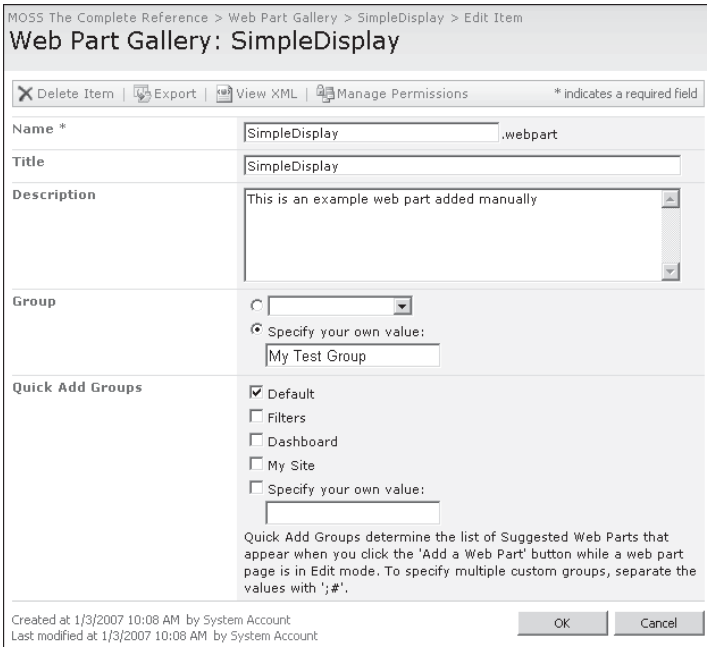
4. SharePoint will read the web.config to find all of the SafeControl entries and then display them on the New Web Parts Page (this list should include the new one created previously):



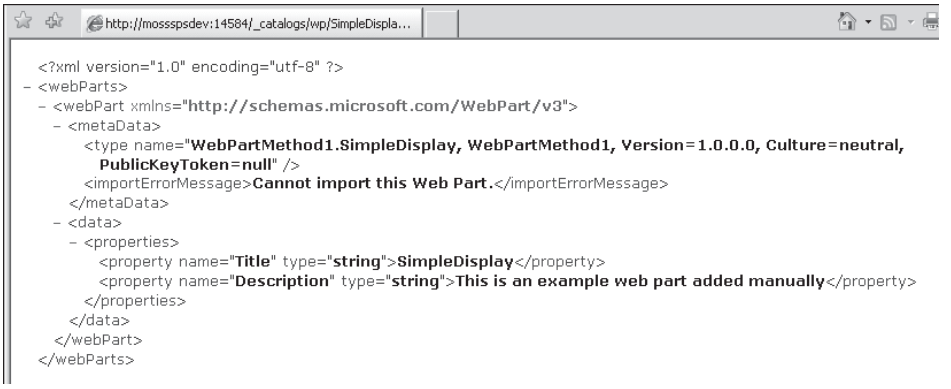
5. Since the web part in this example is WebPartMethod1, you'll have to scroll down to the bottom of this list and check the box next to it to select it:



6. Scroll back to the top of the New Web Parts Page and click the Populate Gallery button; this will then load the web part into the SharePoint Gallery. When the Gallery page is redisplayed, you can scroll down to locate the new webpart file. When first created, only the basic information is added, so you would usually update the properties. Clicking the Edit icon next to the web part name will open the editing page (I've added a description and specified that I want a new group called My Test Group, which is used to group parts displayed in the Add Web Part dialog. I've also set the Quick Add Group to default:



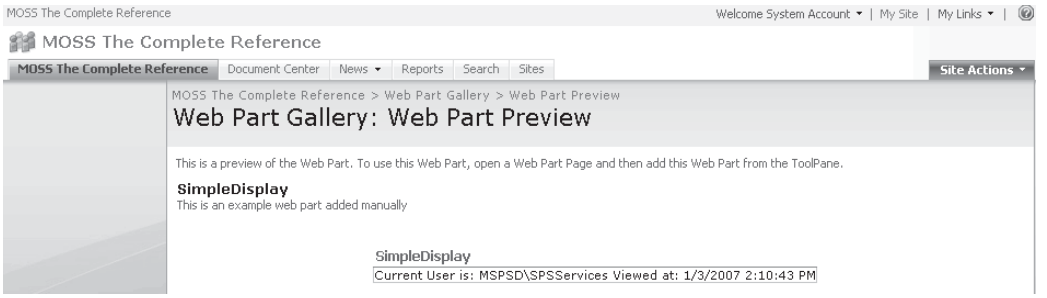
7. Notice that from this page, you can also view the direct XML. If you click on the View XML link, the XML file will be displayed:



Click the Back button on your browser to return to the item; next to the View XML link, you will see the Export link. This will allow you to save the file to disk (as mentioned, this is useful when creating a Solution package).

8. Make the changes and then click OK to save the item.

When you return to the Web Part Gallery Page, you can now test the part. You can either add it to a page or test it in the Gallery itself (note that testing in the Gallery is only good for simple parts that don't need settings or connections). Locate the web part in the Gallery List and then click the name. This will open up the Web Part Preview Page, which looks something like this:



Debugging the Simple Web Part

There are two ways in which to debug web parts once they've been installed. Because our simple example builds the DLL directly into the bin directory of the site, you can use both methods. In both cases, however, the website must be configured to debug (see step 5).

Using the URL The first is to simply modify your Debug settings in the project so as to set a starting URL. This way, when you press the F5 key in Visual Studio 2005 to begin debugging, the DLL will be recompiled and then the browser will be loaded automatically.

To debug in this manner:

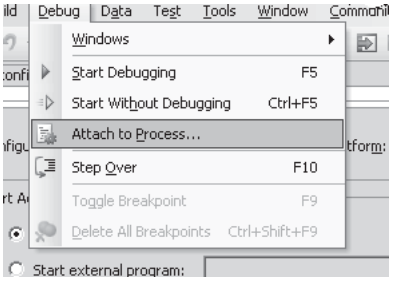
1. In Visual Studio 2005 in the Solution Explorer, right-click on the project name and select Properties.
2. From the Properties page, click the Debug tab.
3. Under the Start Action section, click the radio button next to Start Browser with URL.
4. Enter the URL to the SharePoint site where the part is/will be installed.
5. Click the Save icon to save the file.
6. Press the F5 key to begin debugging.

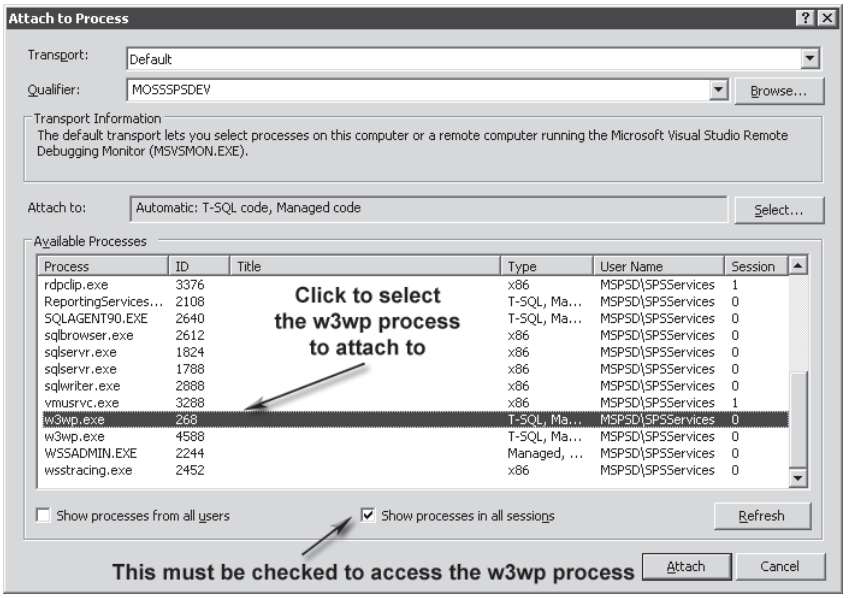
NOTE *If the web part requires installation in the GAC, this method will not work; F5 will rebuild and create a new version of the DLL that won't match one in the GAC.*

Using the Process Debugging by attaching to the web process is the most common method for testing web parts that require installation in the GAC or are non-user facing (such as Event Receivers). Also, this is a good method for one-off installations where the URL method shown previously doesn't work (which happens, I'm sorry to say).

To use this method, do the following:

1. Open the SharePoint site where the part is installed in a browser.
2. In the Visual Studio project, from the top menu, select Debug | Attach to Process.
3. When the Attach To Process window appears, make sure that the Show Processes in all Sessions is checked, and then scroll down the list to locate the w3wp process as shown:





4. Click to select the process name and then click the Attach button. With IIS 6, there is a catch. You may have noticed that there are two (and maybe more) w3wp processes running; one for every application pool. The one to attach to is not obvious, so you can either attach to all or use trial and error. When you are attached to the right process, break points in your code will appear as a solid ball—an empty circle means you are not quite there. In that case, stop debugging and try attaching to a different process.

Building a Web Part Solution

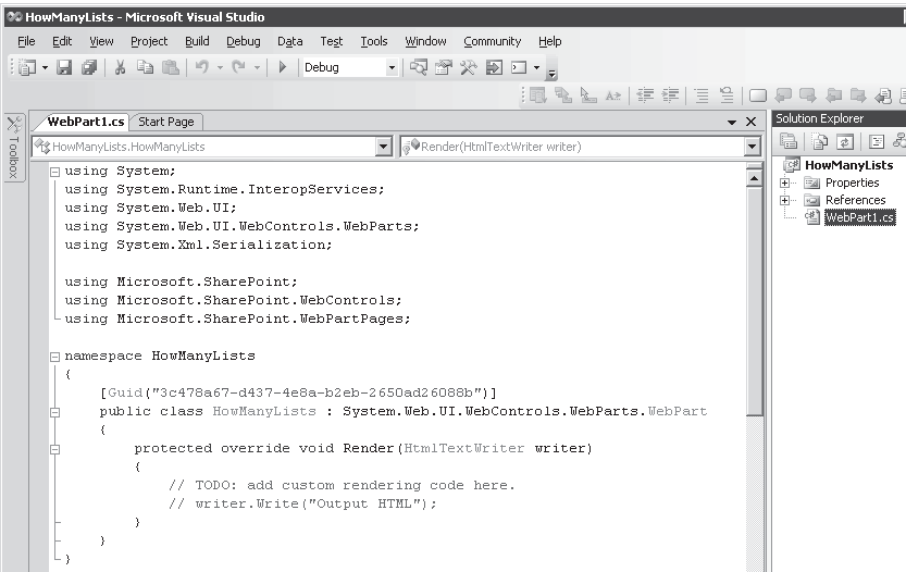
In Visual Studio 2005 (with the Windows SharePoint Services Extensions installed), all tasks save creating the project and coding are handled for you for in the deployment (though not in the removing). With some slight modifications, it also provides you with a Solution package containing a few minor changes that can be used for server deployment. However, knowing

how to build things from scratch is necessary; even with the automation, some manual editing is required and deployment to production servers isn't automated in any case.

To demonstrate the complete process, you're going to create a web part solution from scratch and do all of the steps to deploy it. Then you'll continue with the ever-important process of removing it. After that, you'll take a crack at using the automated process and then another at streamlining the compilation process for deployment. Also note that while this will be a single part, a Web Part Solution and a single .webpart file can contain several web parts. Multiple parts in a single solution is typical for related parts and is one of the methods used for building connectable parts.

Step 1: Create a New Web Part Project

Open Visual Studio 2005 and select New | Project, click + next to C# (or VB), and then click SharePoint. In the right-hand pane, select Web Part. Name the project **HowManyLists** (notice that the Solution name is set automatically), make sure the folder location is correct, leave Create Directory for Solution checked, and then click OK to create the project. A default class file will be created opened for editing as shown.



Notice that the part includes a Global Unique Identifier (GUID) automatically. This becomes the Feature GUID that is used when constructing the Feature.XML file for SharePoint.

Step 2: Add Code to the Class

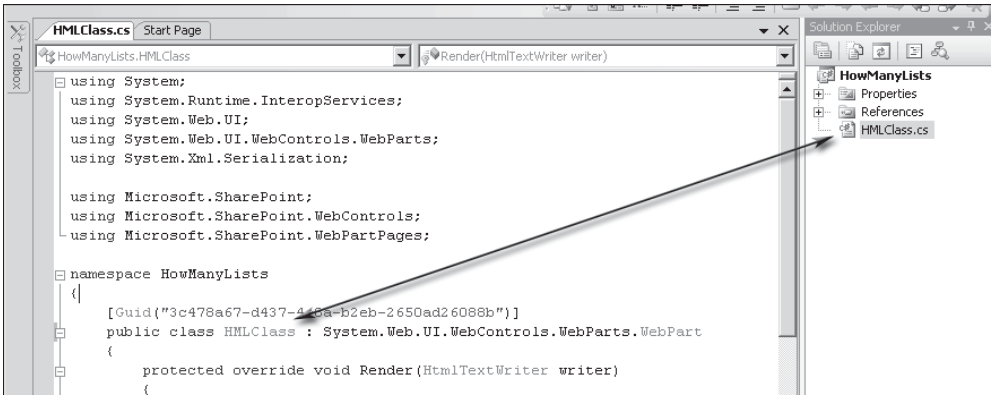
Okay, I'm sure you are just as sick of the Hello World part as I am, so instead you're going to do some actual work here. In addition, I'm going to include the main methods used for development of real parts (not just the Render) and demonstrate how to manage Postback. The end goal of this example is to create a new web part called HowManyLists; the simple

function will be that when a button is pressed, it will determine how many lists are in the site and display the count. To make this complete, you'll track the times the part has been loaded and handle the Postback, and you'll also add a custom property and set it using the .webpart file. Suffice it to say that this one part demonstrates all of the following:

- Order in which methods execute
- Using of custom properties and interaction with the web part properties
- Using of embedded resources (setting the display title)
- Using of ViewState (storing hits)
- Handling Postback events (responding to a button click)
- Error handling (trapping expected errors)

To start building the part, do the following:

1. First, rename the class in the code. Change WebPart1 to **HMLClas**s.
2. Right-click on the WebPart1.cs file in the Solution Explorer window and then select Rename. Change the name to be **HMLClass.cs**; when you're done, your project should look similar to this:



3. Enter in the following code around the existing code you already created previously and remember to leave the GUID assigned by your project as is (don't use the one shown). Note comments, of course, are optional:

```
using System;
using System;
using System.Runtime.InteropServices;
using System.Web.UI;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Serialization;
using Microsoft.SharePoint;
using Microsoft.SharePoint.WebControls;
using Microsoft.SharePoint.WebPartPages;
using System.Web.UI.WebControls; // Added for controls
using System.Drawing;            // Added for color, add a reference too
using System.Text;               // Added for StringBuilder
using System.Resources;          // Added for Resources
```

```
namespace HowManyLists
{
    [Guid("3c478a67-d437-4e8a-b2eb-2650ad26088b")]
    public class HMLClass : System.Web.UI.WebControls.WebParts.WebPart
    {
        // Save for times checked:
        public int HitCount;
        public string OurDisplayTitle = "Get List Count";
        //====================================================================
        // Define Child Controls
        protected Label DisplayCount = new Label();
        protected Button CalcListCount = new Button();
        //====================================================================
        // PROPERTY: ShowHitCount
        // This is a settable flag that turns the Hit Count display
        // On or Off (Checked = True or On). Note that you set the default
        // as 'true' (you will override this in the webpart file).
        // Also notice the XML settings - these set the web part property
        // category to "Custom Settings" (this is what displays when
        // the web part properties are displayed), sets the storage to be
        // Shared (setting applies to all users), sets personalization to
        // be shared and is browsable (visible) in the web part properties.
        private const bool defaultShowHitCount = false;
        private bool showHitCount = defaultShowHitCount;
        [SPWebCategoryName("Custom Settings"),
          WebPartStorage(Storage.Shared),
          WebBrowsable(true),
          Personalizable(PersonalizationScope.Shared),
          WebDescription("Sets true/false to show hit count"),
         WebDisplayName("Check to show hit count")]
        public bool ShowHitCount
        {
            get { return showHitCount; }
            set { showHitCount = value; }
        }
        //====================================================================
        // METHODS:
        //====================================================================
        // The following are the various methods as executed in the order
        // shown (this is EXACTLY the same as a Web Control):
        //====================================================================
        // ONINIT EVENT:
        // Fires 1st: Startup of the Control and where in web parts, the
        // storage system is loaded; also calls LoadViewState (loads the
        // view state for all controls); if you uncomment this event, you
        // MUST include the base.OnInit(e) line. This is the best
        // place to define controls that are added to the part.
        //
        protected override void OnInit(System.EventArgs e)
        {
            base.OnInit(e); // You must include this if you override
            //
            // Create the label:
            //
            DisplayCount.ID = "DisplayCount";
            DisplayCount.Text = "";
            DisplayCount.ForeColor = System.Drawing.Color.Blue;
```

```
    DisplayCount.Font.Bold = true;
    // VERY Important: Be sure to add the control:
    Controls.Add(DisplayCount);
    //
    // Create the button:
    //
    CalcListCount.Click += new EventHandler(CalcListCount_Click);
    CalcListCount.ID = "CalcListCount";
    CalcListCount.Text = "Click Here";
    CalcListCount.Font.Size = FontUnit.XXSmall;
    CalcListCount.Font.Bold = true;
    CalcListCount.CausesValidation = true;
    Controls.Add(CalcListCount);
    //
}
//
//=====================================================================
// ONLOAD EVENT:
// Fires 2nd: OnLoad event - this is where startup functionality/
// processing happens, including trapping PostBack - AFTER THE
// OnLoad events, any User Generated events (Click, and so on) are fired
protected override void OnLoad(System.EventArgs e)
{
    base.OnLoad(e);
    // This is the area to establish DB connections & data retrieval
    if (Page.IsPostBack)
    {
        // PostBack processing:
        try
        {
            string HCountString = (string)this.ViewState["HITCOUNT"];
            if (HCountString != "" || HCountString == null)
            {
                HitCount = Convert.ToInt32(HCountString);
                HitCount++;
                this.ViewState["HITCOUNT"] = HitCount.ToString();
            }
        }
        catch
        {
            this.ViewState["HITCOUNT"] = "0";
            HitCount = 0;
        }
        // Save it either way:
        this.SaveViewState();
    }
}
//
//=====================================================================
// CREATECHILDCONTROLS EVENT:
// Fires 3rd (BEFORE any events and OnPreRender)- this creates any
// controls; you can use this event to do any checking or resetting
// of controls. You can alternately define controls here (instead
// of OnInit) though this can affect how well you can maintain
// view state.
//
```

```
protected override void CreateChildControls(System.EventArgs e)
{
    base.CreateChildControls(e); // You must include this if you override
}       //
//=====================================================================
// EVENT HANDLERS:
// Fires 4th: Event Handlers - event handlers fire just AFTER
// CreateChildControls and before OnPreRender:
void CalcListCount_Click(object sender, EventArgs e)
{
    //
    // On click, you need to calculate the current number of lists
    // in the current Site:
    //
    // Get a ref to the current site:
    //
    SPWeb CurrentSite = SPControl.GetContextWeb(Context);
    SPListCollection SPLC = CurrentSite.Lists;
    //
    // Show the value:
    //
    DisplayCount.Text = "Found: " + SPLC.Count.ToString() + " lists";
    //
}
//
//=====================================================================
// ONPRERENDER EVENT:
// Fires 5th: OnPreRender event - called just before anything is
// drawn via the Render event (here is where Properties can be
// changed before the controls are drawn); this event calls
// SaveViewState to save the view state before rendering
//
//protected override void OnPreRender(System.EventArgs e)
//{
//        base.OnPreRender(e);
//}
//=====================================================================
// RENDER EVENT:
// Fires 6th (last): Render event - Render this control to the
// output (HTML Writer) parameter specified
// <param name="output">The HTML writer to write out to</param>
protected override void Render(HtmlTextWriter writer)
{
    // Make sure Child Controls have been setup:
    EnsureChildControls();
    //-----------------------------------------------------------
    // Best practices tip: When writing out, ALWAYS use a table;
    // this prevents getting out of whack when placed on the page
    //-----------------------------------------------------------
    // Get the title from the Resources:
    //
    string TheButtonText = "Get List Count";
    //
    ResourceManager WPResourceManager =
        new ResourceManager("HowManyLists.HMLResources",
        GetType().Assembly);
    //
```

```
try
{
    string TestButtonText =
          WPResourceManager.GetString("ButtonText");
    if (TestButtonText.Length > 0)
    {
        TheButtonText = TestButtonText;
    }
}
catch (System.Resources.MissingManifestResourceException
        ErrInName)
{
    // Means a problem with the name
    string MMREText = ErrInName.Message.ToString();
}
catch (Exception ErrInResource)
{
    string SomeOtherProbText = ErrInResource.Message.ToString();
}
WPResourceManager.ReleaseAllResources();
// Assign it to the control:
CalcListCount.Text = TheButtonText;
//-------------------------------------------------------------
// You can either use a string to build the display or simply
// write it out as you go - I suggest using a string since
// it will speed up rendering
StringBuilder OSPart1 = new StringBuilder();
StringBuilder OSPart2 = new StringBuilder();
StringBuilder OSPart3 = new StringBuilder();
//
// Build the output strings:
//
// NOTE: HTML attributes should be enclosed in quotes,
// for example:
//      "<table width=\"200\"
// however I have left them out for readability
//
OSPart1.Append("<table width=200 cellspacing=0 border=0>");
OSPart1.Append("<tr width=100% bgcolor=white>");
// Notice that I've used a standard WSS Style:
OSPart1.Append("<td colspan=2 width=100% class='ms-wptitle'>");
OSPart1.Append(OurDisplayTitle);
OSPart1.Append("</td></tr>");
OSPart1.Append("<tr width=100% bgcolor=white>");
OSPart1.Append("<td width=100% align=center class='ms-ls'>");
//
OSPart2.Append("</td></tr>");
try
{
  if (HitCount > 0 && ShowHitCount)
  {
    string DispHits = HitCount.ToString();
    OSPart2.Append("<tr width=100% bgcolor=white>");
    OSPart2.Append("<td width=100% align=center class='ms-ls'>");
    OSPart2.Append("You've checked: " + DispHits + " times");
    OSPart2.Append("</td></tr>");
  }
```

```
            }
            catch { }
            OSPart2.Append("<tr width=100%>");
            OSPart2.Append("<td width=100% align=center valign=center>");
            //
            OSPart3.Append("</td></tr>");
            OSPart3.Append("</table>");
            //
            // Render:
            //
            writer.Write(OSPart1);
            if (DisplayCount.Text == "" || DisplayCount.Text == null)
            {
                writer.Write("<br><br>");
            }
            else
            {
                DisplayCount.RenderControl(writer);
                if (!ShowHitCount) // Fill in space
                {
                    writer.Write("<br><br>");
                }
            }
            writer.Write(OSPart2);
            CalcListCount.RenderControl(writer);
            writer.Write(OSPart3);
        }
    }
}
```

At this point, you should be able to build the project. Make sure you are in Debug Mode and then select Build | Build <project name>, but don't try to run the project at this point. When you are able to build successfully, continue on to the next step.

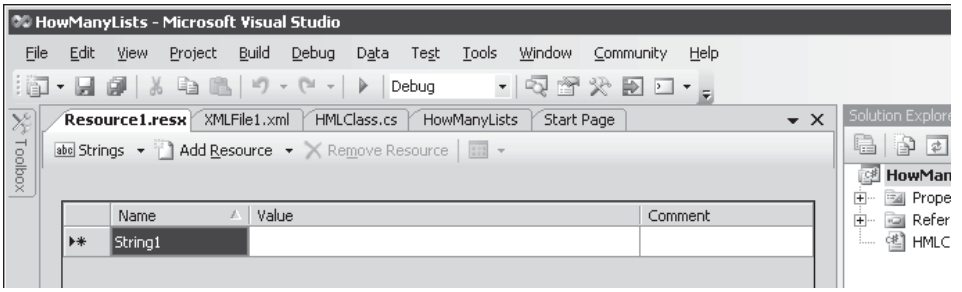
Step 3: Define/Add Resources

Like any .NET application, web parts can use resources such as predefined text, script files, custom CSS files, and images, and can be used for localization (changing language). Combining the web part with these resources is done by creating a Resources file in the web part project. When you compile and build a part, the resources are bundled in with the installation package and deployed with the web part. Resources defined by a Resources (resx) file can be used within your code and even in webpart files. In this next example, you are going to create a single string resource that sets the text on the button. This is simplistic for sure, but is the same method you would use if you were checking for language localization. This resource will be stored in the project (the resx file), and when the project is compiled, will be automatically included in the assembly (DLL).

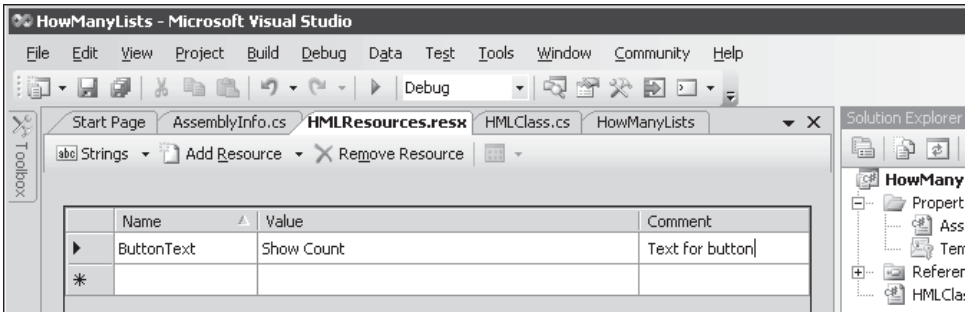
To add resources, you have to start with adding a Resource file to the project. To do this, you'll need to do the following:

1. In Visual Studio, select the top menu's File | New | File to create a file. Note that when WSS Extensions for VS are installed, you cannot add a non-SharePoint item using the Add Item method in Solution Explorer. You must use the File menu to create a new file.

2. This will open the New File dialog box. In the left pane, click on Web to select it, and then click to select Resource File.
3. Click OK to create the Resource file. This will display the Resource file with the first entry as shown:



4. You are going to add a string resource, which you will use for the button text. Replace the default name (string) with **ButtonText**, enter a value of Show Count, and then enter in the comments. It should look something like this:



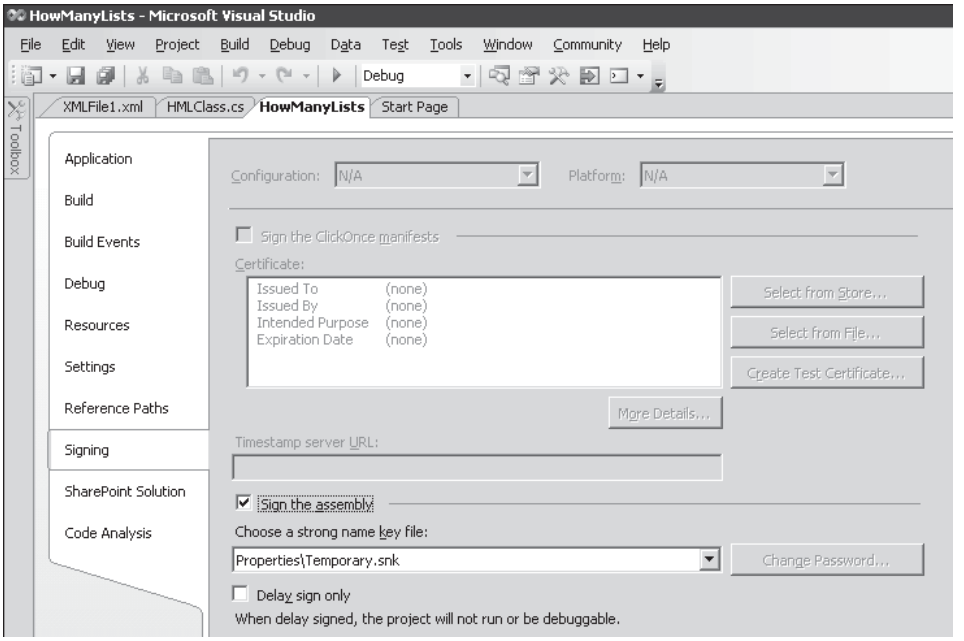
5. Now click the Save icon to save the file. When the Save File dialog appears, make sure you save the file within your project (save it one folder below where the VS Solution (sln) file is and change the name from Resource1.resx to **HMLResources.resx**).
6. In Visual Studio, in the Solution Explorer, look for the new HMLResources.resx file; if it is not included in the project, right-click on it and select Properties.
7. In the Properties window, set the Build Action to be Embedded Resource (if it isn't already), and then click the Save All icon to save all changes.

When you have completed the previous steps, you should have a saved Resource file included in your project; you will use this when building the webpart file later on.

Step 4: Sign the Assembly

Visual Studio 2005 automatically creates a temporary strong name key (snk) file and assigns it to the project. While this is fine in a development environment, you'll likely generate your own or your organization will have a standard one for use in signing applications. Since the temporary.snk file is created by default, you'll have to change this to point to the desired file. To change the SNK file:

1. In the Solution Explorer window, right-click on the project name and select Properties.
2. On the Properties page, click the Signing tab. This will look something like this:



3. Click the Browse button, navigate to your key file, click OK, and then select File | Save All or click the Save All icon to save the property changes.
4. To set the new key, recompile the project by selecting Build | Build HowManyLists or pressing CTRL-SHIFT-B (or F6, depending on your IDE settings in Visual Studio).

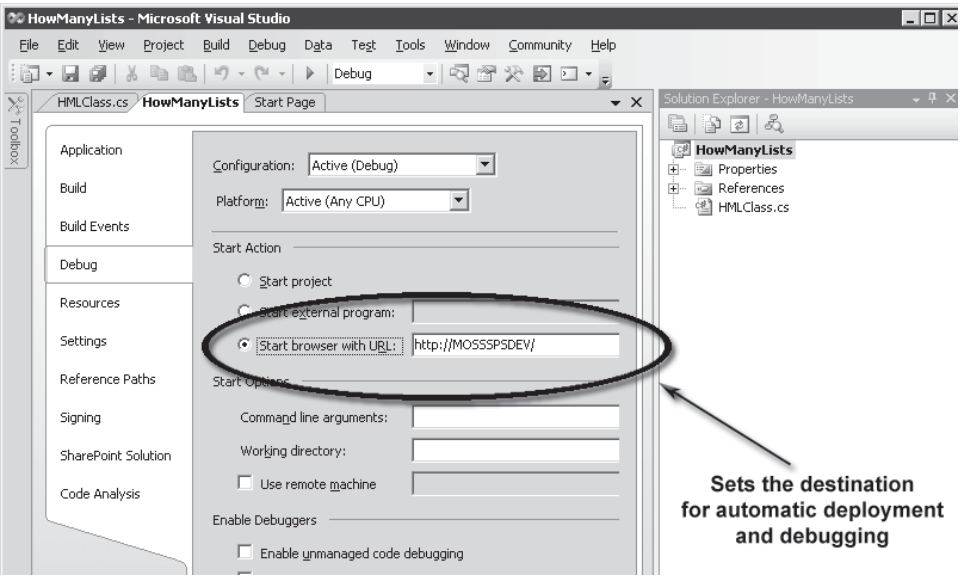
Step 5: Set the SharePoint Solution Parameters

When Windows SharePoint Services Extensions for Visual Studio are installed, it adds a new Property tab to the VS project properties. This tab, called SharePoint Solution, allows entry of the basic web part properties, including the Solution information, description, and so on. When these are set and saved, it creates a solution.XML file in the VS Project. This is later used for automated deployment directly from Visual Studio. If you are building this process manually, these settings have no effect; however, you will need them to test the automated process later.

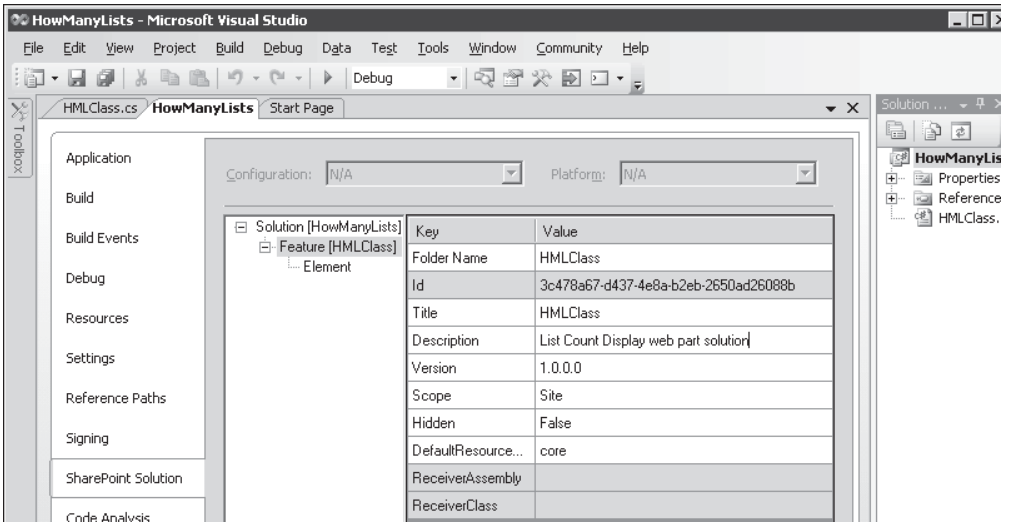
In specific, the values entered (or preset) in the SharePoint Solution settings will be used to create the Manifest, Feature, Element Manifest, and the DDF file needed for creating the WSP file as well as for the creation of a batch file to use for deployment to SharePoint directly. Note that to directly deploy to a SharePoint site, however, in addition to the settings, you must also set the Debug setting to select the site in which you wish to deploy. To set these, do the following:

1. In the project Solution Explorer, right-click on the project name and select Properties.

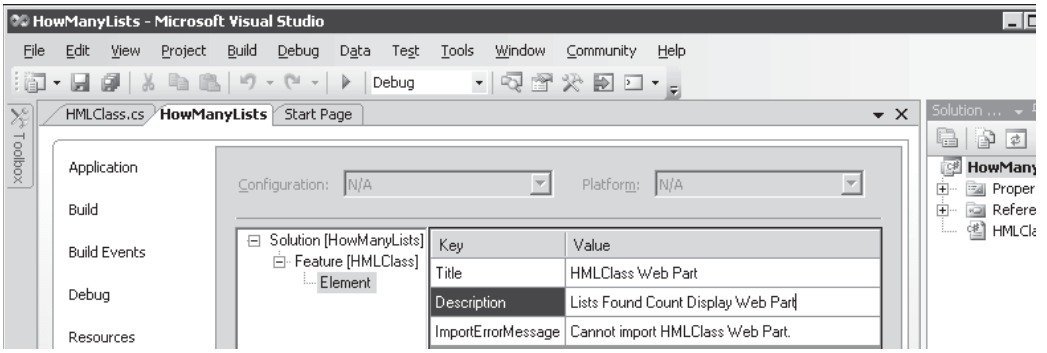
2. On the properties window, click the Debug tab.
3. In the Start browser with URL field, enter the URL to the SharePoint site as shown:



4. Next, click the SharePoint Solution tab.
5. In the left-hand pane, click + next to Solution to expand it and then click Feature; the feature properties are displayed in the right-hand pane.
6. Enter the solution description. (This description is added to the Features.XML file by VS and appears under the Feature Title in the Site Collection Features Administration Page.)



7. Click + next to Feature to expose element.
8. In the left-hand pane, click Element; the Element Properties are displayed on the right.



9. Enter in the element (or class) description:
10. Click OK, and then select File | Save All or click the Save All icon to save the changes.

NOTE *There is a bug in WSS extensions in that the description you enter for the element is not saved; you can add it manually by editing the solution.XML file.*

Step 6: Create a Webpart File

The webpart file is an XML file that SharePoint uses to determine the assembly it is related to and the properties of the web part, including its title and description. This file is the metadata and properties definition of the web part, much as the item information for a document library would describe the document itself. Primary sections include the Metadata Section, which defines the assembly, the Default Import Error Message (displayed when the assembly cannot be loaded in SharePoint), and the Data Section, which defines all of the properties for the web part, both the default items (such as title), and any custom properties that have been defined.

Creating a webpart file is done by simply creating a new XML file and then adding the XML statements as needed. However, you will need to have some information before you start:

1. You must decide whether you will be setting any of the default properties that will change the way the part operates, and if these will be done through the webpart file or set by the code.
2. You must decide if you will be presetting any custom properties through the webpart file.
3. You must get the assembly information for the web part.

Note that throughout this example, I use some random web part names for demonstration purposes. When you create the actual webpart file in Part 4 of step 6 later in the chapter, you will use the correct names associated with the example part. Also note that after you have deployed a solution, you can export, modify, and re-upload the webpart file without deactivating the solution.

The complete outline of the XML in a .webpart file is as follows (I've added comments):

```
<webParts>
<!-- Defines a single web part -->
<webPart>
<metaData>
      <!-- Assembly (DLL) Info -->
<type name="Assembly information…" />
<!-- What to show users when there's an error -->
<importErrorMessage>Error Message</importErrorMessage>
</metadata>
<!-- Begin of Web Part Data (Property definitions) -->
<data>
<!-- Custom Properties -->
<properties>
<!-- Sensitive Properties (see ExportMode below) -->
<ipersonalizable>
<property name="" type=""></property>
</ipersonalizable>
<!-- Non-sensitive Properties -->
<property name="" type=""></property>
</properties>
</data>
</webPart>
</webParts>
```

The sections within this file define the information needed by SharePoint to both enable the web part to be added to the Web Part Gallery and to have it used on a Web Part Page. For most web parts, setting all of the various properties is unnecessary since the default settings are usually sufficient. In fact, for most cases, you will only need to include the very basic sections and define custom properties you wish to set default values for. The purpose for each of these sections is as follows:

- **WebParts** This is an outer node that contains one or more Web Part Definitions.
- **WebPart** This is a node to identify a web part.
- **MetaData** This is a node that contains the Type element, which identifies the assembly (DLL) and the importErrorMessage element (text to display if the part cannot be loaded).
- **Data** This is an outer node that contains one or more Property elements.
- **Properties** This is an outer node that contains Properties.
- **Ipersonalizable** Used in properties, this node identifies properties that are considered sensitive and are not exported if the Web Part Export Setting is set to NonSensitiveData (see ExportMode later on).
- **Property** This defines either a custom or generic property name, type, and value.

While the structure is there, not all of these sections are needed. In fact, only assembly information is absolutely required for a single web part, though you should always specify the title and description (this is what is shown to users on the web part page).

Step 6: Part 1 Setting Default Properties

All web parts within SharePoint have a default set of properties that govern how the part is displayed (or not) on a page. This includes things like the title, the description, and operational features, such as whether the part can be moved between zones, and so on, and whether they are available from the UI when a user selects Edit | Modify Shared Web Part when editing a page. Pre-setting other properties (like height, width, and so on) is optional since all have default values.

Conveniently, you have two options for setting these properties: through the webpart file and/or through the object model in the code (custom properties are set in a similar way). The important distinction is that if set in the webpart file, it is considered a preset and can later be changed using the UI. If set in the code, properties are overridden and ignore UI changes. While many properties can be set, most will not use the extent of them, so I've listed only the most common and what they are used for. You can find a complete definition of the webpart file on the MDSN site, http://msdn2.microsoft.com/en-us/library/ms227561.aspx.

To save you a whole lot of time, you can find all of the properties listed next defined under the System.Web.UI.WebControls.WebParts namespace; enumerations referenced here (like PartChomeType) can be found under the System.Web.UI.WebControls. To use them in your class, you must add a project reference to them and include them with Imports or using.

Property Name: Title

- **What it does** This represents the title of the web part and is shown in the Web Part Selection window. If shown, this is also the title of the web part when placed on a page.
- **Possible values** Any text
- **Object model example**

```
this.Title = "Title Of the Web Part";
```

- **Webpart file example**

```
<property name="Title" type="string">Title Of the Web Part</property>
```

Property Name: Description

- **What it does** The description is shown along with the title in the User Interface; this should be fairly descriptive but not more than 100 or so characters.
- **Possible values** Any text
- **Object model example**

```
this.Description = "A detailed description";
```

- **Webpart file example**

```
<property name="Description" type="string">A detailed description</property>
```

Property Name: Height and Width

- **What it does** This gets or sets the web part display height or width on the page and overrides the actual display of the part itself.
- **Possible values** The default for the web part height and width is to fit within the zone where the web part is placed. If this value is set, this value is a numeric value

plus a two-character abbreviation that indicates a unit of measure set to cm (centimeters), in (inches), mm (millimeters), pc (picas), pt (points), or px (pixels). For example, 500px meaning 500 pixel. Note that if the unit is omitted, px is the default.

- **Object model example**

```
this.Height = "500px";
this.Width = "300px";
```

- **Webpart file example**

```
<property name="Height" type="unit">500px</property>
<property name="Width" type="unit">300px</property>
```

Property Name: ChromeType

- **What it does** This gets or sets the type of border that is displayed around the web part in the web part zone.
- **Possible values** Default or TitleOnly shows only web part title; TitleAndBorder shows the title and puts a border around the entire part; BorderOnly shows a border around the part display; and None hides title and shows no border.
- **Object model example**

```
this.ChromeType = PartChromeType.TitleAndBorder;
```

- **Webpart file example**

```
<property name="ChromeType" type="chrometype">TitleAndBorder</property>
```



Property Name: ChromeState

- **What it does** This gets or sets whether the web part is displayed or minimized on the page.
- **Possible values** Minimized is when the part display is minimized (user must use Restore to display) or Normal (the default).
- **Object model example**

```
this.ChromeState = PartChromeState.Normal;
```

- **Webpart file example**

```
<property name="ChomeState" type="chromestate">Normal</property>
```

Property Name: TitleUrl

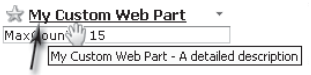
- **What it does** This gets or sets the URL link used when the user clicks on the web part title; for example, the default for SharePoint lists and libraries is the Allitems.aspx page—this link can point to any valid URL.

- **Possible values** This is a full or relative URL; if not specified, the title will be static (no link).
- **Object model example**

```
this.TitleUrl = "/pages/MyWebPartPage.aspx";
```

- **Webpart file example**

```
<property name="TitleUrl" type="string">/pages/MyWebPagePage.aspx</property>
```



TitleUrl sets the link on the web part title; the Title and Description are shown in the Tool Tip

Property Name: HelpUrl

- **What it does** This gets or sets the link used when the user selects the Help option from the Web Part Title drop-down menu.
- **Possible value** A full or relative URL to a page or file that will provide the user help in using the web part; if not specified, the Help option in the Web Part Menu is not displayed.
- **Object model example**

```
this.HelpUrl = "http://mossspsdev/_layouts/help.aspx";
```

- **Webpart file example:**

```
<property name="HelpUrl" type="string">http://mossspsdev/_layouts/
help.aspx</property>
```

Property Name: HelpMode

- **What it does** This sets the way the User Interface will respond when the user clicks on the Help link (defined by HelpUrl talked about previously).
- **Possible values** Modeless, which opens the Help link in a new independent browser window and which the user can leave open or closed; Modal, which opens the Help link in a new independent browser window (but the user must close it before they can return to the original); and Navigate, which opens the Help link in the same browser window, replacing the current (user must use the browser back button to return to the site).
- **Object model example**

```
this.HelpMode = WebPartHelpMode.Modeless;
```

- **Webpart file example**

```
<property name="HelpMode" type="helpmode">Modeless</property>
```

Property Name: TitleIconImageUrl

- **What it does** This gets or sets the URL for the image to display in the title of the web part, when the web part is on the page.

- **Possible values** A full or relative URL to the image file, which can include using images that are stored within Picture Libraries on the site. However, bear in mind that if you specify an image, the image must be accessible (readable) by all site users. If not specified, no image is displayed.
- **Object model example**

```
this.TitleIconImageUrl = "/_layouts/images/ADDTOFAVORITES.GIF"
```

- **Webpart file example**

```
<property name="TitleIconImageUrl"
     type="string">/_layouts/images/ADDTOFAVORITES.GIF</property>
```

NOTE *Keep this on one line. Also, the image I've used previously happens to be in the SharePoint root images folder and shows a single star. I'm using the relative URL that SharePoint defines automatically (you can find this in IIS Manager).*



Property Name: CatalogIconImageUrl

- **What it does** This gets or sets the URL for the image to display in the Web Part Catalog.
- **Possible values** A full or relative URL to the image file is the same as TitleIconImageUrl shown previously; if not specified, the WEBPART.GIF file is used.
- **Object model example**

```
this.CatalogIconImageUrl = "/_layouts/images/WEBPART.GIF";
```

- **Webpart file example**

```
<property name="CatalogIconImageUrl"
    type="string">/_layouts/images/WEBPART.GIF</property>
```



Property Name: Direction

- **What it does** This gets or sets the direction for any text displayed by the web part and is used for localization of the web part display.
- **Possible values** Not Set (the default) means the default display direction will be used; LeftToRight displays text left to right for languages like English; and RightToLeft displays text right to left for languages like Arabic.
- **Object model example**

```
this.Direction = ContentDirection.LeftToRight;
```

- **Webpart file example**

```
<property name="Direction" type="contentdirection">LeftToRight</property>
```

Property Name: ExportMode

- **What it does** This gets or sets whether the end user can export this Web Part Definition and what properties are included.
- **Possible values** All, which means all of the web part properties can be exported; NonSensitiveData, which means only properties set as nonsensitive (not enclosed in <ipersonalizable> tags) can be exported; and None, which means the web part cannot be exported (this is typical for assembly-based web parts, since the webpart definition file is of no use without the assembly being installed).

NOTE *This setting applies to the User Interface only; in the Site Administration Web Part Catalog, the Export Feature exports all properties.*

- **Object model example**

```
this.Direction = WebPartExportMode.All;
```

- **Webpart file example**

```
<property name="ExportMode" type="exportmode">All</property>
```

As a technical note, setting properties as sensitive (at least for default properties) is for special uses in SharePoint. While the properties will be loaded (and visible) when the webpart file is added to the catalog, the properties will revert to their default values when the part is added to a page (often not the desired behavior). There are special cases where this is useful, however. For example, if the Title property was marked as sensitive (surrounded by <ipersonalizable> tags) when you uploaded this to SharePoint the title of the web part would be visible in the Web Part Catalog. However, when you add the part to the page, the title would revert to the default value (blank), and the part title would be displayed as Untitled.

Boolean Properties

- **What they do** There are several simple Boolean properties that also control how the web part will operate. These include the following:

Property Name	If True:	If False:
Hidden (default False)	Part is not displayed.	Part is displayed normally.
AllowConnect (default True)	Part can connect to other parts.	Connections not allowed.
AllowClose (default True)	Part can be closed on a Web Part Page.	Part cannot be closed.
AllowEdit (default True)	Part properties that are set as Personal can be edited by the user.	User cannot modify personal properties.
AllowHide (default True)	Part can be hidden.	Part cannot be hidden.
AllowMinimize (default True)	Part can be minimized in the display.	Part cannot be minimized.
AllowZoneChange (default True)	Part can move between zones on a multi-zone page.	Part cannot be moved from the zone it was added to.

To restore the part back to the page requires more steps than usual. The user must select Edit Page | Add Web Parts, and then from the Add Web Parts dialog box, click the Advanced Web Part Gallery and Options link. Finally, select the Closed Web Parts gallery and then drag the part back to the page.

- **Possible values** True or False (first letter must be capitalized).
- **Object model example** (same format for all)

```
this.Hidden = false;
this.AllowConnect = true;
```

- **Webpart file example**

```
<property name="Hidden" type="bool">False</property>
<property name="AllowConnect" type="bool">True</property>
```

Step 6: Part 2 Setting Custom Properties

For most web parts, you will end up using many custom properties since they enable you to create real-time settings that the user can change without you having to change code. For example, you might want the user to choose their own title or be able to select which list to create a view of. Alernatively, you might want to override a particular feature and so on. As defined in the Web Part Class, the two XML settings—WebBrowsable and Personalizable—determine if and what the user can set. If WebBrowsable is set to True, it is shown in the Web Part Properties and is accessible to the user. If Personalizable is Shared, it can be set once for all users; if Personal, individual users can have their own settings; and if None, the value cannot be set.

To use a custom property, you must first define it in code. For example:

```
//=====================================================================
// PROPERTY: ShowHitCount
private const bool defaultShowHitCount = false;
private bool showHitCount = defaultShowHitCount;
[SPWebCategoryName("Custom Settings"),
 WebPartStorage(Storage.Shared),
 WebBrowsable(true),
 Personalizable(PersonalizationScope.Shared),
 WebDescription("Sets true/false to show hit count"),
 WebDisplayName("Check to show hit count")]
public bool ShowHitCount
{
   get { return showHitCount; }
   set { showHitCount = value; }
}
```

As shown, the default value for this property is false, meaning don't show it. In your code, this is settable, so in theory, once the part was added to the page, you could go back and change it to True. However, that would require manual intervention, so you can use the automatic way by setting this as a Property Element in the webpart file. To override the value, you simply specify a new entry within the <properties></properties> section (this will force the property to be True when added to a page):

```
<property name="ShowHitCount" type="bool">true</property>
```

Though this is a simple example, be aware that overriding a value is hardly the only use for Properties—far from it. Custom properties need not be visible to users, so they can basically be anything (pretty much) that you need to pass from the Web Part Definition to the Web Part Properties. As you get more into them, you'll find they are good for passing statements too, even CAML.

CAUTION *Properties with preset values that have been defined in the sensitive data section (<ipersonalization>) are only loaded into the Web Part Catalog and will have no effect during runtime in SharePoint. The value of the property will be derived at runtime from the default setting in the code or as set by the user in the UI (the webpart file value will be ignored).*

Step 6: Part 3 Getting Assembly Information

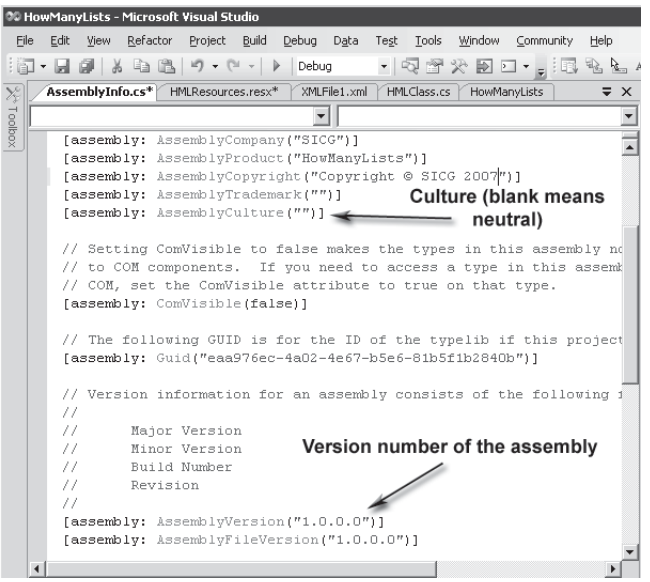
As you saw in the simple example, you had to add a SafeControl entry (by hand) to the web.config file. When using a solution, this is included in the webpart file, and when the solution is installed, SharePoint handles creating the SafeControl entry for you.

However, in this example, you are using a Signed Assembly; this means the SafeControl entry requires a fully qualified assembly definition (including Version, Culture, and PublicKeyToken). The culture and version are available in the AssemblyInfo.cs file created in the project.

To open this file, you should:

1. Open the project Visual Studio 2005 (if it's not open already).
2. In the Solution Explorer, click + next to the Properties folder to open it.

3. Double-click on the AssemblyInfo.cs file to open it for editing. It should look something like this:



You can see the AssemblyCulture is set to nothing, which means neutral (this attribute will only have a value if this were a subassembly used for language-specific resources). The version as shown is 1.0.0.0, which is the default version assigned when a web part is created. This attribute allows you to version web parts (and assemblies in general) so that more than one version can run side by side. Note that web parts don't support mixed versioning (i.e., 1.0.0.*). Lastly, you'll need to get the public key token value that is created using the Strong Name Key. This information is unfortunately not available through Visual Studio, and you must use the strong name utility (SN.exe) to get it. The best way to do this is using the VS command prompt:

1. Select Start | All Programs | Microsoft Visual Studio 2005 | Visual Studio Tools | Visual Studio 2005 Command Prompt.
2. From the command prompt, change your directory to the debug directory in your project.
3. Enter the command **SN -T HowManyLists.dll** (note the *-T* must be capital).
4. Record the 16-character string returned; this is the public key token for the assembly.

Using the previous steps, you can put them all together to create the fully qualified assembly name (though remember, your token will be different):

```
HowManyLists, Version=1.0.0.0, Culture=neutral, PublicKeyToken=9f4da00116c38ec5
```

Step 6: Part 4 Building the Webpart File

The final part of this step is to build the webpart file itself and add both assembly information and desired property settings. Select the top menu, File | New | File, to open the New File dialog box; click General in the left pane and XML File in the right, and then click on Open to create.

This creates a new file called XMLFile1.XML (or similar), which you will save as the webpart filename. To start off with, when the XML Editing window appears, some text is already added for an XML file, which starts out with <?XML version=1.0. You do not need this line, so this should be deleted and you should start with an empty window. Next, simply enter the following XML statements (carefully!) and be sure to replace the Public Key Token as shown with the correct value you found in Part 3. Don't wrap lines:

```
<webParts>
  <webPart xmlns="http://schemas.microsoft.com/WebPart/v3">
    <metaData>
        <type name="HowManyLists.HMLClass, HowManyLists,
            Version=1.0.0.0, Culture=neutral,
            PublicKeyToken=9f4da00116c38ec5" />
        <importErrorMessage>Error importing assembly</importErrorMessage>
    </metaData>
    <data>
        <properties>
            <property name="Title" type="string">How Many Lists</property>
            <property name="Description"
               type="string">Shows number of lists in the site</property>
            <!-- The custom property -->
            <property name="ShowHitCount" type="bool">true</property>
        </properties>
    </data>
  </webPart>
</webParts>
```

Click the Save icon to save the file. When the Save File dialog appears, change the name from XMLFile1.XML (or whatever it is) to **HowManyLists.webpart**, and then save this to your project. Some quick notes on this file: Notice that although in the MetaData section you specify the assembly information, you preface it with the namespace (HowManyLists) and the class name (HMLClass). Below this, you should also specify the importErrorMessage value, which is the message displayed in SharePoint if there is a problem loading the assembly. Also, even minor issues with this file can cause a load failure. See the troubleshooting and debugging information that follows later on in the chapter.

Step 7: Create an Element Manifest File

The Element Manifest file defines where the actual Feature (Solution) elements will be installed and into which module and is used by the Feature file to know where to place the Solution in SharePoint. In the case of web parts, this is in the Web Part Catalog; since you only have a single web part to add here, this file is very short.

Do the following:

1. In your project in Visual Studio, select the top menu File | New | File to create a file.
2. Then, in the New File dialog box, in the left-hand pane, click on General to select it, and then in the right-hand pane, click XML File.
3. When the file opens, delete the default text and enter in the following:

```
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
      <Module Name="WebParts" List="113" Url="_catalogs/wp" >
           <File Url="HowManyLists.webpart" Type="GhostableInLibrary" />
      </Module>
</Elements>
```

4. Save the file, and when prompted, name the file **elementFeature.XML** and be sure that you are saving the file to the project folder (not the solution folder).

Step 8: Create a Feature File

The Feature file defines the location of the Element Manifest and webpart files for installing into SharePoint. Based on a unique GUID, this file will enable SharePoint to uniquely identify the Feature, even if duplicate names exist. In general, when creating a Feature file, you would customarily use the GUIDGEN.exe utility to generate a unique GUID to use. In the case of web parts (and courtesy of Visual Studio 2005), a unique GUID was generated as an XML item at the top of our class file as shown:

```
namespace HowManyLists
{
    [Guid("3c478a67-d437-4e8a-b2eb-2650ad26088b")]
    public class HMLClass : System.Web.UI.WebControls.WebParts.WebPart
    {
```

The GUID created in the class can now be used to create your Feature. You could generate another GUID, but that is unnecessary. The Feature file, like the Element Manifest file, is pretty short. To create the feature file, do the following:

1. To create the Feature file, in your project in Visual Studio, select the top menu File | New | File.
2. Go to the New File dialog box. In the left-hand pane, click on General to select it. Then, in the right-hand pane, click XML File.
3. When the file opens, delete the default text and enter in the following (remember to change the GUID accordingly):

```
<Feature Title="How Many Lists"
  Id="3c478a67-d437-4e8a-b2eb-2650ad26088b"
  Description="How Many Lists Solution Web Part"
  Version="1.0.0.0" Scope="Site" Hidden="FALSE"
  DefaultResourceFile="core" xmlns="http://schemas.microsoft.com/sharepoint/">
      <ElementManifests>
            <ElementManifest Location="elementManifest.xml" />
            <ElementFile Location="HowManyLists.webpart" />
      </ElementManifests>
</Feature>
```

4. Save the file and when prompted, name the file **Feature.XML**. Be sure that you are saving the file to the project folder (not the solution folder).

Step 9: Create a Manifest File

At this point, you have everything you need for the Feature/web part itself, so what remains is how to package it all up for SharePoint. For this, you need a Manifest file you can use to create a WSP file (like a CAB) and that will include the assembly, XML, and webpart files that will be loaded into SharePoint using the STSADM (SharePoint Command Line Administration Tool) tool.

The Manifest file specifically describes the contents of the WSP Package for SharePoint and identifies where the physical assets (files and so on) will go, as well as whether the assembly will be added to the Global Assembly Cache (GAC). It also includes the SafeControl

entry that will be added to the web.config file (the step you did manually in the first example). In reality, you are simply creating a Solution Manifest, the same as if installing any Feature or Solution for SharePoint. Since this particular web part is limited, you are not utilizing all of the options available, such as Resources for images and the like. For the complete reference on the Solution Schema, go to http://msdn2.microsoft.com/en-us/library/ms442108.aspx:

1. In your project in Visual Studio, select the top menu File | New | File to create a file.
2. Go to the New File dialog box. In the left-hand pane, click on General to select it. Then in the right-hand pane, click XML File.
3. When the file opens, delete the default text and enter in the following with some changes:
   - The SolutionId must be a unique GUID (use the GUIDGEN to create a new GUID).
   - The FeatureManifest location specifies the folder name that will be created under the SharePoint Features folder to hold the physical files (the example shows the Visual Studio default method of AssemblyName_GUID, but this is by no means required; use any name appropriate).
   - The SafeControl Assembly information must be the same as the one created in the webpart file:

```
<Solution SolutionId="72C358A7-B2AD-4bd7-9CAE-D2BD560560E3"
          xmlns="http://schemas.microsoft.com/sharepoint/">
  <FeatureManifests>
    <FeatureManifest
   Location="HowManyLists_3c478a67-d437-4e8a-b2eb-2650ad26088b\Feature.xml" />
  </FeatureManifests>
  <Assemblies>
    <Assembly Location="HowManyLists.dll"
              DeploymentTarget="GlobalAssemblyCache">
        <SafeControls>
            <SafeControl
                Assembly="HowManyLists, Version=1.0.0.0, Culture=neutral, PublicKey
Token=9f4da00116c38ec5"
                Namespace="HowManyLists" TypeName="HMLClass" Safe="True" />
        </SafeControls>
    </Assembly>
  </Assemblies>
</Solution>
```

4. Save the file, and when prompted, name the file **Manifest.XML**. Be sure that you are saving the file to the project folder (not the solution folder).

In the Manifest file, notice that you are setting the deployment target of the assembly to be the GlobalAssemblyCache, which will give it full privilege as well as make it accessible to the entire farm. While certainly not required for what you are doing with this part, you will more often than not use this setting. Alternately, this can be set to WebApplication, which will place the assembly in the SharePoint bin directory of the site. For more information, refer to the complete explanation of the Manifest file under Features.

Step 10: Create a DDF File (MAKECAB Instructions)

To create a WSP file using the MAKECAB.exe utility, you need to be able to give it some instructions, specifically what to package up and what to do when it is deployed. The DDF

file serves the same purpose as the .NET INF file for a Windows application. Finding information about DDF files is interesting to say the least; there's a lot of documentation, but not a lot of instruction. Fortunately, the necessary commands needed to install web parts are pretty straightforward:

- **CabinetNameTemplate** This specifies the name of the WSP file.
- **DiskDirectoryTemplate** This is set to CDROM, meaning all files go into a single directory.
- **CompressionType** This is set to MSZIP, meaning all files are compressed in cabinet files.
- **.Set DiskDirectory1** This sets the folder (on disk) where the WSP file will be created. Solution (as shown later) will create a subfolder (from where you run MAKECAB) called Solution and creates the WSP under the name set by CabinetNameTemplate.

The order of how the files are listed will determine the order they are copied in. In addition, the command .Set DestinationDir will create a new directory and any files listed after it will be placed there. In your case, you are instructing the WSP file to do the following:

1. Create a new folder to hold the files (the name is taken from the Manifest file).
2. Copy the Manifest.XML file and the assembly to the top-level directory.
3. Create a new folder to hold the Feature.
4. Copy the Feature, Element Manifest, and webpart files into the new Feature folder.
5. In your project in Visual Studio, select the top menu File | New | File to create a file.
6. Go to the New File dialog box. In the left-hand pane, click General to select it. Then, in the right-hand pane, click Text File.
7. When the file opens, enter the following text exactly (however, you must change the name of the destination folder to either remove the GUID shown or use your own):

```
;
.OPTION EXPLICIT     ; Generate errors
.Set CabinetNameTemplate=HowManyListsSolution.wsp
.set DiskDirectoryTemplate=CDROM ; All cabinets go in a single directory
.Set CompressionType=MSZIP ;** All files are compressed in cabinet files
.Set UniqueFiles="ON"
.Set Cabinet=on
.Set DiskDirectory1=Solution
Manifest.XML
HowManyLists.DLL

.Set DestinationDir=HowManyLists_3c478a67-d437-4e8a-b2eb-2650ad26088b
Feature.XML
elementManifest.XML
HowManyLists.webpart
;
; *** End
```

8. Save the file, and when prompted, name the file **HowManyLists.ddf** and be sure that you are saving the file to the project folder (not the solution folder).

Step 11: Use MAKECAB.exe to Create a WSP File

Making the WSP file is quite easy since you simply need to pass it the DDF file. The MAKECAB.exe utility is part of the Windows SDK and is available for download on the Microsoft site. With the DDF file, the command is quite simply makecab /F <DDF filename>; which takes the instructions in the DDF file and acts accordingly.

Note that in your example, you've built all of the necessary files and placed them in the Project folder. You are missing one (the DLL), which you will copy later on, but it is worth mentioning that it is just as easy to create a new folder, drop all the files needed, and then run the utility from there. In the online chapter, I cover a few ways to do this using Visual Studio.

To run this command, follow these simple steps:

1. Using Windows Explorer, navigate to your Project bin\debug folder (this is where your DLL should be located).
2. Copy the file up to the project folder (the DLL is needed here for the MAKECAB utility).
3. Next, select Start | All Programs | Microsoft Visual Studio 2005 | Visual Studio Tools | Visual Studio 2005 Command Prompt. Note that the VS command prompt includes the path needed to find the MAKECAB.exe file; otherwise, you must find this file yourself; the default location is C:\Program Files\Microsoft Visual Studio 8\SmartDevices\SDK\SDKTools).
4. From the command prompt, change your directory to the project directory; then do a DIR command to verify that you see all of the files, including the DDF and DLL files.
5. Enter the command **MAKECAB -/F HowManyLists.ddf** (note the */F* must be capitalized). Any errors that it encounters will be shown; if you need to see more details, you can use the Verbose switch and set the detail level (1 is lowest, 3 is highest), **V3**, for example.
6. When MAKECAB is complete, exit the command window.

Assuming all is well, a new folder called Package will be created in your project folder and the WSP file inside—and your build is complete! You are now ready to install the Solution in SharePoint; you can either copy the WSP file to another location or simply load it from where it is. The preferred location should be the SharePoint Root WPResources folder.

Step 12: Use STSADM to Add and Deploy

Once a WSP file has been created, it takes just three commands to actually install the web part Solution. Because these all use the STSADM utility, you'll combine them here, since you always do the same steps when deploying for the first time and/or to another server. As you already know, the STSAdm utility is the SharePoint command-line administration tool (see Appendix A for more information). Everything that can be done with the User Interface can be done with this utility and its use is pretty simple.

Since all of these commands are done at the command line, the first step is to go ahead and open up a command window. When open, navigate to the SharePoint Root Bin directory (usually c:\program files\common files\microsoft shared\web server extensions\12\bin). If you enter the command without any parameters (type in **STSAdm** and press ENTER), it will return help.

The first command is to add the Solution to SharePoint. This makes it available for deployment to sites. From the command line enter:

```
STSADM -o addsolution -filename <path to your solution aka WSP file>
```

For example, in my example the command line would be (entered on one line):

```
STSADM -o addsolution -filename
      c:\VSProjects\HowManyLists\HowManyLists
                \Package\HowManyListsSolution.wsp
```

Once the solution has been added, you can actually deploy it to a site using the **-url** switch (if you omit this, it will install on all sites). The command to deploy is

```
STSADM -o deploysolution -name <WSP Name> -local
    -allowGacDeployment -url <TargetURL>
```

In this example, it would be

```
STSADM -o deploysolution -name HowManyListsSolution.wsp -local -allowGacDeployment
-url http://mossspsdev:14584
```

NOTE *If the -url switch is omitted, the deployment is to all sites.*

At this point, the solution is installed and deployed so your last step is to activate it. While this can be done via the Site Administration, you can just as easily do it at the command line using the command:

```
STSADM -o activatefeature -id <solution id> -url <TargetURL>
```

You may recall the Feature ID; this is the GUID set in the Feature.XML file, which was originally created in the Class file with the project. So, going with the example, the command would be

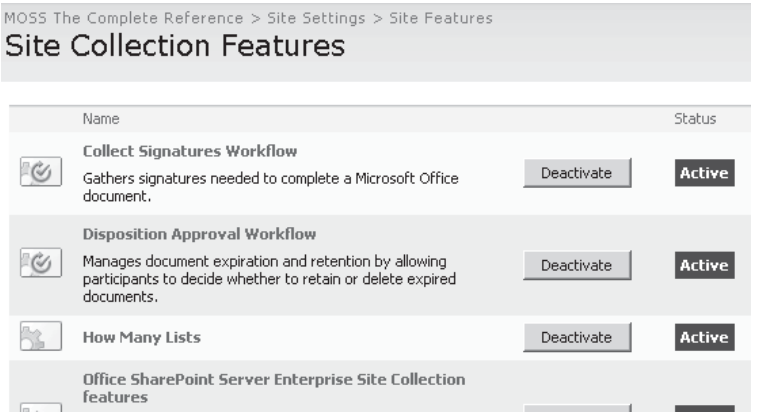
```
STSADM -o activatefeature -id 3c478a67-d437-4e8a-b2eb-2650ad26088b -url
http://mossspsdev:14584
```

To finish up, you need to run the IISReset command (simply type in **IISReset** and press ENTER). This is not mandatory, but it's always a good idea. Another technical tip: it's a really good idea to build a batch file to run the previous installation commands, because it can be pretty daunting to type in a GUID. This applies for removal as well (see the commands later on in the chapter).

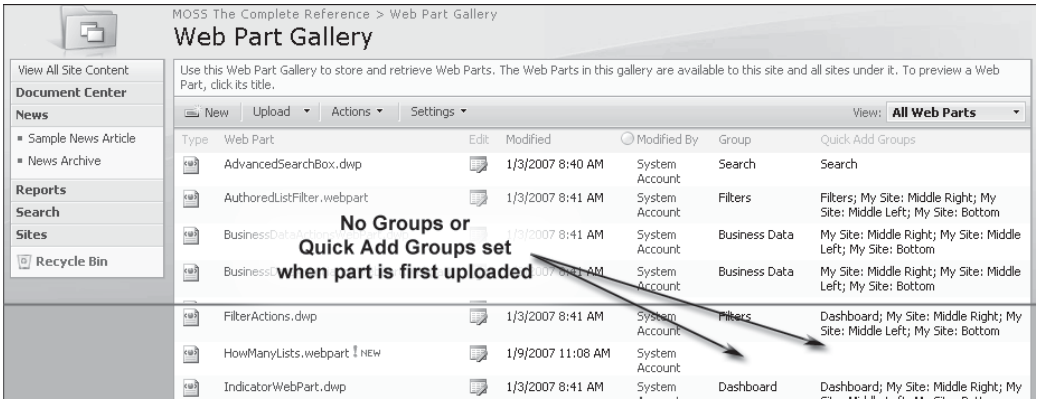
Verifying the Installation

Now it's time to see what you have accomplished with all of this. Since you have already activated the solution, it is actually ready to go, but you can verify the installation through the site administration:

1. Log in to the SharePoint site and from the home page, select Site Actions | Site Settings | Modify All Site Settings.
2. Under the Site Collection Administration section, click on the Site Collection Features link. When the page opens, you should see the solution installed active, as shown:

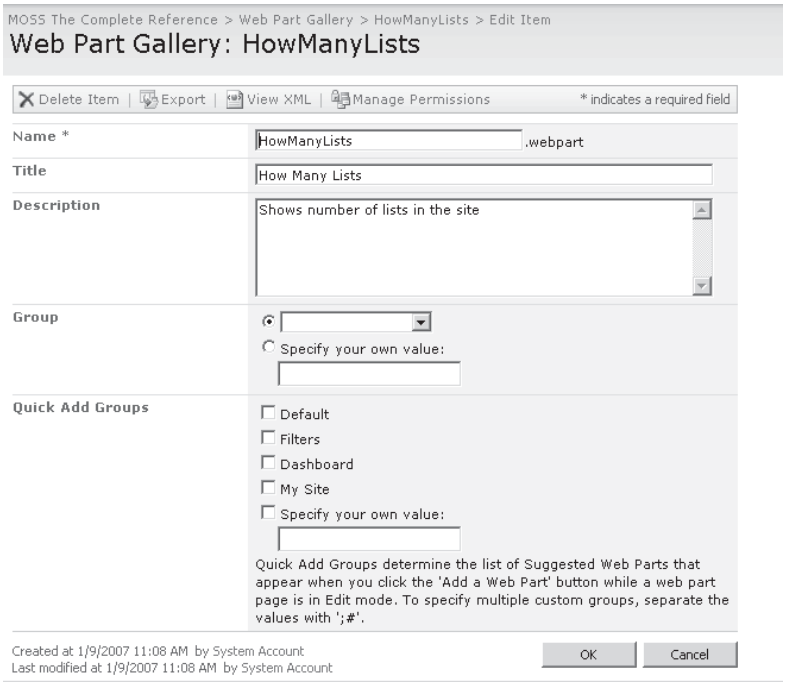


3. Return to the Site Settings page (use the link on the breadcrumbs), and then, under the Galleries section, click the Web Parts link, which opens the Web Part Gallery. Scroll down the list and you should find the new part:



4. Notice that while some of the properties are set, namely the Title and Description (which came from the webpart file), the Group and Quick Add Groups are not set. While these can be set when adding a web part programmatically, there is no way to set these according to the official Microsoft documentation. To modify the settings, you need to edit the Solution properties (not to be confused with web

part properties). Click the Edit icon next to the name to open the Properties page, which should look something like this:

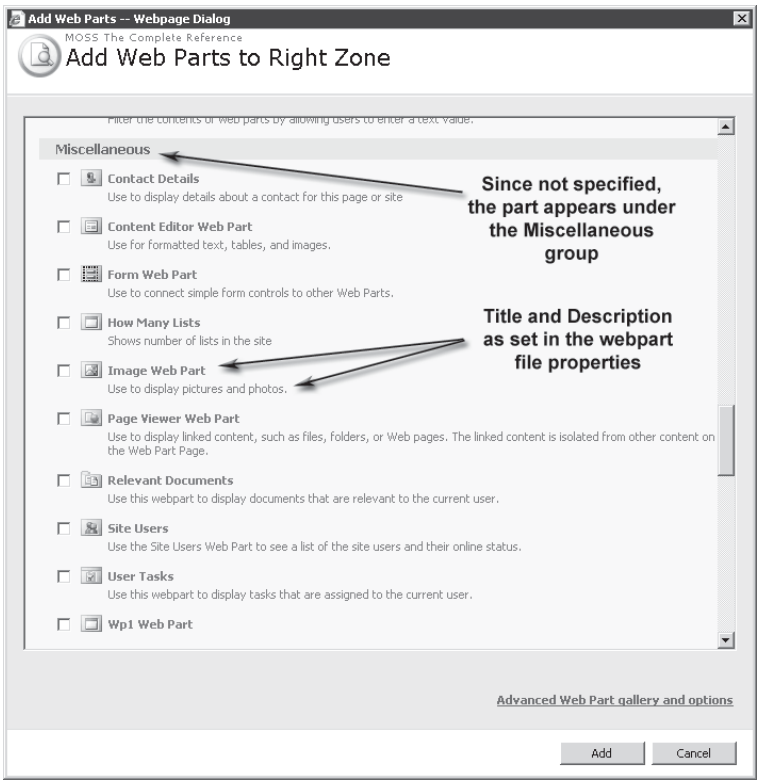


5. From this page, you can now both set groups and quickly add groups. Also note, this page provides you with two more capabilities that you will find quite handy in the course of things:
   - **Export** This allows you to export the webpart definition file.
   - **View XML** This allows you to see the webpart file (in XML form).
6. Since you're about to remove this part, you needn't worry about entering anything here; click Cancel or simply click the link to go back to the home page.

Testing the Web Part

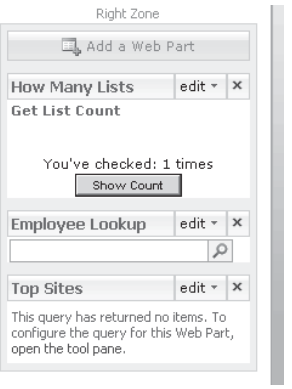
Once the part has been installed and you've verified it, you can add it to a page to see how it operates. If you installed it at the site root, as the example showed, the part is available everywhere, so you can add it to any page you choose:

1. Open the SharePoint site to a web part page and then select Site Actions | Edit Page.
2. When the page opens in Design Mode, click any one of the Add Web Part links.
3. This will open the Web Part Gallery Add Web Parts dialog. Scroll down to the Miscellaneous section to find it:

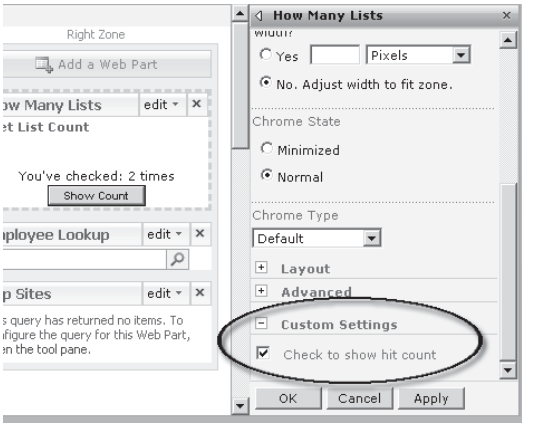


As shown, since you did not specify a group in the Solution properties, it falls to the default group. Also notice that the title and description are shown. These were set by the properties entries you created in the webpart file.

4. Add the part by clicking the check box next to the title and then click the Add button.
5. When the part loads, you should see the display similar to this:



At this point, you can verify that your custom property setting in the webpart file actually took effect. If you recall, you created a custom property called ShowHitCount, and in the code, set the default to be False. In the webpart file you overrode that and set this to be True. From the previous example, you can see that it did indeed set it, because of the default display (the hit count records page hits so one has already occurred when you added the part). To actually verify the setting, click the Edit link on the Web Part Title and then select Modify Shared Web Part. When the Properties panel appears, scroll down to Custom Properties (the name you set in the code) and click + to expand it. There you will see the custom property and that it is indeed checked:



Now you can test out the functionality of the part. To do this, click the Publish link from the Publishing menu to save the page and then publish it. Now you can check out the basic operation of the part:

- When you click the button, the number of lists in the current site will be counted and the total will be displayed.
- Each time you click or refresh (but stay on the same page), you will see the hit count increment.

Fixing Common Loading Problems

There are a few common errors that can occur when you're loading the web part (though I'm hoping that's not happening to you now!). The basic error handling here is user-based and therefore pretty generic. It's further masked by the message displayed since this is simply the message specified in the webpart importErrorMessage. While there can be any number of issues, here are a few tips:

Invalid Entries If you have an invalid entry in the webpart file or if there is something wrong with the assembly, you usually see a message like:

> Unable to add selected web part(s)
>
> My Custom Web Part: This web part cannot be imported; error loading assembly, Contact Support.

In the case of the assembly itself being a problem, it is usually related to permissions, the code itself having an error, or the code referencing an uninstalled feature.

Possible fixes include:

- Making sure that the web.config trust level has been set to WSS_Medium.
- Making sure that the SharePoint Services account has Execute privilege on the SharePoint Root directories.
- Verifying you are not referencing any DLLs or assemblies that don't exist on the system (custom controls are usually a culprit here).
- Double-checking the .webpart file for typos.

Illegal Entries Probably the most common error may *sound* more severe, but is less of an issue:

Unable to add selected web part(s).

My Custom Web Part: The format is not valid. Try importing a Web Part file (.webpart).

This message is telling you that somewhere in your webpart file you have either a bad XML statement (missing an end tag) or that you have a property that was either misspelled, reserved (can't be changed), or simply not a valid property name.

Fixing the Webpart File In both of the previous issues, if the webpart file was the problem, the fix was simple and you didn't have to do anything to the Solution. Go to Site Administration, open the Web Part Gallery, find the web part, and then edit the properties. You can also export the webpart file, delete it from the Gallery, edit the file, return to the Web Part Gallery, and then upload it.

Debugging

When you have installed a solution in this manner, you can still debug the web part. But, unlike the first part you did, the only way to debug an assembly installed this way is by using the Attach to Process method. Refer to the debugging section in that example for instructions. There are a few conditions for installed solutions. First, the assembly you installed must have been built in Debug Mode, and second, you must have debug enabled on the site through the web.config file. if either of these conditions is not met, you'll receive a message stating that, "The following module was built either with optimizations enabled or without debug information."

Removing a Web Part Solution

Removing a Solution can be done completely through the Site and Central Administration, but it is much quicker to do via the command line. It must be removed in a similar order to the way you installed it, and you also have to add an additional command. As with adding a web part, the first step is to go ahead and open up a command window. When it's open, navigate to the SharePoint Root Bin directory (usually c:\program files\common files\microsoft shared\web server extensions\12\bin).

The first command is to deactivate the Solution on the site, and again you use the **-url** switch to remove it from a specific site. The format of this command is

```
STSADM -o deactivatefeature -id <solution id> -url <TargetURL>
```

Again, the ID here is the GUID of the Solution (from Feature.XML) so the command (entered on one line) would be

```
STSADM -o deactivatefeature
    -id 3c478a67-d437-4e8a-b2eb-2650ad26088b -url http://mossspsdev:14584
```

Next, you have to uninstall the feature using the command (note that the **-force** switch is required):

```
STSADM -o uninstallfeature -id <solution id> -force
```

So the command will be

```
STSADM -o uninstallfeature -id 3c478a67-d437-4e8a-b2eb-2650ad26088b -force
```

Now you have to retract the solution from the farm (this does not remove it, but it is no longer available). This command to retract a solution is

```
STSADM -o retractsolution -name <WSP Name> -local -url <TargetURL>
```

In this example, it would be (again entered on one line)

```
STSADM -o retractsolution
    -name HowManyListsSolution.wsp -local -url http://mossspsdev:14584
```

Now that the solution has been retracted, it can be deleted using the Delete Solution command:

```
STSADM -o deletesolution -name <WSP Name>
```

So your command would be

```
STSADM -o deletesolution -name HowManyListsSolution.wsp
```

At this point, the solution has been removed but you have one little problem; although everything is gone, the .webpart XML definition still exists in the Web Part Gallery. In fact, if a user were to attempt to add this part to a page, they will receive an error similar to:

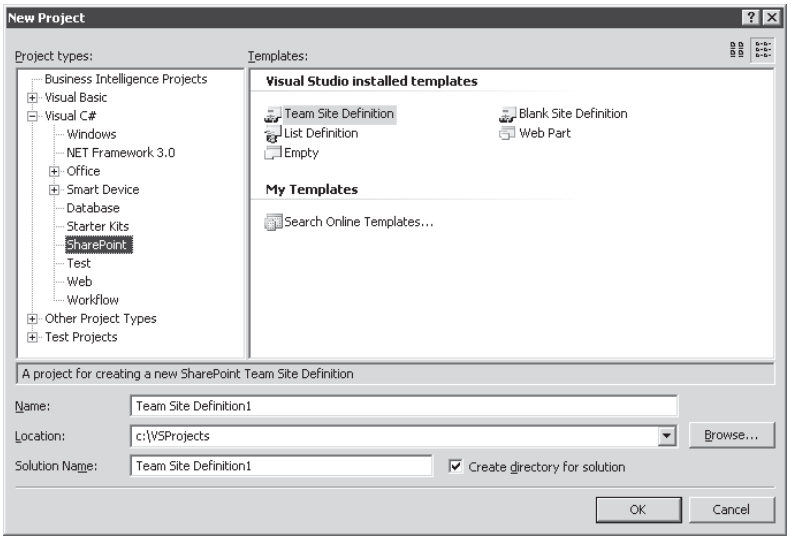
```
Unable to add selected web part(s).
How Many Lists: The Web Part you attempted to add no longer exists in
the Closed Web Parts Gallery. Refresh your browser to update the Web
Part Page and the Gallery.
```

So here's the final step for removing the part completely:

1. Open the SharePoint site and select Site Actions | Site Settings | Modify All Site Settings.
2. On the Site Administration page under the Galleries section, click the Web Parts link to open the Web Part Gallery, and then scroll down until you find the part.
3. Click on the Edit icon next to the name.
4. When the Properties page is displayed, click the Delete link.

Building Web Parts in Visual Studio 2005

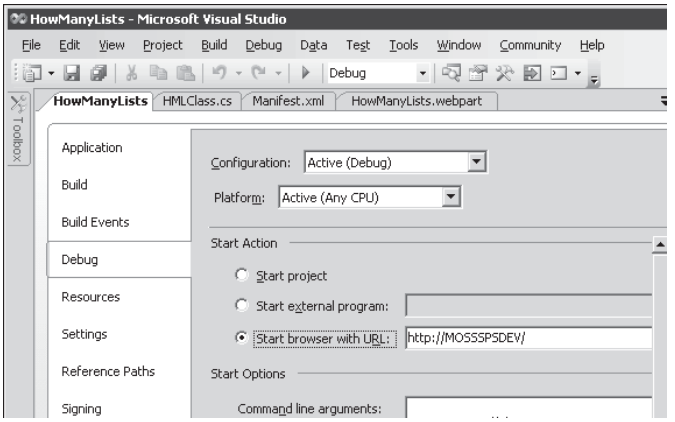
So finally you get to the good stuff: building web parts directly out of Visual Studio. If you followed the previous examples, you can see that it wasn't so tough, but yes, there are a lot of moving parts. Fear not, with the release of Windows SharePoint Services 3.0 Extensions for Visual Studio, building and deploying a web part is a snap! With WSS Extensions, nearly everything you would want to build is included, from web parts to (gasp!) Site Definitions! In fact, WSS Extensions give you several options as shown:



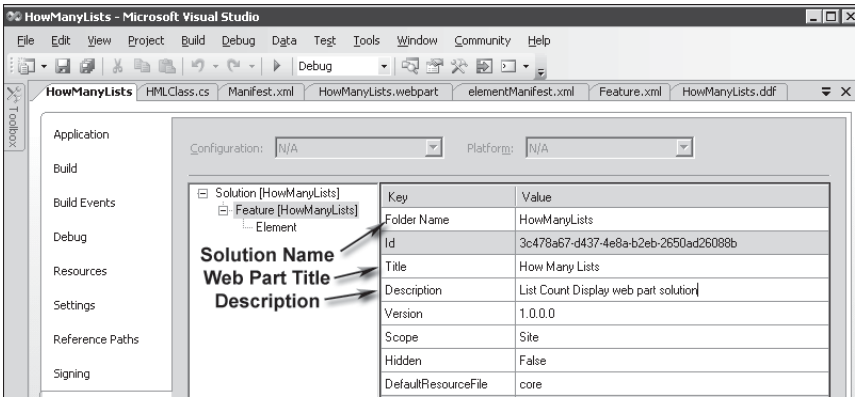
As you can see, WSS Extensions provide the ability to create a new Team Site Definition, Blank Site Definition, List Definition, and yes, the ubiquitous Web Part Definition. In this example, you're going to build a part within Visual Studio and have it deploy directly to the server. You'll basically do everything that you did in the previous example, but I think you will find that the automated method reduces the number of steps (actually to 1!).

To begin with, you can either start with a new project or use the one from the previous example. Since the last step of that example removed the part completely, you are starting from scratch installation wise, so either will do. However, since the part you just created actually does something, I will use it for this example. So let's get started!

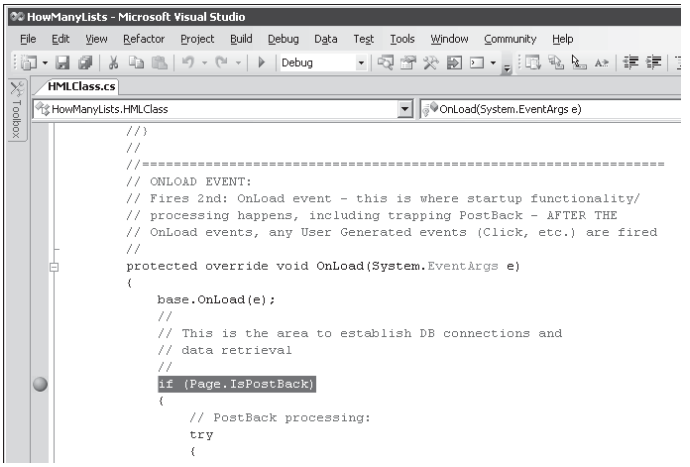
1. Open the HowManyLists project in Visual Studio (or select Open | Project then browse to your project).
2. In the Solution Explorer, right-click on the project name and select Properties.
3. On the Properties page, click the Debug tab.
4. Under Start Browser with URL, enter the path to the SharePoint site:



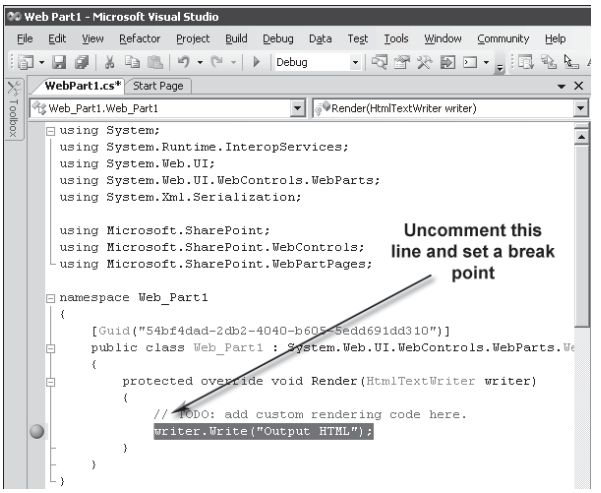
5. Next, click the SharePoint Solution tab.
6. Enter the Folder Name (this becomes the Solution Name), Title, and Description (note that a bug does not save the description; you must edit the .webpart file manually):



7. If you are using a specific strong name key (see the previous example), you can set that now; by default, the temporary.snk file is already assigned, so you don't need to do anything. Close the Properties page and open your class file.
8. If you are using the HowManyLists project, set a break point in the OnLoad() method:



9. If you have created a new project, uncomment the one line of code and set a break point on it as shown:

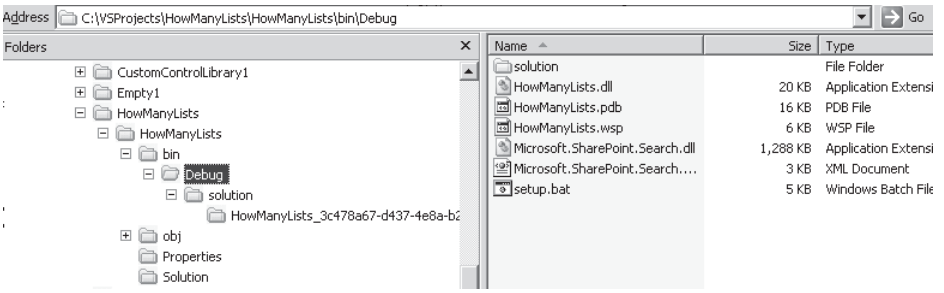


10. Click the Save All icon.
11. To build and deploy the project, press the F5 key or select Build | Deploy <project>.
12. When the SharePoint Site opens, edit a page and add the web part. When the part loads, you will automatically hit the break point in the code and can now step through it.

Yes folks, that's all there is to it!

So what's the magic behind the scenes? Basically, Visual Studio does for you what you did in the previous example; it creates the files needed (Manifest, Feature, Element Manifest, and webpart files), it creates a nice little batch file to do the installation (and uninstall as you will see next), and it creates the WSP file. Not only that, but voila—it installs it directly into SharePoint, deploys the solution, activates it, and even runs IISReset when done—all in one step!

To check this out, use Windows Explorer to open the project folder and then expand out to see the Debug folder. As you can see, in addition to the normal DLL and debug files, you'll see a new folder called Solution:



In this Solution folder, you will find the Manfiest.XML file, DLL, and the Feature/Solution folder to be installed in SharePoint (notice the GUID is appended to the name), which contains the Feature, Elements Manifest, and webpart files. In addition, you may have noticed the Setup.bat file. This is used by Visual Studio to install and uninstall the Solution.

And by the way, Visual Studio also handles removing and redeploying the solution for you if you rebuild and deploy. You can try this out by stopping the debugging and then clicking F5 again to build and redeploy. You will see that the existing solution is deactivated, then it's removed, and then the new one installed.

NOTE *You must actually deploy the solution to the site to create the files and solution folder.*

Using the Web Part Solution to Create a Farm Deployment Package

As mentioned previously, Visual Studio will build all of the necessary files. It will then create a batch file that handles adding the solution to SharePoint and then deploys the solution automatically. While that's very cool for building the part, it is also cool for deployment elsewhere. The solution that Visual Studio creates includes a Setup.bat file it uses to install or remove the Solution. If you view this file, you'll see that all the information is already there, including the URL where it will deploy. To deploy the solution on a new server, simply zip up the Solution folder, copy to a new server, unzip, edit the Setup.bat file to point to the correct URL, and then run the setup.bat file (from command line or using the Run command) as

```
Setup.bat /install
```

To remove the solution

```
Setup.bat /uninstall
```

Removing a Web Part Solution Deployed by Visual Studio

To remove a solution that has been installed by Visual Studio, you can easily go through the Site Administration. Simply deactivate it, delete the solution from the Site Collection Features, and then go to the Web Part Gallery and delete the web part. Alternatively, you can use the Setup.bat file as you did previously by using the /uninstall switch but be aware—the Setup.bat file removes the solution, but just like your manual example, it doesn't remove the web part from the Web Part Gallery. This must be done manually.

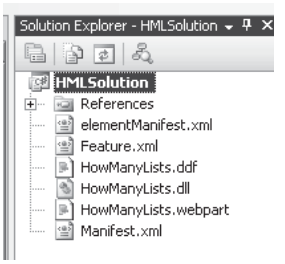
Using MSBuild to Build WSP Files Automatically

Visual Studio nicely handles all of the deployment chores for a web part, but in many cases, you might not want to have to deploy to a server just to create the installation files. Never fear, since you already know a bit about what goes into building the package, and specifically the WSP file, you can easily modify a Visual Studio project to generate this for us automatically.

There are two ways to implement using MSBuild: directly in your project or by creating a solution project that will do the same thing. The difference is that if you make these changes to your project, you will not be able to use the automated deployment method. For this reason, I suggest using a Solution project since it is just as easy. In addition to the project files, MSBuild also needs a Manifest.XML and DDF file (for the MAKECAB utility), as well as a Targets file to be used by Visual Studio. The Manifest and DDF files are exactly the

same ones you built in the example, so if you are following the example, you can reuse these same files. To implement a Solution project that creates a WSP file, do the following:

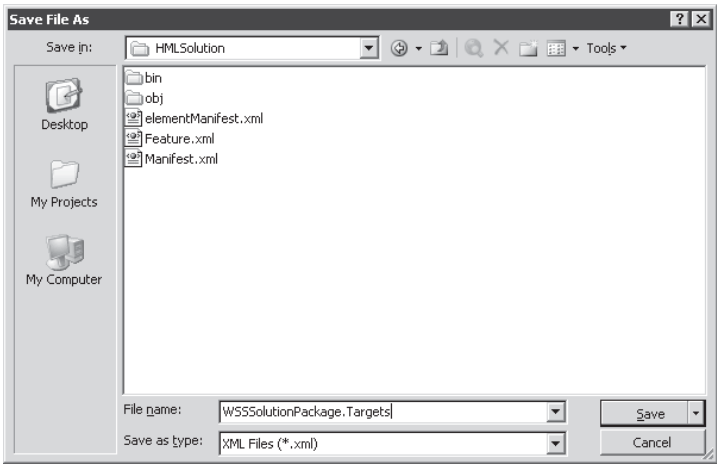
1. Open Visual Studio and create a new Empty project and name it **HMLSolution**.
2. When the project opens, right-click on the project name and select Add Existing Item in the Solution Explorer.
3. Browse to your project folder and add the following files: Manifest.XML, HowManyLists.ddf, HowManyLists.ddl, HowManyLists.webpart, Features.XML, and elementManifest.XML. When you're done, your solution look something like this:



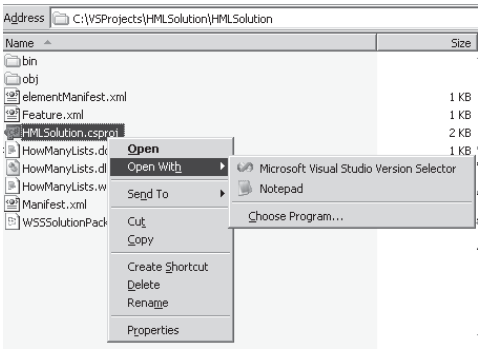
4. Now you need to add the Targets file for Visual Studio. Select New | File; from the dialog box, click General in the left-hand pane, click XML file in the right pane, and then click the Open button.
5. When the file opens, add the following (do not break up the lines as I've done here; this is just to make it readable):

```
<?XML version="1.0" encoding="utf-8"?>
<Project xmlns="http://schemas.microsoft.com/developer/msbuild/2003"
DefaultTargets="WSSSolutionPackage">
<PropertyGroup>
<MakeCabUtilPath>"C:\WINDOWS\system32\MAKECAB.exe"</MakeCabUtilPath>
</PropertyGroup>
<Target Name="WSSSolutionPackage">
<!-- Create a Windows SharePoint Services Solution (WSP) file -->
<Exec Command="$(MakeCabUtilPath) /F HowManyLists.ddf " />
<!-- Create as a CAB file for to see the contents -->
<Exec Condition=" '$(Configuration)|$(Platform)' == 'Debug|AnyCPU' "
   Command="$(MakeCabPath) /F WSSSolutionPackage.ddf " />
</Target>
</Project>
```

6. Save this file back to your project folder (in this case, HMLSolution) and name the file **WSSSolutionPackage.Targets** as shown:



7. This will save the file as WSSSolutionPackage.Targets.XML, so outside of Visual Studio, use Windows Explorer to navigate to your project folder, find the WSSSolutionPackage.Targets.XML file, and then rename it to **WSSSolutionPackage .Targets** (you cannot rename it in Visual Studio).
8. Back in the Visual Studio Solution Explorer, click the Show All Files icon.
9. Find the WSSSolutionPackage.Targets file, right-click on it, and include it in the project.
10. The next step involves editing a project file, so close Visual Studio.
11. Use Windows Explorer to navigate to the solution project folder.
12. Right-click on the file and open it in Notepad.



13. Modify the first line and change DefaultTargets to **WSSSolutionPackage**. Go from

```
<Project DefaultTargets="Build"
   xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
```

To

```
<Project DefaultTargets="WSSSolutionPackage"
   xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
```

14. Next, near the bottom of the file, look for the <Import> entry and change the Project from $(MSBuildBinPath)\Microsoft.CSharp.targets to **WSSSolutionPackage.Targets**.

From

```
<Import Project="$(MSBuildBinPath)\Microsoft.CSharp.targets" />
```

To

```
<Import Project="WSSSolutionPackage.Targets" />
Save the file and exit the editor
```

15. Reopen the solution project in Visual Studio. You may get a warning message about the project being modified; check Load Project Normally and click OK.
16. Select Build | Build Project. The new WSP file will be built in the Solution folder and is ready to deploy!

Summary

This chapter covered the basics for building web parts, which is probably one of the most powerful extensions you can build on top of the SharePoint platform. With a bare minimum of code, you can easily build custom displays, external data displays, data entry forms, and more, and then deploy them with a click. Although the examples in this chapter were kept pretty simple, remember that anything you can do in .NET, you can do in a web part. And while I touched on it briefly, be sure to look into building connectable web parts, which are the gateways to building really complex applications. Finally, web parts are the preferred method for development inside of the SharePoint site environment; for those of you who came from the 2003 environment and either were scared of or didn't like web parts, you'll have to adjust!

APPENDIX A

The STSAdm Command Tool

Command Reference	Purpose
stsadm **-o activatefeature** {-filename <relative path to Feature.xml> | -name <feature folder> | -id <feature Id>} [-url <url>] [-force]	Activates an installed feature for the farm or for the specified site at <URL>.
stsadm **-o activateformtemplate** -url <URL to the site collection> [-formid <form template ID>] [-filename <path to form template file>]	Activates an installed Form Template (InfoPath).
stsadm **-o addalternatedomain** -url <protocol://existing.WebApplication.URLdomain> -incomingurl <protocol://incoming.url.domain> -urlzone <default, extranet, internet, intranet, custom> -resourcename <non-web application resource name>	Adds an alternate domain name to the specified Internet zone.
stsadm **-o addcontentdb** -url <url> -databasename <database name> [-databaseserver <database server name>] [-databaseuser <database username>] [-databasepassword <database password>] [-sitewarning <site warning count>] [-sitemax <site max count>]	Adds a content database to an existing SharePoint site.
stsadm **-o adddataconnectionfile** -filename <path to file to add> [-webaccessible <bool>] [-overwrite <bool>] [-category <bool>]	Adds a data connection file (used to connect to an external data source).

Command Reference	Purpose
stsadm **-o add-ecsfiletrustedlocation** -Ssp <SSP name> -Location <URL|UNC> -LocationType SharePoint|Unc|Http -IncludeChildren True|False [-SessionTimeout <time in seconds>] [-ShortSessionTimeout <time in seconds>] [-MaxRequestDuration <time in seconds>] [-MaxWorkbookSize <file size in Mbytes>] [-MaxChartSize <size in Mbytes>] [-VolatileFunctionCacheLifetime <time in seconds>] [-DefaultWorkbookCalcMode File|Manual|Auto|AutoDataTables] [-AllowExternalData None|Dcl|DclAndEmbedded] [-WarnOnDataRefresh True|False] [-StopOpenOnRefreshFailure True|False] [-PeriodicCacheLifetime <time in seconds>] [-ManualCacheLifetime <time in seconds>] [-MaxConcurrentRequestsPerSession <number of requests>] [-AllowUdfs True|False] [-Description <descriptive text>]	Adds a new Excel Services Trusted File Location and sets the calculation mode (this is stored in Shared Services).
stsadm **-o add-ecssafedataprovider** -Ssp <SSP name> -ID <data provider id> -Type Oledb|Odbc|OdbcDsn [-Description <descriptive text>]	Adds a new data provider connection (and type) to a Shared Service Provider.
stsadm **-o add-ecstrusteddataconnectionlibrary** -Ssp <SSP name> -Location <URL> [-Description <descriptive text>]	Adds a new trusted data connection library in the selected Shared Service Provider.
stsadm **-o add-ecsuserdefinedfunction** -Ssp <SSP name> -Assembly <strong name|file path> -AssemblyLocation GAC|File [-Enable True|False] [-Description <descriptive text>]	Adds a new user defined function (solution), including list definitions and DLLs.
stsadm **-o addexemptuseragent** -name <user-agent to receive InfoPath files instead of a Web page>	Adds a new Exempt User Agent for InfoPath forms.
stsadm **-o addpath** -url <url> -type <explicitinclusion/wildcardinclusion>	Adds or excludes a path from an application.
stsadm **-o addpermissionpolicy** -url <url> -userlogin <login name> -permissionlevel <permission policy level> [-zone <URL zone>] [-username <display name>]	Adds a new Permission policy to a SharePoint site.
stsadm **-o addsolution** -filename <Solution filename> [-lcid <language>]	Adds a solution (solution packages/web parts).

Command Reference	Purpose
stsadm **-o addtemplate** -filename <template filename> -title <template title> [-description <template description>]	Adds a new template (site definition) to SharePoint.
stsadm **-o adduser** -url <url> -userlogin <DOMAIN\user> -useremail <someone@example.com> -role <role name> / -group <group name> -username <display name> [-siteadmin]	Adds a new user to the selected SharePoint site.
stsadm **-o addwppack** -filename <Web Part Package filename> [-lcid <language>] [-url <url>] [-globalinstall] [-force] [-nodeploy]	Adds a Web Part Package (declining).
stsadm **-o addzoneurl** -url <protocol://existing.WebApplication.URLdomain> -urlzone <default, extranet, internet, intranet, custom> -zonemappedurl <protocol://outgoing.url.domain> -resourcename <non-web application resource name>	Adds a new Security Zone for alternate access mapping.
stsadm **-o allowuserformwebserviceproxy** -url <URL of the web application> -enable <true to enable, false to disable>	Enables or disables use of the Forms Web Service proxy on the selected SharePoint site.
stsadm **-o allowwebserviceproxy** -url <URL of the web application> -enable <true to enable, false to disable>	Enables or disables use of the Web Service proxy on the selected SharePoint site.
stsadm **-o associatewebapp** -title <SSP name> [-default | -parent] -url <Web application 1 url,Web application 2 url> [-all]	Associates the selected SharePoint site with the specified Shared Services Provider.
stsadm **-o authentication** -url <url> -type <windows/forms/websso> [-usebasic (valid only in windows authentication mode)] [-usewindowsintegrated (valid only in windows auth mode)] [-exclusivelyusentlm (valid only in windows auth mode)] [-membershipprovider <membership provider name>] [-rolemanager <role manager name>] [-enableclientintegration] [-allowanonymous]	Sets the authentication to use for the SharePoint site.

Command Reference	Purpose
For site collection backup: stsadm **-o backup** -url <url> -filename <filename> [-overwrite] For catastrophic backup: stsadm **-o backup** -directory <UNC path> -backupmethod <full | differential> [-item <created path from tree>] [-percentage <integer between 1 and 100>] [-backupthreads <integer between 1 and 10>] [-showtree] [-quiet]	Performs a SharePoint backup.
stsadm **-o backuphistory** -directory <UNC path> [-backup] [-restore]	Lists the SharePoint backup history.
stsadm **-o binddrservice** -servicename <data retrieval service name> -setting <data retrieval services setting>	Binds a new data retrieval service (for the use with the BDC).
stsadm **-o blockedfilelist** -extension <extension> { -add | -delete } [-url <url>]	Adds or removes file types from the Blocked File Type list.
stsadm **-o canceldeployment** -id <id>	Cancels the deployment of a feature or solution (by its ID).
stsadm **-o changepermissionpolicy** -url <url> -userlogin <DOMAIN\name> [-zone <URL zone>] [-username <display name>] [{ -add | -delete } -permissionlevel <permission policy level>]	Change permission policy in a security zone.
stsadm **-o copyappbincontent**	Copies the application bin content (the bin directory that is automatically created under the application); note that actual use is not documented by Microsoft.

Command Reference	Purpose
stsadm **-o createadminvs** [-admapidname <app pool name>] [-admapidtype <configurableid/NetworkService>] [-admapidlogin <DOMAIN\name>] [-admapidpwd <app pool password>]	Creates a SharePoint application (same as creating it through Application Management in Central Administration).
stsadm **-o createcmsmigrationprofile** -profilename <profile name> [-description <description>] [-connectionstring <connection string>] -databaseserver <server> -databasename <name> -databaseuser <username> [-databasepassword <password>] [-auth windowsauth|sqlauth] -destination <url> [-rootchannel <channelname>] [-destinationlocale <LCID>] [-migrateresources onlyused|all] [-migrateacls yes|no] [-emailto <address1;address2>] [-emailon success|failure|none|both] [-keeptemporaryfiles Never|Always|Failure] [-enableeventreceivers yes|no]	Creates a CMS Migration Profile (for use with CMS systems to convert to MOSS).
stsadm **-o creategroup** -url <url> -name <group name> -description <description> -ownerlogin <DOMAIN\name or group name> [-type member|visitor|owner]	Creates a new SharePoint User Group.
stsadm **-o createsite** -url <url> -owneremail <someone@example.com> [-ownerlogin <DOMAIN\name>] [-ownername <display name>] [-secondaryemail <someone@example.com>] [-secondarylogin <DOMAIN\name>] [-secondaryname <display name>] [-lcid <language>] [-sitetemplate <site template>] [-title <site title>] [-description <site description>] [-hostheaderwebapplicationurl <web app url>] [-quota <quota template>]	Creates a new SharePoint site under the specified application (also sets the type of site template to be used, for example, STS#0 for a Team Site).

Command Reference	Purpose
stsadm **-o createsiteinnewdb** -url <url> -owneremail <someone@example.com> [-ownerlogin <DOMAIN\name>] [-ownername <display name>] [-secondaryemail <someone@example.com>] [-secondarylogin <DOMAIN\name>] [-secondaryname <display name>] [-lcid <language>] [-sitetemplate <site template>] [-title <site title>] [-description <site description>] [-hostheaderwebapplicationurl <web app url>] [-quota <quota template>] [-databaseuser <database username>] [-databasepassword <database password>] [-databaseserver <database server name>] [-databasename <database name>]	Creates a new SharePoint site using the specified database (this is often used when the databases are created by the DBA before the installation).
stsadm **-o createssp** -title <SSP name> -url <Web application url> -mysiteurl <MySite Web application url> -ssplogin <username> -indexserver <index server> -indexlocation <index file path> [-ssppassword <password>] [-sspdatabaseserver <SSP database server>] [-sspdatabasename <SSP database name>] [-sspsqlauthlogin <SQL username>] [-sspsqlauthpassword <SQL password>] [-searchdatabaseserver <search database server>] [-searchdatabasename <search database name>] [-searchsqlauthlogin <SQL username>] [-searchsqlauthpassword <SQL password>] [-ssl <yes|no>]	Creates a new Shared Services Provider and defines the databases it will use.
stsadm **-o createweb** -url <url> [-lcid <language>] [-sitetemplate <site template>] [-title <site title>] [-description <site description>] [-convert] [-unique]	Creates a web (site) at the specified URL, including the template to use (type of site) and whether it will inherit permissions or be unique.
stsadm **-o databaserepair** -url <url> -databasename <database name> [-deletecorruption]	Starts the database integrity checking process that examines the database specified and attempts to repair any problems found.
stsadm **-o deactivatefeature** {-filename <relative path to Feature.xml> | -name <feature folder> | -id <feature Id>} [-url <url>] [-force]	Deactivates a feature that has been activated in SharePoint.

Command Reference	Purpose
stsadm **-o deactivateformtemplate** -url <URL to the site collection> [-formid <form template ID>] [-filename <path to form template file>]	Deactivates the specified Form Template (same as if done from the Site Settings).
stsadm **-o deleteadminvs**	Deletes a SharePoint application.
stsadm **-o deletealternatedomain** -url <ignored> -incomingurl <protocol://incoming.url.domain>	Deletes an Alternate Access Mapping entry.
stsadm **-o deletecmsmigrationprofile** -profilename <profile name>	Deletes a CMS Migration Profile.
stsadm **-o deleteconfigdb**	Deletes the Configuration Database.
stsadm **-o deletecontentdb** -url <url> -databasename <database name> [-databaseserver <database server name>]	Deletes the named content database for the specified aite at <URL>.
stsadm **-o deletegroup** -url <url> -name <group name>	Deletes the specified SharePoint User Group.
stsadm **-o deletepath** -url <url>	Deletes the specified path from the Inclusion/ Exclusion list.
stsadm **-o deletepermissionpolicy** -url <url> -userlogin <login name> [-zone <URL zone>]	Deletes the specified Permissions policy from the zone.
stsadm **-o deletesite** -url <url> -deleteadaccounts <true/false>	Deletes the site at <URL> and optionally deletes the accounts in Active Directory associated with that site.
stsadm **-o deletesolution** -name <Solution name> [-override] [-lcid <language>]	Deletes a solution from SharePoint (it must be retracted before this will work).
stsadm **-o deletessp** -title <SSP name> [-deletedatabases]	Deletes the specified Shared Services Provider.
stsadm **-o deletessptimerjob** -title <SSP Name> -jobid <SSP Timer Job Id>	Deletes the specified Timer Job running under the Shared Services Provider.
stsadm **-o deletetemplate** -title <template title> [-lcid <language>]	Deletes the specified Site Template.
stsadm **-o deleteuser** -url <url> -userlogin <DOMAIN\name> [-group <group>]	Deletes a user from a site or from a specific group.

Command Reference	Purpose
stsadm **-o deleteweb** -url <url>	Deletes the site at <URL>.
stsadm **-o deletewppack** -name <name of Web Part Package> [-lcid <language>] [-url <url>]	Deletes a Web Part Package (declining).
stsadm **-o deletezoneurl** -url <protocol://existing.WebApplication.URLdomain> -urlzone <default, extranet, internet, intranet, custom> -resourcename <non-web application resource name>	Deletes a zone specified for Alternate Access Mapping.
stsadm **-o deploysolution** -name <Solution name> [-url <virtual server url>] [-allcontenturls] [-time <time to deploy at>] [-immediate] [-local] [-allowgacdeployment] [-allowcaspolicies] [-lcid <language>] [-force]	Used to deploy solution packages (including Web Parts); if URL is specified, deploys to the specific site.
stsadm **-o deploywppack** -name <Web Part Package name> [-url <virtual server url>] [-time <time to deploy at>] [-immediate] [-local] [-lcid <language>] [-globalinstall] [-force]	Used to deploy Web Part Packages (declining).
stsadm **-o disablessc** -url <url>	Disables Self-Service Site Creation for the specified SharePoint site.
stsadm **-o displaysolution** -name <Solution name>	Displays the solution details.
stsadm **-o editcmsmigrationprofile** -profilename <profile name> [-description <description>] [-connectionstring <connection string>] [-databaseserver <server>] [-databasename <name>] [-databaseuser <username>] [-databasepassword <password>] [-auth sqlauth|windowsauth] [-emailto <address1;address2>] [-emailon success|failure|none|both] [-excludeschema] [-keeptemporaryfiles Never|Always|Failure] [-enableeventreceivers yes|no]	Alters an existing CMS Migration Profile (content migration).

Command Reference	Purpose
stsadm **-o editcontentdeploymentpath** -pathname <path name> [-keeptemporaryfiles Never|Always|Failure] [-enableeventreceivers yes|no] [-enablecompression yes|no]	Alters an existing Content Deployment Path (for content management); used with the publishing process.
stsadm **-o editssp** -title <SSP name> [-newtitle <new SSP name>] [-sspadminsite <administration site url>] [-ssplogin <username>] [-ssppassword <password>] [-indexserver <index server>] [-indexlocation <index file path>] [-setaccounts <process accounts (domain\username)>] [-ssl <yes|no>]	Alters the properties of an existing Shared Service Provider.
stsadm **-o email** -outsmtpserver <SMTP server> -fromaddress <someone@example.com> -replytoaddress <someone@example.com> -codepage <codepage> [-url <url>]	Sets the Outgoing e-mail server settings for the site at <URL> (usually the SMTP server of a web front end).
stsadm **-o enablecmsurlredirect** -profilename <profile name> -off	Enables or disables a CMS Migration Profile.
stsadm **-o enablessc** -url <url> [-requiresecondarycontact]	Enables Self-Service Site Creation for the specified SharePoint site.
stsadm **-o enumalternatedomains** -url <protocol://existing.WebApplication.URLdomain> -resourcename <non-web application resource name>	Lists the Alternate Access Mappings.
stsadm **-o enumcontentdbs** -url <url>	Lists the Content Databases attached to the Site at <URL>.
stsadm **-o enumdataconnectionfiledependants** -filename <filename for which to enumerate dependants>	Lists the Data Connection File Dependants.
stsadm **-o enumdataconnectionfiles** [-mode <a | u | all | unreferenced>]	Lists the Data Connection Files stored.
stsadm **-o enumdeployments**	Lists the current deployments in SharePoint.
stsadm **-o enumexemptuseragents**	Lists the Exempt User Agents in SharePoint.
stsadm **-o enumformtemplates**	Lists the installed Form Templates in SharePoint.
stsadm **-o enumgroups** -url <url>	Lists the Site Groups in SharePoint.
stsadm **-o enumroles** -url <url>	Lists the available roles for the site at <URL>.
stsadm **-o enumservices**	Lists the installed services.

Command Reference	Purpose
stsadm **-o enumsites** -url <virtual server url> -showlocks -redirectedsites	Lists the SharePoint sites in the current installation.
stsadm **-o enumsolutions**	Lists the installed solutions in SharePoint.
stsadm **-o enumssp** -title <SSP name> [-default | -parent | -all]	Lists the Shared Service Providers.
stsadm **-o enumssptimerjobs** -title <SSP Name>	Lists the current Timer Jobs (active).
stsadm **-o enumsubwebs** -url <url>	Lists the webs (sites) defined under the site at <URL>.
stsadm **-o enumtemplates** [-lcid <language>]	Lists the install templates (all or for the specified language)
stsadm **-o enumusers** -url <url>	Lists current users of the site at <URL>.
stsadm **-o enumwppacks** [-name <name of Web Part Package>] [-url <virtual server url>] [-farm] stsadm **-o enumzoneurls** -url <protocol://existing.WebApplication.URLdomain> -resourcename <non-web application resource name>	Lists the currently installed web parts (declining).
stsadm **-o execadmsvcjobs**	Starts execution of all Administration Service Jobs.
stsadm **-o export** -url <URL to be exported> -filename <export file name> [-overwrite] [-includeusersecurity] [-haltonwarning] [-haltonfatalerror] [-nologfile] [-versions <1-4> 1 - Last major version for files and list items (default) 2 - The current version, either last major or the last minor 3 - Last major & last minor version for files and list items 4 - All versions for files and list items] [-cabsize <integer from 1-1024 MB> (default: 25)] [-nofilecompression] [-quiet]	Exports a definition (list, site).

Command Reference	Purpose
stsadm **-o extendvs** -url <url> -ownerlogin <domain\name> -owneremail <someone@example.com> [-exclusivelyusentlm] [-ownername <display name>] [-databaseuser <database user>] [-databaseserver <database server>] [-databasename <database name>] [-databasepassword <database user password>] [-lcid <language>] [-sitetemplate <site template>] [-donotcreatesite] [-description <iis web site name>] [-sethostheader] [-apidname <app pool name>] [-apidtype <configurableid/NetworkService>] [-apidlogin <DOMAIN\name>] [-apidpwd <app pool password>] [-allowanonymous]	Extends a SharePoint application (Virtual Server)—this is the same as provisioning.
stsadm **-o extendvsinwebfarm** -url <url> -vsname <web application name> [-exclusivelyusentlm] [-apidname <app pool name>] [-apidtype <configurableid/NetworkService>] [-apidlogin <DOMAIN\name>] [-apidpwd <app pool password>] [-allowanonymous]	Extends a SharePoint application (Virtual Server) for a web farm—this is the same as provisioning.
stsadm **-o forcedeletelist (forcedeleteweb)** -url <url>	Forces delete of a list or a web (site).
stsadm **-o formtemplatequiescestatus** [-formid <form template ID>] [-filename <path to form template file>]	Returns the usage status of the selected Form Template.
stsadm **-o getadminport**	Returns the Central Administration Site Port number.
stsadm **-o getdataconnectionfileproperty** -filename <filename of the data connection file> -pn <property name>	Returns a Data Connection File property.
stsadm **-o getformsserviceproperty** -pn <option name>	Returns a Forms Service property.
stsadm **-o getformtemplateproperty** [-formid <form template ID>] [-filename <path to form template file>] -pn <property name>	Returns a Form Template property.

Command Reference	Purpose
stsadm **-o getproperty** -propertyname <property name> [-url <url>]	Returns a SharePoint Property (see list in next section).
stsadm -o getsitedirectoryscanschedule	Returns the site directory scan schedule (how often crawled). See Schedule Parameter below.
stsadm **-o getsitelock** -url <url>	Returns the current lock status for the site (read/write or both).
stsadm **-o geturlzone** -url <protocol://incoming.url.domain>	Returns the URL zone for the selected domain path.
stsadm **-o import** -url <URL to import to> -filename <import file name> [-includeusersecurity] [-haltonwarning] [-haltonfatalerror] [-nologfile] [-updateversions <1-3> 1 - Add new versions to the current file (default 2 - Overwrite the file & all its versions (delete then insert) 3 - Ignore the file if it exists on the destination] [-nofilecompression] [-quiet]	Imports a file into SharePoint.
stsadm **-o installfeature** {-filename <relative path to Feature.xml from system feature directory> | -name <feature folder>} [-force]	Installs a feature in SharePoint (must be activated to use).
stsadm **-o listloggginglevels [-showhidden]**	Lists the current logging levels set for SharePoint.
stsadm **-o listregisteredsecuritytrimmers** -ssp <ssp name>	Lists the registered security trimmers for the Shared Services Provider. The Business Data Catalog Security Trimmer allows custom security trimming of entity instances (search documents) that have been indexed by the Business Data Catalog before they are returned to the user.
stsadm **-o localupgradestatus**	Returns the Local Upgrade Status of the database (used in conversions only).

Command Reference	Purpose
stsadm **-o managepermissionpolicylevel** -url <url> -name <permission policy level name> [{ -add | -delete }] [-description <description>] [-siteadmin <true | false>] [-siteauditor <true | false>] [-grantpermissions <comma-separated list of perms>] [-denypermissions <comma-separated list of permissions>]	Alters a Permission Policy.
stsadm **-o migrateuser** -oldlogin <DOMAIN\name> -newlogin <DOMAIN\name> [-ignoresidhistory]	Migrates a user (converts login name).
stsadm **-o osearch** [-action <list|start|stop>] required parameters for 'start' (if not already set): role, farmcontactemail, service credentials [-f (suppress prompts)] [-role <Index|Query|IndexQuery>] [-farmcontactemail <email>] [-farmperformancelevel <Reduced|PartlyReduced|Maximum>] [-farmserviceaccount <DOMAIN\name> (service credentials)] [-farmservicepassword <password>] [-defaultindexlocation <directory>] [-propagationlocation <directory>]	Sets parameters for Office Search (these same settings can be found in the Shared Service Provider Administration Site).
stsadm **-o osearchdiacriticsensitive** -ssp <ssp name> [-setstatus <True|False>] [-noreset] [-force]	Turns diacritic (accent-sensitive) search on or off (usually off for English and on for languages such as Spanish and Romanian).
stsadm **-o preparetomove** {-ContentDB <DatabaseServer:DatabaseName> | -Site <URL>} [-OldContentDB <uniqueidentifier>] [-undo]	Prepares a content database for moving (used when changing SQL Servers).
stsadm **-o profilechangelog** -title <SSP Name> -daysofhistory <number of days> -generateanniversaries	Displays the User Profile Change log for the specific Shared Service Provider.
stsadm **-o profiledeletehandler** -type <Full Assembly Path>	Adds a profile delete handler (a custom assembly that handles processing of a deleted user profile).
stsadm **-o provisionservice -action <start/stop>** -servicetype <servicetype namespace or assembly qualified name if not SharePoint service)> [-servicename <servicename>]	Provisions (starts/stops) a selected service in SharePoint.

Command Reference	Purpose
`stsadm -o quiescefarm` `-maxduration <duration in minutes>`	Starts a Quiesce process that disables new logins and slowly stops operations in SharePoint (ideal when creating full backups).
`stsadm -o quiescefarmstatus`	Shows the current Quiesce status.

`stsadm -o quiesceformtemplate`
`[-formid <form template ID>]`
`[-filename <path to form template file>]`
`-maxduration <time in minutes>`
	Starts a Quiesce process on a specific Form Template.

`stsadm -o reconvertallformtemplates`	Reconverts all Form Templates (this was used for the BETA to Production conversion).
`stsadm -o refreshdms` `-url <url>`	No documentation available on this command.
`stsadm -o refreshsitedms` `-url <url>`	No documentation available on this command.

`stsadm -o registersecuritytrimmer`
`-ssp <ssp name>`
`-id <0-2147483647>`
`-typename <assembly qualified TypeName of ISecurityTrimmer implementation>`
`-rulepath <crawl rule URL>`
`[-configprops <name value pairs delimited by '~'>]`
	Registers a new Security Trimmer for the specified Shared Service Provider. Tthe Business Data Catalog Security Trimmer allows custom security trimming of entity instances (search documents) that have been indexed by the Business Data Catalog before they are returned to the user.

`stsadm -o registerwsswriter`	Registers a Windows SharePoint Services Writer (no additional information available).
`stsadm -o removedataconnectionfile` `-filename <filename to remove>`	Deletes a Data Connection File.

`stsadm -o removedrservice`
`-servicename <data retrieval service name>`
`-setting <data retrieval services setting>`
	Removes a Data Retrieval Service.

`stsadm -o remove-ecsfiletrustedlocation`
`-Ssp <SSP name>`
`-Location <URL|UNC>`
`-LocationType SharePoint|Unc|Http`
	Removes the specified Excel Services Trusted File Location.

Command Reference	Purpose
stsadm **-o remove-ecssafedataprovider** -Ssp <SSP name> -ID <data provider id> -Type Oledb|Odbc|OdbcDsn	Removes the specified data provider from the Shared Service Provider.
stsadm **-o remove-ecstrusteddataconnectionlibrary** -Ssp <SSP name> -Location <URL>	Removes a Trusted Data Connection Library from the Shared Service Provider.
stsadm **-o remove-ecsuserdefinedfunction** -Ssp <SSP name> -Assembly <strong name|file path> -AssemblyLocation GAC|File	Removes an Excel Services User Defined Function from the Shared Service Provider.
stsadm **-o removeexemptuseragent** -name <user-agent to receive InfoPath files instead of a Web page>	Removes an Exempt User Agent.
stsadm **-o removeformtemplate** [-formid <form template ID>] [-filename <path to form template file>]	Removes the specified Form Template.
stsadm **-o removesolutiondeploymentlock** [-server <server> [-allservers]	Removes a Solution Deployment lock (enables).
stsadm **-o renameserver** -oldservername <oldServerName> -newservername <newServerName>	Renames the SharePoint SQL Server.
stsadm **-o renameweb** -url <url> -newname <new subsite name>	Renames a SharePoint web (site).
For site collection restore: stsadm **-o restore** -url <url> -filename <filename> [-hostheaderwebapplicationurl <web application url>] [-overwrite] For catastrophic restore: stsadm **-o restore** -directory <UNC path> -restoremethod <overwrite | new> [-backupid <Id from backuphistory>] [-item <created path from tree>] [-percentage <integer between 1 and 100>] [-showtree] [-suppressprompt] [-username <username>] [-password <password>] [-newdatabaseserver <new database server name>] [-quiet]	Restores SharePoint from a backup.

Command Reference	Purpose
stsadm **-o restoressp** -title <SSP name> -url <Web application url> -ssplogin <username> -mysiteurl <MySite Web application url> -indexserver <index server> -indexlocation <index file path> [-keepindex] -sspdatabaseserver <SSP database server> -sspdatabasename <SSP database name> [-ssppassword <password>] [-sspsqlauthlogin <SQL username>] [-sspsqlauthpassword <SQL password>] [-searchdatabaseserver <search database server>] [-searchdatabasename <search database name>] [-searchsqlauthlogin <SQL username>] [-searchsqlauthpassword <SQL password>] [-ssl <yes|no>]	Restores a Shared Service Provider from a backup.
stsadm **-o retractsolution** -name <Solution name> [-url <virtual server url>] [-allcontenturls] [-time <time to remove at>] [-immediate] [-local] [-lcid <language>]	Retracts a solution that has been deployed (makes it unavailable and must be done before a solution can be deleted).
stsadm **-o retractwppack** -name <Web Part Package name> [-url <virtual server url>] [-time <time to retract at>] [-immediate] [-local] [-lcid <language>]	Retracts a Web Part that has been deployed.
stsadm **-o runcmsmigrationprofile** -profilename <profile name> [-skipanalyzer] [-onlyanalyzer] [-startover] [-migratesincetime <DateTime string>] [-migrationfolder <path>] [-exportonly] [-importonly] [-htmldiff <path>]	Runs a CMS (Content Management Server) Content Migration Profile.
stsadm **-o runcontentdeploymentjob** -jobname <name> [-wait yes|no] [-deploysincetime <datetime>] The <datetime> parameter should be in the format of: "MM/DD/YY HH:MM:SS"	Runs a content deployment job.
stsadm **-o scanforfeatures** [-solutionid <Id of Solution>] [-displayonly]	Lists sites where a particular solution has been deployed.

Command Reference	Purpose
stsadm **-o setadminport** -port <port> [-ssl] [-admapcreatenew] [-admapidname <app pool name>]	Sets the Central Administration site port number.
stsadm **-o setapppassword** -password <password>	Sets the application password.
stsadm **-o setbulkworkflowtaskprocessingschedule** -schedule <recurrence string>	Sets the Bulk Work Flow Task Processing Schedule, which is basically when workflow tasks will be processed. See Schedule Parameter below.
stsadm **-o setconfigdb** [-connect] -databaseserver <database server> [-databaseuser <database user>] [-databasepassword <database user password>] [-databasename <database name>] [-exclusivelyusentlm] [-farmuser] [-farmpassword] [-adcreation] [-addomain <Active Directory domain>] [-adou <Active Directory OU>]	Sets the configuration database name for the SharePoint Instance (usually only done when joining an existing configuration).
stsadm **-o setcontentdeploymentjobschedule** -jobname <name> -schedule <schedule>	Sets the schedule for automated content deployment. See Schedule Parameter below.
stsadm **-o setdataconnectionfileproperty** -filename <filename of the data connection file> -pn <property name> -pv <property value>	Sets a Data Connection file property.
stsadm **-o setdefaultssp** -title <SSP name>	Sets the specified Shared Service Provider as the default (all new sites are assigned to the default).
stsadm **-o set-ecsexternaldata** -Ssp <SSP name> [-ConnectionLifetime <time in seconds>] [-UnattendedServiceAccountName <account name>] [-UnattendedServiceAccountPassword <account password>]	Sets the connection information needed for an Excel Services External Data connection.
stsadm **-o set-ecsloadbalancing** -Ssp <SSP name> [-Scheme WorkbookURL|RoundRobin|Local] [-RetryInterval <time in seconds>]	Sets the Excel Services Load Balancing method and retry interval.
stsadm **-o set-ecsmemoryutilization** -Ssp <SSP name> [-MaxPrivateBytes <memory in MBytes>] [-MemoryCacheThreshold <percentage>] [-MaxUnusedObjectAge <time in minutes>]	Sets the memory and caching options for Excel Services processing.

Command Reference	Purpose
stsadm **-o set-ecssecurity** -Ssp <SSP name> [-FileAccessMethod UseImpersonation|UseFileAccessAccount] [-AccessModel Delegation|TrustedSubsystem] [-RequireEncryptedUserConnection False|True] [-AllowCrossDomainAccess True|False]	Sets the security for Excel Services.
stsadm **-o set-ecssessionmanagement** -Ssp <SSP name> [-MaxSessionsPerUser <number of sessions>]	Sets the session limits for Excel Services.
stsadm **-o set-ecsworkbookcache** -Ssp <SSP name> [-Location <local or UNC path>] [-MaxCacheSize <storage in Mbytes>] [-EnableCachingOfUnusedFiles True|False]	Sets the memory and caching options for Excel Workbooks.
stsadm **-o setformsserviceproperty** -pn <option name> -pv <option value>	Sets a Forms Service property.
stsadm **-o setformtemplateproperty** [-formid <form template ID>] [-filename <path to form template file>] -pn <property name> -pv <property value>	Sets a Form Template property.
stsadm **-o setholdschedule** -schedule <recurrence string>	Sets a hold schedule job processing. See Schedule Parameter below.
stsadm **-o setlogginglevel** [-category < [CategoryName | Manager:CategoryName [;...]] >] {-default | -tracelevel < None; Unexpected; Monitorable; High; Medium; Verbose; > [-windowsloggginglevel < None; ErrorServiceUnavailable; ErrorSecurityBreach; ErrorCritical; Error; Warning; FailureAudit; SuccessAudit; Information; Success; >] }	Sets the SharePoint logging level.
stsadm **-o setpolicyschedule** -schedule <recurrence string>	Sets a Policy Schedule (for example, different policies for day/night). See Schedule Parameter later on in table.

Command Reference	Purpose
stsadm **-o setproperty** -propertyname <property name> -propertyvalue <property value> [-url <url>]	Sets a SharePoint Property.
stsadm **-o setrecordsrepositoryschedule** -schedule <recurrence string>	Sets the Records Repository schedule (when records are processed). See Schedule Parameter later on in the table.
stsadm **-o setsearchandprocessschedule** -schedule <recurrence string>	Sets the Search and Process Schedule (when search will run). See Schedule Parameter later on in the table.
stsadm **-o setsharedwebserviceauthn** -ntlm | -negotiate	Sets the Shared Web Services authentication type (NTLM or negotiate, and NTLM is the default).
stsadm **-o setsitedirectoryscanschedule** -schedule <recurrence string>	Sets the Site Directory (search) Scan Schedule. See Schedule Parameter later on in the table.
stsadm **-o setsitelock** -url <url> -lock <none | noadditions | readonly | noaccess>	Sets a site lock for a specific SharePoint site.
stsadm **-o setsspport** -httpport <HTTP port number> -httpsport <HTTPS port number>	Sets the port number for a Shared Services Provider.
stsadm **-o setworkflowconfig** -url <url> {-emailtonopermissionparticipants <enable|disable> | -externalparticipants <enable|disable> | -userdefinedworkflows <enable|disable>}	Sets configuration settings for workflow.
stsadm **-o siteowner** -url <url> [-ownerlogin <DOMAIN\name>] [-secondarylogin <DOMAIN\name>]	Sets the site owner (and optional secondary owner) for a SharePoint site.

Command Reference	Purpose
stsadm **-o spsearch** [-action <list|start|stop|attachcontentdatabase| attachcontentdatabase|fullcrawlstart|fullcrawlstop>] [-f (suppress prompts)] [-farmperformancelevel <Reduced|PartlyReduced|Maximum>] [-farmserviceaccount <DOMAIN\name> (service credentials)] [-farmservicepassword <password>] [-farmcontentaccessaccount <DOMAIN\name>] [-farmcontentaccesspassword <password>] [-indexlocation <new index location>] [-databaseserver <server\instance>] [-databasename <database name>] [-sqlauthlogin <SQL authenticated database user>] [-sqlauthpassword <password>] -action list -action stop [-f (suppress prompts)] -action start -farmserviceaccount <DOMAIN\name> (service credentials) [-farmservicepassword <password>] -action attachcontentdatabase [-databaseserver <server\instance>] -databasename <content database name> [-searchserver <search server name>] -action detachcontentdatabase [-databaseserver <server\instance>] -databasename <content database name> [-f (suppress prompts)] -action fullcrawlstart -action fullcrawlstop	Starts the Search Service with optional parameters.
stsadm **-o spsearchdiacriticsensitive** [-setstatus <True|False>] [-noreset] [-force]	Enables or disables diacritic (accent) sensitive search.
stsadm **-o sync** {-ExcludeWebApps <web applications> | -SyncTiming <schedule(M/H/D:value)> | -SweepTiming <schedule(M/H/D:value)> | -ListOldDatabases <days> | -DeleteOldDatabases <days>}	Synchronizes SharePoint processing across servers. See Schedule Parameter later on in the table.

Command Reference	Purpose
stsadm **-o syncsolution** -name <Solution name>] [-lcid <language>] [-alllcids] stsadm **-o syncsolution** -allsolutions	Synchronizes solutions across servers.
stsadm **-o unextendvs** -url <url> [-deletecontent] [-deleteiissites]	Removes an application.
stsadm **-o uninstallfeature** {-filename <relative path to Feature.xml> | -name <feature folder> | -id <feature Id>} [-force]	Uninstalls a feature (must be deactivated first).
stsadm **-o unquiescefarm**	Un-quiesce SharePoint to make it available again.
stsadm **-o unquiesceformtemplate** [-formid <form template ID>] [-filename <path to form template file>]	Un-quiesce a Form Template to make it available again.
stsadm **-o unregistersecuritytrimmer** -ssp <ssp name> -id <0 - 2147483647>	Remove a Security Trimmer from a Shared Service Provider.
stsadm **-o unregisterwsswriter**	Removes a WSS Writer.
stsadm **-o updateaccountpassword** -userlogin <DOMAIN\name> -password <password> [-noadmin]	Updates an account password.
stsadm **-o updatealerttemplates** -url <url> [-filename <filename>] [-lcid <language>	Updates the Alert Templates for a specific site.
stsadm **-o updatefarmcredentials** [-identitytype <configurableid/NetworkService>] [-userlogin <DOMAIN\name>] [-password <password>] [-local [-keyonly]]	Adds a new user or group to the Server Farm Administrators group.
stsadm **-o upgrade** {-inplace | -sidebyside} [-url <url>] [-forceupgrade] [-quiet] [-farmuser <farm user>] [-farmpassword <farm user password>] [-reghost] [-sitelistpath <sites xml file>]	Upgrades a SharePoint site (used for Migration from V2 to V3).

Command Reference	Purpose

`stsadm -o upgradeformtemplate`
`-filename <path to form template file>`
`[-upgradetype <upgrade type>]`
	Upgrades an installed Form Template.

`stsadm -o upgradesolution`
`-name <Solution name>`
`-filename <upgrade filename>`
`[-time <time to upgrade at>]`
`[-immediate]`
`[-local]`
`[-allowgacdeployment]`
`[-allowcaspolicies]`
`[-lcid <language>]`
	Upgrades an installed solution.

`stsadm -o upgradetargetwebapplication`
`-url <URL to upgrade>`
`-relocationurl <new URL for non-upgraded content>`
`-apidname <new app pool name>`
`[-apidtype <configurableid/NetworkService>]`
`[-apidlogin <DOMAIN\name>]`
`[-apidpwd <app pool password>]`
`[-exclusivelyusentlm]`
	Upgrades a web application (used for migration from V2 to V3).

`stsadm -o uploadformtemplate` `-filename <path to form template file>`	Adds a new form template to SharePoint.

`stsadm -o userrole`
`-url <url>`
`-userlogin <DOMAIN\name>`
`-role <role name>`
`[-add]`
`[-delete]`
	Adds or removes a User Role from the SharePoint site.

`stsadm -o verifyformtemplate` `-filename <path to form template file>`	Validates the Form Template (done before uploading).

Schedule Parameter

In several of the commands above, a schedule parameter is required that sets the schedule for the particular option. These are verbose ways of scheduling, as shown in the following examples:

```
Schedule Parameter example values:
       "every 5 minutes between 0 and 59"
       "hourly between 0 and 59"
       "daily at 15:00:00"
       "weekly between Fri 22:00:00 and Sun 06:00:00"
       "monthly at 15 15:00:00"
       "yearly at Jan 1 15:00:00"
```

Extending the STSAdm Tool

The STSAdm tool provides you with most everything you want to do with SharePoint, but what if you want to do something special, say, implement your own backup processing? Also, what if you needed this to be a class so that it could be accessible elsewhere? The answer is extending the STSAdm command-line tool. This happens to be an extensible tool (like everything else in SharePoint), and it enables you to add your own custom command in just a few steps! The magic happens in a set of configuration files under the SharePoint Hive and your own custom class. Rather than wasting extra time writing about it, you're going to jump right in and create an extension; your command in this case will simply check the existence of a site showing how to do both the extension and how the class can access the SharePoint Object Model.

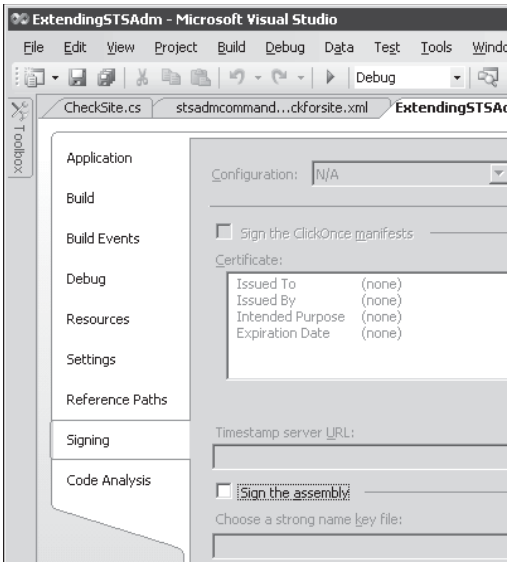
1. Open Visual Studio 2005 and create a new Class Library project called **ExtendingSTSAdm**.
2. When the project opens, in the Solution Explorer, rename the file from Class1.cs to **CheckSite.cs.**
3. Add the following code (I've noted the additional namespaces needed; again, this is C# so VB folks will have to translate). Be sure to add a project reference to SharePoint:

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Collections.Specialized;   // Need for StringDictionary
using Microsoft.SharePoint;             // Need for SharePoint Obj Model
using Microsoft.SharePoint.StsAdmin;    // Need for ISPStsadmCommand

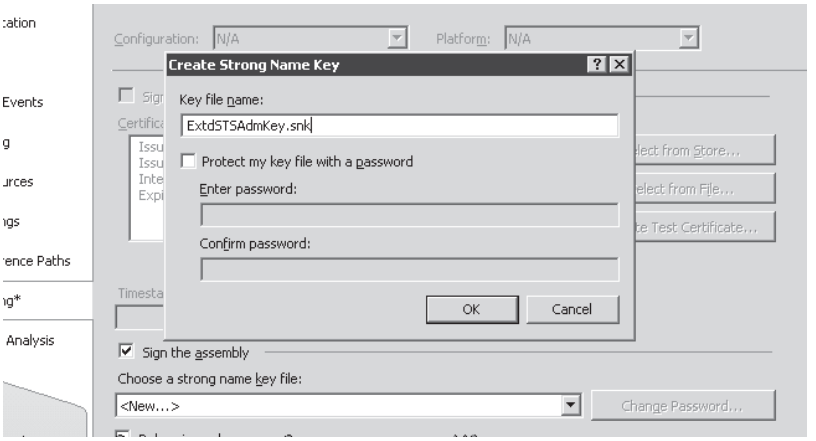
namespace ExtendingSTSAdm
{
    // To wrap us in the STSAdm tool, you inherit from the
    // ISPStsadmCommand class; this has only two methods:
    //   GetHelpMessage returns a message when the users enters
    //      stsadm -help <Command>
    //   Run executes the actual functionality
    public class CheckSite : ISPStsadmCommand
    {
        public string GetHelpMessage(string CommandPassed)
        {
            string ReturnMessage = "stsadm -o CheckForSite -url <-url>\r\n";
            return ReturnMessage;
        }
        public int Run(string command,
           System.Collections.Specialized.StringDictionary keysPassed,
           out string output)
        {
            output = "";
            string TestURL = "";
            bool SiteFound = false;
            SPSite Check4SiteSite = null;
            SPWeb TopSite = null;
            //
```

```
            if (keysPassed.ContainsKey("url"))
            {
                TestURL = (string)keysPassed["url"];
            }
            if (keysPassed.ContainsKey("URL"))
            {
                TestURL = (string)keysPassed["URL"];
            }
            // If URL was passed, check it or return error msg:
            if (TestURL.Trim() != "" && TestURL.Trim() != null)
            {
                try
                {
                    Check4SiteSite = new SPSite(TestURL);
                    TopSite = Check4SiteSite.OpenWeb();
                    SiteFound = true;
                }
                catch (Exception ErrInSite)
                {
                    SiteFound = false;
                }
            }
            if (SiteFound) { output = "Found " + TopSite.Title.ToString(); }
            if (!SiteFound) { output = "No site found at url: " + TestURL.ToString(); }
            return 1;
        }
    }
}
```

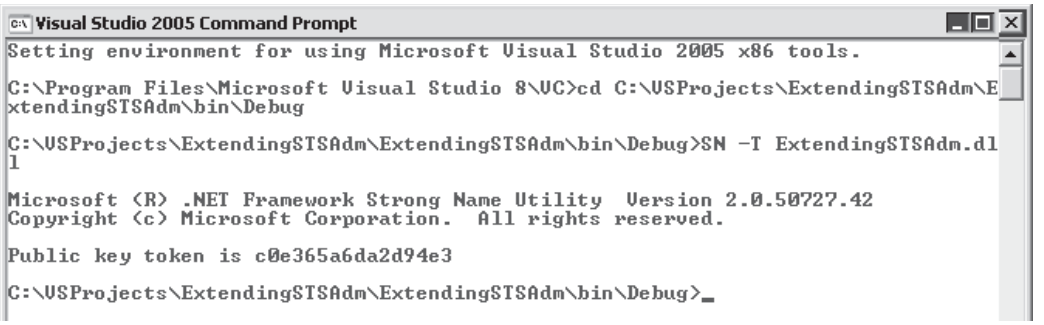
4. Compile the project to be sure there are no typos or references needed.
5. Right-click on the project name in the Solution Explorer and select Properties.
6. Any Assembly to be used by STSAdm must be strongly named, so click the Signing Tab, which should look like this by default:



7. Click Sign This Assembly, and in the drop-down menu, select <New…> to create a new key.
8. In the Create New Key dialog, enter **ExtdSTSAdmKey.snk** and uncheck Password protected:



9. Build the project again.
10. From Windows, open a command window via Start | All Programs | Microsoft Visual Studio 2005 | Visual Studio Tools | Visual Studio Command Prompt, change your directory to your project folder and then the folder bin\debug or bin\release (depending on how you built the project—wherever the DLL was created when you compiled), and get the public key token ID using the SN utility by entering in the command **sn -T ExtendingSTSAdm.dll**:



11. Record the ID returned.
12. Next, any assembly to be used with STSAdm must be loaded into the GAC (Global Assembly Cache) using the GACUtil utility; in this case (while still in the command window and folder), enter:

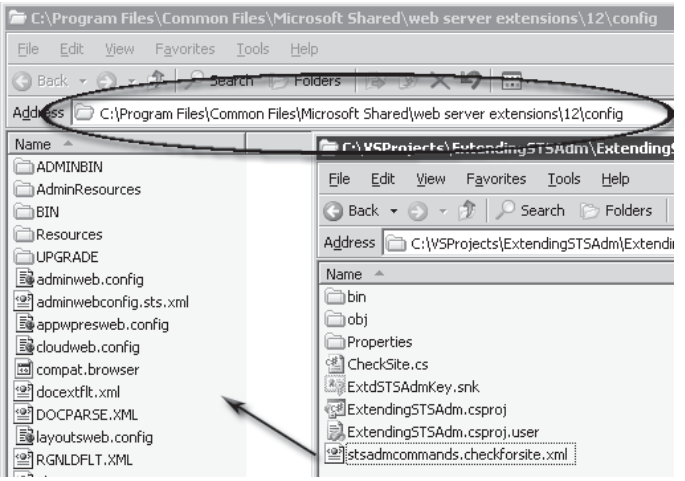
```
gacutil /i ExtendingSTSAdm.dll
```

13. You should see a message that it was successfully installed.

14. Back in Visual Studio, you want to create an STSAdm command Xml file (this will be read in by STSAdm when run), so select File | New | File, and on the New File dialog, click General in the left-hand pane and Xml File in the right, and then click the Open button and enter in the Xml as follows (keep the command entry on one line as I've done here and be sure to use your public key token, not this one):

```
<?xml version="1.0" encoding="utf-8"?>
<commands>
      <command name="checkforsite"
 class="ExtendingSTSAdm.CheckSite,ExtendingSTSAdm,
Version=1.0.0.0,Culture=neutral,PublicKeyToken=c0e365a6da2d94e3"/>
</commands>
```

15. The file name format for STSAdm command files is stsadmcommands.<name>.xml. Save this file in your project under the name of stsadmcommands.checkforsite.xml (notice that the name matches the command name shown in the Xml command file shown previously).
16. Finish the installation by copying the stsadmcommands.checkforsite.xml file to the SharePoint Hive Config folder.
17. You can now test your command as shown:



APPENDIX B

Setting Up Application Authentication

Delivering something new in this release, Windows SharePoint Services 3.0 now supports most kinds of user authentication methods available, including Windows-based authentication services (Windows Integrated, Active Directory), non-Windows services (LDAP), and Forms-based authentication. In fact, SharePoint can now even support more than one source. The primary methods fall into three groups:

- **Windows authentication** Includes Anonymous, Basic Authentication, Digest Authentication, NTLM, Kerberos, and Certificates.
- **Forms-based authentication** Includes any LDAP service or database, such as SQL Server or Oracle.
- **Web Single Sign-on (SSO)** Provides off-box forms authentication through federated services (ADFS); this applies to third-party systems.

NOTE *Regardless of what kind of authentication is used, the accounts used to run SharePoint (SPServices, SPSearch, and so on) must be either Windows Domain accounts or local machine accounts.*

While these methods are supported by SharePoint, it is important to know that SharePoint has nothing to do with user authentication. Information provided by some of these sources, like Active Directory, are used to maintain SharePoint user profiles, but the actual job of authenticating (including storing passwords and so on) is maintained by the authenticating source. This is done natively through the Windows network (NTLM/Kerberos), remotely to a third-party source through Active Directory Federation Services (ADFS), or to IIS via Forms authentication. Because of this, all of the types of authentication that would be available to any .NET application are now available in SharePoint.

On this same subject, it is important not to confuse user authentication with user site security, which *is* maintained by SharePoint. For example, a domain administrator in Active Directory or even a local system administrator may have a high level of access to the servers, but in SharePoint they are just another user and are granted only the rights permitted by SharePoint. This is a bit of a shift for those familiar with the previous versions, in which a local administrator had full rights by default.

Scope of Authentication

Since the SharePoint architecture supports any collection of sites for any purpose, user authentication is managed at the application level. From a technical perspective, a SharePoint application plays the same role as a virtual directory does in IIS, as in reality, it is the same thing, just managed by SharePoint instead of IIS. This setup enables a single SharePoint instance to host one application as a Publishing Site to support external users authenticating using forms and another application to support a Collaboration Site for internal users using NTLM.

This granularity is not by happenstance; by providing application-level authorization, the SharePoint platform can support a fully hosted environment giving individual subscribers with complete stand-alone environments complete data security, while allowing them to utilize Shared Services. By the same token, an individual company can provide hosted customer sites, each utilizing the customer's own authentication method and eliminating the need to create or maintain external user accounts.

In addition to being set at the application level, different authentication methods can be assigned to specific zones. For example, the same application could use Windows as the default provider and then use an external provider for Internet access. This would be a typical setup to allow intranet users direct (no login) access to the site, yet permit extranet users (such as partners or vendors) to authenticate against their own identity management system.

Selecting the Authentication Method

Of the methods supported, there are obviously advantages and disadvantages between them. The basic choice depends on your network, how your internal users authenticate, any third-party authentication providers you will be using, and what is likely most important, the kinds of application(s) you will use with SharePoint.

Authentication can be a fairly tricky subject, and there are a lot of environmental issues in large enterprise environments. There are a number of external variables that must be accounted for, including VPNs, so I can't hope to cover every scenario. However, Table B-1 is a general guide that will apply to most environments.

Of course, the table is a generalization—you can (and most likely will) have multiple configurations to meet your needs. As a general rule for the most functionality in SharePoint, Windows authentication via NTLM or Kerberos is the best.

The biggest decision you will have to make is what you wish to sacrifice in terms of user functionality. Obviously, Windows authentication provides the most and Web SSO provides the least; this is because though authenticated through Forms or SSO, SharePoint does not have the cookie it needs to provide personalization. This means users can only be permitted role-based security. The key usability features include:

- **Targeting** This is the ability to push content to a specific user role.
- **Role-based security** This is the ability to assign group-level permissions.
- **Personalization** This is the use of My Sites, My Tasks, and so on.
- **Client Integration** This is the integration of client applications like Word to launch from SharePoint.

Method	Application Type	Advantages	Disadvantages
Windows (includes: Anon/Basic, Digest, Certificates, NTLM, Kerberos)	—Team Sites —Workspaces —Project Sites —Internal Blogs —Internal WIKI Sites —Collaboration Sites —SInternal Publishing Sites	—Uses local Active Directory —No development required	—Some browsers have problems with IIS authentication protocols —Not good for external users —Impractical for Internet
Forms	—Public blogs —Public WIKI sites —Mixed-use Collaboration Sites —Publishing Sites —Anonymous read-only sites —Customer/partner sites	—Can use any database —Can target content by roles —Unlimited internal/external users	—Requires custom development —Users are by role and do not have individual security in SharePoint
Web SSO	—Extranet blogs —Extranet WIKI sites —Mixed use Collaboration Sites —Publishing Sites —Customer/partner sites	—Utilizes ADFS —Can utilize a non-AD source —Can use more than one source	—Requires SSL for ADFS —Requires a Federated Authentication System (ISA or similar)

Table B-1 Authentication Methods, Site Types, and Requirements

Of course, you can assign more than one method to an application to get the best of both worlds by using zones, but it is the way a given user authenticates that dictates what is available to them, and they can't have it both ways. The features compromise by the method shown in Table B-2.

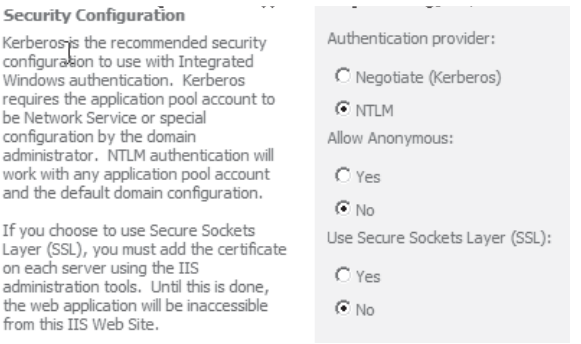
Method	Targeting	Role-Based Security	Personalization	Client Integration*
Windows	Yes	Yes	Yes	Yes
Forms	Yes	Yes	No	Maybe**
Web SSO	Yes	Yes	No	No

*** Client integration determines whether a user can open a document within SharePoint for editing; if it's not available, users can work locally with documents and upload them to SharePoint.**
**** Forms authentication does not always play well with client integration, often ending up with mixed results (much of this depends on the browser in use; for some users, it will work fine, with others, it will cause problems, like constantly prompting for login).**

Table B-2 Features Available by Authentication Type

Choosing the Authentication Method on Create or Extend

As mentioned previously, setting up authentication is done on an application by application basis, so the initial security settings are set when an application is first created or extended (you will see this below). On the Application Settings Page, there is a section for security as shown:



As you can see, there is not much there. You can only set the basics for Windows-based authentication. Fortunately for most, these will be the only settings you need to make; however, if you intend to use Forms or Web SSO, you'll have to accept the defaults here and modify the authentication settings after the application is created; you can easily go back and modify them from the Application Management Page in Central Administration.

Note that in order to set up authentication methods, the account you use to log in to SharePoint must be a member of the SharePoint Farm Administrators Group. If you intend to use an external identity management system, the account you use must have the necessary administrative rights for it.

Setting Non-Windows-Based Authentication Methods

Methods for setting up Forms authentication, external LDAP Services, and Windows Single Sign-on are explicitly documented in the SharePoint Software Development Kit and in the product installation guide. Since the processes are subject to change, it is best to get these directly from Microsoft. One of the best sources for a step-by-step guide on the actual installation process can be found on the Microsoft TechNet site (enter this all on one line):

```
http://technet2.microsoft.com/windowsserver/WSS
/en/library/b6bc8fec-c11c-4ed7-a78d-3ad61c7ef6c01033.mspx?mfr=true.
```

INDEX

A

AAM (Alternate Access Mapping), 179–180, 227–228
Access, 567
accessing
  database and query, 238
  pages, 365
  Sites, 149
  SSP failures in, 136–137
  web part connections, 260
  web part properties, 408
accounts
  adding memberships to existing, 292–293
  adding SharePoint Installation, 69–70
  Administrative Groups and Users OU, 66–67
  application pool, 110
  configuring SharePoint from SPServices, 128
  correcting access issues for, 149
  determining mode of, 40
  enabling local log on, 76–77
  establishing before installation, 63
  failures in SSP, 138–139
  granting domain rights for, 67
  password policies for, 63, 65–66
  required for creating SSP, 154
  retrieving impersonation information from Registry, 615–618
  setting up SQL Server, 80–81
  SharePoint Search, 175
  signing up for Single Sign-On service, 217–218
  supported modes for Active Directory user, 43–44
  types of SharePoint, 64–65
actions
  defined, 581–582
  setting up, 587–589
  types of Core, List, and Task, 582–583
Active Directory
  account installation checklist, 59
  authentication handled by, 21
  defining OU for, 62–63
  development system installations of, 75–76
  installing e-mail services after, 77–78
  issues to avoid for, 54
  modifying user profile import path to, 164–167
  user account modes supported by, 43–44
ADACM (Active Directory Account Creation Mode), 43, 44
administration. *See also* administrators group; Central Administration Site; Site Administration
  about Site Administration, 197, 198
  levels of Site, 285
  overview of Central Administration, 197–198
  required after Advanced installations, 107–108, 116
Administration Object Model, 638
administrators
  adding or removing from Site Collections, 258–259
  adding or removing Site, 297
administrators group
  adding SharePoint Installation account to, 69–70
  creating Administrative Groups and Users OU account, 66–67
  designating SharePoint Server, 104–105
  Farm Administrators group, 214–206, 744
  rights of Administrator Site groups, 289
Advanced installations. *See also* installing SharePoint Server; post-install configuration
  Central Administration features for, 198–199
  choosing, 97

completing, 103–104
confirming settings for, 102–103
connecting to server farm, 99–101
creating first application for WSS, 148–149
designating SharePoint Server Administrator's group, 104–105
features of, 96–97
overview, 39, 53, 86
proxy server settings, 104
selecting authentication method in, 101
setting file installation location, 98
steps for, 97–105, 109–110
tasks required after, 107–108, 116
Advanced Web Part properties, 407
alerts, 451
Alternate Access Mapping (AAM), 179–180, 227–228
antivirus protection, 186–187, 212–213
API. *See* Object Model
_app_bin directory, 669
appearance properties for web part, 406
application pools
adding to SQL, 110
adjusting after installation, 92–93
setting up accounts for, 211
Application Programming Interface. *See* Object Model
application servers, 15, 60
application templates. *See* Site Definitions
applications. *See* creating applications and Shared Services; and specific applications
Applications Management page (Central Administration Site), 241–283
Application Security section, 242, 260–264
Authentication Providers option, 263–264
Check Services Enabled in this Farm option, 276
Configure Connection to Records Center option, 264–265
Configure InfoPath Forms Services option, 271–272
Configure Session State option, 276–277
Create or Configure This Farm's Shared Services option, 275
Create or Extend Web Application option, 242–246
Create Site Collection option, 254–255
Define Managed Paths option, 248
Delete Site Collection option, 255–256
Delete Web Application option, 247
Document Conversions option, 266–267
External Service Connections section, 242, 264–267
Grant or Configure Shared Services Between Farms option, 275–276
HTML Viewer option, 265–266
InfoPath Forms Services option, 242, 270–275
Manage Content Databases option, 251–252
Manage Data Connection Files option, 274
Manage Form Templates option, 270–271
Manage the Web Service Proxy option, 274–275
Manage this Farm's Shared Services, 278–280
Manage Web Application Features option, 252–253
Office Search option, 268–270
Office SharePoint Server Shared Services option, 242, 275–277
opening, 241–242
Policy for Web Application option, 262–263

Quota Templates option, 257
Remove SharePoint From IIS Web Site, 246
Security For Web Part Pages option, 260
Self-Service Site Management option, 260–261
SharePoint Site Management section, 242, 254–259
SharePoint Web Application Management section of, 242–254
Site Collection Administrators option, 258–259
Site Collection List option, 259
Site Collection Quotas and Locks option, 258
Site Use Confirmation and Deletion option, 256
Upload Form Template option, 272–274
User Permissions for Web Application option, 261–262
Web Application General Settings option, 250–251
Web Application List option, 253–254
Web Application Outgoing E-Mail Settings option, 249
WorkFlow Management settings, 242, 277–283
Workflow Settings option, 277–278, 569
Approval dialog (Office applications), 595
Approval workflows
attaching, 570–573
creating in Office applications, 595–596
defined, 569
approving page content, 481
architecture, 11–26
Business Data Catalog Service, 23–24
Central Administration Site, 14–15, 16
core Microsoft services of, 13–15
developing for portal methodology, 33–34
devising formal portal, 32
Document Conversion Service, 24–25
Excel Web Services, 22–23
Features and Solutions Framework, 18–19
implementation of WSS, 5–6
influences from other technologies on, 11–12
InfoPath Forms Services, 25
Office Search Services, 21–22
personalization features, 19–21
security independent of Site, 613
services provided by other Microsoft products, 21
Shared Services Administration Site, 16
SharePoint server roles, 15
Sites and pages, 16–18
WSS view of SharePoint, 12–13
areas, 5
.asmx web services, 629–630
ASP.NET. *See also* Master Pages
aspx pages for SharePoint Sites, 517
Forms authentication, 43
Master Pages and Page Layouts in, 16–18, 476, 554–555
standard Master Pages controls in, 482–483
structure of Page Layouts in, 483
aspx files
aspx pages for SharePoint Sites, 517
displaying aspx page templates, 521, 522
Master Pages and Page Layouts, 554–555

assemblies
getting information for .webpart files, 695–696
identifying safe, 671–672
locating in bin directory, 669
referencing Object Model, 612–613
signing, 684–685
Attach to Process window (Visual Studio), 660, 675–676
attachments
enabling/disabling list, 379
options for library, 381
audiences, 177–179
auditing
audit log report, 329
audit settings, 328
implementing Information Management Policies with, 536
overview, 472
policies for, 329
authentication. *See also* Forms authentication; Windows authentication
choosing method of, 40, 42–43, 59, 101, 742–744
designating providers for, 263–264
InfoPath Forms Service configuration for, 272
methods of, 741
Mixed Mode vs. Windows, 80
non-Windows-based, 744
scope of, 742
setting for new web applications, 244
SSO, 43, 741, 742–743
user account modes with Active Directory, 43–44
authoring
role in content deployment, 502
Smart Client, 492–493
Web Content, 489
automating content deployment, 505–506

B

Backup and Restore options (Central Administration Site), 231–237
backups, 231–237
making SQL Server, 232
methods for, 233
performing, 231–233
planning system, 48
restoring from, 234–236
SSP restored from, 283
status of current, 236–237
types of, 231–232
viewing history of, 234
barcodes
configuring, 215
enabling, 381
implementing Information Management Policies with, 536
removing, 329, 330
Base Activity Library, 565
Basic installations. *See also* installing SharePoint Server; post-install configuration
adjusting application pools after, 92–93
changing authentication modes not allowed for, 53
correcting SQL reports after WSS install, 93–96
directories and files with, 90–91
features of Central Administration for, 198–199
overview, 39, 53, 86–87
running SharePoint Products and Technologies Configuration Wizard, 87–89
single servers using, 45
steps for, 87–96, 104, 108–109, 115, 116
tasks required after, 107–108
uninstalling SharePoint from, 195
Basic pages, 364, 365–367

BDC (Business Data Catalog) Service, 23–24
bin directory, 669
blocking file types, 213
Blog Sites
  about, 526–528
  creating posts from applications, 528–529
  template for, 522
blogs
  My Blog Web Part for, 446, 447
  posting to, 526, 527–529
  Recent Blog Posts Web Part, 443, 447, 450
Boolean properties, 694
branding
  business analysis and, 30, 32, 33
  Master Pages and, 476–477
  types of modifications for, 553–554
building SharePoint Designer workflows, 581–594
  about steps, actions, and conditions, 581–583
  creating workflow steps, 586
  defining actions, 587–589
  editing workflow, 594
  listing conditions for steps, 586–587
  preparatory steps, 584
  verifying and publishing workflow, 590–594
  workflow design considerations, 583–584
business considerations, 27–37
  branding and business analysis, 30, 32, 33
  creating portal deployment methodology, 30–37
  developing feature function matrix, 37
  governance considerations, 28–29
  tips for implementing SharePoint Server, 28
Business Data Catalog (BDC) Service, 23–24
business sponsors, 28

C

CAB files, 664, 665
caches. *See also* page caching
  configuring page, 510–514
  defined, 507
  disabling for volatile Sites, 511
  enabling disk-based, 510
  object, 332, 513–514
  page, 319–320, 330
  page item, 509, 513–514
  policies for, 509, 514–515
  profiles for, 509–510, 512
  setting up Site Collection output, 511–512
  Site Collection profiles for, 330–331
  types of SharePoint, 508–509
  user, 330
CAL (Client Access Licenses), 2
calendar modifications, 313
CAML (Collaborative Application Markup Language), 518, 552, 624
Cascading Style Sheets. *See* CSS
CatalogIconImageUrl property, 692
categories
  creating XSL-styled list view of, 426, 440
  listing sub sites by, 426, 439–440
  web part, 405
Categories Web Part, 426, 439–440
Central Administration Site. *See also* Applications Management page; Operations page
  about, 197–198
  application security, 260–264
  Applications Management in, 241–242
  architecture of, 14–15, 16

backing up and restoring, 231–237
configuring settings during SharePoint install, 96
connection to records center site, 264–265
construction of, 285
content deployment options, 200, 238–241, 502–503
data configuration for, 237–238
global configuration options, 224–230
Home page features for, 199–200
illustrated, 198
installed features of, 198–199
Operations page of, 200–201
organization of, 113–114
post-install tasks listed on, 107–108, 114–116
Shared Services, 275–277
SharePoint Site Management, 242, 254–259
unable to start, 111
workflow settings for MOSS, 277–283, 569
Central Records Management Site, 537
checklists for installation, 58–61
child sites. *See* sub sites
ChromeState property, 690
ChromeType property, 690
classes
about web part, 663, 665
adding code for web part solution, 677–683
adding code to web part, 670–671
renaming web part, 668–669
Client Access Licenses (CAL), 2
client applications, creating published links to Office, 171–172
client integration for authentication, 742, 743
clustering, 60
Code Access Security, 601
Collaboration Portal Site Definition, 530, 531
Collaboration Site Type, 519, 557
Collaborative Application Markup Language (CAML), 518, 552, 624
Colleague Tracker Web Part, 447, 448
Colleagues Web Part, 447, 449
Collect Feedback workflows, 570
Collect Signature workflows, 570
columns. *See also* list columns; Site columns
adding from Site columns, 385–386, 455–456
creating list, 383–384
Dublin Core, 468
editing list item, 385
grouping by content type, 386–387
indexed, 386
list, 376, 377, 623
modifying Site, 306, 460
ordering, 386
translatable, 334
command-line tool. *See* STSAdm
commands for STSAdm command-line tool, 715–736
communications between servers and with SQL Server, 42
conditions in SharePoint Designer
defined, 581, 582
defining for steps, 586–587
listing of, 582
confidentiality of content, 29
Configurable option (Properties window), 93
Configuration Successful window, 89
Configure Your Server Wizard, 72
configuring servers, 44–50
medium server farm, 46–47
server farm roles, 44–45
single servers, 45
small server farm, 45–46

connections
between Central Administration and Shared Services, 275–277
configuring to records center Sites, 264–265
establishing between web parts, 260, 408, 411–412
to library and extracting binary file, 627–628
to lists, 622–623
making Web Service, 630–632
from portal Sites, 328
to server farm, 99–101
to SharePoint Sites, 621
Contact Details Web Part, 426, 441
content. *See also* content deployment; content management
adding with Content Editor Web Part, 413, 414–415
approving page, 481
checking status of deployment, 241
configuring MOSS Site, 358–360
creating custom user queries of, 427–429
deleting content databases, 247
deploying to server farm, 240–241
deployment paths for, 239–240, 502, 503–504
displaying with XML Web Part, 413, 423–424
expiration of, 536
guidelines for, 29
managing, 320–321
migrating from MCMS to MOSS, 221
publishing approval workflow for MOSS, 372
references to stored, 299
search engine indexing of, 316
storage of WSS, 521–522
types of, 307, 459–461
using core web parts for, 412–413
viewing modification logs to, 321
working with databases containing, 251–252
content deployment
automatic, 505–506
Central Administration options for, 200, 238–241, 502–503
checking status of, 241
creating jobs for, 504
overview of, 502
to server farm, 240–241
setting paths for, 239–240, 502, 503–504
tools for, 501
Content Editor Web Part, 413, 414–415
content management. *See also* content deployment; ECM; MCMS; Publishing pages
branding with Master Pages, 476–477
building Page Layouts, 485–489
converting Word document to SharePoint page, 493–497
custom field controls, 506–507
developing vary-by-custom handlers for, 515
modifying toolbars, 491–492
page caching and custom field controls, 507–515
reusable columns and, 453–454
Smart Client Authoring, 492–493
tools for content deployment, 501
updating converted documents, 497
variations, 497–501
Web Content Authoring process, 489
working with Site Content Types, 459–465
content query cache, 509
Content Query Web Part, 427–429
content source, 643
copyrighted information, 29
Crawler Impact Rules, 269
crawling, 635

Create Site page, 346–347, 348–349
creating applications and Shared Services, 128–149
creating MOSS SSPs, 134–135
enabling SSPs, 130–131
process for creating applications, 131–134
setting up first Site, 143–148
SSPs, 128–130
cross-referencing links in Site Directory, 226–227
CSS (Cascading Style Sheets)
changes overriding Site theme settings, 303
setting URLs to alternate, 298
custom field controls, 506–515
custom search scripts, 639–642
custom search tools, 638
Customer Experience Improvement Program, 218
customizing
forms with Form Web Part, 413, 416–417
Page Layouts, 557
policies, 537–538
Search Scopes in MOSS, 322
workflows, 572–573

D

data. *See also* lists
creating and editing list views of, 387–391
creating relationships between documents and libraries for, 399
ECS settings for external, 156
propagating index, 636
setting up trusted file location external, 157–158
SharePoint Designer variable, 582
storage for SharePoint, 7
data connection files, 274
databases
access and queries for, 238
Active Directory use of, 62–63
backup methods for, 233
creating for Visual Studio personalization, 84–85
deleting content, 247
designating for new web application, 244
issues to avoid for, 54
managing content, 251–252
setting default server for, 237–238
specifying instance and port number for SQL, 110–111
specifying server farm configuration, 99–100
storage of Master Pages and Page Layouts in, 518
DCMI (Dublin Core Metadata Initiative), 468
.DDF files
about, 663–664, 665
creating for web part solutions, 699–700
role in creating WSP for solution, 701
debugging
event receivers, 659–661
one-off web parts, 675–676
web parts, 707
default MOSS web parts, 425
default permission groups, 290
default SSPs, 280–281
default.master Master Page, 557
Define Your New Workflow page (SharePoint Designer), 584–585
deleting
content databases, 247
content types, 307, 463
current Site, 318–319
current Sites, 318–319
event receivers, 661–662
links from Top Link Bar, 303

list views, 376
remnants after SQL Server Service Pack installs, 83
Site Collections, 255–256
Site columns, 308
Site Content Types, 307, 463
unused Sites, 256
users or groups from Site Collection, 293–294
web applications, 139–140, 247
WSS Master Pages, 306–307
deploying. *See also* content deployment; portal deployment methodology
event receivers, 657–658
Features, 607–609
web part DLL to Site directory, 669–670
web part solutions, 701–702
Description property, 689
development systems
AD, DNS, and DHCP installations for, 75–77
adding SharePoint Extensions for Visual Studio to, 105–106
deployment methodology for, 35, 36
e-mail service installations, 77–78
enabling Allow Log On Locally option for, 76–77
including test user accounts for, 67
installing Office, 86
special operating system installation steps for, 70–72
tools needed for, 51
Visual Studio installation for, 83–85
DHCP (Dynamic Host Configuration Protocol), 75–76
diagnostic logging settings, 182–183
DIP (Document Information Panel), 458, 463
Direction property, 693
directories
bin and _app_bin, 669
created during Basic installation, 90–91
deploying web part DLL to Site, 669–670
including or excluding SharePoint control for folders, 248
specifying settings for Site, 326
disaster recovery
backup and planning for, 48
configuring stand-by servers for, 110
large server farms, 47–48
medium server farm, 47
planning for, 40
single server, 45
small server farm, 45–46
disk-based cache
about, 508
enabling, 510
resetting, 332, 514
Disposition Approval workflows, 570
distribution groups, 207–210
DMZs and network planning, 41, 42
DNS
development system installations of, 75–76
importance of planning, 41
Document Center Site Template, 531–532
document conversion
architecture of, 24–25
Central Administration Site, 266–267
enabling/disabling for Site content type, 465
handling errors for, 125–126
managing from Central Administration, 203
overview of, 493
starting, 124–125
updating converted documents, 497
used with Smart Client Authoring, 492–493

from Word to SharePoint page, 493–497
Document Conversion Launcher Service, 203
Document Conversion Load Balancer Service, 124–126, 203
Document Conversion Processing feature, 537
Document Information Panel (DIP), 458, 463
document libraries
applying Information Management Policy to, 541
building project to check document status, 650–657
concept of folders vs., 399, 400
creating custom user queries of, 427–429
list columns in, 382–383
notifying document creator with event receiver, 650
Shared Documents library, 364, 365
Document Policies feature, 537
Document Workspace Sites, 361–363, 522, 545–546
documents. *See also* document conversion
Approval workflows for, 569–570
automating deployment of, 505–506
converting from Word to SharePoint, 493–497
creating summary lists of, 424, 426–427
folders vs. document libraries, 399, 400
handling and storing, 538–539
links between workflow and, 574–575
list of relevant, 420–421
managing with Records Center and Information Management Policies, 536
migrating to SharePoint, 401
project to check status of, 650–657
restoring from Recycle Bin, 325–326
storing with libraries, 398–399
updating converted, 497
Domain Account Mode, 43, 44
domain accounts, 61–67
Administrative Groups and Users OU creation, 66–67
creating before installation, 63
defining Active Directory OU, 62–63
enabling Allow Log On Locally option for development, 76–77
granting domain rights, 67
mailbox creation for, 78
password policies for, 63, 65–66
domains
installing e-mail services after joining, 77–78
joining existing, 69
downloading
application templates, 519
HTML Viewer, 266
NewSid utility, 50
Visual Studio Extensions for Windows SharePoint Server, 85
Dublin Core Metadata Initiative (DCMI), 468
Dynamic Host Configuration Protocol (DHCP), 75–76

E

ECM (Enterprise Content Management). *See also* content management; IFS; Master Pages
about, 474–475
components of, 475
Document Conversion services in, 124–126

ECS (Excel Calculation Services), 153–162. *See also* trusted file locations
about, 153–154
adding user-defined assemblies, 161–162
configuring, 154, 155–156
designating trusted data providers, 160–162
setup in SSP, 154
starting, 203
trusted file locations for, 156–159
using trusted data connection libraries, 159–160
verifying service running, 154
editing
list item columns, 385
list views, 387–391
Master Pages, 555–556
public zone URLs, 179–180
Publishing pages, 372–374
Registry, 86
Wiki pages, 525–526
workflows in SharePoint Designer, 594
Elements Manifest file
about, 597
code for complex Feature, 609
creating for event receivers, 656, 657
elements of, 598–600
packaging event receivers in, 650
specifying for Feature, 604–605
using for web part solution, 665, 666, 697–698
e-mail
configuring incoming, 126–127, 205–207
creating Document Workspace Site via, 362
handled by Exchange Server, 21
installing services for, 77–78
library options for, 381–382
modifying outgoing SMTP settings for web applications, 249
setting outgoing configuration for, 127–128, 204–205
specifying in workflow, 589–590
End User Agreement License, 87
Enterprise Content Management. *See* ECM
Enterprise Edition of MOSS
about, 3–4
enabling features, 189–190, 222–223
upgrading to, 222
Enterprise Site Definitions, 531–535
Document Center Site Template, 531–532
My Site Host, 535
Report Center Site Template, 534–535
Search Center/Search Center with Tabs, 535
Site Directory Template, 533–534
Enterprise Templates, 17, 519
error reports, 218
errors. *See also* failures
after uninstalling SharePoint, 194–195
checking for install, 110
correcting common SQL, 110–112
document conversion, 125–126, 496
handling Search, 175
illegal entries in .webpart file, 706
installing Shared Services, 136–140
tracking with System Application Event log, 619–620
using Single Sign-On service first time, 183–184
when installing Office 12 products, 86
event receivers
adding GUID with Feature Manifest file, 655
building project to use, 650–657

creating Elements Manifest file for, 656, 657
defined, 645
deleting, 661–662
deploying, 657–658
event properties, 649–650
list events and method fired by, 647–649
methods for, 646–647
setup required for testing, 658
testing and debugging, 659–661
uses for, 645–646
workflows vs., 645
event throttling, 219
events
error tracking for, 619–620
methods fired by list, 647–649
properties of, 649–650
Excel. *See also* ECS
about ECS, 153–154
starting workflows in, 594–596
Excel Calculation Services. *See* ECS
Excel Web Services, 22–23
Exchange Server, 21
execution cycle for web parts, 666
exporting documents from libraries, 627–628
ExportMode property, 693
extending web applications, 245–246
extensions. *See also* Windows SharePoint Server Extensions
STSAdm command-line tool, 737–740
Workflow, 85
extranets for SharePoint, 41, 42

F

failures
"Page Not Found", 193
SSP access, 136–137
SSP account, 138–139
SSP service, 138
Farm Administrators group, 214–206, 744
farms. *See* server farms
Feature files
about, 665
adding GUID to, 655
creating for web part solutions, 698
feature function matrix
developing, 37
planning installation using, 39–40
Feature Manifest file
about, 597
code for complex Feature, 609
elements of, 598
features
activating/deactivating Site, 318
activating/deactivating Site Collection, 326–327
assigning GUID for, 597, 604
building, 602–607
defined, 18, 597
deploying, 607–609
developing portal's feature function matrix, 37
Elements Manifest file for, 597, 598–600, 604–605, 609
Feature Manifest file, 597, 598
framework for Solutions and, 18–19
installations of, 598
managing web application-level, 252–253
MOSS Enterprise, 189–190, 222–223
personalization, 19–21
setting up complex, 609–610
SharePoint Server, 340
testing page added by, 608
using farm-wide, 229
workflows as, 568
XML files of, 597–598
files. *See also* trusted file locations; and specific files
blocking types of, 213
building .webpart, 696–697

created during Basic installation, 90–91
displaying with Page Viewer Web Part, 413, 419–421
found in web parts, 663–665
including Feature in project, 603
managing data connection, 274
Master Page and Page Layout aspx, 554–555
MDF and LDF backup, 233
reducing Windows Event Log size, 70–71
setting up installation locations for SharePoint, 98
SNK, 684–685
web.config, 517, 618–619
Filter Web Parts, 425, 437–438
flushing object caches, 332
folders
concept of document libraries vs., 399, 400
SharePoint control for, 248
form templates
managing, 270–271
uploading InfoPath, 272–274
Form Web Part, 413, 416–417
Forms authentication
about, 43, 741
client integration and, 743
features of, 742–743
setting up non-Windows-based, 744
Free Text Search Object, 643–644
front-end servers
configuring for load balancing, 59
running Windows Search on, 119
starting Windows Search on, 123

G

GAC (Global Assembly Cache)
debugging web parts installed in, 675
installing event receivers in, 657
placing compiled DLLs in, 659
as usual destination for assemblies, 669
galleries
List Template, 309
Master Page, 306–307
Site Column, 308
Site Content Type, 307
Site Template, 308–309, 549
Web Part, 310–311
Get Started with My Site Web Part, 447, 449
Global Assembly Cache. *See* GAC
global configuration options, 224–230
disabling farm-wide features, 229
managing solutions for server farms, 229–230
setting path to new Site collections, 226
shutting down farm services gradually, 228–229
timer job definitions, 225–226
timer job status display, 224–225
validating and cross referencing links in Site Directory, 226–227
Global Unique Identifiers. *See* GUIDs
governance, 28–29, 261
groups
adding security, 292–293
creating for list columns by content type, 386–387
deleting from Site Collection, 293–294
designating permissions for MOSS Site, 358
Farm Administrators, 214–206, 744
managing distribution, 207–210
managing Permission Level, 296–297
overcoming workflow limitations using, 568
Site Column, 454
viewing existing, 291–292
GUIDGEN.exe utility, 655, 698, 699

GUIDs (Global Unique Identifiers)
added to HowManyLists solution, 677
adding to Feature file, 655, 698, 699
assigned to lists, 376
assigning for Feature, 597, 604
identifying Sites with, 350
page assembly using, 364
used by Publishing pages, 484

H

handlers. *See* event receivers
hardware requirements, 49–50
Height and Width property, 689–690
HelpMode property, 691
HelpUrl property, 691
history, backup, 234
History Lists, 567
Home Directory tab (Properties window), 92
Home page
adjusting Master Page to update only, 482
Central Administration's, 199–200
My Home page, 443, 444
HOSTS files, 70
HTML Viewer, 265–266

I

I Need To... Web Part, 429–431
icons
changing Site, 298–299, 553
placing in picture library, 299
IFS (InfoPath Forms Service), 270–275
architecture of, 25
component of ECM, 475
configuring settings for, 271–272
creating Site Content Types for, 468–472
managing form templates, 270–271
managing Forms Service proxy, 274–275
setting session throttles for, 151–153, 272
updating Site Content Types, 472–473
uploading InfoPath Form Template, 272–274
IIS (Internet Information Services)
adjusting IIS Manager, 91–96
as basis for SharePoint Server, 6, 14–15
configuring, 73
configuring to use .NET 2.0 Framework, 74–75
installing, 72–73
verifying isolation mode setting for, 73
IIS Manager, 91–96
IIS Web Sites, 247
IISReset command, 667, 702
illegal entries in .webpart file, 706
image conversion errors, 496
Image Web Part, 412–413, 417–419
IMP (Information Management Policies)
applying policy to document library, 541
assigning policy to content type, 541–542
creating custom policies, 537–538
enabling/disabling, 215
features within SharePoint, 473
Records Center Site policies, 539–541
settings for, 465
specifying for lists and libraries, 381
types of, 536
impersonation
getting account information from web.config file, 618–619
retrieving account from Registry, 615–618
Site vs. in-code, 613
using, 613–615
importing user profiles, 164–167
In Common Between Us Web Part, 447, 449

incoming e-mail configuration, 126–127, 205–207
Index Engine for Office Search Service, 632–633
indexing
columns, 386
Search content, 120–121
Site content, 316
individual's roles in workflows, 564, 568
InfoPath Forms Services. *See* IFS
Information Management Policies. *See* IMP
Information Rights Management, 211–212
infrastructure for portal methodology, 33–34
inheritance
Master Pages and, 476
navigational, 352
page layout and Site template, 300–301
permissions, 351
Site columns, 457
installing MAKECAB.exe utility, 668
installing SharePoint Server, 53–106. *See also* planning installation
adding AD, DNS, and DHCP for development systems, 75–77
advanced installations, 86, 96–105
basic installations, 86, 87–96
checklists for, 58–61
defining domain accounts, 61–67
e-mail service installation, 77–78
installing Workflow Extensions for Visual Studio, 85
issues to avoid, 53–54
load balancing, 55–58
.NET Frameworks installation tasks, 74–75
Office installations, 86
operating system installation, 68–72
overview, 53
running Windows Update, 78
selecting method for, 87–88
setting up and testing network, 55
SharePoint Extensions for Visual Studio, 105–106
SQL Server installation, 79–83
tasks for IIS installation, 72–73
Visual Studio installation, 83–85
instances
Central Administration at level of, 197
number of users per workflow, 569
internal name for list column, 623
internal URLs, 180, 228
Internet Explorer Enhanced Security, 70
Internet Information Services. *See* IIS
Internet Site planning, 41, 42
Internet zones
mapping URLs to, 227–228
setting user policies by, 262
intranets Site planning, 41, 42
invalid entries in .webpart file, 706
isolation mode setting for IIS, 73

J

jobs
checking status of timed, 224–225
creating for content deployment, 504
setting up content deployment paths for, 239–240, 502, 504
viewing or managing timer, 224–225

K

keywords, 325
kickoff process for projects, 30, 31

L

labels
creating or removing for variations, 333–334, 499
enabling/disabling, 329

implementing Information Management Policies with, 536
using, 539–540
large server farms, 47–48
Layout Starter Pages, 486
layouts. *See* Page Layouts
LDAP (Lightweight Directory Access Protocol), 62–63
LDF files, 233
libraries
about, 375, 398–399
about SharePoint, 339
concept of folders vs. document, 399, 400
connecting to, 622–623
extracting documents from, 627–628
included for WSS Sites, 523–524
item scheduling for, 380
list columns in, 382–383, 623
list type, 375
modifying Site, 313–314
options for e-mail and attachments, 381–382
permissions for, 380
referencing Site content stored in, 299
role in storing and displaying Site content, 521, 522
saving as template, 380
Shared Documents library, 364, 365
storage of Publishing pages in, 483, 484
trusted data connection, 159–160
licenses
Client Access, 2
converting type of, 224
End User Agreement, 87
Lightweight Directory Access Protocol (LDAP), 62–63
Line Item Security feature, 537
links
adding or deleting from Top Link Bar, 303
adding Site Directory, 361
available on Personal Settings and My Links menus, 450–451
listing with Summary Link Web Part, 425, 433–434
managing Search Scope for related, 319
validating and cross referencing in Site Directory, 226–227
between workflow and document, 574–575
list columns, 382–387
adding from Site columns, 385–386, 455–456
creating, 383–384
editing, 385
grouping by content type, 386–387
indexing of, 386
ordering, 386
using, 623
List Settings options
Advanced Settings options, 379–380
Communications options, 381–382
General Settings options, 379
managing lists from, 378
Permissions and Management options, 380–381
list templates
about, 309
adding or removing from List Template Gallery, 309
saving, 380
List View Web Parts
connections using, 411–412
creating, 375, 394–398, 410
displaying Site content with, 521, 522
effect of deleting list on, 376
managing lists from, 378
list views, 387–391
about, 339–340, 377, 387
creating and editing, 387–391
deleting, 376
setting, 394–398

lists. *See also* libraries; list columns; List View Web Parts; list views
about, 375
accessing items on, 622–623
adding links, 425, 433–434
based on templates, 377
characteristics of SharePoint, 391
connecting to, 622–623
creating and adding items to, 625–627
custom, 408–410
deleting list views vs., 376
displaying in I Need To... drop-down menu, 429–431
elements of, 376–377
GUIDs assigned to, 376
included for WSS Sites, 523–524
libraries and, 375
list columns, 382–387
List Settings options for, 379–382
managing, 378
menu options for, 392–393
modifying Site, 313–314
object model of, 376
permissions for, 288, 380
Relationship List, 497, 500–501
relevant documents, 420–421
role in storing and displaying Site content, 521, 522
rows and columns of, 376
RSS feeds from, 382
saving as template, 380
searching for items in, 624–625
SharePoint, 339
showing user's task, 413, 422–423
summary document, 424, 426–427
types of SharePoint, 401–403
user interface features of, 392
Web Services, 629–630
load balancing, 55–58
configuring front-end servers for, 59
configuring up Excel Service settings for, 155
setting up, 55–57
starting Windows, 193
stopping, 58
testing, 57–58
using Document Conversion Load Balancer services, 124–126, 203
loading web parts, 706–707
locale settings, 313
locks for Site Collections, 258
Login/Logout feature for My Site, 450–451
logs
checking system event, 190
enabling usage analysis, 176–177, 219–220
errors with System Application Event, 619–620
options managed in Central Administration, 218–219
reviewing Feature deployment, 609
settings for diagnostic, 183–183
variation, 334, 500
viewing Site content and structure in modification, 321
Look and Feel section for Site Administration, 297

M

mailboxes for domain account, 78
MAKECAB.exe utility, 668, 701
Manifest files
creating for web part solutions, 698–699
web part, 664, 665
mapping
Alternate Access, 179–180, 227–228
portal topology, 32
URLs to Internet zones, 227–228
Master Page Gallery options, 306–307, 312, 481
Master Pages
about, 16, 476

adding or removing in MOSS, 312
branding and, 476–477
building in SharePoint Designer, 477–483
changes overriding Site theme settings, 303
component of ECM, 475
creating, 556–557
custom modifications to, 553–555
developing custom pages with, 515
inheritance in, 476
modifying core pages, 557
modifying My Site default master, 561–562
one-off changes to, 555–556
setting options for MOSS, 298
storage of, 518
uploading or deleting WSS, 306–307
using, 340, 341
WSS and MOSS use of, 476–477
master/peer relationship of variants, 497
MCMS (Microsoft Content Management Server)
about, 2
options for upgrading and migrating to MOSS, 220–224
replaced by ECM, 474
MDF files, 233
medium server farms, 47
Meeting Site Type, 557
Meeting Workspaces
creating, 545, 546–548
Site templates for, 522–523
supported by WSS, 344
Member Site group, 289
memberships, 291–293
Memberships Web Part, 447, 449
memory
configuring Excel Service settings for, 156
virtual server requirements for, 50
menu options for lists, 392–393
metadata
about, 400
defined, 398
showing in Document Information Panel, 463
standard definition for document, 468
methods
listing of Event Receiver, 646–649
overriding Event Receiver, 646, 652
used to render web parts, 363
Microsoft Commerce Server, 2
Microsoft Content Management Server. *See* MCMS
Microsoft Excel. *See* Excel
Microsoft Office. *See* Office applications
Microsoft Office SharePoint Server. *See* MOSS; SharePoint Server
Microsoft Office SharePoint Server 2007 Software Development Kit, 501, 744
Microsoft Project, 8
Microsoft technologies. *See also* IIS; Office applications; SQL Server; Visual Studio
IIS influence on SharePoint Server, 6, 14–15
.NET influence on SharePoint Server, 14
overview of influences on SharePoint Server, 11–12
services provided by other, 21
SQL Server influence on, 13
Microsoft Visual Studio. *See* Visual Studio
Microsoft Word. *See* Word
migrating documents to SharePoint, 401
Mixed Mode authentication, 80
modifications
custom, 553–554
out-of-the-box, 553
MOSS (Microsoft Office SharePoint Server 2007). *See also* Master Pages; MOSS Sites; MOSS web parts; My Site
additional Site groups for, 289–290

adjusting Excel Services Trusted Locations, 153–162
all Sites supported by, 344
audit log report for, 329
audit settings, 328
cache profiles for Site Collections, 330–331
completed installation steps for, 109
composition of websites and pages, 16–18
configuring SSPs, 140–142
connecting/disconnecting from portal Site, 328
content management for current Site, 320–321
converting license type after upgrading, 224
core web parts available, 413
creating SSPs, 134–135
deploying Site content, 200, 238–241
Document Conversion Service within, 24–25, 203
enabling Enterprise features for, 189–190, 222–223
Enterprise-level content types, 307
errors installing Shared Services, 136–140
establishing Shared Service Providers, 128–131
Excel Web Services in, 22–23
extra configuration tasks for, 115–116
Features and Solutions Framework for, 18–19
features of, 8
hierarchy of pages in, 484
InfoPath Forms Services in, 25
inheriting page layouts and Site templates, 300–301
keyword management, 325
Master Pages in, 298, 312, 476–477
Microsoft product mix for, 4–5
modification logs for content and structure in, 321
modifying Site navigation, 299–300
MOSS-specific web parts, 424, 425–426
My Site Web Parts, 446–450
object caches, 332
options for Search Scopes, 322–325
page caching policies for publishing Sites, 319–320
Page Layout Templates, 312
paths for Site Collections, 226
permission groups in, 290
personalization for, 19–21
Publishing pages, 369–371
removing and setting Site Collection policies, 329–330
running on domain controller, 75
scope for related links, 319
SE and EE editions of, 3–4, 222
Search for SE/EE editions, 4–5
Search Scopes in, 322
searchable Site columns, 306
service level settings for Shared Services, 162–179
setting up Office Search, 120–122, 204
shutting down farm services gradually, 228–229
Site Collection usage reports for, 326
Site variations created in, 332–333
specifying settings for Site directory, 326
starting Excel Calculation Services for, 203
suggestions for web parts in, 405
support for Publishing pages, 17–18
translatable columns, 334
types of SharePoint lists, 401–403
upgrading to, 220–224
use of Share Services Administration Sites in, 197, 199
user and page output cache policies in, 330
user interface for, 335, 336, 337

viewing variation publishing logs, 334
Windows Search in, 21, 204
MOSS My Site. *See* My Site
MOSS Sites
about, 355
Basic pages for, 365–367
connecting Central Administration with Shared Services, 275–277
content and structure management for, 358–360
creating portal level, 357–360
creating sub Sites in Site Directory, 360–361
Enterprise Edition features enabled for, 189–190, 222–223
enterprise Site definitions, 531–535
features of, 549
Information Management Policy enabled for, 215
managing InfoPath Forms Services in, 270–275
overview of, 529
portal Site definitions for, 530–531
publishing approval workflow for content on, 372
Publishing pages for, 369–371
Records Center and Information Management Policies for, 536–544
scheduling and running policy usage reports, 220
support for Basic, Web Part, and Publishing pages, 364–365
topology of, 355–356
using Office Search for, 268–270
Web Part Page creation for, 367–369
workflow management for, 277–283
MOSS web parts
about, 424
Filter, 425, 437–438
I Need To..., 425, 429–431
list of, 425–426
RSS Viewer, 425, 431–433
Search, 425–426, 438–439
Summary Link, 425, 433–434
Table of Contents, 425, 434–435
This Week in Pictures, 425, 435–436
MSBuild to create WSP files, 712–714
My Blog Web Part, 446, 447
My Calendar Web Part, 446, 447
My Contacts Web Parts, 447, 448
My Home page, 443, 444
My Inbox Web Part, 447, 448
My Links menu, 450–451
My Links Web Part, 447, 449
My Mail Folder Web Part, 447, 448
My Pictures Web Part, 447, 449
My SharePoint Sites Web Part, 447, 449
My Site
adding Personalization Site links, 172
alerts, 451
capabilities of, 529
changing default master for, 561–562
construction of, 285
content governance for, 29
creation of, 444
customizing Site Template for, 561
defined, 9
designating trusted locations for, 171
designing profile page for, 169–170
establishing settings for, 168–170
overview of, 443–444, 452
pages of, 443
Personal Settings menu, My Links menu, and Login/Logout features, 450–451
public view of Profile Page, 445
removing Organization Display Web Part from, 562
sub sites in, 450
using Site Aggregator Web Part for, 424, 426–427
web parts for, 446–450

My Site Administration, 285
My Site Host Site Definition, 535
My Tasks Web Part, 447, 448
My Workspaces Web Part, 447, 449

N

namespaces
    accessing parts of object model via, 613
    publishing, 505–506
naming server farm configuration databases, 100
navigation
    changing top navigation bar, 303
    choosing user interface options for, 351–352
    configuring MOSS Site, 358–360
    from Content to Create Site page, 348–349
    enabling/disabling Tree View, 302, 553
    features for top-level Sites, 341–344
    modifying Site, 299–300
navigation node cache, 509
.NET Framework
    configuring IIS for version 2.0, 74–75
    Event Receiver classes in, 646
    influence on SharePoint, 14
    installing, 74
    using impersonation, 613
    WWF workflow powered by, 564–565
Network Load Balancing Manager (NLBMgr) utility, 58, 193
networks
    placement of, 42
    planning, 40–41
    pre-installation set up and testing of, 55
NewSid utility, 50
NLBMgr utility (Network Load Balancing Manager) utility, 58, 193
non-Site groups, 296–297
non-Windows-based authentication, 744
notifications
    of unused Sites, 256
    workflow task, 277–278

O

object caches
    flushing, 332
    settings for, 332, 513–514
Object Model
    accessing through Windows application, 611–612
    accessing with impersonation, 613–619
    connecting to list, 622–627
    error tracking with System Application Event log, 619–620
    event receivers' access to, 645
    exporting documents from libraries, 627–628
    lists, 376
    referencing, 612–613
    setting web part solution properties in code, 689
    SharePoint extensibility and, 611
    SharePoint Sites referenced via, 621
    SPItemEventProperties object, 646
    SPSite vs. SPWeb objects, 620–621
Office applications. *See also* Excel; Outlook; SharePoint Designer; Word
    Access, 567
    accessing SharePoint Object Model via applications in, 611–612
    adding links to client applications, 171–172
    creating Sites using, 344–345, 361–363, 545–548
    Document Information Panel in, 463
    error messages when installing products from, 86
    installing, 86

integration with SharePoint Server, 361
Project, 8
role in SharePoint workflow process, 567
setting specific user for workflow, 577–581
Site content type integrated with, 458
starting workflows in, 594–596
using workflows in, 576–577
Office SharePoint Server Search Service. *See also* Windows SharePoint Server Search Service
about, 632
architecture of, 21–22
building custom search tools, 638
components of, 632–635
configuring, 173–175
crawling, 635
ECM and, 475
functions and services of, 21–22
handling errors in, 175
indexing content for, 120–121
MOSS settings for, 268–270
propagation of index data, 636
search queries, 635–636
searching items in list, 624–625
setting up, 120–122, 204
tweaking, 636–638
usage reports for, 175–177
one-off web parts, 667, 668–676
adding code to class and compiling, 670–671
adding references and modifying class, 668–669
adding to SharePoint and testing, 672–675
creating project, 668
debugging, 675–676
overview, 667, 668
setting debug/release destination, 669–670
ONET.XML file, 525, 552, 561
operating systems
installing for development and QA systems, 70–72
running Windows Update, 78
steps for installing, 68–72
Operations page (Central Administration Site)
Alternate Access Mappings option, 227–228
Antivirus option, 212–213
Backup and Restore section, 200, 231–237
Blocked File Types option, 213
Configuring Incoming E-Mail Settings option, 205–207
Content Deployment Object Status option, 241
Content Deployment Paths and Jobs option, 239–240, 502, 504
Content Deployment section, 200, 238–241
Content Deployment Settings option, 240–241
Convert License Type option, 224
Data Configuration section, 200, 237–238
Data Retrieval Service option, 238
Default Database Server option, 237–238
Diagnostic Logging option, 218–219
Distribution Groups option, 207–210
Enable Enterprise Features option, 222
Enable Features on Existing Sites option, 222–223
Global Configuration section, 224–230
illustrated, 200
Information Management Policy Configuration option, 215
Information Management Policy Usage Reports option, 220
Information Rights Management option, 211–212

Logging and Reporting section, 200, 218–220
Manage Farm Features option, 229
Manage Microsoft Content Management Server Migration Profiles option, 221
Manage Setting for Single-Sign-on option, 216–218
Master Site Directory Settings option, 226
Outgoing E-Mail Settings option, 204–205
overview, 200–201
Quiesce Farm option, 228–229
Security Configuration section of, 200, 210–218
Servers in Farm option, 201–202
Service Accounts option, 210–211
Services on Server option, 202, 203–204
Site Directory Links Scan option, 226–227
Solution Management option, 229–230
Timer Job Definitions option, 225–226
Timer Job Status option, 224–225
Topology and Services section of, 200–210
Update Farm Administrator's Group option, 214–206
Upgrade and Migration section, 220–224
Usage Analysis Processing option, 219–220
Organization Display Web Part, 562
OUs (organizational units)
Administrative Groups and Users, 66–67
defining SharePoint AD, 62–63
outgoing e-mail
configuring, 127–128, 204–205
modifying SMTP server settings for, 249
Outlook
creating Meeting Workspaces, 545, 546–548
role in SharePoint workflow process, 567
viewing assigned workflow tasks in, 578
web parts used for Outlook, 446, 447, 448, 529
Outlook Web Access, 446, 447, 448, 529
overriding
Event Receiver methods, 646, 652
security during development and testing, 611–612
Site theme settings, 303
OWSTIMER service, 138, 139

P

page caching, 507–515
about, 509
advantages of SharePoint, 507–508
configuring, 511–514
disabling for volatile Sites, 511
policies for, 509, 514–515
profiles for, 509–510, 512
publishing Site policies for, 319–320
setting page output cache policies, 330
page item cache, 509, 513–514
Page Layout Templates, 312
Page Layouts
about, 483
adding or removing templates for, 312
choosing Site Content Types for, 486
creating, 486–489, 557–560
custom Site modifications with, 553–554
customizing, 557
default Publishing pages, 370
developing custom pages with, 515
inheriting, 300–301

publishing page using, 489–491
setting up, 485–486
storage of, 518
using cache policies on, 514–515
web part properties for, 407
"Page Not Found" failures, 193
Page Viewer Web Part, 413, 419–421
pages. *See also* Master Pages; Publishing pages; and specific pages
about, 363–365
accessing, 365
approving content of, 481
ASP.NET, 9
Basic, 364, 365–367
converting Word documents to SharePoint, 493–497
defined, 338–339
displaying with Page Viewer Web Part, 413, 419–421
editing and creating Wiki, 525–526
execution cycle for web parts and, 666
hierarchy of MOSS, 484
illustrated, 364
inheriting layouts and Site templates, 300–301
modifying navigation for, 299–300
My Site, 443
providing contact details for owner of, 426, 441
publishing from template, 489–491
restoring to original Site definition, 305
Site Welcome Page, 301
steps for creating and publishing, 489–491
uses for, 363
Web Part, 364, 367–369
workflow and Publishing, 372–374
parameters
setting for web part solutions, 685–687
STSAdm schedule, 736
parent Sites
accepting permissions from, 358
inheriting navigation from, 352, 358
page layouts and Site templates inherited from, 300–301
permissions inherited from, 351
passwords
changing for services, 210
entering Single Sign-On service, 218
policies for SharePoint accounts, 63, 65–66
paths
defining SharePoint control for folder, 248
managing content deployment, 239–240, 502, 503–504
Permission Level groups, 296–297
permissions
advanced Site Administration, 297
assigning for personalization services, 173
default permission groups, 290
defining for user interface, 351
enabling/disabling Site, 295–296
library, 380
list, 379, 380
setting for MOSS Sites, 358
SharePoint, 288–289
Site groups and their rights, 289–290
user web application, 261–262
viewing existing, 291–292
Personal Settings menu, 450–451
personal Sites
about groups on, 289
permission level groups for, 291
Personal View (Web Part Page), 368
personal web part property values, 406
personalization
adding Personalization Site links, 172
assigning permissions for services, 173
authentication and, 742, 743
default My Home page, 444

enabling Visual Studio features for, 84–85
handling page, 19–21
using Filter Web Parts, 437–438
picture libraries, 425, 435–436
pilot productions, 28
planning installation, 39–52. *See also* installing SharePoint Server
backup and disaster recovery plans, 48
feature function matrix for, 39–40
hardware requirements, 49–50
network connectivity, 40–41
selecting authentication method, 42–43
software requirements, 50–51
types of installation, 39
types of server configurations, 44–50
policies. *See also* IMP
creating custom, 537–538
enabling information management, 215
page caching, 319–320, 330
removing and setting expirations for, 329–330
setting user web application, 262–263
Site, 29
policy usage reports, 220
portal deployment methodology, 30–37
business analysis and branding, 30, 32, 33
defining business problem and implementation for, 27–28
developing infrastructure and architecture for, 33–34
devising architecture and topology for, 32
feature function matrix for, 37
initial planning for, 30, 31
mapping topology of intended use for, 32
overview, 31
planning rollout of configuration, development, and training, 35, 36
setting scope on requirements and delivery, 32–33
user deployment strategy for, 35, 36
portal Sites. *See also* portal deployment methodology
accessing Site Collection Site Administration from, 285, 286, 287
Collaboration Portal Site Definition for, 530
connecting or disconnecting from, 328
MOSS Sites as, 355
Publishing Portal Site Definition for, 530–531
structural design of MOSS, 337
using Records Center as sub Site of, 543
portals
defined, 9
origins of technology for, 1, 2
Sites created under, 341
unable to restore, 353
using restore to reproduce, 236
ports
designating for new web application, 243–244
setting for web applications, 132
specifying for SQL, 110–111
Posting Document Template, 529
posting to blogs, 526, 527–529
post-install configuration, 107–195. *See also* service levels for Shared Services
about Central Administration Site, 113–114
adding servers to farm, 117–118
adding SQL application pool accounts, 110
adjusting Excel Services Trusted Locations, 153–162
administrative tasks in, 107–108, 114–116

assigning services to servers, 118–126
checking for install errors, 110
configuring workflow settings, 149–150
creating applications and Shared Services, 128–149
enabling Enterprise features for MOSS, 189–190, 222–223
enabling Windows Load Balancing, 193
incoming e-mail configuration, 126–127
joining existing server farm, 190–192
outgoing e-mail settings, 127–128
repairing SharePoint Server, 193–194
reviewing system event logs, 190
service level settings for Shared Services, 162–179
setting session throttles for InfoPath Forms Service, 151–153, 272
SharePoint Server Administration command-line tool, 112–113
SQL settings and correcting common errors, 110–112
starting Document Conversion services, 124–126, 203
summary of completed installation steps, 108–110
uninstalling SharePoint Server, 194–195
updating web application settings, 187–188
PowerPoint workflows, 594–596
previewing web parts, 310–311
Profile Page, 443, 445
Profile Service Policies, 167
profiles
cache, 509–510, 512
managing MCMS migration, 221
options for output caches, 511–513
Site Collection cache options, 512
Project, 8
projects
building to check document status, 650–657
creating new web part, 677
designing templates for Sites in, 29
initial planning for portal, 30, 31
kickoff process for, 30, 31
plans for developing SharePoint, 30
setting portal requirements and delivery, 32–33
properties
Boolean, 694
event, 649–650
setting custom, 694–695
web part, 406–407, 408
.webpart file, 689–694
proxy servers
Forms Service, 274–275
setting up, 104
public key token, 671, 696, 697
publishing namespace, 505–506
Publishing pages, 17–18. *See also* document conversion
about, 364, 483–484
converting Word document to, 493–497
creating, 369–371
editing and workflows for, 372–374
elements of, 484–485
GUID for, 484
illustrated, 485
Master Page and Page Layouts for, 17–18
steps for creating and publishing page, 489–491
web parts specific to, 489
Publishing Portal Site Definition, 530–531
Publishing Portal with Workflow Site Definition, 530–531
Publishing Wizard
creating InfoPath Site Content Types in, 468–471

updating Site Content Types in, 472–473
viewing content type published in, 464

Q

QA systems
adding test user accounts for, 67
installing Office, 86
special operating system installation steps for, 70–72
Visual Studio installation for, 83–85
queries
creating with Content Query Web Part, 427–429
enabling/disabling, 238
search, 635–636
using search parameters in URL, 638–639
Query Engine
options for, 633
search queries with, 635–636
Query Web Service, 644
Quick Launch
enabling/disabling, 302
modifying properties for WSS Sites, 304
Quiesce Farm options (Central Administration Site), 228–229
quotas, 257, 258

R

receivers. *See* event receivers
Recent Blog Posts Web Part, 443, 447, 450
Records Center Sites
applying policy to document library, 541
assigning policy to content type, 541–542
Central Administration connection to, 264–265
configuring and using, 543–544
creating new information management policy, 539–541
document handling and storage at, 538–539
managing documents via, 536
rerouting records, 78–79
viewing records on, 544
Recycle Bin, 325–326, 352–354
references
adding in web part class project, 668
command reference chart for STSAdm, 715–736
to SharePoint Sites via Object Model, 621
stored content, 299
regional settings, 312–313
Registry
cautions on editing, 86
retrieving impersonation information from, 615–618
Relationship List, 497, 500–501
Relevant Documents Web Part, 420–421
Relevant Documents With Part Web Part, 413
Remote Access, 71–72
renaming virtual servers, 50, 69
repairing SharePoint Server, 193–194
Report Center Site Template, 534–535
reports
audit log, 329
correcting SQL Reports after WSS install, 93–96
error, 218
fixing SQL Reports, 111–112
policy usage, 220
Search usage, 175–177
viewing Site usage, 314
Resource files, 663, 665, 683–684
restoring
Central Administration Sites, 231–237
data from backups, 234–236
page to original Site definition, 305

Site Collections, 256
SSP from backup, 283
viewing status of current restore, 236–237
retrieving impersonation information, 615–619
Reusable Content libraries, 575
RMS (Rights Management System), 211–212
role-based security, 742, 743
root web cache, 508
RSS feeds
displaying, 425, 431–433
enabling/disabling, 315
list options for, 382
providing with blogs, 528
RSS Viewer Web Part, 431–433
rules for Search Scopes, 324
Runtime Object Model, about, 638

S

SafeControl
adding to web.config file, 671–672
included in solution's .webpart file, 695
scalability and workflow limitations, 568
scripts for custom search, 639–642
Search. *See* Office SharePoint Server Search Service; Windows SharePoint Server Search Service
Search Center Site Definition, 535
Search Center with Tabs Site Definition, 535
Search Object Model, 638, 643–644
Search Scopes, 633–634
adding and removing, 322–325
custom MOSS, 322
defined, 175
managing for related links, 319
rules for, 324
Search Server role, 15
Search Web Parts, 425–426, 438–439
searchable Site columns, 306, 553
security
antivirus options, 212–213
application, 260–264
authentication vs., 741
blocking file types, 213
configuring Excel Service settings for, 155
independent of architecture, 613
information management policies, 215
Information Rights Management options, 211–212
list, 379
Operations page options for, 200, 210–218
overriding during development and testing, 611–612
setting for new web applications, 244
updating Farm Administrators group, 214–206
using impersonation, 613–615
security groups, 292–293
server farms. *See also* disaster recovery; servers
Active Directory accounts installation checklist, 59
adding servers to, 117–118
Advanced installation features for, 96–97
changing settings for Shared Services, 275
checking and enabling services for, 181–182
checking Shared Services operating for, 276
choosing Advanced installations, 97
configuring workflows for all servers, 569
content deployment to, 240–241
deploying solutions for, 229–230
deploying with web part solution, 712

enabling/disabling farm-wide features, 229
flushing object caches for all servers, 332
installation steps for connecting to, 99–101
joining existing, 190–192
large, 47–48
managing Shared Services for, 278
medium, 47
recommended accounts for, 63
removing server from, 192
restoring, 234–236
role in content deployment, 502
server roles for, 44–45
setting file installation location, 98
setting Records Center Site to, 543
setting search options for, 268–269
shutting down gradually, 228–229
single servers, 45
Single Sign-On service for, 183–186, 216–218
small, 45–46
SQL Server installation checklist for, 59–60
upgrade path for, 49
using Shared Services between, 275–276
viewing existing and removed servers in, 201–202
web/application server installation checklist for, 60
servers
adding to farm, 117–118
application, 15, 60
assigning services to, 118–120
communications between SQL Server and, 42
enabling for Search queries, 121–122
hardware requirements for, 50
indexing Search content, 120–121
installing operating system for, 68–72
joining existing domain, 69
joining existing server farm, 190–192
load balancing for front-end, 59
managing services on, 203–204
minimum size of, 40
planning roles for, 15, 40
removing from server farm, 192
Rights Management System on, 211–212
selecting default database, 237–238
settings for proxy, 104
SharePoint configurations for, 44–50
software requirements for, 50–51
stopping services before removing, 201–202
updating HOSTS files, 70
virtual, 50, 69
service levels for Shared Services, 162–179
adding Personalization Site links, 172
adjusting settings for, 162–163
antivirus protection, 186–187
assigning personalization services permissions, 173
checking and enabling farm's, 181–182
configuring Search settings, 173–175
creating published links to Office client applications, 171–172
designating My Site trusted locations, 171
diagnostic logging settings, 182–183
establishing My Site settings, 168–170
handling Search errors, 175
managing audiences, 177–179
managing Single Sign-On service, 183–186, 216–218
modifying Profile Service Policies, 167
usage reports for Search, 175–177
User profiles and properties, 163–167
service packs
SQL Server Service Pack 1, 81–82
Visual Studio 1 and 2, 85

services
changing passwords for, 210
failures in SSP, 138
managing on servers, 203–204
stopping before removing server, 201–202
sessions
configuring Excel Service settings for, 155
configuring state for, 276–277
managing for trusted file locations, 158
Shared Documents library, about, 364, 365
Shared Service Providers. *See* SSPs
Shared Services. *See also* service levels for Shared Services; SSPs
changing settings for server farms, 275
checking current status for server farm, 276
connecting MOSS Central Administration Site with, 275–277
managing farm's, 278
using between server farms, 275–276
Shared Services Administration Site, 16, 197, 198
Shared View (Web Part Page), 368
shared web part property values, 406
SharePoint 3.0 Central Administration. *See* Central Administration Site
SharePoint Data Store, 6, 7
SharePoint Designer. *See also* building SharePoint Designer workflows
building workflows, 584–594
creating Site Definitions with, 550–551
customizing Page Layouts in, 557
designing Master Pages in, 477–483
editing workflows, 594
modifying Master Pages in, 555–556
role in workflow process, 567, 568
saving Sites as Site Template, 551
steps, actions, and conditions, 581–583
using, 518–519, 581–582
verifying workflow in, 590–594
workflow design considerations in, 583–584
SharePoint Documents Web Part, 447, 450
SharePoint Portal Server, 1–2
SharePoint Products and Technologies Configuration Wizard, 98–99
Advanced Settings button for Configuration Wizard, 102
basic installations using, 87–89
confirming settings for, 102–103
connecting to server farm, 99–101
joining existing server farm, 100–101, 190–192
selecting authentication method, 101
starting, 98–99
SharePoint Search account, 175
SharePoint Server. *See also* MOSS; Object Model; user interface; WSS
architecture of, 5–9
configuring page caching for, 510–514
core web parts for, 412–413
extensibility and, 611
goals for workflows, 563–564
implementation of WWF, 565–566
integration of Office products with, 361
Internet Information Server as basis for, 6
Microsoft product mix for, 4–5
origins of, 1–2
out-of-the-box and custom branding options, 553–554
referencing Object Model, 612–613
repairing, 193–194
server roles in, 15
services found in, 3–5
Sites and pages of, 16–18
technologies used in, 2–3

as technology platform, 3
tools for content deployment, 501–502
types of caches, 508–509
uninstalling, 194–195
workflow tools in, 566–568
SharePoint Server Administration command-line tool. *See* STSAdm
SharePoint Server Administrator's group, 104–105
Shutdown Event Tracker, 71
SIDs (system IDs), 50, 69
single servers
Central Administration Site for, 113
configuration and recovery for, 45
installation checklist for, 60–61
single sign-on authentication. *See* SSO authentication
Single Sign-On service
enabling, 183–186
first-time errors using, 183–184
managing from Central Administration, 216–218
Site Administration
about, 197, 198
accessing Site Collection Site Administration, 285–287
adding new Site Group, 294–295
adding users, security groups, or memberships, 292–293
advanced permissions options, 297
Audit Log Reports option, 329
creating sites via, 346–347
Delete This Site option, 318–319
deleting users or groups from Site Collection, 293–294
enabling/disabling Quick Launch and Tree View navigation, 302
enabling/disabling Site Permissions, 295–296
Galleries section, 306–312
levels of, 285
List Template Gallery option, 309
Look and Feel section, 297–306
managing Permission Level groups, 296–297
Master Page Gallery option, 306–307, 312, 481
Master Page options, 298
Page Layouts and Site Templates Settings option, 300–301
People and Groups options for, 287
Portal Site Connection option, 328
Publishing Site Output Cache Settings option, 319–320
Quick Launch option, 304
Recycle Bin option, 325–326
Regional Settings option, 312–313
Related Links Scope Settings option, 319
Reset Page to Site Definition Version option, 305
role in SharePoint workflow process, 567
RSS option, 315
Save Site as Template option, 304–305, 548–549
Search Keywords option, 325
Search Scopes option, 322–325
Search settings for fields option, 306
Search Settings option, 322
Search Visibility option, 316
Site Administration section, 312–321
Site Collection Administration section, 322–334
Site Collection Administrators option, 297
Site Collection Cache Profile option, 330–331
Site Collection Features option, 326–327
Site Collection Object Cache option, 332, 513–514
Site Collection Output Cache Settings option, 330
Site Collection Policies option, 329–330

Site Collection Usage Reports option, 326
Site Collection Workflows option, 311
Site Column Gallery option, 308
Site Connection Audit Settings option, 328
Site Content and Structure Logs option, 321
Site Content and Structure option, 320–321
Site Content Type Gallery option, 307
Site Directory Settings option, 326
Site Features option, 318
Site Hierarchy option, 327
Site Libraries and Lists option, 313–314
Site Navigation Settings option, 299–300
Site Template Gallery option, 308–309, 549
Site Theme option, 302–303
Site usage summary option, 314
Site Welcome Page, 301, 553
Sites and Workspaces option, 316–317
summary of variation settings in, 498
Title, Description, and Icon option, 298–299, 553
Top Link Bar option, 303, 553
Translatable Columns option, 334
User Alerts option, 315
Users and Permissions section, 287–297
Variation Labels option, 333–334
Variation Logs option, 334
Variations option, 332–333
viewing existing users and groups, 291–292
Web Part Gallery, 310–311
Site administrators, 319
Site Aggregator Web Part, 424, 426–427
Site Collection administrators, 297
Site Collections, 254–259
about, 16
accessing Site Administration page for, 285–287
activating/deactivating features in, 326–327
adding or removing administrators from, 258–259
administration of, 285
applying quotas and storage limits on, 258
configuring output caches, 511–514
creating, 254–255
deactivating and removing InfoPath forms for, 270–271
defined, 338
deleting, 255–256
management of, 198
modifying regional settings for, 312–313
object caches, 332, 513–514
relationships as SPWeb objects, 620
removing and setting policies for, 329–330
restoring items from Recycle Bin, 325–326
setting path for MOSS, 226
setting user and page output cache policies, 330
summarizing user's task lists for, 413, 422–423
viewing details of, 259
viewing status of Site's current workflows, 311
viewing usage reports for, 326
Site columns
adding columns from, 385–386, 455–456
cautions using, 457
changing, 308, 455
creating, 308, 454–455
defined, 453
deleting, 308

developing for Site Content Types, 458
groups for, 454
modifying, 306, 460
using, 455–457
Site Content Type Gallery, 307
Site Content Types
about, 457
applying information management policy to, 541–542
assigning settings for, 461–465
choosing for Page Layouts, 486
creating InfoPath, 468–472
deleting, 307, 463
developing Site columns for, 458
Dublin Core Columns, 468
exercises creating, 457–467
grouping list columns by, 386–387
Page Layouts based on, 483
setting up new, 459–461
updating InfoPath, 472–473
using, 307, 465–467
workflows as, 575
Site Creation Permissions, 347–348
Site Definitions
about, 518
application templates available for download, 519
Blog, 526–529
creating from within User Interface, 548–549
creating in SharePoint Designer, 550–551
deploying from Visual Studio, 552
MOSS enterprise, 531–535
MOSS portal, 530–531
My Site Host, 535
My Sites as type of, 561–562
Office-based, 545–548
Records Center and Information Management Policies for MOSS Sites, 536–544
saving, 549
Search Center/Search Center with Tabs, 535
using user-defined Site templates, 550
Visual Studio for creating, 551–552
Wiki, 525–526
WSS, 524–525
for WSS Sites template types, 522–523
Site Directory
about, 356
creating MOSS Sites with, 360–361
validating and cross-referencing links in, 226–227
Site Directory Template
about, 533
Categories Web Part included in, 426, 439–440
creating Sites in, 533–534
Top Sites Web Part in, 426, 440–441
Site Groups
adding, 294–295
additional MOSS, 289–290
permission groups vs., 290
Visitor, Member, and Administrator, 289
Site impersonation, 613
Site Master Page option, 298
Site Master Page Settings option, 298, 482
Site pages. *See* pages
Site permissions, 295–296
Site Template Gallery option, 308–309, 549
Site templates. *See also* Site Definitions
about, 308–309, 340–341
adding or removing from Site Template Gallery, 308–309
choosing for Site Collections, 254, 255
creating first web application with, 148–149
customizing My Site, 561–562
defined, 9
designing, 29
saving SharePoint Designer Sites as, 551
saving Site as, 304–305, 548–549
selecting for new Sites, 351
selection options for MOSS Sites, 357

Site Directory, 356
using user-defined, 550
Site Theme options, 302–303, 553
Site types, 519
Site usage reports, 314
Site Users Web Part, 413, 421–422
Sites. *See also* MOSS Sites; sub sites; WSS Sites; and specific Sites
about, 9
about Central Administration, 14
about Site Directory, 356
accessing Site Collection Site Administration, 285–287
activating/deactivating installed features for, 318
architecture of Shared Services Administration, 16
changing Site Welcome Page, 301
changing title, description, and icons on, 298–299
composition of WSS and MOSS page and, 16–18
configuring connections to records center, 264–265
connecting to SharePoint, 621
creating in Site Directory Template, 533–534
creating summary lists of documents, 424, 426–427
creating through user interface, 344–345
creating XSL-styled list view of, 426, 440
custom user queries of, 427–429
defining variations for, 332–333
deleting current, 318–319
developing policies for, 29
displaying top, 426, 440–441
elements of WSS, 520–521
enabling/disabling user's ability to create, 260–261
Enterprise Edition features on MOSS, 189–190, 222–223
files used to create pages, 518
governance of, 29
guidelines for content, 29
hierarchy of pages and Sites in MOSS, 484
managing content deployment for, 200, 238–241
MOSS topology for, 355–356
New SharePoint Site page settings for, 350–352
notifications and deletions of unused, 256
pages for, 363–365
permissions for, 288–289
personalizing, 437–438
principals of WSS, 7–8
relationships as SPSite objects, 620
restoring, 234–236
saving as Site Template, 551
security independent of Site architecture, 613
setting Site Creation Permissions, 347–348
SharePoint vs. HTML, 517
showing list of users, 413, 421–422
Site directory settings for, 326
templates for, 29
theme of, 302–303
unable to restore from Recycle Bin, 353
user interface components for, 338–340
uses for web.config file, 517
using Content Page to reach Create Site page, 348–349
using Site Administration to create, 346–347
WSS topology for, 345–346
WSS vs. MOSS, 336–337
Sites in Category Web Part, 426, 440
slideshows, 425, 435–436
small server farm, 45–46
Smart Client Authoring. *See also* document conversion
document conversion in, 492–493
updating converted documents, 497

SMTP, installing service for, 77–78
SMTP servers, modifying settings for web applications, 249
SN.exe (strong name utility), 696
SNK (strong name key) files, 684–685
Software Development Kit (SDK), 501, 744
software requirements for servers, 50–51
Solution Manifest file, 600–601
  elements of, 600–601
solutions. *See also* features; web part solutions
  about, 310
  building in Visual Studio, 667–668, 676–706
  defined, 19, 597
  Feature Manifest file, 597, 598
  framework for Features and, 18–19
  installations and components of, 598
  installing with STSAdm command-line tool, 230
  managing farm, 229–230
  removing web part, 707–708
  Solution Manifest file, 600–601
  XML files of, 597–601
Source Lists, role in SharePoint workflow process, 567
Spell Check button (Publishing pages), 370
SPItemEventProperties object, 646
SPItemEventReceiver class, 646
SPQuery object, 624
SPServices account, configuring SharePoint from, 128
SPSite object, 620–621
SPWeb object, about, 620–621
SQL Reports, fixing, 111–112
SQL Server
  backups for, 232
  fixing SQL Reports, 111–112
  influence on SharePoint, 13
  installation checklist for server farms, 59–60
  installing, 79–83
  planning communication between servers and, 42
  planning requirements for, 49, 50–51
  post-installation adjustments and corrections, 110–112
  roles in SharePoint server architecture, 15
  unable to connect to, 111
SQL Server installation, 79–83
  account setup, 80–81
  adding Service Pack 2, 82–83
  applying Service Pack 1, 81–82
  steps for, 79–80
SQL Server Service Pack 1, 81–82
SQL Server Service Pack 2, 82–83
SQLExpress, 53
SSO (single sign-on) authentication
  about, 43, 741
  features of, 742–743
SSP Admin Site
  adding new trusted file locations, 157–159
  configuring Excel Service settings from, 154, 155–156
  setting User profiles and properties, 163–167
SSPs (Shared Service Providers). *See also* service levels for Shared Services
  about, 128–130
  account for creating and logging into, 154
  adjusting service levels for, 162–163
  administration Site for, 285
  changing associations with web applications, 281–283
  creating, 278–280
  default, 280–281
  enabling use of, 130–131
  Excel Calculation Services setup in, 154
  function of, 16
  handling installation errors for, 136–140
  illustrated, 129
  MOSS, 134–135, 140–142
  restoring, 283

Standard Edition of MOSS, 3–4, 222
starting workflow manually, 575–576
status
approval workflow, 573–574
building project to check document, 650–657
current backup, 236–237
deployed content, 241
timed jobs, 224–225
viewing Site's current workflow, 311
steps
about, 581–582
defining, 586
listing conditions for, 586–587
stopping load balancing, 58
strong name key (SNK) files, 684–685
STSAdm (SharePoint Server Administration) command-line tool
about, 112–113
adding and deploying web part solution, 701–702
command reference chart for, 715–736
extending, 737–740
installing solutions with, 230
schedule parameter for commands, 736
sub sites
about Site Directory, 356
adjusting navigation for, 299–300
administering, 285
creation rights for, 316–317
defined, 338
inheriting page layouts and site templates, 300–301
listing by category, 426, 439–440
modifying regional settings for, 312–313
MOSS, 360–361
My Site, 450
overview of, 198
relationships as SPWeb objects, 620–621
templates available for, 341
viewing and managing, 327
WSS, 345, 520
suggestions, web part, 405
Summary Link Web Part, 425, 433–434
System Account My Site settings, 444
System Application Event log, 619–620
system IDs (SIDs), 50, 69
System Master Page option, 298

T

Table of Contents Web Part, 425, 434–435
targeting, 742, 743
Team Services, 1
Team Site model, 285
Team Site Site Definition, 524
templates. *See also* list templates; Site templates
adding or removing MOSS Page Layout Templates, 312
assigning for content type settings, 461–462
displaying Site content and aspx page, 521, 522
Document Center Site, 531–532
downloading application, 519
lists and libraries saved as, 380
lists based on, 377
managing form, 270–271
Quota, 257
Report Center Site, 534–535
Site Content Types, 457
Site Directory, 533–534
uploading InfoPath Form, 272–274
workflows saved as, 575
testing
approval workflow, 573–574
event receivers and debugging, 658, 659–661
load balancing, 57–58
networks before SharePoint installation, 55
one-off web parts, 675
page added by Feature, 608

SharePoint Designer workflow, 591–594
web parts solution, 704–706
themes, 302–303
third-party web parts, 405, 431, 442
This Week in Pictures Web Part, 425, 435–436
Three-State Approval workflows, 570
time zones, 313
timer job definitions, 225–226
timer job status display, 224–225
title of Sites
adding to New SharePoint Site page, 350
modifying, 298–299, 553
Title property, 689
TitleIconImageUrl property, 691–692
TitleUrl property, 690–691
toolbar modifications, 491–492
tools
content deployment, 501
custom search, 638
workflow, 566–568
Top Link Bar, 303, 553
Top Sites Web Part, 426, 440–441
top-level Sites
Collaboration Site Type used only for, 519
defined, 338
no relationship between WSS, 520
user interface features for, 341–344
topology
devising formal, 32
mapping portal, 32
MOSS Site, 355–356
WSS Site, 345–346
trace logs, 219
tracking errors, 619–620
training, 35, 36
translatable columns, 334
Tree View navigation, 302, 553
troubleshooting
errors installing Office products, 86
Feature deployment, 609
installation issues to avoid, 53–54
loading problems for web parts, 706–707
repairing SharePoint Server, 193–194
steps before installing e-mail services, 77–78
uninstalling SharePoint, 194–195
Windows Search, 123
trust levels for web part DLLs, 672
trusted data connection libraries, 159–160
trusted data providers, 160–162
trusted file locations, 156–159
adding new, 157
allowing external data for, 157–158
configuring workbook properties for, 158
designating My Site, 171
enabling user-defined functions for, 159
entering address for, 157
selecting calculation behavior for, 158
setting session timeouts for, 158
turning off
Internet Explorer Enhanced Security, 70
Shutdown Event Tracker, 71
tweaking searches, 636–638

U

uninstalling SharePoint, 194–195
updating
converted documents, 497
HOSTS files on servers, 70
InfoPath Site Content Types, 472–473
web application settings, 187–188
with Windows Update, 78
upgrading
converting type of license after, 224
from MOSS Standard to Enterprise Edition, 222
upgrade path for server farms, 49
uploading
InfoPath Form Templates, 272–274
WSS Master Pages, 306–307

URLs
adding internal, 180, 228
debugging using, 675
editing public zone, 179–180
Image Web Part for setting image, 417–419
mapping to Internet zones, 227–228
options on New SharePoint Site page for, 350–351
setting Record Center Site, 543
setting to alternate CSS, 298
using query parameters as part of, 638–639
usage analysis logging, 176–177, 219–220
usage reports
Search, 175–177
viewing Site Collection's, 326
user alerts, 315
user interface, 335–354
about, 335
branding, 553
components of, 338–340
core web parts for, 412–413
creating Site Definitions from within, 548–549
creating Sites through, 344–345, 346–347, 357–360
defining for Site, 350–352
features of top-level Sites, 341–344
illustrated, 341, 342, 343, 344
lists, 392
navigation options for, 351–352
Recycle Bin, 352–354
selecting Site templates for new, 351
setting permissions for, 351
WSS and MOSS, 335–337
User profiles. *See also* personalization
designing profile page for My Site, 169–170
handling personalization in, 19–21
importing, 164–167
modifying Profile Service Policies, 167
user-based workflows, 569–573
user-defined assemblies, 161–162
user-defined functions, 159
user-defined workflows, 277
users
adding to group, 214–206, 292–293
advantages of web parts for, 405
alerts for, 315
assigning to workflow in Office, 577–581
content guidelines for, 29
controlling public view of Profile Page, 445
creating custom queries of content and Site, 427–429
deleting from Site Collection, 293–294
deployment strategy for, 35, 36
enabling/disabling ability to create Sites, 260–261
governance for, 28–29
number per workflow instance, 569
permissions for, 261–262, 289
personal web part property values for, 406
personalizing Site for, 437–438
policies for web application, 262–263
profiles and properties for, 163–167
roles in workflows, 564, 568
showing list of Site's, 413, 421–422
summarizing Site Collection task lists for, 413, 422–423
uploading documents to SharePoint, 401
viewing, 291–292

V

validating links in Site Directory, 226–227
variable data for SharePoint Designer workflows, 582
variation logs, 500
variations, 497–501
about, 497–498
defining Site, 332–333
enabling, 498–501
labels for, 333–334, 499
Relationship List, 497, 500–501
Site Administration settings for, 498

viewing publishing logs for, 334
viewing variation logs, 500
vary-by-custom handlers, 515
verifying
ECS service running, 154
IIS isolation mode setting, 73
SharePoint Designer workflow, 590–594
solution installation, 702–704
viewing
assigned workflow tasks, 578
backup history, 234
content modification logs, 321
content type published, 464
permissions, 291–292
records on Records Center Sites, 544
rights for workspaces, 316–317
servers in server farms, 201–202
Site Collections details, 259
Site content and structure in modification logs, 321
Site usage reports, 314
Site workflows, 311
sub sites, 327
timer jobs, 224–225
usage reports for Site Collection, 326
user alerts, 315
users and groups, 291–292
web part solution, 703–704
views. *See* list views
virtual directories, 93–96
Virtual Directory Creation Wizard, 94
virtual servers, 50, 69
Visitor Site group, 289
Visual Designer, 565
Visual Studio
adding Workflow Extensions for, 85
applying Service Packs 1 and 2 for, 85
building web parts in, 667, 708–712
creating Site Definitions in, 551–552
customizing Page Layouts in, 557
debugging using Attach to Process window, 675–676
declaration line added to XML files, 602
deploying Site Definitions from, 552
designing Master Pages in, 477
event receivers project in, 650–657
Feature building in, 602–607
installing, 83–85
referencing Web Services, 630–631
Windows SharePoint Server Extensions with, 85, 105–106, 667
working with workflows in, 563, 567, 568, 581

W

web applications
about, 131
application pools accounts for, 211
changing associations with SSP, 281–283
correcting Site access issues for, 149
creating, 131–134, 242–245
deleting, 139–140, 247
designating authentication providers for, 263–264
entering port settings for, 132
extending, 245–246
outgoing e-mail settings for, 249
security for, 260–264
selecting or switching, 253–254
setting up first, 143–148
settings options for individual, 250–251
Site Collections for, 254, 255–256
unused Sites in, 256
updating settings for, 187–188
user permissions for, 261–262
user policies for, 262–263
working with application-wide features, 252–253
Web Content Authoring, 489
web pages. *See* pages
Web Part Gallery
about options in, 310–311
access to, 260
loading web part to, 672–675
viewing web part solution in, 703–704

Web Part Pages
about, 364, 664
creating, 367–369
security options for, 260
Web Part Properties tool pane, 408
web part solutions, 676–706. *See also* .webpart files
adding and deploying, 701–702
adding code to class, 677–683
building .webpart files, 696–697
creating new web part project, 677
DDF file for, 699–700
defining and adding resources for, 683–684
Element Manifest file for, 665, 666, 697–698
Feature file for, 665, 698
Manifest file for, 698–699
overview, 676–677
removing, 707–708, 712
setting parameters for, 685–687
signing assembly, 684–685
steps in building, 667–668
testing, 704–706
verifying installation of, 702–704
.webpart file for, 687–697
WSP created for, 701
Web Part Zones, 367, 368, 664
web parts, 663–714. *See also* List View Web Parts; one-off web parts; solutions; web part solutions; .webpart files
about, 9, 340, 405
accessing properties of, 408
building, 664–665, 667–668
Categories, 405, 426, 439–440
Colleague Tracker, 447, 448
Colleagues, 447, 449
components of, 663–665
connecting, 408, 411–412
Contact Details, 426, 441
Content Editor, 413, 414–415
Content Query, 427–429
core, 412–413
creating custom lists for use with, 408–410
debugging, 707
defined, 663
direct deployment of assemblies, 669
Filter, 425, 437–438
fixing loading problems for, 706–707
Form, 413, 416–417
Get Started with My Site, 447, 449
I Need To..., 425, 429–431
how they work, 665–666
Image, 412–413, 417–419
In Common Between Us, 447, 449
included with WSS Site templates, 523
installed as solutions, 19
list view, 410
managing solutions properties for server farms, 229–230
Memberships, 447, 449
methods for rendering, 363
MOSS-specific, 424, 425–426
MSBuild to create WSP files, 712–714
My Blog, 446, 447
My Calendar, 446, 447
My Contacts, 447, 448
My Inbox, 447, 448
My Links, 447, 449
My Mail Folder, 447, 448
My Pictures, 447, 449
My SharePoint Sites, 447, 449
My Tasks, 447, 448
My Workspaces, 447, 449
Object Model reference added to, 612
one-off, 667, 668–676
Page Viewer, 413, 419–421
previewing or modifying definitions of, 310–311
properties of, 406–407
Publishing pages, 489
Recent Blog Posts, 443, 447, 450
Relevant Documents, 420–421
Relevant Documents With Part, 413
removing Organization Display web part from My Site, 562
removing solution, 707–708, 712
RSS Viewer, 425, 431–433
SafeControls for, 600, 671–672

Search, 425–426, 438–439
SharePoint Documents, 447, 450
Site Aggregator, 424, 426–427
Site Users, 413, 421–422
Sites in Category, 426, 440
solutions for, 667–668, 676–706
suggestions for, 405
Summary Link, 425, 433–434
Table of Contents, 425, 434–435
third-party, 405, 431, 442
This Week in Pictures, 425, 435–436
Top Sites, 426, 440–441
Users Tasks, 413, 422–423
using Visual Studio to build, 708–712
XML, 413, 423–424
web servers
installation checklist for server farms, 60
planning software requirements for, 50–51
Web Services
functions of, 628
list of, 629–630
referencing, 630–632
SharePoint extensibility and, 611
websites. *See* Sites
web.config file
about, 517
adding SafeControl to, 671–672
retrieving impersonation information using, 618–619
.webpart files, 687–697
about, 663, 665, 687
building, 696–697
custom properties for, 694–695
default properties for, 689–694
further information on, 689
getting assembly information, 695–696
outline of XML for, 688
webserver role, 15
Welcome Page, 301, 553
wfc.exe compiler, 583
Wiki Sites
about, 525
editing and creating Wiki Pages, 525–526
Site template for, 522
Windows authentication
about, 43, 741
features of, 742–743
Mixed Mode authentication vs., 80
specifying, 264
Windows Event Logs, 70–71
Windows Identity object, 613
Windows Search. *See* Windows SharePoint Server Search Service
Windows SharePoint Server Extensions
adding, 245–246
adding parameters from SharePoint Solution tab, 685–687
building web parts with, 708–712
downloading, 85
installing, 105–106
removing from IIS Web Site, 246
using, 667
Windows SharePoint Server Search Service
running on front-end servers, 119, 123
searching items in list, 624–625
setting up, 122–123
troubleshooting, 123
using in MOSS, 21
Windows SharePoint Services. *See* WSS
Windows Update, 78
Windows Workflow Foundation. *See* WWF
Word
converting documents to SharePoint page, 493–497
creating workspace Site from, 361–363
Document Workspaces in, 84, 361–363, 545–546
launching to allow assigned workflow tasks, 579–580
starting workflows in, 594–596

workbooks
ECS cache settings for, 156
selecting calculation behavior for trusted file locations, 158
setting size for trusted file locations, 158
trusted file locations for, 156–159
Workflow Extensions for Visual Studio, 85
workflows, 277–283, 563–596. *See also* building SharePoint Designer workflows
assigning specific user in Office, 577–581
avoiding event receivers to create, 646
basics of, 566–568
building in SharePoint Designer, 581–594
changing SSP/application associations, 281–283
component overview of, 567
configuring in Central Administration, 277–283, 569
content type configurations, 462
creating new SSP, 278–280
defined, 340
design considerations for SharePoint Designer, 583–584
editing in SharePoint Designer, 594
event receivers vs., 645
goals for, 563–564
limitations of, 568–569
managing Shared Services for server farms, 278
options for customizing, 572–573
Publishing pages and, 372–374
restoring SSP from backup, 283
reusing as template, 575
roles of individuals in, 564, 568
settings for, 149–150
starting, 575–576, 594–596
testing, 573–574, 591–594
types of user-based, 569–570
user-defined, 277
verifying and publishing SharePoint Designer, 590–591
viewing status of Site's current, 311
Workflows dialog (Office applications), 595
workspaces
creation rights for, 316–317
Document and Meeting, 545–548
Office Sites as, 545
Site templates for, 522
WSP files
about, 598, 601, 664
building automatically, 712–714
creating with MAKECAB.exe utility, 701
defined, 665
WSS (Windows SharePoint Services). *See also* WSS Site templates; WSS Sites
about, 3
accessing Site Collection Site Administration, 286
additional features of MOSS, 8
Advanced Settings button for Configuration Wizard, 102
architecture of, 12–13
Collaboration and Meeting Sites supported by, 344
composition of Sites and pages, 16–18
configuration options for, 39
core services of, 4, 6–7
core web parts available, 413
creating first application for, 94–95, 148–149
creating sub sites for, 345
data storage illustrated, 7
development of, 1–2
Features and Solutions Framework for, 18–19
features of user interface for, 335–337
integrating workflow into user interface, 564
modifying top navigation bar, 303
permission groups in, 290
pre-installation decisions for, 59

principals for Sites, 7–8
running on domain controller, 75
setting up Windows SharePoint Server Search Service, 122–123
Site topology in, 345–346
types of SharePoint lists, 401–403
WSS Site templates
types of, 522–523
web parts included for, 523
WSS Sites
content storage for, 521–522
creating Basic pages, 365–367
elements of, 520–521
list and library types included for, 523–524
modifying Quick Launch properties for, 304
organization of, 355
saving as template, 304–305, 548–549
Site Definitions for, 524–525
support for Basic and Web Part Pages, 364–365
types of templates for, 522–523
uploading or deleting Master Pages, 306–307
using Master Pages in, 476–477
Web Part Page creation for, 367–369
WWF (Windows Workflow Foundation)
about, 563, 564
features of, 564–565
illustrated, 565
role in SharePoint workflow process, 567
SharePoint implementation of, 565–566

X

XML files. *See also* .webpart files
building .webpart, 696–697
creating .webpart, 687–688
declaration line added in Visual Studio, 602
Elements Manifest file, 597, 598–600, 604–605, 609
Feature Manifest file, 597, 598, 609
further information on .webpart, 689
Solution Manifest file, 600–601
XML Web Part, 413, 423–424

Z

zones
illustrated, 525
locating for installed web part, 704–705
Web Part, 368, 664