Chapter 9

Content Management

Without content, a portal would be a lonely place. The volume and richness of web content are what brings users back to a portal again and again, whether it is an internal portal for knowledge management or an external portal to create a virtual community and a market for goods and services. While portal content could be created as most web sites are created, one page at a time through authoring tools such as Microsoft FrontPage or Macromedia Dreamweaver or through HTML coding in a simple editor, major portals rely on content management systems to automate the creation, editing, and maintenance of thousands or hundreds of thousands of web pages. You should consider content management for your organization, as it can have the biggest payoff of all your portal initiatives.

In many ways, content management is at the heart of a portal, whether it is an enterprise portal or an external portal, because so much content resides in web pages. The content management system is also an embodiment of the site taxonomy, and it governs the site navigation. It creates the consistent "look-and-feel" that is essential to giving users a positive experience in the portal. If all you did to implement a portal was to implement content management, migrate all pages to the system, and enable users to create their own fresh content, you would be providing quite an enhancement over what most web sites offer.

The first content management systems were created as custom solutions, and some served as the basis for what later became commercial content management systems. Content management systems consist of a repository where content is stored, one or more frontends for authoring and other management tasks such as review and approval, and additional components to enforce content management business rules and provide services such as notification to authors and reviewers of changes in the status of a page. The repository is typically a relational database, and the frontend systems are most often browser-based.

Measuring Return on Investment for Content Management

Automating content management is one of the portal initiatives with the highest return on investment. While portal content does not necessarily bring revenue in the door like the commerce section of a portal, organizations with large, complex web sites spend a great deal of time and money on maintaining that content, and a content management system (CMS) can result in significant cost savings.

The first value in the equation is to determine how much you are already spending on content management. How many hours of your web team's time are spent on creating or editing content? Is your webmaster responsible for fixing typographical errors in content? What is the cost of this time? What is the hourly rate or salary for the people engaged in content management? Don't forget to include the time of content authors who submit items to be posted on the web site. How much time do they spend on this activity?

Next, what is the lag time in posting content, from authoring to going live? The delay is often significant because the web team can be a bottleneck that slows down the process of updating and adding content. What is the value of more timely updates to your web site? Would improved communication on the web site increase customer satisfaction or make other constituents happier? How much would that improvement be worth to you?

Now calculate the cost of implementing a content management system. What would be the cost for hardware, software, maintenance, implementation, and conversion services? Over how many years should the investment be amortized?

This return on investment exercise may be enough to interest management in the development of your portal. By adding content management capabilities, you can significantly increase the growth of portal content, thereby leading to higher use of the portal.

Content Management Options

Content management products run the gamut from simple and inexpensive (or even free) to complex, powerful, and expensive. So it is within the

Microsoft product pantheon, which provides three choices for content management.

First, you can continue in the tradition of custom content management and create your own system by building .NET controls along with a database repository to store the underlying content. This is the best choice if you are being paid by the hour and have an unlimited budget and schedule. You can tailor the system to fit your unique requirements.

The downside to the custom CMS is the same as for any other custom solution. It requires a significant amount of resources to design, build, and maintain compared to an off-the-shelf solution. The blank slate offered by custom development may encourage users to be more creative in their requirements than the business case may warrant, or to ask for functionality that in the long run would be unnecessary or counterproductive. Custom software is only as good as the quality assurance process applied to it, and a surprising amount of work is required to bring software defects down to an acceptable level.

In short, building a custom CMS made sense only until commercial products were available to fill this niche. Now that the price of a CMS is dropping, the functionality of a custom system built with the same investment as a product purchase is dropping correspondingly, making only the simplest custom CMS worth building. For instance, Microsoft announced a small business edition of Microsoft Content Management Server that is only one-fourth the price of the full product (already competitively priced).

A second product option for content management has emerged. SharePoint Portal Server (SPS) has limited content management functionality built in, and with some creativity you can build an enterprise portal with a broad range of content and functionality without a full-bore CMS. For instance, SPS contains prebuilt pages with content management capability for such items as news, contact (name and address) listing, calendars, document libraries, threaded discussions, and others.

For most enterprise portals, you will need an industrial-strength content management solution. Fortunately for .NET portal developers, Microsoft has an off-the-shelf content management product built on .NET that is tightly integrated with .NET security, Visual Studio, ASP.NET, and web services. It is rather unimaginatively named Microsoft Content Management Server (MCMS).

This chapter provides an overview of how to plan content management for your portal, ways to implement that plan on the Microsoft platform, and the integration points for content management with other elements of the

portal and .NET. I start by describing the planning and development process. The bulk of the chapter is devoted to implementation in MCMS, Microsoft's enterprise content management system. I also describe the content management features of SharePoint Portal Server 2003, as these are quite interesting for building internal portals. I discuss the option of a custom .NET content management solution. In every case, the place to begin is with a site framework.

Site Framework for Content Management

A web site or portal needs a foundation, and content management allows the developer or architect to lay a site framework that will ensure a solid site in the future. This section describes the approach for laying this foundation.

Our methodology is based on the Microsoft Solutions Framework (MSF) and the Rational Unified Process. MSF is a spiral process model. Over the course of a project, all the steps are carried out several times. Figure 9.1 shows the key steps in the development methodology, showing only one trip around the spiral. For more information on the Microsoft Solutions Framework, go to www.microsoft.com/business/services/mcsmsf.asp.

The MSF methodology consists of four broad phases:

1. *Inception*
    - Major Activities
        - Formulate the scope of the project, that is, capture the context and the most important requirements and constraints to derive acceptance criteria.
        - Plan and prepare a business case and evaluate alternatives for risk management, staffing, project plan, and trade-offs between cost, schedule, and features.
        - Synthesize a candidate architecture, evaluate trade-offs in design, and assess make/buy/reuse decisions so that cost, schedule, and resources can be estimated.
    - Work Products
        - Vision document containing a general vision of the core project's requirements, key features, and main constraints.
        - Initial project glossary.

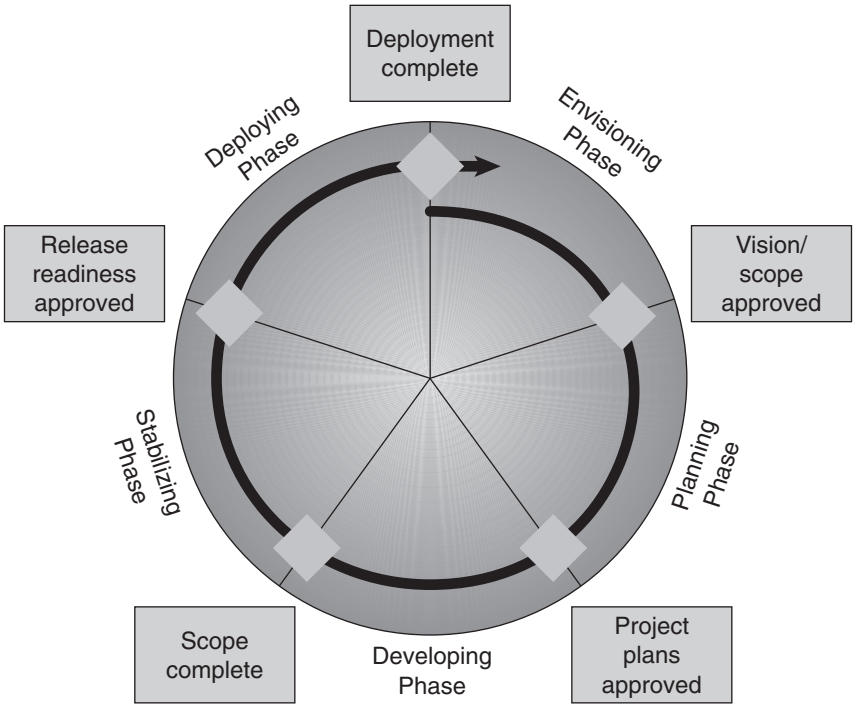

Figure 9.1 MSF Methodology Overview

- Initial business case that includes the business context and success criteria.
- Initial risk assessment.
- Project plan that shows the phases and iterations.

2. *Elaboration*

- Major Activities
  - Vision is elaborated, and a solid understanding is established of the most critical features that drive the architectural and planning issues.
  - The process, infrastructure, and the development environments are elaborated, and the process, tools, and automation support are put into place.

  - The architecture is elaborated and the components are put into place. Potential components are evaluated, and the make/buy/reuse decisions are sufficiently understood to determine the construction phase cost and schedule with confidence. The selected architectural components are integrated and assessed against the primary behavior scenarios.

- Work Products
  - Requirements document that captures in detail functional, maintainability, extensibility, scalability, security, and usability requirements.
  - Software architecture description.
  - Revised business case.
  - Revised risk assessment.
  - A development plan for the overall project, including the coarse-grained project plan, which shows iterations and evaluation criteria for each criterion.

3. *Construction*

- Major Activities
  - Resource management, resource control, and process optimization.
  - Complete component development and testing against the defined evaluation criteria.
  - Assessment of product releases against acceptance criteria for the vision.
- Work Products
  - The software product integrated on the adequate platforms.
  - The user manuals.
  - A description of the current release.

4. *Transition*

- Major Activities
  - Beta testing to validate the new system against user expectations.
  - Training of users and administrators.
  - Rollout of the product.
- Work Products
  - Software acceptance and rollout.

All this methodology information may seem a bit much for a portal project, but it is quite important to manage a portal as you would manage any other significant development effort. Some customers have approached us to build portals without asking users about their requirements or verifying whether the capabilities the portal will provide will help users. This is a risky use of expensive resources.

Through the lens of our methodology, the MCMS portion of the portal project begins with review meetings and analysis to create a vision for the site in the inception phase. The elaboration or design phase calls for detailed functional specifications and design for elements including the templates and the navigation. The construction phase witnesses the implementation of channels, templates, and resources as well as supporting infrastructure implementation, including hardware and software installation. The transition phase involves putting the site into production, training, maintenance of the site, and preparation for the next round of development. The entire cycle may be iterated a number of times.

You may have your own methodology and want to map these activities accordingly. Having a solid methodology will reduce the risk of the project and improve the quality of the results. It also ensures continuity of a project when personnel change and safeguards against finger-pointing when schedule slips or scope changes.

MCMS lends itself to an iterative team development process much more than a typical page-at-a-time web site. You can approach the channels and templates in a modular fashion. While one team is fielding the templates and testing the workflow for the marketing department, another can be independently developing the human resources channel and integrating job application web pages with an existing PeopleSoft installation. When channels and templates pass user acceptance testing, the real work of cranking out or converting content begins, but developers have already been freed to focus on other development tasks.

Site Vision

The first step in planning the site is to hold a design review session to map the content and organization of the site with the stakeholders of the portal. Most portals start from existing web sites rather than from scratch, so it is essential to identify the pages in the web sites that will be converted to the CMS. Some dynamic content such as online stores or interfaces to databases and other applications will not be converted to MCMS; instead, this content will be integrated into the larger .NET Framework for the portal. A good place to start is with the static pages of the site, which typically constitute the vast majority by page count.

You can use the current site homepage as the baseline for a master template. Use it as the basis for suggestions during the design review meetings when graphic design and navigation decisions are made based on user input. During those meetings, participants provide graphic design and graphic elements for use in MCMS templates. The client freezes the graphic design at the conclusion of this phase. A storyboard, or static mockup of web pages, is a useful tool to capture this design.

Next, audit the existing web site to determine which components will run without modification in the new architecture, which will be migrated, and which will have to be rewritten entirely. At this time, measure the performance of the current site or sites as a benchmark for the new site. Include measures such as average response time for various operations.

> ***TIP*** *Microsoft Visio provides an excellent tool for this stage of designing your portal. Use the Conceptual Web Site Shapes under the Web Diagram provided with Visio to sketch out the general plan for the site.*

The deliverable from this phase is a site migration plan that enumerates the pages to be migrated, the target navigation, and the high-level template design. The site migration plan shows which pages will be migrated and how they map to sections in the new site. It also lists the templates to be created and describes the graphics and other elements to be used in the templates.

Taxonomy and Container Hierarchies

Once the overall vision for the site is set, you implement the site taxonomy using MCMS channels. A **taxonomy** is a hierarchy of categories under which content is organized. It helps users navigate and search in a portal. Chapter 10 provides guidance on how to create a taxonomy and some sample taxonomies. For the examples in this chapter, I use simple site taxonomies.

Microsoft recommends that you create channels before building and populating the other containers in MCMS because the channels are the vertebrae of your content management system. Later in this chapter I describe the steps in defining the channels. At this point you develop the template and resource galleries.

Security Implementation

This stage is a good time to choose the security implementation for the portal. Start by creating a simple login page to grant developers access to MCMS, and then add content creators and others as the site progresses.

Template Creation

You need to develop a base or skeleton template that includes shared page elements and navigation. It can be the basis of other templates for the site. Develop custom placeholders as required at this stage. You want to create the base templates as quickly as possible to allow the content creation and migration process to begin. Subsequent template development may be carried out in parallel to the content creation process.

Content Creation and Migration

The next step is to create content based on the templates and migrate existing content to the site. This is also the time to clean up the site by removing dated content, and editing and refreshing content as identified in the vision phase of the project. In the case of SharePoint Portal Server, this means migration of documents such as Office documents as well as the creation of HTML pages.

Site Testing

Site testing for functionality and performance is a continuous process that can be initiated once the framework is complete and content is being added to the site. Use whatever testing methodologies and automated tools you choose for the testing process.

Some vendors offer site-testing products integrated with MCMS. For instance, Coast (www.coast.com) provides testing and quality assurance tools for web sites during development and postdeployment. Their product becomes part of the MCMS routing and approval process, checking for errors such as broken links, inappropriate content, and compliance with accessibility guidelines such as Section 508 (http://www.section508.gov/). Section 508 is a U.S. government regulation which amended the Rehabilitation Act in 1998. It calls for Federal government agencies to make electronic and information technologies available to people with disabilities. HiSoftware (www.hisoftware.com) also integrates its solution with MCMS.

This package, called Solution for Online Accessibility, includes Content Management Server and HiSoftware products to help customers achieve Section 508 compliance (http://www.hisoftware.com/soa/).

Content Management Server Overview

Microsoft Content Management Server is designed to provide enhanced authoring and management of web sites of all sizes, including routing and approval processing of content, freshness dating for content, and tight integration with .NET. Therefore, in many ways it is the center of the overall .NET portal framework. For outward-facing portals, the bulk of content consists of web pages, and keeping those pages up to date is the most daunting task for a webmaster.

MCMS has evolved from a product called Resolution created by the nCompass, which was purchased by Microsoft in May 2001. The 2002 version of MCMS is the first major overhaul of the product since the acquisition, and this effort dramatically remolded nCompass into the .NET Framework and web services paradigm. The 2001 version was rebranded by Microsoft but was not very different from nCompass's product.

The following sections of this chapter introduce these MCMS features:

- Site framework
- Content creation and approval process
- Site navigation
- Site management
- Template creation
- .NET and web services integration

There is insufficient space here to provide a feature-by-feature tutorial on MCMS. Our goal is to introduce the key concepts and then go beyond the product help and documentation with practical guidance on how to use the product to achieve the overall goals of the portal.

MCMS Architecture

MCMS provides a repository to store content, along with content creation and site management tools as well as APIs to connect to other services in

the portal. It consists of the following logical elements that make up its architecture:

- *Content Repository.* SQL Server is the database where content is stored for use in the site.
- *Site Manager.* Webmasters are provided a rich Windows client to manage many aspects of the MCMS site.
- *Authoring Connector.* This component allows authors to create content in Word XP and post it directly to MCMS. Note that this feature works only with Word and not with other Office programs.
- *Content Connector.* The Content Connector component allows programs to access content in MCMS without risking data corruption with a direct connection to the MCMS database tables.
- *Publishing API.* MCMS 2002 contains a number of APIs that are wrappers around COM (Component Object Model) APIs used in earlier versions, and also new APIs to be used within the managed .NET environment. The managed APIs in the Microsoft.Content-Management.Publishing namespace are known as the publishing API.
- *ISAPI*. The Internet Server API is the API for Internet Information Server. It provides one of the means of extending MCMS and also provides security services for HTTP requests by means of ISAPI filters.
- *APIs*. Programs can access the CMS content by means of APIs, such as the publishing API.
- *Word XP*. Users can publish directly from Microsoft Word XP without using the web user interface to MCMS. Developers can create authoring connectors to streamline this process. These are essentially wizards that define the format of the content and the location where it will be published. Once the user creates the content, it enters the standard MCMS routing and approval process.

Additional elements in the architecture (Figure 9.2) are outside of MCMS per se, but remain critical elements in the total portal picture:

- *Visual Studio.NET.* For developers, Visual Studio.NET is the integrated development environment for MCMS, and it is difficult to imagine implementing MCMS without Visual Studio. This is where templates are created, and .NET web services are created and

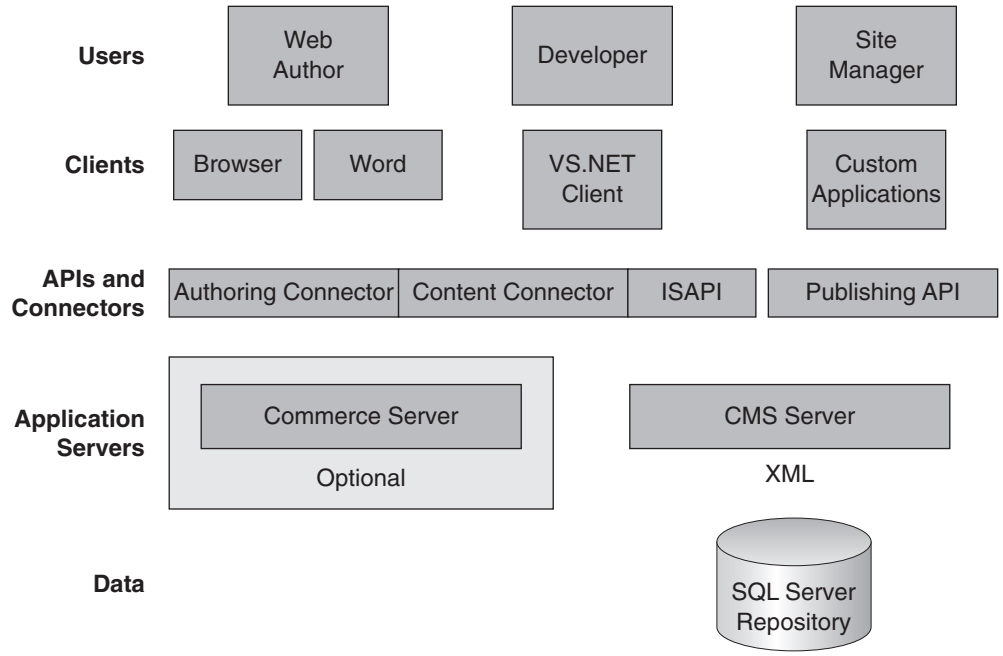

Figure 9.2 MCMS Architecture

maintained. With MCMS 2002, sites can be saved as Visual Studio projects. This means that .NET tools are used to directly manipulate every aspect of MCMS and extend it to provide new functionality.

- *Commerce Server.* While MCMS provides authorization-based personalization, it does not provide rule-based personalization capabilities, which are available in Commerce Server 2002. See Chapter 7, "User Profiles," and Chapter 8, "Personalization," for detailed information on how to implement personalization. As with MCMS 2002, Commerce Server 2002 sites can be saved as Visual Studio projects.

The MCMS architecture is extensible because it has been built on the .NET Framework and integrated with other Microsoft enterprise servers such as Commerce Server, SharePoint Portal Server, and BizTalk. In the next product generation, this integration will probably be even tighter, and code reuse will be even greater.

Site Framework for Content Management Server

A web site or portal needs a foundation, and MCMS allows the developer or architect to lay a site framework that will ensure a solid site in the future. The MCMS framework consists of three main elements. An MCMS site is stored in a hierarchical logical structure based on channels, template galleries, and resource galleries. The channels are used to organize content for site administration and also to create navigation. Channels should be based on the taxonomy you develop for your site, as explained in Chapter 10, "Developing a Taxonomy."

Channels

Channels are containers that can contain web pages or other channels (subchannels). For instance, a human resources site could have channels such as leave policy, benefits, health care, and salary history. The health care channel could have subchannels for medical, dental, and optical insurance.

One way that channels are revealed to users is in the navigation. Unfolding menus may belie underlying channels, as may left navigation with a tree structure. For instance, on the Information Strategies corporate web site, there is a page devoted to books written by the company's consultants (Figure 9.3). The left navigation shows that this page is in the Books subsection of the Resources section of the web site. Users appreciate as many navigational aids as you can provide to keep track of where they are, where they have been, and where they are headed.

You can expand and refine your channel structure as your project proceeds, but you will save time and effort by starting with a well-developed group of channels (Figure 9.4).

Do not use a root folder in the channels to store content. You should leave at least one level to store only child folders to make expansion of the site simpler. For instance, if you later choose to implement localization, you will want to create parallel channels for each of the languages you support, such as English, French, and Spanish channels.

> ***TIP*** *Do not use the Site Manager to move pages from one channel to another. This is better accomplished through the browser interface as one of the edit functions on a page menu.*

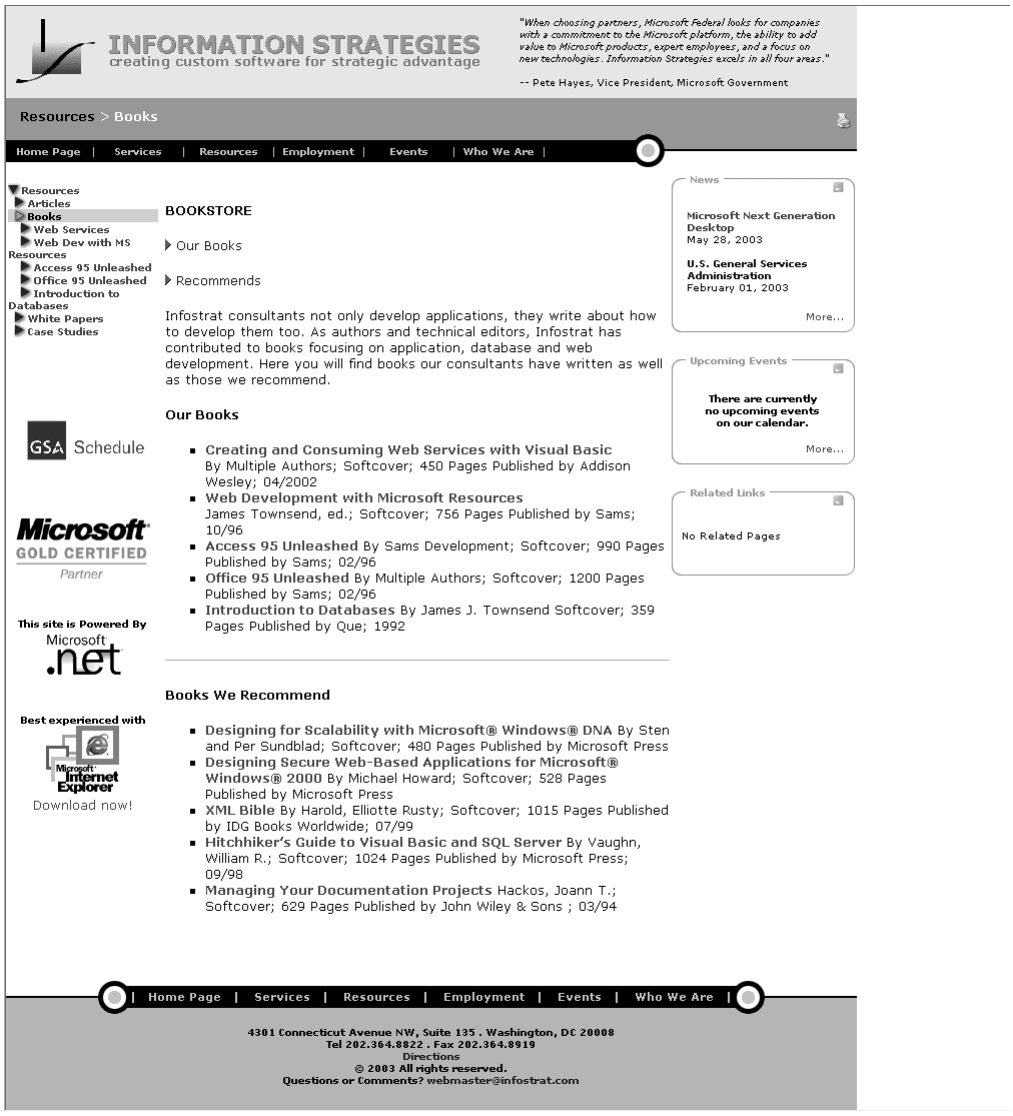

Figure 9.3 Taxonomy as Shown in Left Navigation

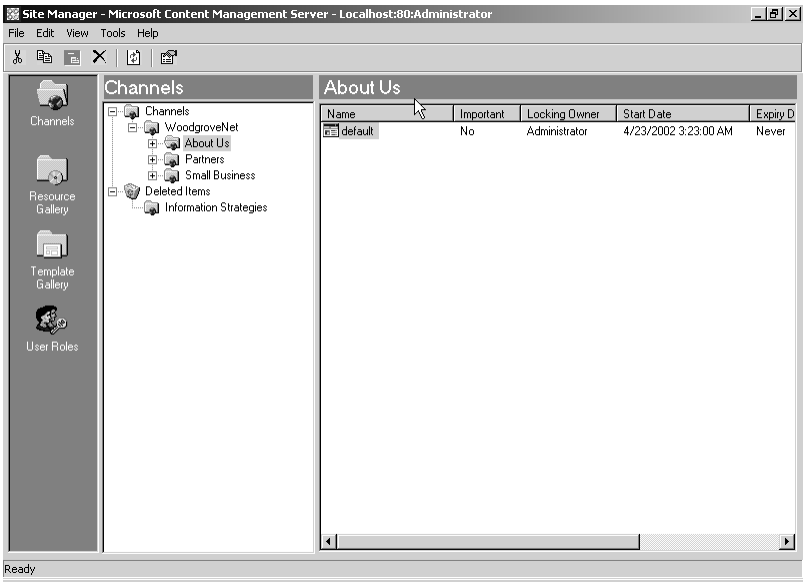

Figure 9.4 Channels

Template Galleries

Template galleries provide a place for site administrators to store the templates used for each section of the site. A channel is likely to have multiple templates associated with it. For an enterprise portal, the human resources channel would have templates for job postings, benefits, and policies such as leave. As with channels, template galleries may have subfolders and be arbitrarily deep. While the template gallery structure often mirrors the channels, it can be different from the channel structure because some templates are used across multiple channels.

You access template galleries from Site Manager and from Visual Studio.NET. Installing MCMS 2002 creates the Template Explorer in Visual Studio.NET, which displays the templates and their properties (Figure 9.5).

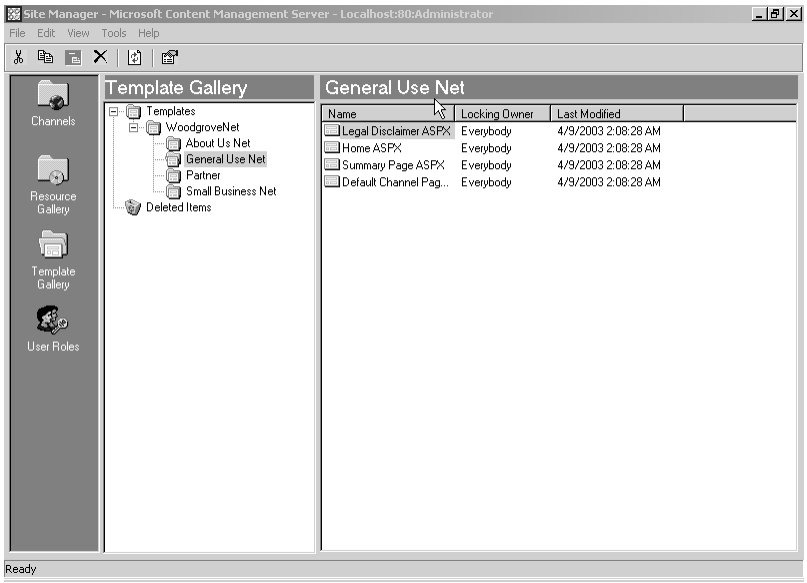

Figure 9.5 Template Gallery

In a typical web site, each template would have dozens or hundreds of pages associated with it. For instance, a product or service listing would be composed of many pages based on the same template. Each template might have several shared resources such as graphics or other design elements that would come from the resource gallery.

Resource Galleries

A resource gallery stores files used by the site, such as shared graphics, audio, or video files. Company logos and trademarks should be stored in a gallery to make them easily available to site editors and template creators. Like a template gallery, the resource galleries can contain subfolders (Figure 9.6). The resource gallery structure may reflect the structure of the template galleries, because the resources are typically used in the context of templates.

The resource gallery gives access to objects to many people in the organization. Therefore it is not the place to store files that must be secured or

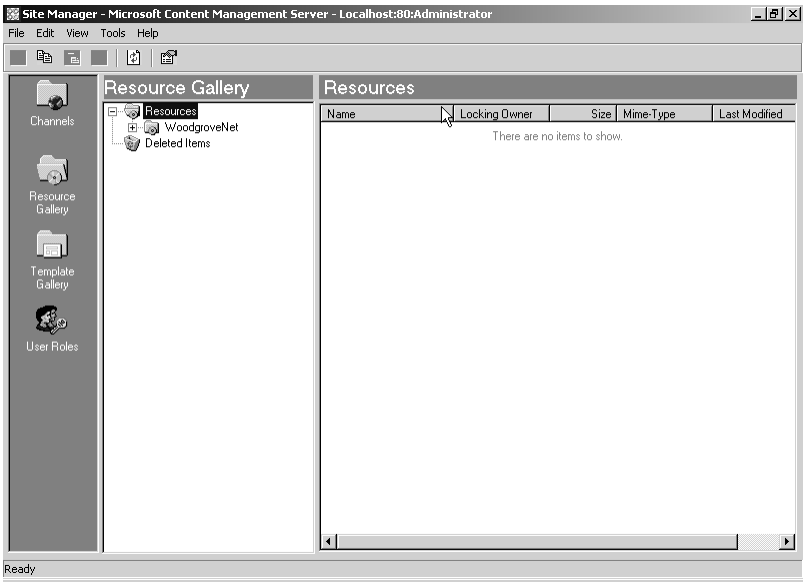

Figure 9.6 Resource Gallery

locked to prevent updates. It does not allow granular permissions for users to be managed in the same way as if the resources appeared on a single page in a channel.

Starting the Portal Site in MCMS

As with all .NET development, the starting point for MCMS is Visual Studio.NET. This is the tool you use to create templates for your site. To do so:

1. In Visual Studio .NET, select **Other Windows** from the View menu.
2. Click **MCMS Template Explorer** (Figure 9.7).

The Template Explorer reveals the template gallery structure of the site. To enhance maintainability, you should create an intelligible hierarchy of templates rather than lumping them all into a single folder.

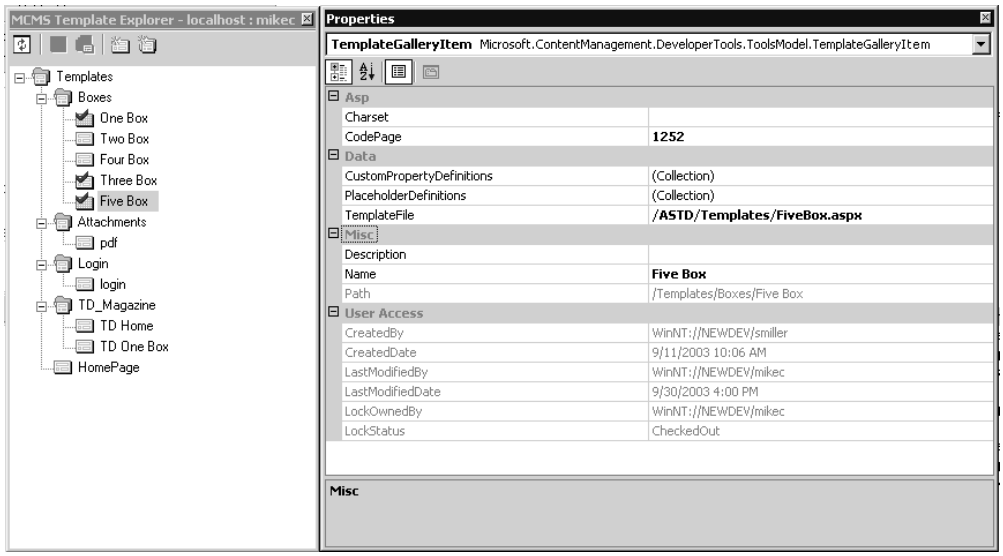

Figure 9.7 Template Explorer in Visual Studio.NET

Figure 9.7 displays two types of templates. First are generic templates with standard layout, labeled "One Box," "Two Box," and so on. These contain the standard page elements such as header, footer, and navigation, along with placeholders for content in single or multicolumn format. They are used throughout the site in many different channels. The second type of template is tailored for the special requirements of a subsite or section of the site. These special templates are used for one-off pages such as the login page or a page to list and view attached PDF files. The TD_Magazine template has a layout that is unique to that publication.

Users see the list of templates, organized in the template gallery, when they create new pages (Figure 9.8).

Selecting the gallery displays a list of templates. After a user chooses a template, he enters content in the content placeholders and saves the page. The page is stored in the appropriate channel based on where the user began in the authoring process.

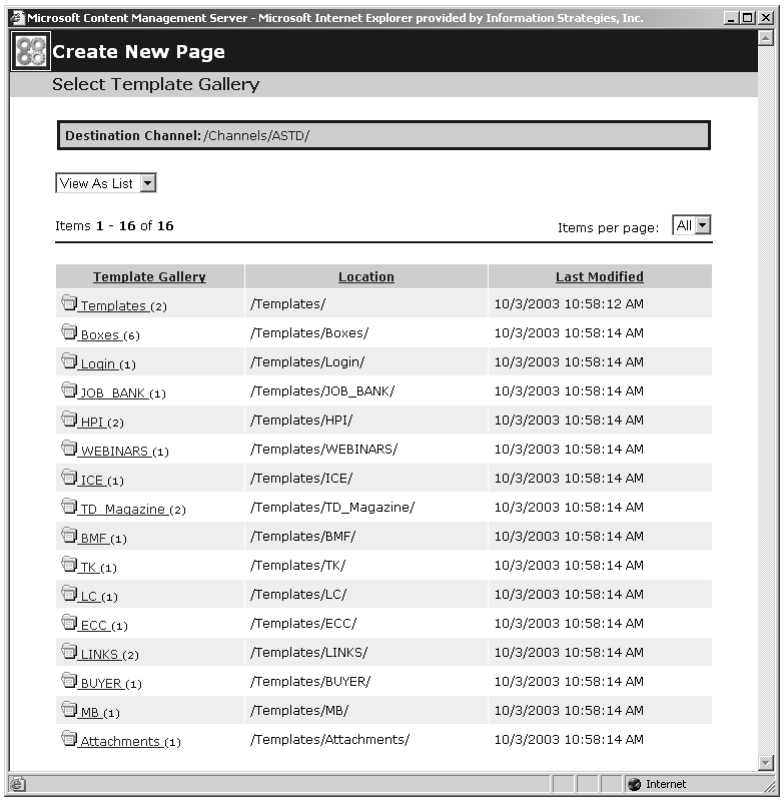

Figure 9.8 Template List

Creating Templates

Authors use templates to create new web pages. Templates consist of page elements such as text and graphics along with placeholders that are used to capture input from the authors. For instance, a press release template would have placeholders for the date, title, text, and contact information. The same template would be used for all press releases, so updating the template would update the appearance of all press release pages. One way

to define a new template is to right-click the template gallery in MCMS Template Explorer, and select New Template. Another way is to create the template in Visual Studio. You can rename a template you have just created. In either case, you use Visual Studio.NET to edit templates.

In MCMS 2002, templates are stored in ASP.NET files. They are checked in and out of the template library by the developer. This functionality is important to coordinate the efforts of multiple developers working on the portal. All developers share a common MCMS repository, and they check out templates as they need to access them.

Placeholders are ASP.NET server controls. This tight integration with .NET gives developers close control over the behavior of the templates and the resulting web pages. It also means that template developers need to learn and master .NET programming. These skills carry over into other portal products. For instance, SharePoint Portal Server now uses the same .NET approach to creating web parts.

The standard placeholders that ship with MCMS are:

- *HTML placeholder.* Probably the most common placeholder; accepts HTML strings including text, formatting, and hyperlink tags.
- *XML placeholder.* Designed for data formatted as XML.
- *Attachment placeholder.* Allows user to insert a file as an attachment; displays resulting link to attachment.
- *Image placeholder.* Stores image file such as JPEG.
- *Office Attachment placeholder.* Contains Office file as an attachment.
- *Office HTML placeholder.* Stores HTML generated by an Office product such as Microsoft Word.

You can also create your own new placeholder definitions with the Custom Property Definition Collection Editor shown in Figure 9.9. You can change the properties of these placeholders or even create your own custom properties. The example in the figure shows a custom property called Roles. This property determines whether the page is visible to public users (anonymous users) or whether it requires authentication to get access to it. The value stored in this property is editable for page authors on the Page Properties page. Be careful about proliferating the placeholder definitions and check to see whether an appropriate placeholder already exists before you create your own. There is no reason to needlessly complicate maintenance of your templates by reinventing the placeholder.

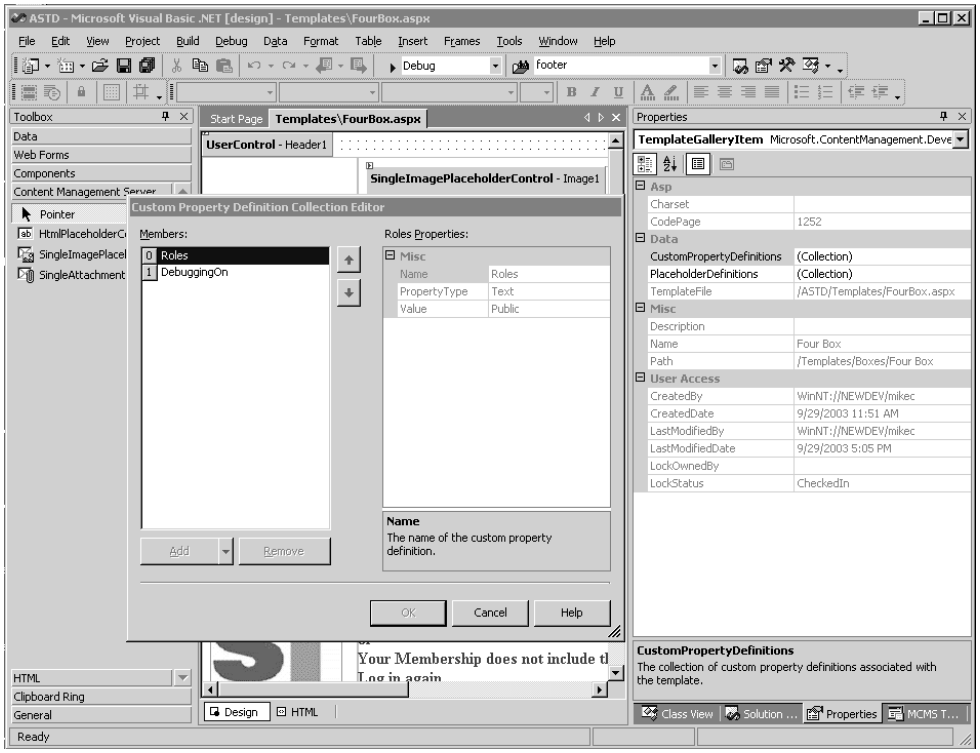

Figure 9.9 Placeholder Definition Collection Editor

MCMS offers quite a range of template design functionality and extensibility. A prerequisite for tapping this power is familiarity with Visual Studio .NET. You can create templates that use nearly anything you could build in a custom .NET web page.

Content Creation and Approval Process

In a mature web site, content goes through a process from creation to live publication, as shown in Figure 9.10. The process is iterative due to period

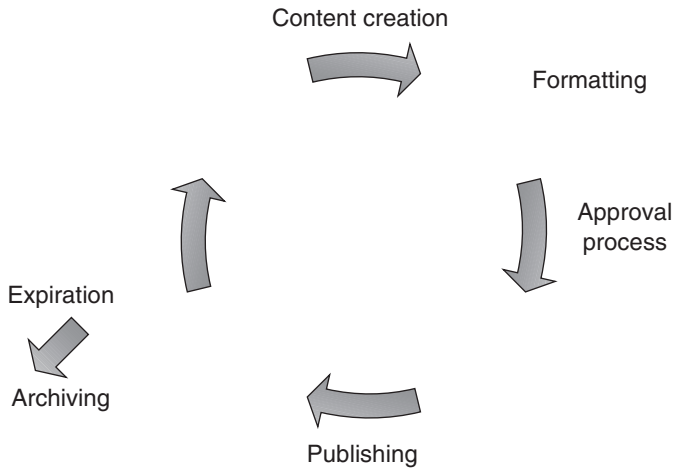

Figure 9.10 Content Management Cycle

updates. The first step is for an author to create the content, typically in a word processor such as Microsoft Word. Artwork may be created from digital cameras or illustration and graphics software. Content may originate in a database or a report that is generated from a line-of-business system such as accounting or customer relationship management. For this example, let's assume a content creator is using Word to write text that will appear on a web page.

In MCMS, content creation can be handled in one of two ways. Authors can continue to work in Word, with direct links between Word documents and the web pages where the content will be published. Creating or editing files puts them into MCMS version control and the approval process. The second approach is to use the browser client for MCMS to create the content. The editing controls on the template contain tools such as formatting, fonts, and spellchecking to make content creation as simple as possible. The template author determines how much control the content creator has over the page by, for example, determining whether HTML tags can be entered or leaving placeholders for photographs or other graphic elements.

The second step in the lifecycle is the formatting of the content into a web page that fits the look-and-feel of the site and incorporates the navigation and other shared elements of the site (such as copyright notice). Before content management, this step would have been handled by a web developer or webmaster who would have received an email with an attached Word document containing the text to be added. This content would have been cut and pasted into an HTML file with an HTML editor or a web-authoring tool.

With MCMS, the formatting step is accomplished by means of the page template. A graphic designer and web developer can create the container that holds the content, leaving the content creator with the job of creating the text itself.

Once the page is formatted, it is sent for one or more stages of approval. It might be sent back to the original author for feedback, to a supervisor, or to legal review. An accessibility review may be conducted to determine whether the page meets government accessibility guidelines such as those in Section 508 of the Disabilities Act. To learn more about Section 508, start with the official government site at www.section508.gov/. Microsoft is committed to making its products accessible to people with disabilities, and you can learn more about these efforts at www.microsoft.com/enable/. For a large number, perhaps the majority, of web sites, there is neither a review process nor a staging server where pages are placed before they go into production. Instead, the webmaster edits the live site directly. Approval processes may be simple or quite complex.

After approval, the page is either published or scheduled for publication. In MCMS, a page property determines the date that a page will go live (Figure 9.11). Embargoed content such as press releases can be prepared in advance and then displayed automatically when the publication date arrives.

When the end of the useful life of content is reached, the content disappears from the web site and is archived. MCMS allows content creators or web administrators to specify the shelf life of a page. Many web sites would benefit from freshness dating.

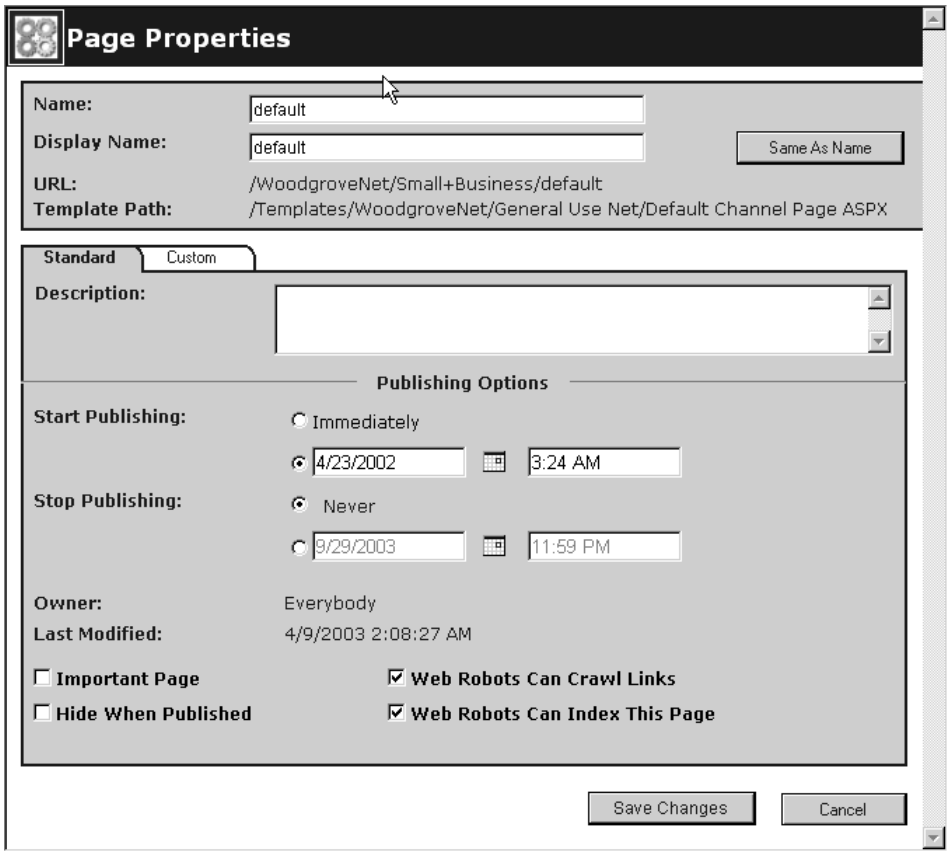

Figure 9.11 Page Properties Page

Defining Channels

Channels are defined in the MCMS Site Manager application. Once you have defined channels, you can start building templates and adding content to your site. For an example, I use the Woodgrove Bank sample site that ships with MCMS. This program is located in the MCMS program group on the Start menu. You will be instructed to log into the site. You can download a trial version of the software at www.microsoft.com/cmserver/default.aspx?url=/cmserver/evaluation/trial/.

1. Select **Channels** in Site Manager.
2. Expand the Channels pane.
3. Right-click **WoodgroveNet** and select **New Channel**.
4. Fill in the information requested on the New Channel dialog box. Name the channel **LoanApplication**. Enter **Loan Application** as the display name and **Forms and background information for submitting loan applications** as the description.

Be sure to follow the naming rules for channels:

- Channel names must be unique and contain only US-ASCII characters (alphanumeric characters and dashes, underscores, commas, and quotation marks).
- Do not include #, &, %, +, /, or | symbols.
- Channel names are limited to 100 characters.
- Do not include spaces in channel names.
- Do not add the HTM extension to the channel name.

Once you have created the channel, you need to fill in the properties for the channel. Here are the steps to follow:

1. Add rights groups to the channel you have created. Start with the subscribers group, and then add another entry for authors.
2. Display the channel properties again by right-clicking the channel and choosing Properties from the menu, then click the **Publishing** tab.
3. In the Channel Rendering section of the channel properties, click **Use First Page** to make the first page the default for the channel.
4. Click **OK** to close the channel-rendering properties dialog box.
5. Set the publishing schedule for the channel. From the channel properties dialog box, click the **Publishing** tab, and then click **Set** in the Lifetime section of the tab. Click **Immediately** to make items published in the channel visible immediately.
6. Click **Never Stop Publishing** as the Stop Publishing entry.
7. Establish the web author default galleries, which are the template and resource galleries that are available to this channel. In the Channel properties, click the **Web Authoring** tab.
8. Browse to the desired locations in the template and resource galleries.

9. Set additional channel properties listed on the **Publishing** tab of the channel properties (Figures 9.12 and 9.13). These include designation as an important channel, hiding the channel, and allowing the channel to be crawled by web robots.

Repeat these steps for all the channels you need to create. You can always create new channels as your site evolves, and content may be moved from one channel to another.

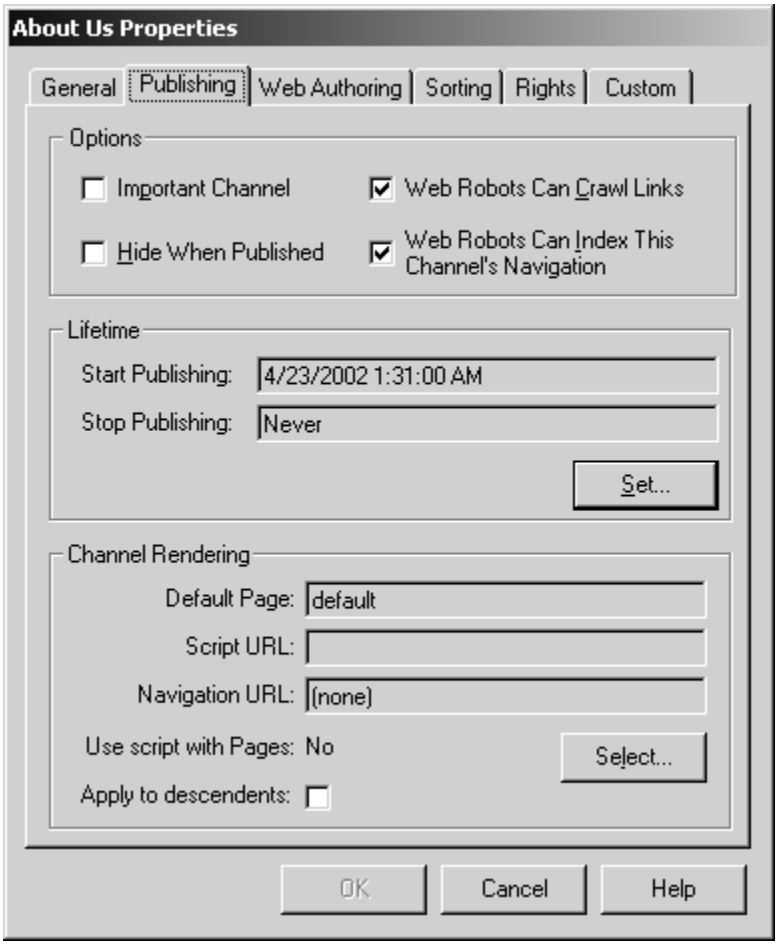

Figure 9.12 Channel Publishing Properties

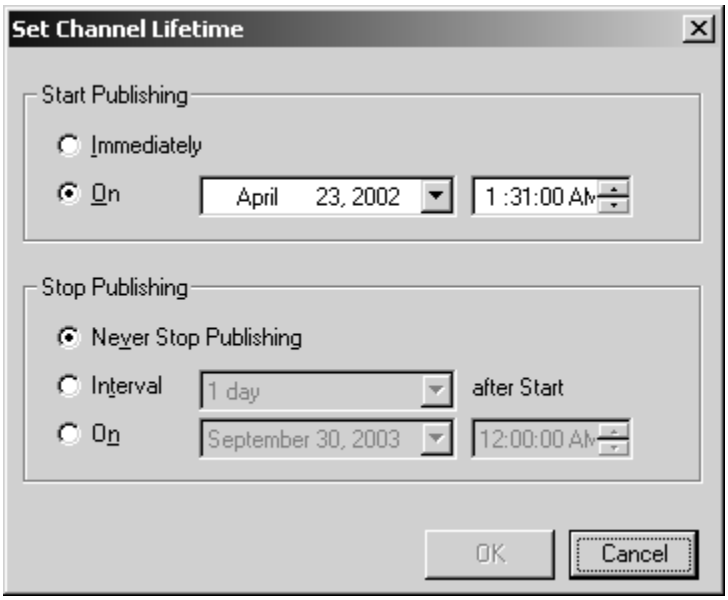

Figure 9.13 Channel Lifetime

The next step is to develop the templates identified in the design review session. You will reuse the templates as much as possible from one section of the site to another to simplify template and site maintenance. Developers must make trade-offs in template design between general use (fewer templates) and specific requirements (more templates). While it is possible to create a single master template that would handle nearly any contingency, this mother of all templates (MOAT) would be difficult to create and maintain, and its superset of page layout and functionality might be daunting to users. On the other hand, the more templates you build, the more templates you have to maintain (see the following box "One Template or Many?").

Multilingual Sites

MCMS 2002 allows you to develop multilingual sites to support the needs of your users. All languages that support the Unicode standard are supported in MCMS. To create a multilingual site, start by building the

templates, channels, and pages for the primary language supported by the site. This work provides the site structure.

The second step is to translate resources used in your templates to the second target language. You do not want to have a Spanish site that uses images with English captions, nor a Lithuanian site with Russian page elements.

Next, create templates using the translated resources to map content from one language to another. MCMS uses a feature called connected pages that lets you link two or more pages that reference the same content source and use related templates. Therefore you could end up with an English press release template and connected Spanish and French pages.

One Template or Many?

Defining templates is the most challenging and time-consuming step in implementing a CMS. Programming, functional requirements, and graphic design converge here, often creating a long approval process or frequent revision. Nearly all web designers have experienced the phenomenon of five department representatives having six opinions on the color, font, or layout of a page.

Inexperienced CMS users are often tempted to allow templates to proliferate so each nook and cranny of the portal uses a different template. For instance, separate templates might be built for FAQs, articles, press releases, and product announcements.

Fight this instinct to create new templates and ask yourself whether a generic template might not be suitable in more than one section. Template discipline significantly reduces the maintenance burden of the site. The big payoff comes when users decide to change the colors, layout, and general appearance of the site. The fewer the number of templates you have, the quicker such changes, which are usually requested about one week before the site is scheduled to go into production, will be made.

Web Page Workflow Implementation

With the navigation and templates in place, the next step is to develop a workflow process for content based on information provided by the users at the design review. You may want to begin with a pilot to learn more about how your organization will choose to implement these new business processes. Start with just one section of the site, and identify the parties involved in the approval process. This workflow is based on standard CMS routing and approval functionality.

The task of the pilot is implementation of a sample workflow. For instance, a document may be routed from an author to a content editor and to an additional approval level such as a manager or legal review. I recommend that a representative workflow be chosen, as it may be copied and customized for subsequent workflows that are established.

Some routing and approval processes are better in theory than in practice. Complicated business rules can create content bottlenecks and delay content publication, thereby undermining the portal's raison d'être. This concept is not easy to explain but becomes quite apparent when a sample workflow is tested. People in the approval loop quickly learn how important it is to stay on top of their content responsibilities.

There are five levels in the MCMS approval process. You can omit some levels, but note that some levels have multiple steps. These levels are:

1. Template designer
2. Author
3. Editor
4. Moderator
5. Subscriber

Start by writing out the business rules for approval in plain English, also known as pseudocode. For instance, the press release section might have the following rules:

- All press releases for the organization will be based on the press release template. This template has been created by the template designer and is available in the Press Releases channel.

- Authors for press releases include the offices of the president, chief operating officer, vice president of sales, marketing department, public affairs, and investor relations. Each of these offices has designated one or more users as authorized to create press releases. The press releases shall conform to the corporate style guide. After a press release is created, it is sent to the editor.
- The public affairs office is responsible for editing press releases. If a press release draft requires additional rewriting, it is returned to the originating office. If normal editing is sufficient, it is edited and forwarded to the web site moderator.
- The legal department and webmaster serve as moderators for press releases. They make a final check for substance and formatting. When they approve content, it is released into production.

Now let's take a closer look at how the steps are implemented in CMS.

Template Designer

The template designer is a web programmer with skills in graphic design and HTML programming. Using Visual Studio.NET, the template designer creates a template for a press release containing placeholder controls for content to be provided by the authors. The template contains standard elements such as masthead, copyright information, and navigation, as shown in Figure 9.14.

If multiple channels are supported, the designer may create multiple templates to match each of the channels. For instance, the default template could be for a browser on a computer or workstation, but a second template could be for a PDA or a cellular phone.

Each of the placeholders in a template has properties that determine the content that is allowed. For instance, a placeholder may restrict the author to enter plain text only without hyperlinks, graphics, or even a specific font. The appearance of the text entered is determined by the template itself. This approach would make sense for fields such as the press release title or date, for instance. On the other hand, a placeholder may allow images to be inserted, such as photographs, or file attachments. The power to insert such objects is one of the reasons why an approval process is needed, as these could create security or performance problems for users.

Templates have to be set up only once, so template building is intense at the beginning of projects and tapers off as the portal reaches maturity.

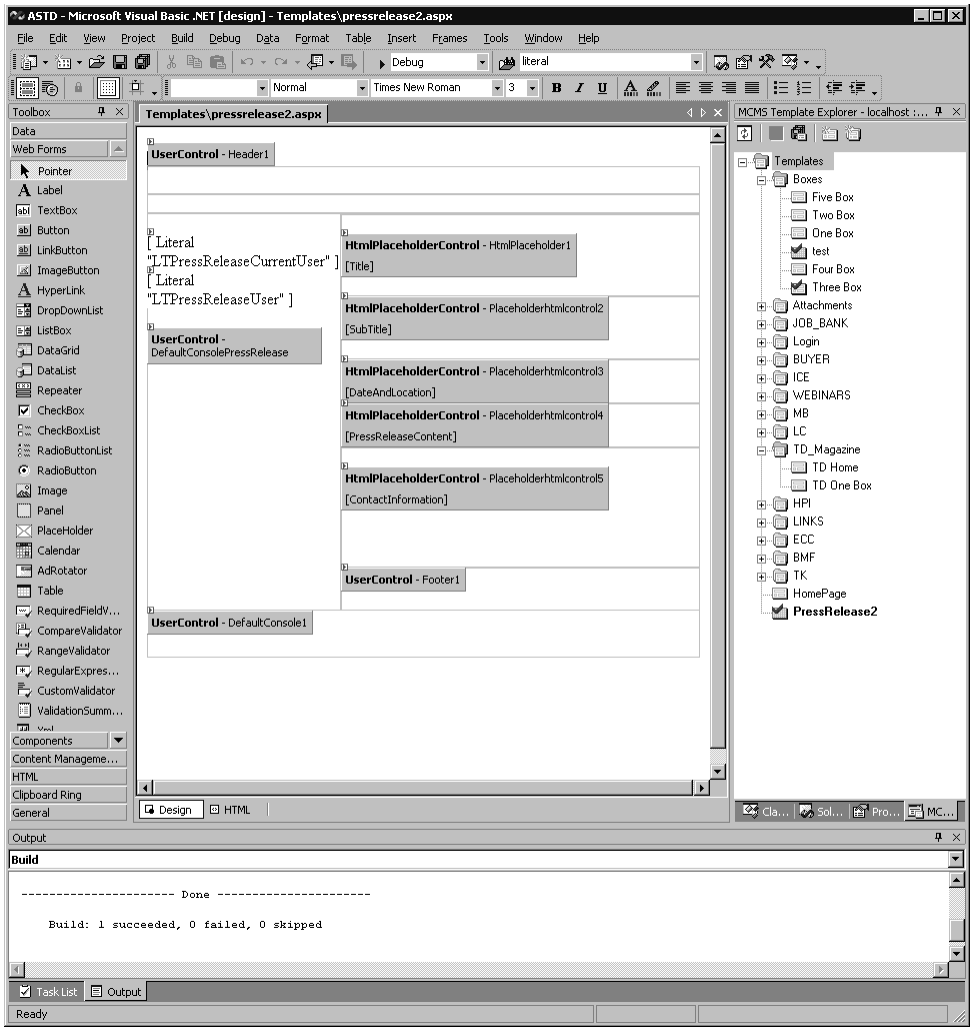

Figure 9.14 Template Design for a Press Release

Moreover, design elements are likely to be shared among templates by means of the resource library, so the tenth template is easier to create than the first.

If authoring will be performed in Word rather than in the browser interface to the template, the template designer must allow for this choice

in the process and plan for how the Word authoring will map to the template, using the MCMS Content Connector wizard.

Author

With the template in place, the author fills in the placeholders in the template. For a press release, this would mean entering information such as the date, title, contact person, phone number, and body of the press release, as shown in Figure 9.15.

The placeholders are now rendered as fields of the appropriate type. Icons on the fields indicate the type of entry that is allowed, such as plain text, hyperlinks, formatted text, or attachments. If prompted, the user can upload graphics to reside on a page or documents to be stored as attachments to a page.

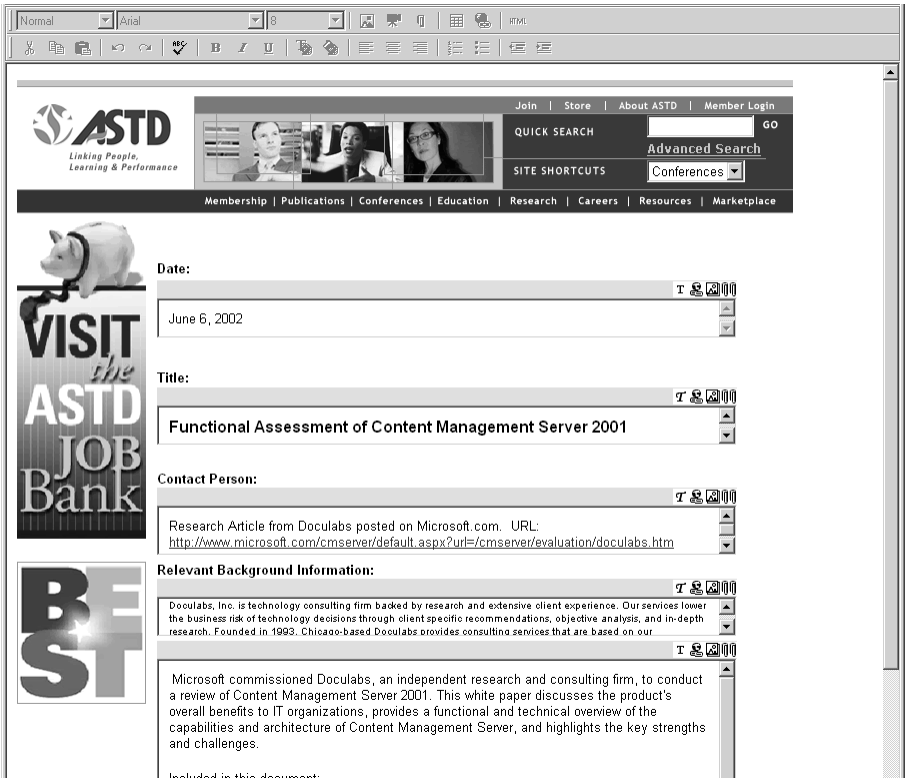

Figure 9.15 Press Release Template in Author View

Finally, the author specifies additional page properties such as the start and end date for publishing the content. Each time the page is saved, the old version is stored as well to allow rollback to a previous version. The author finally submits the finished page to the editor.

As soon as the page is submitted by the author, the editor is notified. She views the page by means of the web interface and edits the material directly. Again, each saved version is stored, so the author can review the editing marks and differences between the original and edited versions. Figure 9.16 shows the page status for a document that has been flagged by the author as complete and ready for editorial review.

If there are no editors or moderators, pages are automatically published when approved by the author. You may also have scenarios with editors or moderators or both. In the latter case, if the editor approves the page, it is forwarded to the moderator for final approval.

Editors and Moderators

The moderator checks the page and the publication date, and then approves or disapproves the page. The moderator can change the page or its properties. Moderators have special tools available in MCMS. The Production Manager is a web interface that allows moderators to see a list of

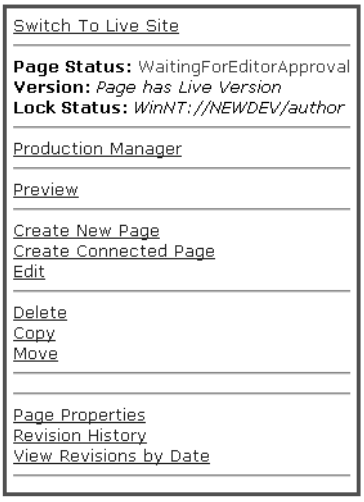

Figure 9.16 Editor Notification

pages in the product process, and then go from page to page to move it through the process (Figure 9.17).

If you want to publish pages without editor or moderator approval, you can simply not assign an editor or moderator to the channel where the page is published. This means that the author makes the decision on when the page is approved. The author can defer publishing by entering the start date in the page properties.

Subscriber

As soon as the publication date for an approved page arrives, it becomes visible to all the users of the site. Subscribers do not have access to any of the administrative features of MCMS. From their perspective, content management is invisible.

To define the approval process, go to the Channel Manager program. Select the channel on which you would like to define the approval process. If needed for your workflow, add editors and moderators to the process so they will receive the pages and approve them. Figure 9.18 shows the user rights assigned to a specific channel.

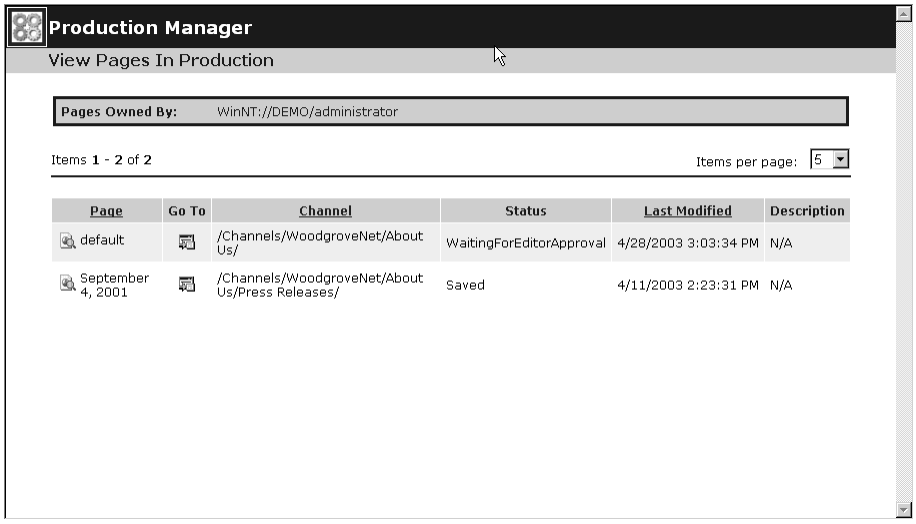

Figure 9.17 MCMS Production Manager Page

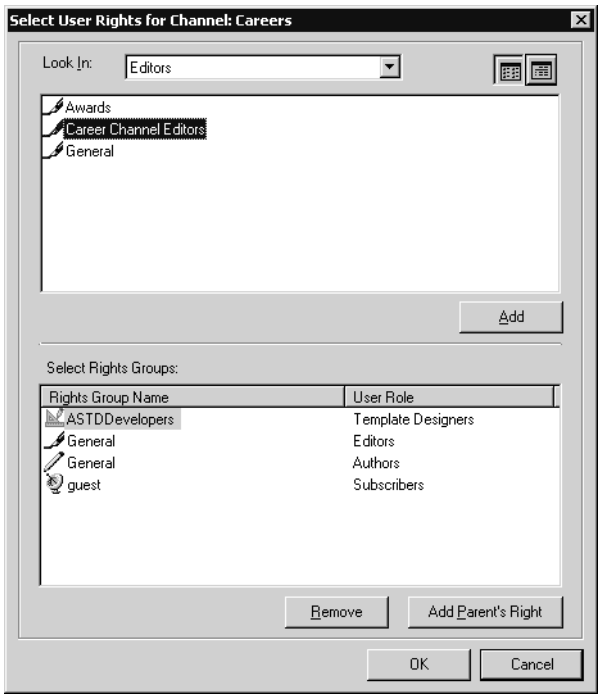

Figure 9.18 Adding an Approval to the Channel

Rights groups are assigned roles along with the permissions that are mapped to these roles. In the example shown in the figure, a group called ASTDDevelopers has the rights to design templates. A special group called Career Channel Editors has been created just for this channel to grant rights for maintaining a single channel without elevating the users to this status for the entire site.

Adding Workflow Email Notification

The out-of-the-box tools of MCMS provide web interfaces to monitor content management tasks, such as Production Manager and Approval Assistant. The problem with these tools is that they require users to go to a URL to see whether they have any pending web pages to approve. Your users will be thankful if you take this a step further and enable what MCMS calls

workflow email notification. This function advises reviewers of events as they occur in the approval cycle. The following code shows how to create email notification to reviewers when a page is submitted for approval. It uses the Microsoft Internet Information Services (IIS) Simple Mail Transfer Protocol (SMTP) service and Collaboration Data Object (CDO) COM components to send email programmatically.[1]

NOTE *Workflow email notification requires the ASP.NET version of Microsoft Content Management Server (MCMS 2002).*

To implement workflow email notification:

1. In the **global.asax** file, add a reference to **Microsoft.Content-Management.Publishing**.
2. At the top of the global.asax file, add the following **using** statements:

```
using System.Web.Mail;
using Microsoft.ContentManagement.Publishing;
using Microsoft.ContentManagement.Publishing.Events;
```

3. Within the **Global** class in the global.asax file, add the following event handler:

```
public void CmsPosting_Submitted( Object source, _
ChangedEventArgs e ) {
    MailMessage mail = new MailMessage();
    mail.From ="MCMS Web Author";
    Posting submittedPosting = (Posting)e.Target;
    mail.Subject = submittedPosting.Name & " has been _
    submitted for approval.";
    mail.Body = submittedPosting.Name & " has been _
    submitted for approval. ";
    foreach (User approver in _
    submittedPosting.Approvers)
    {
        mail.To = approver.ClientAccountName & _
        "@youknow.net";
        SmtpMail.Send(mail);
    }
}
```

[1]Microsoft, Content Management Server help file, "Implementing Workflow E-Mail Notification."

To register the event, follow the steps outlined in the help topic "Registering for Publishing Events Using HTTP Modules." Be sure to check the help file to get the code that matches your release of the product.

Enabling Security on MCMS

Because MCMS is so important for all types of portals, it is likely to be the first line of defense in your security scheme. The information and guidance offered in Chapter 6 apply here as well. This section provides special notes regarding security implications for MCMS. You must configure access in three locations: Internet Information Server (IIS), ASP.NET Web.config file, and the MCMS Server Configuration application. The entries you make in these locations depend on the usage scenario and your security preferences. These guidelines are for the most common scenarios.

Internet

A public web site should be available to anonymous users, with no authentication required. By default, a new MCMS site requires user authentication, even to view pages. To make your site visible to guests (anonymous users), you must change the configuration from the default as follows:

1. In the ASP.NET Web.config file, remove or comment out the following line:

```
<add type="Microsoft.ContentManagement.Web.Security. _
CmsAuthorizationModule, Microsoft.ContentManagement. _
Web, Version=5.0.1200.0, Culture=neutral, _
PublicKeyToken=31bf3856ad364e35" _
name="CmsAuthorizationModule" />
```

2. In the MCMS Site Manager, create an account to use as the guest account, and add this account to the subscribers rights group.
3. Go to the MCMS Server Configuration application in the MCMS program group and click Security.
4. In the Guest Visitors section of the Security Configuration dialog box, click the **Yes** radio button for **Allow Guests on Site**. Enter the name of the account to use as the guest account for the site.

If you want to provide some content exclusively to authenticated users, you can follow the guidelines in the following extranet section in addition to allowing anonymous users.

Intranet

Intranet sites use integrated Windows authentication. MCMS supports both Active Directory and Windows NT authentication. Integrated Windows authentication is suitable under the following conditions:

- All users have Windows NT or Active Directory accounts.
- The intranet is located inside the firewall.
- The entire site requires authentication and does not allow anonymous users.
- All users use Internet Explorer to access the intranet.

As you might imagine, Integrated Windows security is quite secure and even allows users to authenticate the servers that are attempting to authenticate them. It also spares the user from entering a username and password by using the current user information on the client computer for authentication. Of course integrated security also means one fewer username and password combination to manage for the user.

Extranet

Now that you know how to handle security for an intranet and an Internet site, the case of the extranet is simple, as it is nothing more than a combination of the two, with a portion of the site allowing anonymous access along with a private, secured channel that is accessed via the Internet.

The baseline configuration for the extranet is the same as for an Internet site. Handle all the pages that are unrestricted in the same way as you would for any public site. This part of the site should support multiple browsers and not be based on browser-specific technology such as ActiveX controls.

The restricted access sections should be protected with MCMS rights groups. This means that the administrator must grant access to the user of a particular channel, folder, or gallery using the MCMS Site Manager. For

instance, you could grant author rights to the legal notices section of a site to the legal department.

TIP *You may want to configure MCMS as an extranet to enable access for content managers. By doing so, you allow your site to be maintained from remote locations rather than just by users in your domain. For instance, content authors could create new pages and editors could participate in page review from home.*

There are three general approaches to authentication on an extranet: forms-based authentication, digital certificates, and custom authentication schemes. You can combine these approaches with one another (for more on choosing an approach, see Chapter 6). The following examples show how you can use these authentication approaches to grant access on an extranet to MCMS administrative functions.

NOTE *Forms-based authentication uses a cookie that stores a ticket indicating that a user has been authenticated. The system checks the cookie first on subsequent access attempts. You should use secure sockets layer (SSL) to protect the contents of the cookie so users cannot falsify that they have been authenticated. Remember that SSL imposes a performance penalty, so don't apply it to pages where anonymous access is granted anyway.*

Forms-Based Authentication

Forms-based authentication routes unauthenticated users to an HTML form in which they enter their credentials. When authentication succeeds, a cookie is written and they are granted access to subsequent pages upon request. Chapter 6 provides examples of how to implement forms-based authentication. The basic steps are:

1. Design a logon page in Visual Studio.NET. Define fields for the username and password along with a Submit button. Be sure to use the Password field as the control to accept the password.
2. Right-click each of the controls dropped onto the page, and select **Run As Server Control**.

3. Add the following code to the **Submit1_ServerClick** event handler:

```
CmsAuthenticationTicket ticket;
ticket = _
CmsFormsAuthentication.AuthenticateAsUser _
(strServerAccount,
    strServerPassword,
    txtUserName.Text,
    "FirstName");
if( ticket != null )
{
    string strReturnUrl = _
    Request.QueryString["ReturnUrl"];
    CmsFormsAuthentication.SetAuthCookie(ticket, true, _
    false);
   StringBuilder strUrl = new StringBuilder();
   strUrl.Append("http://");
   strUrl.Append(Environment.MachineName);
   strUrl.Append(strReturnUrl);
   Response.Redirect(strUrl.ToString());
}
else
{
    Label1.Text = "Your username or password are _
    incorrect. Please re-enter your username and _
    password.";
}
```

4. Add the following code to the **Page_Load** handler:

```
if(!Request.IsSecureConnection)
{
    StringBuilder strSSL = new StringBuilder();
    strSSL.Append("https://");
    strSSL.Append(Environment.MachineName);
    strSSL.Append("/");
    strSSL.Append(Request.Url.PathAndQuery);
    Response.Redirect(strSSL.ToString());
}
```

5. Save your changes and build the solution.

Digital Certificates

Certificates are digital keys installed on a particular workstation. They contain information that uniquely identifies the user to applications accessed by the workstation. Certificates can be mapped to Active Directory or Windows domains. Users are issued certificates by a certificate-issuing authority, and these certificates are downloaded to their workstations. Windows 2000 comes with a certificate server so you can manage certificates.

The key difference between forms-based security and digital certificates is that certificate-based authentication does not require the user to enter authentication information on a login page. Authentication is more seamless and virtually invisible to the user. Therefore, certificates are more appropriate for processes that have no human user at all, such as the automation of the supply chain to post orders from a buyer to a seller.

You can set up your site so users without certificates are treated as anonymous users while those with certificates are mapped to the permissions and roles of the certificate user. You can establish unique certificates for each individual user or map many users to a single account. For more information on certificates and Windows security, see TechNet security how-tos at www.microsoft.com/technet/itsolutions/howto/sechow.asp and Michael Howard's book *Designing Secure Web-Based Applications for Microsoft Windows 2000* (Microsoft Press, 2000).

Custom Authentication

The third form of authentication for extranets is custom authentication. This approach uses another directory service and maps users back to related Active Directory accounts. For instance, user information might be stored in Commerce Server for means of site personalization. Users might be authenticated by a third party such as Microsoft Passport. The custom security scheme would then look up the appropriate Active Directory account and grant permissions based on that account.

Custom authentication is the least commonly used form of the authentication approaches discussed here. This approach may be selected when there is a limited amount of content or functionality to provide to authenticated users, and the site is therefore rather simple to administer.

Although it is technically possible, I do not recommend using databases to store authentication information. Active Directory and other directory servers are more hardened to security threats.

.NET and Web Services Integration

Up to this point, we have been implementing standard MCMS functionality, albeit in the larger context of our portal master plan. With .NET and web services integration, the true potential of the portal is realized. While MCMS is the heart of the portal platform, it is .NET that provides the overarching framework. Some services in the portal exist entirely outside MCMS, others are displayed within pages based on MCMS templates, and still others are web services either inside or outside the enterprise.

.NET integration means the use of the .NET Framework to extend functionality beyond MCMS to include additional server platform products such as SharePoint Portal Server, SQL Server, and Commerce Server, or non-Microsoft products based on the .NET platform. Not all .NET integration requires the use of web services. The fact that Microsoft has moved or is moving all its enterprise server products to a common application platform provides tremendous benefits in terms of your taking advantage of developer skills and component reuse.

With web services integration, you are free to connect to an even broader range of applications beyond the .NET world, such as Java, WebSphere, and any other service that jumps on the web services bandwagon. You now have the opportunity to syndicate content from news sources or tap into financial services for transactions, currency conversions, online marketplaces, and a host of other solutions.

> ***NOTE*** *I have omitted coverage of the Site Stager component of MCMS, as it is not compatible with ASP.NET-based sites. It is used to push static copies of HTML and ASP pages to other servers, including non-Windows servers.*

.NET Integration

MCMS provides a wealth of power with its out-of-the box user interface for content management and the Windows administrative clients for webmasters. You are likely to rely on these extensively in your implementation. They are not the only options for clever .NET programmers, however. You can create custom .NET applications that take advantage of all MCMS features, but with an interface of your choosing. For instance, you may find that some of the options available to users in the standard browser-based

Web Author are more (or less) than what your users need to accomplish their tasks. You can hide or simplify options. You can create specialized interfaces to run in an Office application that guide users through the steps of content creation. These are called MCMS web applications to distinguish them from MCMS web services. Within Visual Studio.NET, projects are stored in solutions, which are stored with the file extension .SLN.

Another way to extend MCMS is to customize the approval process discussed earlier in the chapter, providing email notification to users when an item awaits their approval. Similarly, you could implement more sophisticated routing rules for the approval process as a custom MCMS application.

MCMS provides several project templates in Visual Studio.NET (Table 9.1). You can use both Visual Basic and Visual C# to create these projects.

These three template types appear in the Templates list in Visual Studio.NET when you create a new project. The MCMS projects are like other project templates, except that they automatically run some preliminary steps to make MCMS components available in the project. The standard console is copied into the project as the basis for a custom console, changes are automatically made to the Web.config file, and users are authenticated against MCMS and their rights are checked.

MCMS projects provide additional toolbox items, the Template Explorer menu bar, and menu items on the Tools menu. The Content Management Server tab in the Toolbox window contains the placeholder controls you need to design MCMS templates. The Template Explorer window provides a hierarchical view of the templates along with the ability to view template properties. The Content Management Server menu item on the

Table 9.1 Visual Studio Content Management Projects

Project Template Name	Description
MCMS Web Application	Used to create an MCMS application with a web user interface
MCMS Web Service	Used to create MCMS XML web services that can be used from other applications
MCMS Empty Web Project	Used to create an empty project for an MCMS web application

Tools menu provides direct links to Site Manager, Database Configuration Application, Server Configuration Application, and Web Author.

You can convert an existing Visual Studio.NET project to an MCMS project. To do so, open the project and select Enable as CMS Project from the Project menu.

> ***NOTE*** *Do not remove references to MCMS components that have automatically been generated by Visual Studio.NET. Removing these references will produce unpredictable and undesirable results.*

Web Services Integration

Web services are probably the most widely touted benefit of the .NET platform because they provide the ability to integrate heterogeneous systems and even make Microsoft and Unix solutions talk to one another. MCMS 2002 brings web services into the fold in a content management product. Along with adopting Visual Studio.NET as the development environment for MCMS, you can create and consume web services with MCMS. Taking this approach results in new openness and the promise of even greater interoperability. It also raises many architectural and design options that require sorting through.

If you already know how to create web services, you know how to create them for MCMS. The big difference is that you must invest the time to learn the MCMS object model to get the most out of your web services.

> ***NOTE*** *Web services configured to directly access the MCMS publishing API create a potential security vulnerability, as they make it possible for the service to write changes to your MCMS repository, overriding a read-only configuration. Be sure to practice safe coding on your web service to prevent dangerous or malicious changes.*

Consuming Web Services

While MCMS may more commonly be used to provide a web service to another application, there are excellent opportunities to have MCMS play the other role and consume web services. This is the case for content syndication. MCMS can periodically retrieve news or other data from another web service.

It is possible to configure MCMS to consume a web service in real time. For instance, a user could view a web page with current weather information, prompting MCMS to request weather data in XML from a weather web service. MCMS would package the resulting data using the page template and display it for the user. In most cases, this is not the most attractive option. Users of your portal would be exposed to additional latency of the Internet and would be vulnerable to any outage or performance degradation of the web service. As usage rises, the load on the web service also rises, increasing the processing cost for the service and potentially increasing the financial cost depending on the revenue model. Real-time concurrency is not often worth this price.

Another problem with posting content directly from a web service is that you have fewer options for error-handling or human intervention to correct errors. You may expose your organization to legal liability by publishing incorrect information on your site, so you need a review process for content.

A better option is to take the data consumed by the service and store it in the CMS repository. This function can be performed on a scheduled basis to ensure a reasonable degree of concurrency. For instance, an hourly update of a four-day weather forecast might be acceptable, or an update on sports scores every minute. For even less volatile data, you could take the data from the web service and launch the content management routing and approval process.

Content Management in SharePoint

The SharePoint product family contains its own content management features, allowing users and webmasters to create new pages on their intranet by filling in web forms. These features were introduced in version 1 of SharePoint Portal Server and SharePoint Team Services and have been significantly enhanced in the current version. Although I am focusing on SharePoint Portal Server, many of the features are also available in Windows SharePoint Services, the successor to SharePoint Team Services. Figure 9.19 shows a portal home page in SharePoint. It consists of several elements called web parts. This example shows web parts such as News, Events, Links, and Topics. Some web parts include content management features.

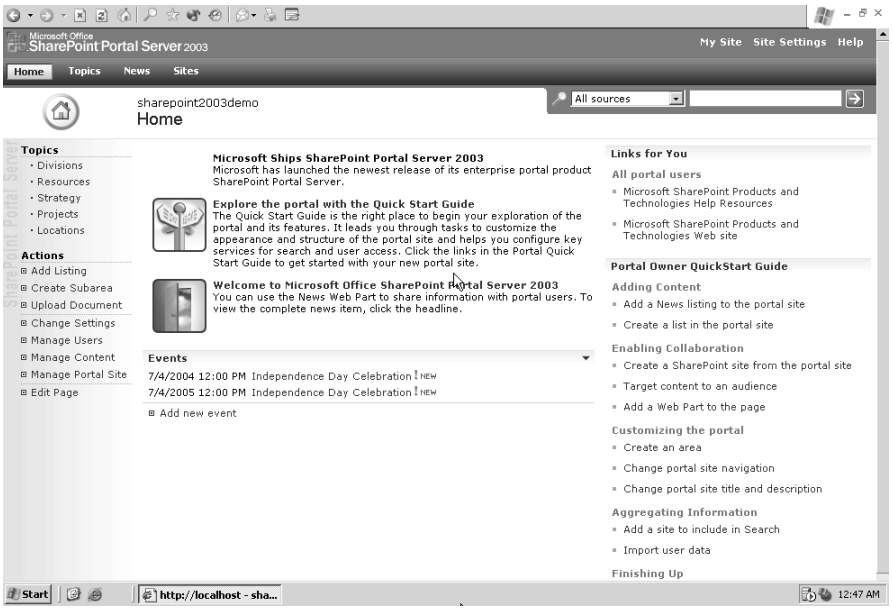

Figure 9.19 SharePoint Portal Server Home Page

For instance, if you want to add a new event to the portal, you can click Add New Event to display the page shown in Figure 9.20.

When the record is saved, the new page is created. The event now appears in the Events web part on the home page of the portal and on the Events page, as shown in Figure 9.21.

This is virtually the simplest content management one can imagine. The user fills out a form with a handful of fields and then clicks Save to enter the content item. The data from the form is stored in the database repository for SharePoint (in this case, SQL Server). The web part controls the appearance of the content. You could build additional web parts based on the same data to present a different view of the user, such as a calendar view of events rather than a list view.

Several other web parts have equally simple content management capability. Others are enhanced to offer control over more properties of the content. For instance, the News Item web part offers much more flexibility than the Events web part. The first page is similar, with the addition of a text editor to control the appearance of the item such as its colors and fonts (Figure 9.22).

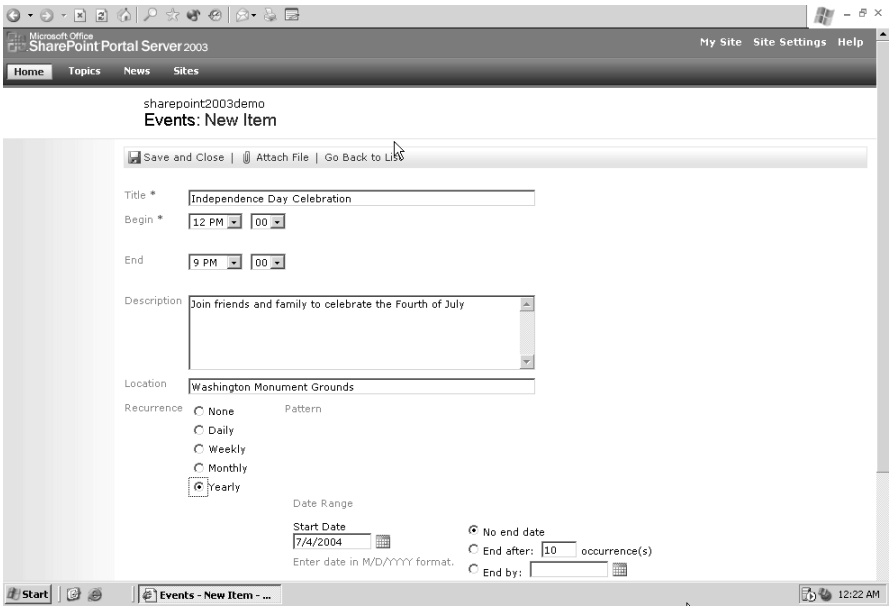

Figure 9.20 SharePoint Portal Server Events Form

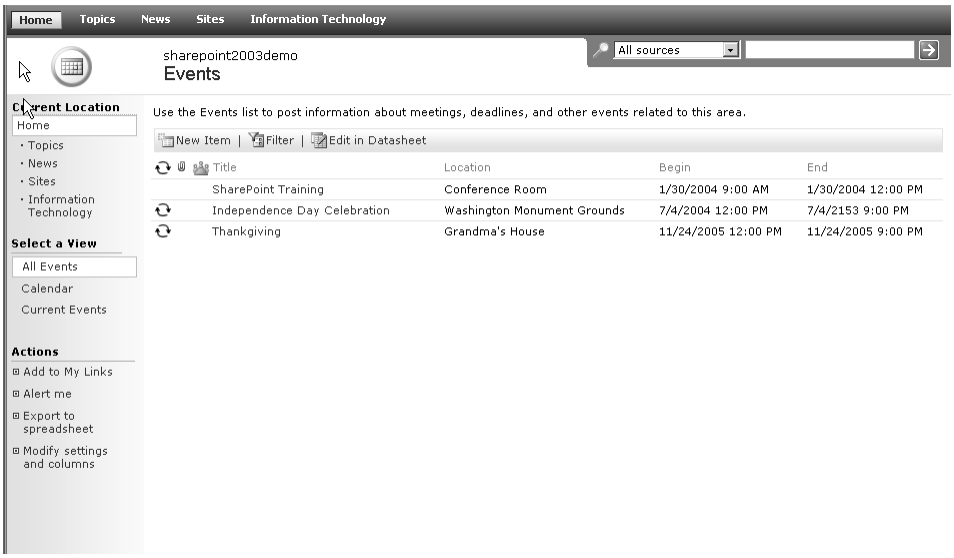

Figure 9.21 SharePoint Portal Server Home Events Page

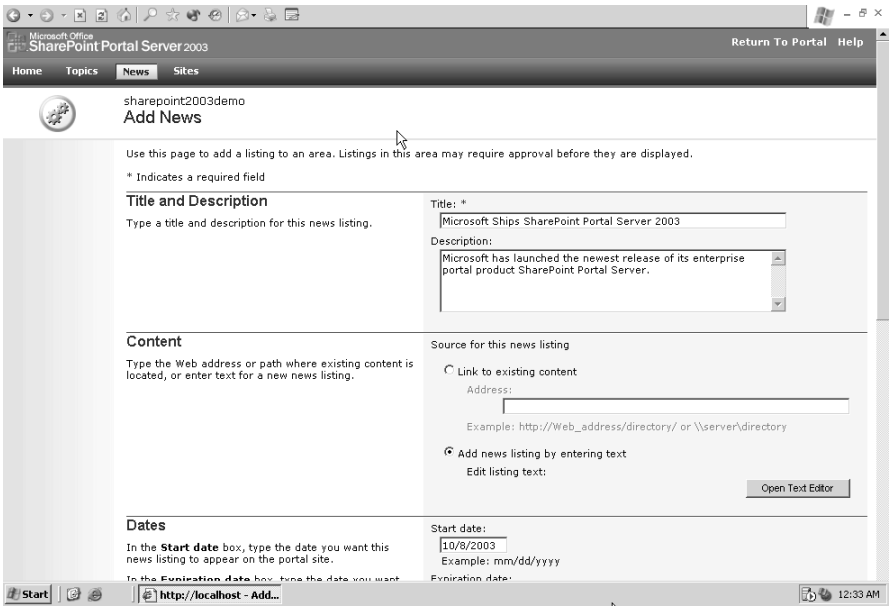

Figure 9.22 Add News Page

After you save an item, you can edit it at any time. This web page has tabs with more and different settings than those for a calendar entry (Figure 9.23).

The Publishing tab (Figure 9.24) controls the starting and ending dates for publication of an item. On this tab, users can also change the publishing status of an item to advance it through the approval process.

The next tab on the News Item page (Figure 9.25) allows an item to be assigned to a group, relates an image to an item, and grants permission to users for the item. The field that stores the filename of an image is used to display images or thumbnails that correspond to the page.

The final tab (Figure 9.26) controls whether the item is indexed by the SharePoint search engine. If the item is not included, it will not be included in search results.

SharePoint content management features are simple and direct, providing just enough to support the standard pages generated on SPS sites. They may be adapted and enhanced through custom web parts and custom site templates, both of which are discussed at greater length in Chapter 12. Compared to no content management, what SharePoint provides is a great leap forward.

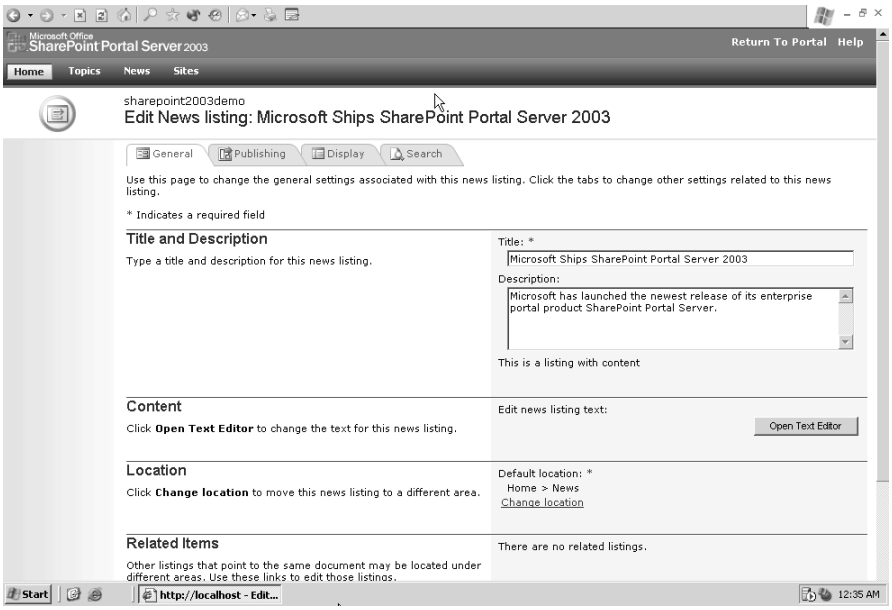

Figure 9.23 Edit News Item–General Tab

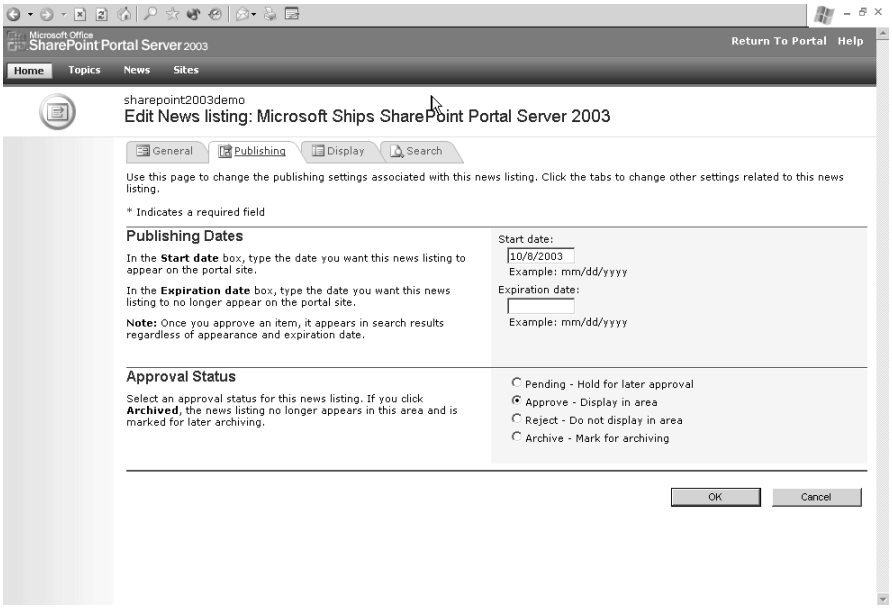

Figure 9.24 Edit News Item–Publishing Tab

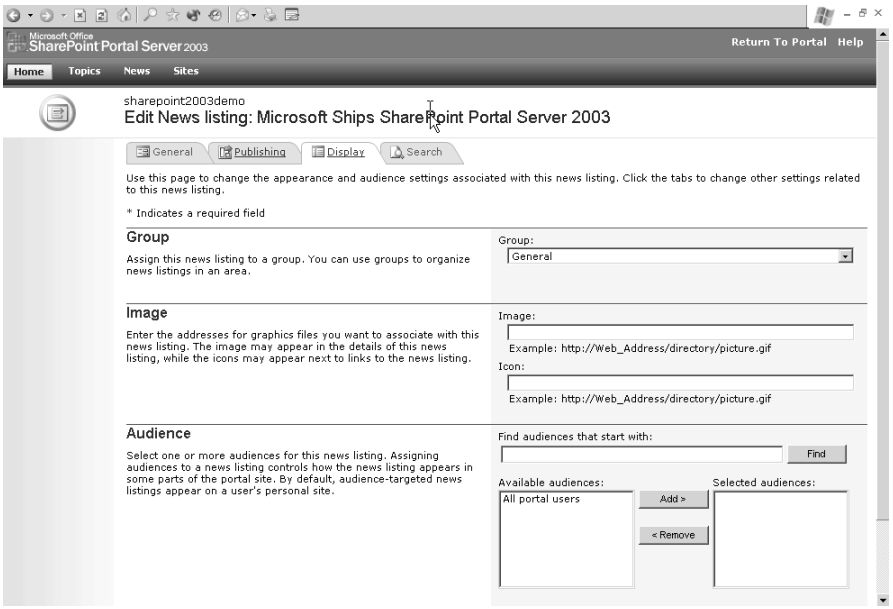

Figure 9.25 Edit News Item–Display Tab

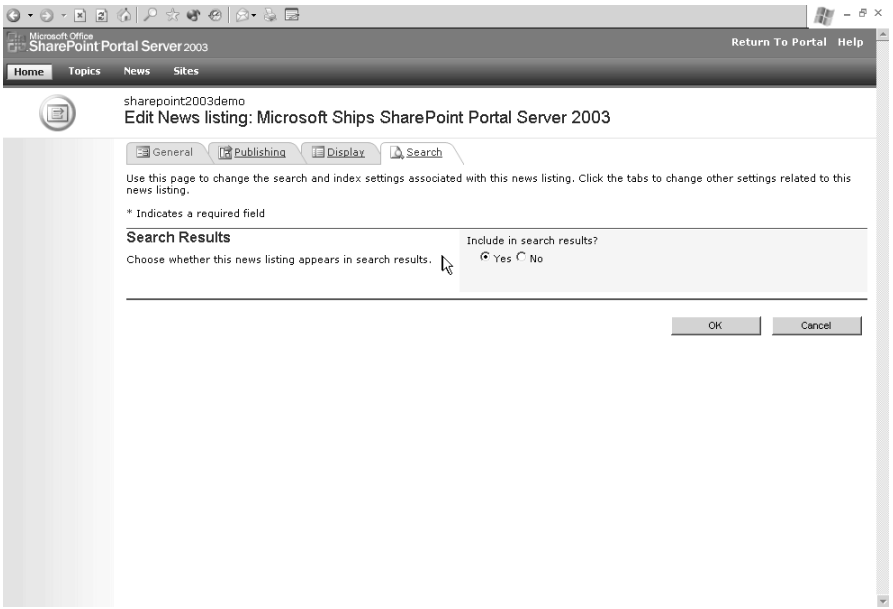

Figure 9.26 Edit News–Search Tab

Integrating SharePoint with Microsoft Content Management Server

Now that we have examined content management both in Microsoft Content Management Server and SharePoint Portal Server, what about harnessing both at once for your site? Microsoft provides integration points that support at least four scenarios:

Using SharePoint portal search with CMS sites. SharePoint Portal Server offers Microsoft's most powerful web search technology, a vital asset for internal and external portal sites. MCMS lacks native search capability, and therefore wedding the two makes sense for many organizations. You can implement a SharePoint search and then add appropriate links in the MCMS templates to reach simple and advanced search capabilities.

Publishing WSS/SPS document libraries to CMS sites. SPS can augment CMS by providing document management capabilities by means of its document libraries. This functionality is similar to that offered in the connector between SPS 1.0 and CMS.

Publishing MCMS content within SPS sites. While SPS provides some rudimentary content management, you are likely to want to include rich content pages from MCMS within your SPS site.

Using MCMS authoring within SPS. Imagine that your public web site was implemented with MCMS and you have just implemented an intranet based on SharePoint Portal Server 2003. How do the twain meet? You can install web parts for MCMS in your intranet that connect to the content on your public web site. This step puts content management in a more visible place for users and integrates it with other line-of-business applications.

You may choose among these integration scenarios or implement features of more than one in your solution. Integration such as adding MCMS web parts to a SharePoint site does not really affect the underlying architecture of your CMS site, so the risk and impact are relatively low.

Custom Content Management

We have taken a look at two content management solutions in the Microsoft server family, MCMS and SPS. Another alternative is to create your own custom .NET content management solution. This approach is a throwback to the days before off-the-shelf content management existed. The value in products such as MCMS and SPS is such that it would be quite difficult to

get anywhere near the same level of functionality, dependability, and support from a custom solution. For general-purpose content management, I discourage you from attempting to reinvent the CMS wheel.

On the other hand, a highly specialized content management solution might be better served with a custom approach. For instance, imagine that your entire web presence was constituted of pages dynamically generated from an existing database. Your site would essentially consist of forms for entering data and reports for extracting that data from the database. In such a case, your requirement is not really for content management at all, but rather for a custom application that happens to have a web user interface. In this scenario, a custom approach might be the quickest path to fulfilling the project requirements.

Conclusion

Nearly every portal needs content management; and the implementation of content management along with related page migration consumes a significant amount of time and labor, at least for large sites. Content management ultimately offers one of the strongest returns on investment for portal technology, as it places control over content in the hands of those who truly own it, freeing up webmasters, HTML programmers, and developers to turn their attention to more challenging tasks than coding static web pages.

The Microsoft .NET platform offers several content management options. SharePoint includes content management along with collaboration, a search engine, and document management. For most large portal sites, you need more content management than SharePoint alone can deliver. Including MCMS gives you tremendous control over the appearance of your web pages and supports a collaborative authoring environment so you can easily add new content and keep existing content fresh. The extensibility of the product along with its melding into the Visual Studio.NET integrated development environment means that you can build a significant amount of your portal solution in MCMS.

By starting with the taxonomy and the templates for static web pages, you can create a basis on which to hang not only static web content but the other applications that make your portal rich in functionality. Chapter 10 provides more detail on how to develop the taxonomy and make it concrete.