Microsoft®

Microsoft® Windows® SharePoint® 3.0 od środka

Ted Pattison
Daniel Larson

Microsoft® Windows® SharePoint® 3.0 od środka
Edycja polska Microsoft Press

Tytuł oryginału: Inside Microsoft® Windows® SharePoint® 3.0

Polish edition by APN PROMISE Sp. z o. o. Warszawa 2007

APN PROMISE Sp. z o. o., biuro: 00-108 Warszawa, ul. Zielna 39
tel. (022) 351 90 00, faks (022) 351 90 99
e-mail: mspress@promise.pl

ISBN: 978-83-7541-010-5.

Przekład: Natalia Chounlamany, Andrzej Bańkowski
Redakcja: Stefan Turalski, Marek Włodarz
Korekta: Magdalena Kalina-Swoboda
Skład i łamanie: Marek Włodarz

Dla Annabelle, Daisy, Sophie i Amy. Jesteście jedynymi komponentami, które w mojej farmie będą zawsze działały w trybie
z Pełnym zaufaniem i Bezwarunkową miłością.

– Ted Pattison

Dla Salliny. Jesteś moją pasją, moją inspiracją, moją muzą,
i moją miłością. Dla naszego nienarodzonego dziecka. Jesteś obdarzone większą miłością
niż można wyrazić słowami.

– Daniel Larson

Spis treści

Podziękowania

Podziękowania Teda

Prawdopodobnie powinienem zacząć od samego początku. Mike Fitzmaurice (vel Fitz) zaprosił mnie do kampusu firmy Microsoft w Redmond w czerwcu w roku 2005. Przez telefon Fitz zdradził mi, że będę miał okazję zobaczyć ich „nowe dzieło", które nadal znajdowało się w ściśle tajnym cyklu alfa. Gdy przyjechałem, Fitz poznał mnie z kilkoma menedżerami programu w zespole SharePoint, którzy, jak się później okazało, byli nieocenionymi źródłami informacji, gdy wraz z Danem prowadziliśmy prace badawcze prowadzące do napisania tej książki. Ze wszystkich pomocnych osób, które poświęciły czas na udzielanie odpowiedzi na nasze pytania, chcielibyśmy w szczególności podziękować Mikowi Ammerlaan, osobie która odpowiedziała na najwięcej naszych pytań i zna zasady działania Windows SharePoint Services wszerz i wzdłuż.

Latem 2005 roku, wraz z grupą Microsoft Developer Platform Evangelism, rozpocząłem pracę nad projektem stworzenia i przeprowadzenia szkoleń dla programistów z zakresu technologii SharePoint 2007. Stałem się członkiem tego zespołu i podczas tworzenia slajdów i ćwiczeń praktycznych zaprzyjaźniłem się z legendarnym Patrickiem Tisseghem z firmy U2U. Chciałbym podziękować Mauricio Ordonez i Scottowi Burmester za umożliwienie mi współuczestniczenia w tym projekcie. Podziękowania należą się także moim współpracownikom m.in. Robowi Barker, Chrisowi Predeek, Dougowi Mahugh, Christin Boyd oraz Donowi Campbell – wszyscy oni pomogli mi w poznaniu usług Windows SharePoint Services i związanych z nimi technologii.

Chciałbym także podziękować wszystkim osobom, które poświęciły swój czas na przeglądanie niniejszej książki. David Robinson przeprowadził techniczną recenzję wszystkich rozdziałów i zaprezentował szereg interesujących spostrzeżeń. Miałem szczęście, gdyż rozdział poświęcony przepływom pracy zaopiniowali czołowi eksperci z branży przemysłowej, tacy jak Eilene Hao, Dharma Shuklaa, Gabe Hall, Scot Hillier, Robert Bogue oraz Ken Getz. Los się do nas uśmiechnął i tak cenieni specjaliści jak Morgan Everett, Mark Collins oraz Stephen Terlecki dokonali dogłębnej i uwzględniającej ostatnie trendy recenzji tworzonego przez Dana rozdziału dotyczącego składników Web Part AJAX.

Mam ogromny dług wdzięczności wobec społeczności MVP produktu SharePoint. Po pierwsze, chciałbym podkreślić, że Lawrence Liu i April Dalke wykonali kawał dobrej roboty w celu zintegrowania społeczności. Mamy prywatną listę dystrybucyjną, za pośrednictwem której systematycznie dzielimy się swoimi doświadczeniami. Chciałbym także podziękować tym ekspertom SharePoint MVP, którzy regularnie nadsyłali posty i tak wiele mnie nauczyli. Zaliczają się do nich m.in. Adam Buenz, Amanda Murphy, Andrew Connell, Bob Fox, Bil Simser, Bill English, Bob Mixon, Carlos Segura Sanz, Cornelius J van Dyk, Darrin Bishop, Brad Smith, Dustin Miller, Eli Robillard, Erol Giraudy, Gary Bushey, Gšran Husman, Heather Solomon, Jan Tielens, Jason Medero, Joel Ward, John Holliday, Joris Poelmans, Kevin Laahs, Loke Kit Kai, Luis Du Solier Grinda, Mads Nissen, Mark Kruger, Matthew McDermott, Mei Ying Lim, Michael Greth, Michael Noel, Mike „the Enforcer" Walsh, Nick Swan, Renaud Comte, Robert Bogue, Shane Perran, Shane Young, Spencer Harbar, Stacy Draper, Stephane Cordonnier, Stephen Cummins, Steve Smith, Steven Collier,

Todd Baginski, Todd Bleeker, Todd O. Klindt, Pierre Vivier-Merle, Woody Windischman oraz Xi Chen.

Chciałbym również podziękować grupie wsparcia technicznego, zwanej również „paczką absolwentów DevelopMentor". Zawsze z czułością będę wspominał czas spędzony w grupie DevelopMentor, w drugiej połowie lat 90., gdy guru był Don Box, COM'y stanowiły nadal uwielbianą technologię, a bańka internetowa rosła każdego dnia. Miałem ogromne szczęście zaprzyjaźnić się i współpracować z prawdziwymi geniuszami, do których należą Don Box, Chris Sells, Tim Ewald, Brent Rector, John Lam, Ingo Rammer, Aaron Skonnard, Ian Griffiths, Keith Brown, Fritz Onion, Jon Flanders, Mike Woodring, Ted Neward, Andrew Gayter, Bob Beauchemin, Brian A. Randell, Brian Maso, Brock Allen, Cal Caldwell, Craig Andera, Dan Sullivan, Dan Weston, Daniel Sinclair, Dominick Baier, Doug Turnure, Dr. Joe Hummel, George Shepherd, Henk de Koning, Jason Masterman, Jason Whittington, Jim Wilson, Kent Tegels, Kevin Jones, Kirk Fertitta, Marcus Heege, Mark Taparauskas, Matt Milner, Niels Berglund, Scott Bloom, Simon Horrell, Steve Rodgers oraz Martin Gudgin. Wszyscy jesteśmy wdzięczni Craigowi Andera za to, że zgodził się przez pierwszych kilka lat utrzymywać serwer pocztowy z listą dystrybucyjną absolwentów DevelopMentor we własnej piwnicy, dzięki czemu grupa mogła dzielić się swoimi technicznymi spostrzeżeniami oraz unikalnym poczuciem humoru.

Chciałbym podziękować także tym pracownikom Microsoft Press, którzy pomogli mi w opublikowaniu niniejszej książki. Szczęśliwy los zesłał mi Bena Ryan jako redaktora prowadzącego i Kathleen Atkins jako redaktora technicznego. Są to te same dwie osoby, z którymi współpracowałem w poprzednim tysiącleciu przy okazji publikacji w wydawnictwie Microsoft Press mojej pierwszej książki dotyczącej VB oraz COM w roku 1998.

Na zakończenie chciałbym podziękować Danowi Larson, współwinnemu powstania tej książki. Dziękuję za chęć wejścia ze mną w spółkę i za pomoc w dostarczeniu książki w roku 2007. Dziękuję za wgłębianie się w schematy list, definicje witryn ustawienia zabezpieczeń CAS oraz za znaczące poprawki w końcowym manuskrypcie. Szczególnie zaimponowała mi twoja biegłość w integracji WSS z rozszerzeniami AJAX dla ASP.NET oraz umiejętność opisania tej technologii bez korzystania z jakichkolwiek dodatkowych zasobów w tak krótkim czasie. Z całą pewnością zasłużyłeś na pseudonim „Kapitan AJAX".

Podziękowania Dana

Chociaż większość moich podziękowań pokrywa się z podziękowaniami Teda, dużą część swojego doświadczenia zawdzięczam zespołom Microsoft Consulting i społeczności programistów z Denver w Colorado. Anthony Petro oraz Marcus Hass odegrali znaczącą rolę, udzielając mi wskazówek i dzieląc się dogłębną wiedzą dotyczącą platformy WSS, a także pierwszych, poważniejszych wdrożeń internetowych rozwiązań WSS oraz SharePoint Portal Server. Dzięki nim dowiedzieliśmy się wiele o tym, jak rozłożyć platformę WSS na podstawowe elementy i jak wykorzystać nowo zdobyte doświadczenia – wiele z tych zagadnień zostanie poruszonych w niniejszej książce.

Chciałbym również podziękować zespołom Microsoft Consulting, z którymi miałem okazję współpracować, a w szczególności Scottowi Short. Scott nauczył mnie, jak powinny być pisane składniki SDK, z których później skorzystają programiści. Umiejętność ta okazała się bardzo pomocna i znalazła odzwierciedlenie w wielu moich projektach oraz zapewne także

w tej książce. Scott nauczył mnie tworzenia szkieletów zamiast aplikacji i pracy z zespołami programistów w celu zapewnienia pomyślnej adaptacji technologii.

Klienci, z którymi współpracowaliśmy przez lata, wnieśli swoje praktyczne doświadczenia, na podstawie których zbudowana została ta książka. Chciałbym podziękować Davidowi Levstik, Davidowi Weinstein, Johnowi Daniels, Lei Yu, Jackowi Deverter i pozostałym klientom, z którymi miałem okazję pracować ostatnimi czasy.

Społeczność programistów z Denver, do której należą Joe Mayo oraz Roy Ogborn, była bardzo pomocna, szczególnie podczas stawiania kolejnych kroków kariery. To jej zawdzięczam większość wiedzy dotyczącej C#, ASP.NET i innych technologii firmy Microsoft i to waśnie tej społeczności chciałbym złożyć szczególne wyrazy wdzięczności. Bardzo pomogli mi również Chris Wallace i grupa użytkowników Visual Studio .NET z Denver oraz grupa użytkowników SharePoint Rocky Mountain, w tym Kris Syverstad i Matt Passannante, którym to zawdzięczam możliwość przeprowadzenia pierwszych dyskusji dotyczących technologii .NET i SharePoint.

Chciałbym również podziękować mojemu zespołowi programistów w firmie *NewsGator* za przedstawienie wielu doświadczeń ze świata rzeczywistego oraz zaopiniowanie wzorców architektury AJAX zaprezentowanych w tej książce. Specjalne podziękowania kieruję do Lane Mohler, Briana Agnes, Sherstin Lauman, Darrina Long, Josha Aragon oraz Toma McIntyre.

Jestem także ogromnie wdzięczny mojej żonie Sallinie za cierpliwość i wsparcie udzielone w czasie pracy nad tą książką. Sallino, bez ciebie nie zdołałbym jej napisać.

Na zakończenie chciałbym podziękować Tedowi za wkład w moją część publikacji i za wspólną pracę nad niniejszą książką. Spędziliśmy wspaniały czas, tworząc razem tę książkę i sądzimy, że widać to na każdej z jej stron. Mamy nadzieję, że lektura tej książki dostarczy tyle przyjemności ile jej pisanie.

Przedmowa

Pojawienie się Teda Pattison na scenie technologii SharePoint w roku 2004 stanowiło wyjątkowo miły kamień milowy. Znany szkoleniowiec z poważnymi referencjami programistycznymi w technologii .NET docenił potencjał platformy Windows SharePoint Services. Ted zaczął publikować artykuły dotyczące produktów i technologii SharePoint w swojej kolumnie w magazynie *MSDN*. Tworzył materiały szkoleniowe. Prowadził szkolenia i wysokiej jakości prezentacje na konferencjach. Zauważyliśmy to.

Ted jest jedną z osób, które zaangażowaliśmy do współpracy z naszym zespołem inżynierskim już w bardzo wczesnej fazie cyklu rozwoju Windows SharePoint Services 3.0 w celu napisania wstępnej dokumentacji dla pierwszych użytkowników beta, która została później opublikowana w postaci artykułów SDK. Ted stworzył i dostarczał kursy dla klientów i partnerów wytypowanych do początkowych programów adaptacyjnych. Doświadczenia zdobyte podczas szkolenia wielu programistów pozwoliły mu uzyskać wyjątkowo dogłębny obraz ich potrzeb oraz tego, czego muszą się oni nauczyć, aby z powodzeniem wykorzystywać technologię SharePoint.

A co najważniejsze, nie postrzegamy Teda jako „programisty SharePoint". Ted jest programistą .NET, który odkrył, jak dołączyć technologię SharePoint do wachlarza swoich umiejętności i narzędzi. Wie, kiedy warto skorzystać ze stworzonej przez nas technologii oraz jak użyć jej w taki sposób, aby zharmonizować ją ze standardowymi praktykami programistycznymi wykorzystywanymi do rozwijania rozwiązań .NET, a w szczególności ASP.NET. Czas, jaki spędził on na przeprowadzaniu szkoleń w DevelopMentor oraz we własnej firmie, pogłębił jego umiejętności w zakresie objaśniania technologii i przekładania wiedzy na praktyczne działanie.

Oprócz Patricka Tisseghem (twórcy książki Microsoft Press dotyczącej programowania w Microsoft Office SharePoint Server 2007 stanowiącej idealne uzupełnienie niniejszej pozycji), z którym Ted często podejmował współpracę, nie istnieją osoby, które spędziłyby więcej czasu myśląc, prowadząc szkolenia i mówiąc o programowaniu z wykorzystaniem produktów i technologii SharePoint.

Daniel Larson jest idealnym współautorem dla Teda, ponieważ posiada komplementarne doświadczenie. Dan jest zaangażowany w rozwój naszej technologii od pięciu lat i regularnie prowadzi blog na temat programowania w technologii SharePoint. Spędza czas z naszym zespołem inżynierskim, szukając sposobów na zintegrowanie najnowszych osiągnięć ASP.NET ze stronami dostarczanymi w witrynach SharePoint. Jest entuzjastą technologii internetowych skierowanych do społeczności, takich jak RSS czy OPML.

Ale co ważniejsze, Dan realizuje w praktyce głoszone przez siebie koncepcje. W firmie NewsGator zajmuje się tworzeniem aplikacji portalowych na światowym poziomie, wykorzystując produkty i technologie SharePoint zintegrowane z RSS, AJAX i siecią społeczną. Dan osobiście doświadczył, jakie rozwiązania sprawdzają się w praktyce i wie, jak z sukcesem tworzyć wspaniałe produkty przy użyciu technologii SharePoint.

Niniejsza książka zawiera wiele wartościowych informacji. Opisuje internetowy interfejs użytkownika, technologię magazynowania dokumentów i danych oraz gotowe do użycia w aplikacji funkcje, takie jak przepływy pracy czy typy zawartości. A co ważniejsze, prezentuje zalecenia dotyczące stylu organizowania projektów kodu, które znacznie podnoszą efektywność pisania, utrzymywania, rozmieszczania i uaktualniania kodu źródłowego.

Rozwijając usługi Windows SharePoint Services, w rzeczywistości tworzymy rozszerzenia i elementy wykorzystywane przez istniejącą aplikację sieci Web. Ted i Dan zrozumieli tę zależność i dostosowali do tej rzeczywistości zaprezentowane przykłady i porady. Efekty są rewelacyjne. A zatem - miłej lektury!

Derek Burney
Dyrektor generalny Windows SharePoint Services
Microsoft

Wstęp

Przyglądając się zmianom zachodzącym na platformie Windows, trudno nie zauważyć, że popularność produktu „SharePoint" stale rośnie. Technologie SharePoint oferują efektywne rozwiązania wykorzystywane podczas tworzenia witryn zespołów, usprawniając współpracę w środowisku bazującym na sieci LAN. Z drugiej strony technologie SharePoint ułatwiają zarządzanie zawartością witryn internetowych, które mogą być skalowane tak, aby obsłużyć nawet tysiące użytkowników w środowisku farmy sieci Web. I chociaż już wbudowana funkcjonalność technologii SharePoint stanowi ogromną wartość, można osiągnąć o wiele więcej, gdy potrafi się wykorzystać Windows SharePoint Services jako platformę programistyczną.

Niniejsza książka ma za zadanie pomóc w projektowaniu i rozwijaniu niestandardowych rozwiązań biznesowych opartych na Windows SharePoint Services 3.0 (WSS). Naszym celem jest nauczenie czytelnika tworzenia, debugowania i instalowania podstawowych elementów wchodzących w skład platformy, takich jak funkcje, definicje witryn, szablony stron, składniki Web Part, schematy list, typy zawartości, programy obsługi zdarzeń i szablony przepływów pracy. Gdy już włoży się trochę wysiłku i nauczy tworzenia tych elementów, możliwości budowania różnego typu aplikacji i rozwiązań na platformie Windows SharePoint Services są praktycznie nieograniczone. Ponadto pozyskane umiejętności pozwolą na rozwijanie funkcjonalności dostępnej w programie Microsoft Office SharePoint Server 2007 (MOSS).

Dla kogo jest ta książka

Niniejsza książka została napisana z myślą o programistach, którym znane są platformy Visual Studio 2005, Microsoft .NET 2.0 Framework oraz ASP.NET. Przykłady kodu dołączone do tej książki zostały napisane w języku C#. Niniejsza publikacja dostarcza programistom i architektom pełnego przeglądu technologii Windows SharePoint Services 3.0 oraz prezentuje specjalistyczne porady dotyczące rozwijania aplikacji na tej platformie. Książka ta może przynieść korzyści zarówno programistom, którzy nie mieli wcześniej okazji wykorzystywać Windows SharePoint Services, jak i doświadczonym programistom WSS.

Wymagania systemowe

Do zbudowania i uruchomienia przykładów kodu dołączonych do niniejszej książki potrzebny będzie następujący sprzęt i oprogramowanie:

- Microsoft Windows Server 2003 z Service Pack 1 lub Microsoft Windows Server 2003 R2 (instalacja natywna lub maszyna wirtualna)
- Microsoft Windows SharePoint Services 3.0
- Microsoft Visual Studio 2005 Standard Edition lub Microsoft Visual Studio 2005 Professional Edition lub Microsoft Visual Studio 2005 Team Suite

- W przypadku instalacji natywnej zalecamy komputer z procesorem minimum 1 GHz Pentium i pamięcią RAM o rozmiarze przynajmniej 1 GB
- W przypadku instalacji na maszynie wirtualnej Virtual PC zalecany jest komputer z procesorem minimum 2 GHz Pentium i pamięcią RAM o rozmiarze przynajmniej 2 GB

Przykłady kodu

Wszystkie przykłady kodu omówione w niniejszej książce mogą zostać pobrane ze strony z materiałami pomocniczymi dla tej książki pod następującym adresem:

http://www.microsoft.com/mspress/companion/9780735623200/

Wsparcie techniczne

Dołożyliśmy wszelkich starań, aby zapewnić wysoką jakość tej książki i jej materiałów pomocniczych. Wydawnictwo Microsoft Press dostarcza wsparcia dla niniejszej książki za pośrednictwem następującej witryny:

http://www.microsoft.com/learning/support/books/

Podstawowe aktualizacje i erratę znaleźć można pod adresem:

http://www.TedPattison.net/InsideWSS

Pytania i komentarze

Komentarze, pytania lub pomysły dotyczące niniejszej książki oraz materiałów pomocniczych lub pytania, na które odpowiedzi można nie znaleźć w podanych powyżej witrynach, prosimy przesyłać do wydawnictwa Microsoft Press za pośrednictwem poczty e-mail na adres:

mspinput@microsoft.com

lub poczty tradycyjnej na adres:

Microsoft Press
Attn: *Inside Microsoft® Windows® SharePoint® Services 3.0* Editor
One Microsoft Way
Redmond, WA 98052-6399

Pod żadnym z powyższych adresów nie jest udzielane wsparcie techniczne dla produktów firmy Microsoft.

Rozdział 1

Wprowadzenie

- **Poznaj podstawowe zagadnienia i terminy dotyczące usług Windows SharePoint Services (WSS).**
- **Poznaj podstawy inicjowania obsługi administracyjnej witryn.**
- **Poznaj możliwości dostosowywania usług WSS do własnych potrzeb.**
- **Zbadaj funkcje współpracy wbudowane w usługi WSS.**
- **Poznaj możliwości rozwijania i rozszerzania usług WSS.**

Inicjowanie obsługi administracyjnej witryn

Jeszcze dziesięć lat temu duża część firm dopiero poznawała sieć WWW i odkrywała możliwość dotarcia do konsumentów za pośrednictwem treści prezentowanej w postaci stron HTML. Większość firm stworzyła pojedynczą witrynę udostępnianą w Internecie, która spełniała wszystkie potrzeby związane z reklamowaniem produktów i usług.

Od tamtej pory wiele się zmieniło. W dzisiejszych czasach firmy wykorzystują witryny sieci Web nie tylko jako metodę dotarcia do klientów, ale również w celu zapewniania dostawcom, pracownikom i zarządowi dostępu do aplikacji. Nierzadko duże korporacje tworzą setki lub nawet tysiące witryn sieci Web. Niektóre z nich wymagają na przykład stworzenia nowej witryny na potrzeby każdej nowej kampanii marketingowej lub nowo zatrudnionego pracownika.

W świecie bez produktów i technologii SharePoint utrzymywanie i zarządzanie nieustannie tworzonymi witrynami sieci Web firmy może być drogie i pracochłonne. Przyjrzyjmy się na przykład, z czym wiąże się proces tworzenia witryny sieci Web zaprojektowanej z myślą o śledzeniu źródeł kontaktów i kontrahentów handlowych.

Po pierwsze, proces ten wymaga, aby administrator baz danych stworzył na serwerze SQL Server lub w innym systemie zarządzania bazą danych nową bazę danych i tabele konieczne do przechowywania danych handlowych. Następnie, programista musi stworzyć witrynę ASP.NET wraz ze stronami sieci Web i wymaganym kodem służącym do wyświetlania i edycji tychże danych. Po przygotowaniu witryny ASP.NET do publikacji, administrator systemu musi skopiować pliki aplikacji ASP.NET na docelowy serwer sieci Web i skonfigurować Internetowe Usługi Informacyjne 6.0 (IIS) poprzez stworzenie nowego katalogu wirtualnego. Jeśli środowiskiem, w którym ma zostać zainstalowana aplikacja ASP.NET, jest farma sieci Web, wymagania wzrastają, ponieważ powyższe procedury administracyjne muszą zostać powielone na wszystkich serwerach frontonu sieci Web w farmie.

Jak łatwo sobie wyobrazić, proces przygotowywania i uruchamiania nowej witryny sieci Web przy użyciu powyższego rozwiązania może ciągnąć się tygodniami, a nawet miesiącami – ze względu na konieczność skoordynowania działań administratora baz danych, programisty Web oraz administratora systemu. Usługi WSS zostały zaprojektowane tak, aby zwiększyć efektywność i przyspieszyć proces tworzenia witryn sieci Web. Programista WSS może tworzyć komponenty, które są następnie wykorzystywane do tworzenia witryn i obszarów roboczych.

Jądro usług WSS stanowi silnik służący do tworzenia i obsługi witryn. Architektura usług WSS została zaprojektowana w szczególności z myślą o środowisku farmy sieci Web. Dzięki usługom WSS dowolny członek działu IT może zainicjować obsługę administracyjną witryny (w uproszczeniu stworzyć witrynę) w czasie mniejszym niż jedna minuta, wprowadzając wymagane informacje za pomocą formularza z poziomu przeglądarki. Administrator baz danych nie musi tworzyć nowej bazy danych ani żadnych tabel. Programista ASP.NET nie musi tworzyć nowej witryny sieci Web. Administrator systemu nie musi kopiować żadnych plików ani konfigurować ustawień usług IIS na serwerze frontonu sieci Web.

Silnik WSS do inicjowania obsługi administracyjnej witryn bazuje na zintegrowanym modelu składowania, który obejmuje wiele baz danych SQL Server służących do przechowywania zawartości i danych konfiguracyjnych. Instalując usługi WSS, można wybrać opcję wykorzystania serwera SQL Server 2005 lub SQL Server 2000. W przypadku prostych scenariuszy rozwoju i wdrożenia można wykorzystać również serwer SQL Express, co eliminuje konieczność wykupienia licencji na oprogramowanie SQL Server.

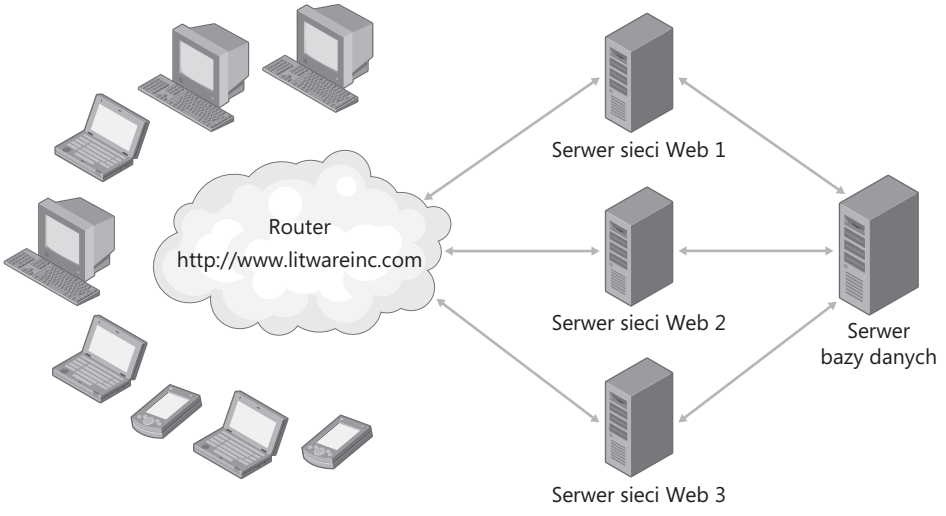
Microsoft Office SharePoint Server 2007

Istotne jest zrozumienie różnicy pomiędzy usługami Windows SharePoint Services (WSS) a programem Microsoft Office SharePoint Server 2007 (MOSS). Zarówno WSS, jak i MOSS stanowią fragmenty oprogramowania stworzone przez zespół Microsoft Office. Jednak usługi WSS stanowią element systemu operacyjnego Windows Server 2003, natomiast MOSS to osobny produkt ze swoją własną licencją SKU. Usługi WSS należy postrzegać jako platformę wewnętrzną, a MOSS jako dodatkowy zestaw komponentów i usług, które wzbogacają tę platformę.

Usługi WSS nie zawierają własnego modelu licencjonowania, ich wykorzystanie obejmuje licencja systemu Windows Server 2003. To sprawia, że zastosowanie aplikacji zaprojektowanych i zbudowanych w oparciu o platformę WSS może być dla firm bardzo opłacalne. Natomiast program MOSS ma swój własny model licencjonowania, który obejmuje licencje po stronie serwera i licencje dostępowe (CAL). Model licencjonowania MOSS jest dodatkowo podzielony na edycję Standard i Enterprise.

Każde wdrożenie WSS opiera się na koncepcji farmy. *Farma* stanowi w uproszczeniu zbiór jednego lub więcej komputerów współpracujących ze sobą w celu dostarczenia funkcjonalności WSS do klientów. W najprostszym scenariuszu wdrożenia farma WSS składa się z pojedynczego serwera, który pełni rolę zarówno serwera frontonu sieci Web, jak i serwera baz danych SQL Server. Bardziej skompilowana farma WSS składać się może z kilku serwerów frontonu sieci Web i dedykowanego serwera baz danych, jak pokazano na Rysunku 1-1.

Każda farma WSS obejmuje jedną bazę danych SQL Server zwaną *bazą danych konfiguracji*. Baza danych konfiguracji śledzi ważne informacje dotyczące całej farmy. Na przykład informacje, które serwery frontonu sieci Web są powiązane z farmą oraz którym użytkownikom nadane zostały uprawnienia administratora usług WSS na poziomie farmy.



Rysunek 1-1 Farma WSS dostarcza aplikacje sieci Web.

Usługi WSS są zbudowane w oparciu o platformę Internetowych Usług Informacyjnych 6.0 (IIS). A w szczególności powierzają witrynom sieci Web obsługę przychodzących żądań HTTP. W związku z tym trzeba zrozumieć, czym tak naprawdę jest witryna sieci Web usług IIS. *Witryna sieci Web usług IIS* stanowi punkt wejścia w infrastrukturze serwera IIS. Na przykład, Domyślna Witryna Sieci Web tworzona automatycznie przez serwer IIS nasłuchuje przychodzących żądań HTTP na porcie 80. Można tworzyć kolejne witryny sieci Web usług IIS, aby zapewnić dodatkowe punkty wejścia wykorzystujące inne numery portów, inne adresy IP lub inne nagłówki hosta.

Istotną cechą charakterystyczną witryn sieci Web usług IIS jest to, że ich ustawienia zabezpieczeń są konfigurowane niezależnie od pozostałych witryn sieci Web usług IIS. Na przykład można skonfigurować Domyślną Witrynę Sieci Web jako upublicznioną witrynę, która dopuszcza wykorzystanie uwierzytelniania podstawowego i zezwala na anonimowy dostęp i równocześnie stworzyć drugą witrynę sieci Web wykorzystywaną w obrębie korporacyjnej sieci LAN na innym porcie np. 1000. Po stworzeniu na serwerze IIS witryny intranetowej można skonfigurować ją tak, aby wymagała ona zintegrowanego uwierzytelniania systemu Windows i nie zezwalała na dostęp anonimowy.

Jeśli witryna sieci Web usług IIS ma być wykorzystywana do obsługi witryn WSS, musi zostać specjalnie skonfigurowana. Z omówieniem technicznych szczegółów tej konfiguracji wstrzymamy się do kolejnego rozdziału. Teraz chcemy wprowadzić jedynie pewne najistotniejsze pojęcia i terminy.

Witryna sieci Web usług IIS, która została skonfigurowana specjalnie z myślą o uruchamianiu witryn WSS, nazywana jest *aplikacją sieci Web*. Każda witryna WSS działa w kontekście określonej aplikacji sieci Web. Jest to istotne, ponieważ aplikacja sieci Web pełniąca rolę hosta dla witryny WSS odpowiada za pewne ważne aspekty środowiska WSS m.in. konfigurację zabezpieczeń uwierzytelniania użytkowników.

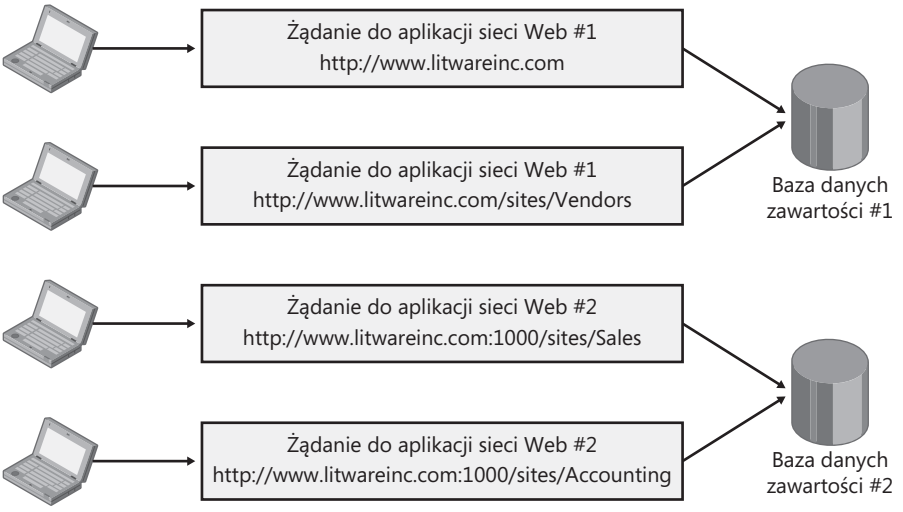
Proces instalacji usług WSS obejmuje stworzenie i konfigurację aplikacji sieci Web o nazwie *Administracja centralna programu SharePoint 3.0*. Aplikacja Administracja Centralna Programu SharePoint 3.0 dostarcza strony, które pozwalają na realizowanie podstawowych zadań administratorskich, takich jak konwertowanie standardowej witryny sieci Web usług

IIS na aplikację sieci Web usług WSS. Administracja Centralna Programu SharePoint umożliwia również stworzenie nowej witryny sieci Web usług IIS i zapewnia jej automatyczną konfigurację na potrzeby aplikacji sieci Web usług WSS, bez konieczności bezpośredniego wykorzystywania żadnego z narzędzi administracyjnych usług IIS.

Aplikacje sieci Web kontra Serwery wirtualne

W ostatniej wersji usług WSS zespół ds. produktu wykorzystywał termin *serwer wirtualny (ang. virtual server)* do opisania witryny sieci Web usług IIS, która została wzbogacona o funkcjonalność WSS. W aktualnej wersji usług WSS i jej dokumentacji pomocniczej termin *serwer wirtualny* został zastąpiony terminem *aplikacja sieci Web* – głównie po to, aby uniknąć nieprawidłowych skojarzeń z innym produktem firmy Microsoft o tej samej nazwie. Jednak programiści WSS powinni pamiętać, że nowy termin *aplikacja sieci Web* oraz poprzedni termin *serwer wirtualny* mogą być często używane zamiennie. Na przykład model obiektowy WSS zawiera klasę SPVirtualServer służącą do programowania obiektów aplikacji sieci Web.

W typowej farmie WSS istnieje zazwyczaj kilka różnych aplikacji sieci Web. Administracja Centralna Programu SharePoint jest konfigurowana podczas instalacji jako osobna aplikacja sieci Web wykorzystywana przez usługi WSS. Potrzebna jest jedna lub więcej innych aplikacji sieci Web do tworzenia i zarządzania witrynami, które będą dostępne dla użytkowników końcowych. Można, na przykład, skonfigurować Domyślną Witrynę Sieci Web jako aplikację sieci Web usług WSS, tak aby witryny WSS były udostępnianie za pośrednictwem standardowego portu 80 protokołu HTTP. Można również stworzyć w farmie dodatkowe aplikacje sieci Web, jak np. intranetową aplikację sieci Web na porcie 1000. Dane konfiguracyjne usług WSS dla całej farmy są przechowywane w bazie danych konfiguracji, a dane powiązane z witrynami WSS są umieszczane w innego typu bazie danych, zwanej *bazą danych zawartości*. Podczas tworzenia nowej aplikacji sieci Web za pośrednictwem Administracji Centralnej Programu SharePoint 3.0, usługi WSS tworzą nową bazę danych zawartości. W prostym modelu wdrażania farma zawiera tylko jedną bazę danych zawartości dla każdej aplikacji sieci Web, jak pokazano na Rysunku 1-2. W scenariuszach wymagających wyższego poziomu zabezpieczeń lub bardziej szczegółowego planowania składowania można wykorzystywać bardziej zaawansowane procedury administracyjne w celu umieszczenia witryn znajdujących się w aplikacji sieci Web w wielu bazach danych zawartości.



Rysunek 1-2 Farma WSS dostarcza aplikacje sieci Web przechowujące dane w bazach danych zawartości.

Wskazówka Tworzone przez programistów oprogramowanie dla usług WSS nie może uzyskiwać bezpośredniego dostępu do bazy danych konfiguracji ani baz danych zawartości. Dlatego trzeba oprzeć się pokusie napisania na przykład kodu ADO.NET, który odczytuje lub zapisuje dane w tabelach wewnątrz tych baz danych. Zamiast tego należy wykorzystać w kodzie interfejsy programistyczne API usług WSS. Dzięki temu usługi WSS uruchomią kod systemowy, który będzie pośredniczył w uzyskaniu dostępu do bazy danych konfiguracji i/lub bazy danych zawartości.

Witryny i zbiory witryn

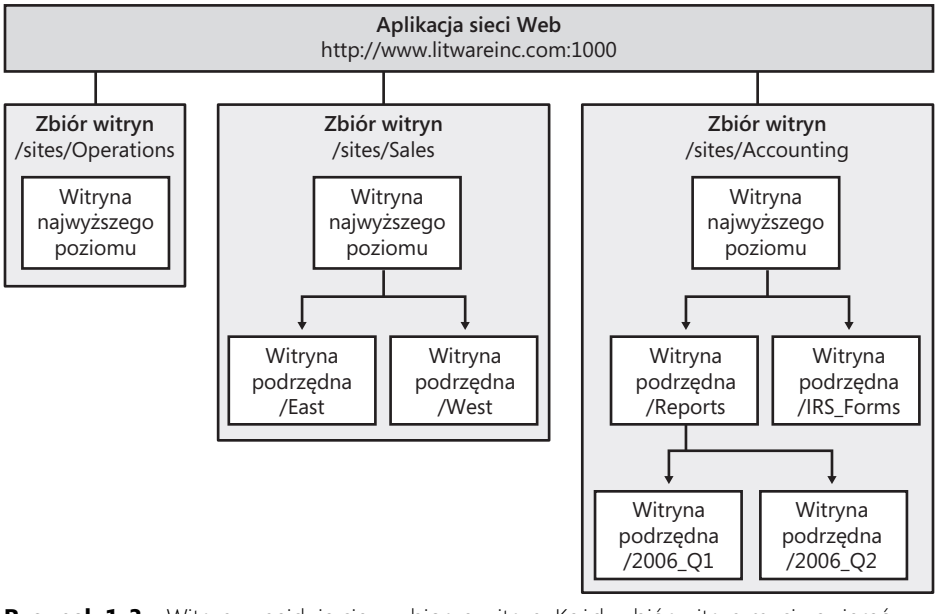
Warto zatrzymać się na chwilę, aby doprecyzować termin *witryna WSS*, który wykorzystywany był w przedstawionym do tej pory omówieniu. Po pierwsze, *witryna WSS* to pojemnik do przechowywania zawartości. Zawartość witryny jest przechowywana głównie w postaci list, bibliotek dokumentów i witryn podrzędnych. Po drugie, witryna stanowi jednostkę zabezpieczeń, której zawartość jest dostępna dla konfigurowalnego zbioru użytkowników. Witryna może definiować swój własny zbiór użytkowników lub dziedziczyć ustawienia z witryny nadrzędnej. Witryna może także zawierać konfigurowalny zestaw grup i uprawnień, które definiują poziom dostępności poszczególnych użytkowników dla list i bibliotek dokumentów znajdujących się w witrynie.

Należy zauważyć, że usługi WSS w rzeczywistości nie dokonują uwierzytelniania użytkowników, pozostawiają to serwerowi IIS i infrastrukturze dostawcy uwierzytelniania ASP.NET. Jednak przejmują przywództwo, gdy do głosu dochodzi autoryzacja. Usługi WSS dostarczają elementy interfejsu użytkownika oraz kod pomocniczy umożliwiający uprzywilejowanym użytkownikom konfigurowanie autoryzacji w odniesieniu do różnych elementów witryn. Usługi WSS 3.0 wprowadzają także mechanizmy stosowania zabezpieczeń, które sprawiają, że polecenia i łącza do elementów witryny są pokazywane wyłącznie użytkownikom, którzy mają do nich autoryzowany dostęp.

Po trzecie, witryna stanowi aplikację z rozszerzalnym, w pełni dostosowywalnym interfejsem użytkownika. Administrator witryny może tworzyć strony i dostosowywać ich układ oraz wygląd. Administrator witryny może także modyfikować strukturę nawigacyjną witryny przy użyciu przeglądarki.

Witryna stanowi fundament dla wykorzystania technologii stron i składników Web Part firmy Microsoft. Administratorzy witryn mogą dostosowywać strony składników Web Part, dodając i konfigurując poszczególne składniki. Użytkownik może personalizować stronę składników Web Part, modyfikując, dodając i/lub usuwając składniki Web Part. Wszystkie dane dostosowań i personalizacji powiązane ze składnikami Web Part na stronie składników Web Part są automatycznie składowane w bazie danych zawartości.

Dla każdej witryny WSS musi zostać zainicjowana obsługa administracyjna w zakresie istniejącej aplikacji sieci Web. Jednak witryna nie może istnieć jako niezależna jednostka w aplikacji sieci Web, każda witryna WSS musi być stworzona w zakresie *zbioru witryn*. Zbiór witryn stanowi pojemnik dla witryn WSS. Każdy zbiór witryn wymaga obecności witryny najwyższego poziomu. Poza wymaganą witryną najwyższego poziomu zbiór witryn może zawierać hierarchię witryn podrzędnych. Na Rysunku 1-3 zaprezentowane zostały różne, dozwolone zbiory witryn. Pierwszy zbiór witryn zawiera jedynie witrynę najwyższego poziomu. Drugi zbiór witryn zawiera jeden poziom witryn podrzędnych pod witryną najwyższego poziomu. Trzeci zbiór witryn zawiera bardziej złożoną hierarchię witryn podrzędnych.



Rysunek 1-3 Witryny znajdują się w zbiorze witryn. Każdy zbiór witryn musi zawierać witrynę najwyższego poziomu i może zawierać hierarchię witryn podrzędnych.

Jedną z pierwszych wątpliwości, które pojawiają się, gdy firma rozpoczyna wykorzystywanie technologii WSS lub MOSS, jest sposób rozmieszczenia witryn w zbiorach witryn. Czy należy na przykład stworzyć jeden duży zbiór witryn z wieloma witrynami podrzędnymi, czy może lepiej stworzyć osobne zbiory witryn? Aby podjąć najtrafniejszą decyzję, należy rozważyć wszystkie związane z nią kwestie, które zostaną omówione w kilku kolejnych

akapitach. Trzeba zrozumieć wpływ rozmieszczenia witryn w zbiorach witryn na zakres przywilejów administracyjnych, granice zabezpieczeń, operacje tworzenia i przywracania kopii zapasowych oraz projekt witryny.

Być może niektórzy czytelnicy zadają sobie pytanie, dlaczego architektura WSS wymaga istnienia specjalnego pojemnika do przechowywania witryn. Przede wszystkim zbiory witryn reprezentują zakres przywilejów administracyjnych. Użytkownik, któremu przypisana została rola administratora zbioru witryn, ma pełne administracyjne uprawnienia we wszystkich witrynach istniejących i tworzonych w przyszłości w tym zbiorze witryn.

Łatwo sobie wyobrazić ciężar zarządzania witrynami w dużej korporacji, która obsługuje tysiące witryn rocznie. Obowiązek administrowania tymi wszystkimi stronami w rozsądnym czasie przekracza możliwości większości działów IT. Koncepcja zbioru witryn jest tak ważna, ponieważ umożliwia działowi IT przeniesienie tego obciążenia na działy biznesowe stanowiące właścicieli poszczególnych witryn.

Weźmy pod uwagę następujący przykład. W wielu firmach wykorzystujących usługi WSS w dziale IT istnieje osoba odpowiedzialna za obsługę zbiorów witryn na żądanie jednego z działów biznesowych. W procesie tworzenia dział IT przypisuje jednemu lub kilku użytkownikom z działu biznesowego uprawnienia administratorskie na poziomie nowego zbioru witryn. Od tego momentu użytkownicy-administratorzy z działu biznesowego mogą dodawać użytkowników oraz tworzyć elementy witryny, takie jak listy czy biblioteki dokumentów bez jakiejkolwiek pomocy działu IT. Mogą także dodawać witryny podrzędne do witryny najwyższego poziomu. To umożliwia im rozbudowywanie hierarchii witryn podrzędnych i konfigurowanie zabezpieczeń w sposób odpowiadający ich potrzebom.

Klucz deszyfrujący WSS

Programiści, którzy rozpoczynają pracę z usługami WSS, często czują się nieco zdezorientowani w związku z inną terminologią wykorzystywaną w usługach WSS oraz w ich prekursorze – usługach SharePoint Team Services (STS) w wersji 1.0. Na przykład występujący w usługach WSS termin *zbiór witryn (ang. site collection)* stanowi odpowiednik znanego z STS terminu w*itryna (ang. site)*. Nowy termin WSS *witryna (ang. site)* stanowi odpowiednik starego terminu STS *sieć Web (ang. Web)*. Nowy termin WSS w*itryna najwyższego poziomu (ang. top-level site)* stanowi odpowiednik starego terminu STS *główna sieć Web (ang. root Web)*.

Chociaż zespół WSS konsekwentnie wykorzystywał nową terminologię WSS w dokumentacji produktu, nadal istnieje sporo wystąpień terminu *sieć Web* zamiast oczekiwanego terminu *witryna* oraz terminu *witryna* zamiast oczekiwanego terminu *zbiór witryn*.

Na starej terminologii STS bazują na przykład nazwy klas w modelu obiektowym WSS. W efekcie programując zbiór witryn, wykorzystuje się obiekt SPSite, a programując witrynę obiekt SPWeb. Obiekt SPSite zawiera publiczną właściwość o nazwie RootWeb, która zwraca obiekt SPWeb reprezentujący witrynę najwyższego poziomu w zbiorze witryn. Gdy już rozpozna się potencjalne źródło rozbieżności, opanowanie różnych aspektów usług WSS staje się łatwiejsze.

Drugim argumentem przemawiającym za wykorzystywaniem zbiorów witryn jest to, że stanowią one zakres dla członkostw i uwierzytelnień. Z założenia każdy zbiór witryn jest niezależny od innych zbiorów witryn pod względem zdefiniowanych grup zabezpieczeń, użytkowników dodanych w roli członków oraz autoryzacji wybranych użytkowników do wybranych operacji.

Wyobraźmy sobie na przykład, że dział IT w firmie Litware posiada jeden zbiór witryn dla działu Sprzedaży i jeden dla działu Księgowości. Mimo iż użytkownicy w dziale Księgowości mają uprawnienia administratorskie w swoim zbiorze witryn, nie mogą oni w żaden sposób wpływać na konfigurację zabezpieczeń zbioru witryn Sprzedaży. Jest tak, ponieważ architektura WSS widzi każdy zbiór witryn jako jednostkę niezależną pod względem konfiguracji zabezpieczeń.

Trzecim powodem wykorzystania zbiorów witryn jest to, że zapewniają one wygodny zakres dla operacji wykonywania i przywracania kopii zapasowych. Można wykonać kopię zapasową zbioru witryn i w późniejszym czasie przywrócić ją z zachowaniem pełnej wierności. Przywracanie zbioru witryn może odbywać się w tej samej lokalizacji, w której wykonana została kopia zapasowa. Ewentualnie zbiór witryn może zostać przywrócony w innej lokalizacji, a nawet na innej farmie. Technika wykonywania kopii zapasowej zbioru witryn i przywracania jej w innej lokalizacji stanowi wsparcie dla strategii przenoszenia witryn WSS ze środowiska programistycznego do środowiska publikującego, a następnie do środowiska produkcyjnego.

Narzędzie wiersza polecenia STSADM.EXE

Usługi WSS są dostarczane wraz z przydatnym narzędziem wiersza poleceń o nazwie STSADM.EXE. To narzędzie umożliwia wydawanie komend z poziomu wiersza poleceń systemu Windows i tworzenie plików wsadowych realizujących zadania administracyjne, takie jak tworzenie zbiorów witryn oraz wykonywanie i przywracanie ich kopii zapasowych. Po uruchomieniu narzędzia z wiersza polecenia lub pliku wsadowego należy przekazać parametr –o wraz z jedną ze wspieranych operacji. Oto przykład instrukcji wiersza polecenia służącej do stworzenia nowego zbioru witryn pod określonym adresem URL.

```
STSADM.EXE -o CreateSite -url http://localhost/sites/Sales
                         -ownerlogin LitwareServer\BrianC
                         -owneremail brianc@litwareinc.com
                         -sitetemplate STS#0
```

W zaprezentowanym przykładzie w celu poprawienia czytelności wprowadzone zostały załamania linii między parametrami. Jednak w rzeczywistości uruchamiając narzędzie STSADM z wiersza polecenia lub pliku wsadowego, nie można wykorzystywać znaków końca linii.

Należy pamiętać, że instalacja usług WSS powoduje dodanie narzędzia STSADM.EXE do katalogu systemowego WSS wewnątrz katalogu Windows Program Files. Aby móc wywoływać to narzędzie bezpośrednio z wiersza polecenia na stacji roboczej, należy dodać następującą ścieżkę do zmiennej systemowej Path.

```
c:\program files\common files\microsoft shared\web server extensions\12\bin\
```

Pisząc plik wsadowy, należy również przyjąć założenie, że może być on uruchamiany na maszynie, na której nie zostały odpowiednio skonfigurowane zmienne środowiskowe. Z tego względu należy pisać pliki wsadowe, które w sposób jawny określają lokalizację narzędzia STSADM.EXE.

```
@SET STSADM="c:\program files\common files\microsoft shared\
            web server extensions\12\bin\stsadm"

%STSADM% -o CreateSite -url http://localhost/sites/Sales
                        -ownerlogin LitwareServer\BrianC
                        -owneremail brianc@litwareinc.com
                        -sitetemplate STS#0
```

Kolejny raz dla poprawy czytelności wprowadzone zostały załamania linii. Tworząc rzeczywisty plik wsadowy, należy je usunąć.

Ostatnim aspektem, który warto wziąć pod uwagę, rozważając wykorzystanie zbiorów witryn, jest to, że stanowią one zakres dla różnego typu elementów witryny i niestandardowych kwerend. Na przykład model obiektowy WSS pozwala na uruchamianie kwerend, które obejmują wszystkie listy w zbiorze witryn. Jednak ten wygodny mechanizm wykorzystywania kwerend nie może obejmować zakresem wielu różnych zbiorów witryn. Dlatego jeśli projekt aplikacji zakłada potrzebę wywoływania kwerend agregujących dane pochodzące z list znajdujących się w różnych witrynach, jest to argument przemawiający za umieszczeniem tych witryn w roli witryn podrzędnych w tym samym zbiorze witryn.

Użytkownicy mogą również tworzyć różnego typu niestandardowe elementy witryny służące do wielokrotnego wykorzystania we wszystkich stronach w zbiorze witryn. Przykładowo, gdy stworzymy kolumnę witryny w witrynie najwyższego poziomu, kolumna ta będzie mogła zostać wykorzystana we wszystkich jej witrynach podrzędnych. Dzięki temu po jednokrotnym zdefiniowaniu cech charakterystycznych kolumny, takich jak formatowanie, sprawdzanie poprawności lub lista wyboru, taki typ kolumny może być wielokrotnie wykorzystywany przez wiele list w zbiorze witryn. Następnie wystarczy uaktualnić kolumnę witryny, aby modyfikacja ta wpłynęła na wszystkie listy, w których kolumna ta została zastosowana. Kolumny witryny zostały wykorzystane po raz pierwszy w wersji WSS 3.0 i zostaną omówione w sposób bardziej szczegółowy w Rozdziale 6 „Listy i typy zawartości".

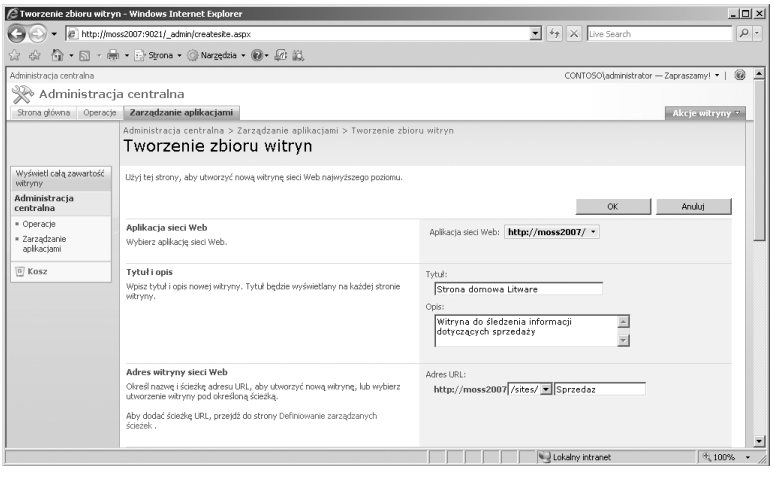
Tworzenie zbioru witryn

Nie można w pełni docenić prostoty usług WSS, dopóki nie stworzy się swojego pierwszego zbioru witryn. Teraz krok po kroku omówimy czynności zmierzające do tego celu. Czytelnik będzie pełnił rolę administratora farmy SharePoint, dlatego niezbędne są odpowiednie uprawnienia.

- W menu Start systemu Windows należy zlokalizować grupę Narzędzia Administracyjne, a następnie kliknąć opcję menu Administracja Centralna Programu SharePoint 3.0, aby

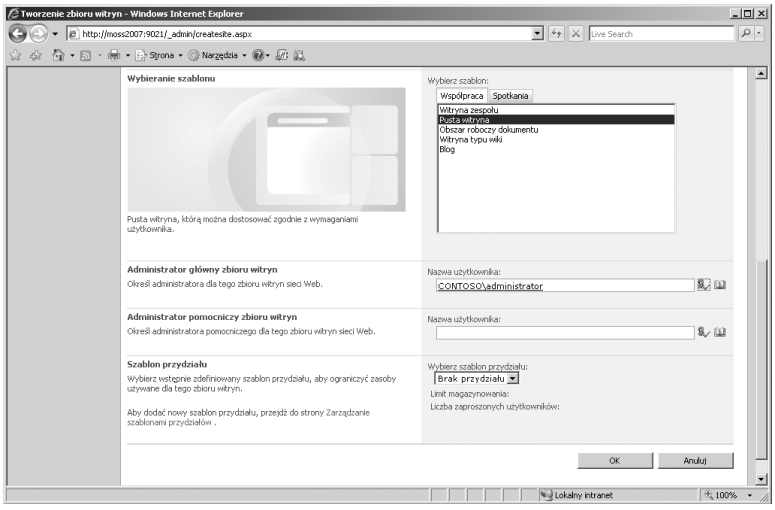
uruchomić aplikację Administracja Centralna Programu SharePoint. W kilku kolejnych krokach procedury aplikacja ta będzie wykorzystywana do tworzenia nowej witryny.

- W górnej części strony głównej aplikacji Administracja Centralna znajdują się trzy karty: Strona Główna, Operacje oraz Zarządzanie Aplikacjami. Należy wybrać kartę Zarządzanie Aplikacjami.
- Na stronie Zarządzanie Aplikacjami należy zlokalizować i kliknąć łącze zatytułowane Tworzenie Zbioru Witryn w sekcji Zarządzanie Witrynami Programu SharePoint. Dzięki temu zostaniemy przeniesieni do strony, na której można wpisać szczegółowe informacje, jakich usług WSS wymagają w celu stworzenia nowego zbioru witryn.
- Spójrzmy na Rysunek 1-4, który prezentuje górną część strony Tworzenie Zbioru Witryn. Rysunek ten ilustruje przykład wypełnienia pól Tytuł, Opis, Adres URL w procesie tworzenia nowego zbioru witryn.



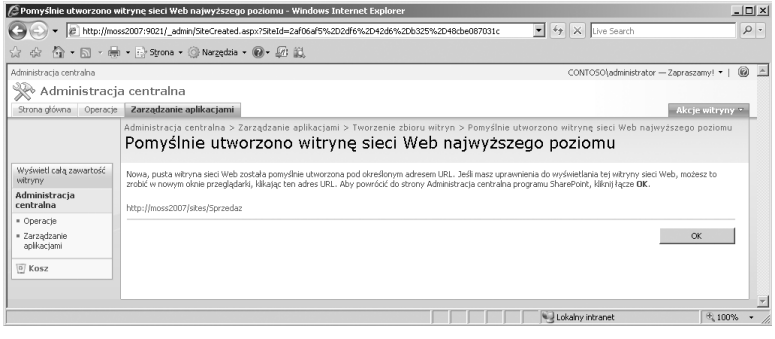
Rysunek 1-4 Podczas tworzenia nowego zbioru witryn trzeba określić nazwę, docelowy adres URL oraz informacje o koncie użytkownika dla jednego lub dwóch administratorów zbioru witryn.

- Rysunek 1-5 prezentuje dolną część strony Tworzenie Zbioru Witryn. Jak widać, strona ta umożliwia użytkownikowi wybranie szablonu witryny, który ma zostać wykorzystany do stworzenia witryny najwyższego poziomu. Można wybrać jeden z kilku różnych szablonów, takich jak np. Witryna Zespołu czy Pusta Witryna. W tym przykładzie należy wybrać szablon Pusta Witryna.



Rysunek 1-5 Usługi WSS umożliwiają wybranie szablonu witryny dla witryny najwyższego poziomu.

- Po wprowadzeniu wszystkich wymaganych informacji, można kliknąć przycisk OK, aby rozpocząć proces tworzenia. Usługi WSS stworzą nowy zbiór witryn zawierający witrynę najwyższego poziomu. Witryna najwyższego poziomu zostanie wygenerowana przy użyciu wybranego szablonu witryny. Po pomyślnym zakończeniu procesu tworzenia usługi WSS wyświetlą stronę zaprezentowaną na Rysunku 1-6.



Rysunek 1-6 Usługi WSS wyświetlają komunikat o pomyślnym utworzeniu nowego zbioru witryn.

- Nowy zbiór witryn WSS zawierający witrynę najwyższego poziomu wygenerowaną na podstawie szablonu witryny Pusta Witryna został pomyślnie utworzony. Teraz wystarczy kliknąć łącze np. *http://<nazwa_serwera>/sites/Sprzedaz*, aby przejść do strony głównej tej witryny najwyższego poziomu, jak pokazano na Rysunku 1-7. Witryna nie zawiera jeszcze żadnych list, ponieważ została utworzona na podstawie szablonu Pusta Witryna. Jednak można z łatwością dodać do niej nowe listy i biblioteki dokumentów.

Rysunek 1-7 Nowa witryna utworzona na podstawie szablonu Pusta Witryna

Dostosowywanie witryn

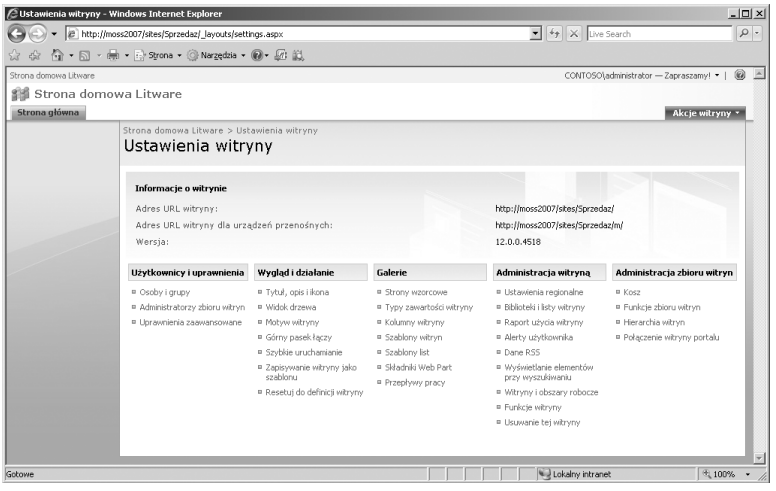
Po stworzeniu pierwszego zbioru witryn musimy zadecydować, w jaki sposób chcemy go skonfigurować i dostosować witrynę najwyższego poziomu. Usługi WSS dostarczają wiele opcji konfigurowania i dostosowywania witryn. Rozpoczniemy od zbadania różnych opcji dostępnych dla użytkowników, którym przypisane zostały uprawnienia administratora na poziomie zbioru witryn.

W prawym górnym rogu strony znajduje się menu Akcje Witryny. Zawiera ono polecenia, które umożliwiają tworzenie nowych elementów witryny, dostosowywanie aktualnej strony oraz nawigację do strony Ustawienia Witryny. Warto zauważyć, że to menu wspiera stosowanie zabezpieczeń. Oznacza to, że osoba o uprawnieniu administratora zobaczy wszystkie polecenia menu Akcje Witryny, podczas gdy mniej uprzywilejowany użytkownik zobaczy jedynie ograniczony zestaw poleceń. Odwiedzający tę stronę użytkownicy, którzy mają dostęp uprawniający wyłącznie do odczytu, w ogóle nie zobaczą menu Akcje Witryny.

Strona Ustawienia Witryny

Strona Ustawienia Witryny zawiera łącza do stron, które umożliwiają wykonywanie różnych zadań administracyjnych i dostosowujących. Standardowa strona Ustawienia Witryny dla witryny najwyższego poziomu została zaprezentowana na Rysunku 1-8.

Czytelnikom, którzy nie mieli wcześniej styczności z usługami WSS, zaleca się zapoznanie ze wszystkimi stronami administracyjnymi dostępnymi za pośrednictwem strony Ustawienia Witryny. Warto zauważyć, że strona Ustawienia Witryny dla witryny najwyższego poziomu zawiera sekcję Administracja witryną oraz dodatkową sekcję Administracja zbioru witryn, której nie zawierają strony Ustawienia Witryny dla witryn podrzędnych.



Rysunek 1-8 Każda witryna zawiera stronę Ustawienia Witryny, która udostępnia różne opcje konfiguracyjne i dostosowujące.

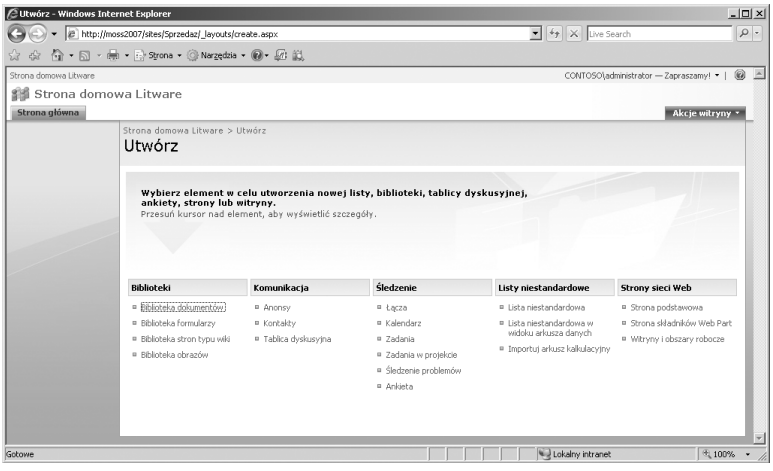
Jak widać, istnieje także sekcja Użytkownicy i Uprawnienia służąca do konfiguracji zabezpieczeń oraz sekcja Wygląd i Działanie służąca do modyfikowania wyglądu interfejsu użytkownika oraz opcji nawigacji. Istnieje również sekcja Galerie, która pozwala na przeglądanie i konfigurowanie różnych elementów aktualnej witryny, takich jak strony wzorcowe, kolumny witryny i typy zawartości. Wszystkie te galerie zostaną szczegółowo omówione w kilku kolejnych rozdziałach.

Strona Utwórz

Menu Akcje Witryny zawiera polecenie Utwórz, które prowadzi do strony Utwórz zaprezentowanej na Rysunku 1-9. Funkcjonalność, którą gwarantuje użytkownikom strona Utwórz, stanowi jeden z aspektów WSS o największych możliwościach. A to dlatego, że strona Utwórz umożliwia użytkownikom dostosowywanie aktualnej witryny poprzez tworzenie na żądanie nowych elementów witryny, takich jak listy, biblioteki dokumentów, strony składników Web Part oraz witryny podrzędne.

Strona Utwórz, zaprezentowana na Rysunku 1-9, zawiera łącza dla standardowych typów elementów, które są domyślnie dostępne w środowisku WSS. Te standardowe typy elementów zostały zaprojektowane tak, aby ułatwiać współpracę na poziomie zespołu. Zaliczają się do nich typy listowe służące do współpracowania przy użyciu elementów list m.in. anonsów, kontaktów, łącz, zdarzeń kalendarzowych, zadań i śledzenia problemów.

Standardowe funkcje współpracy w WSS zapewniają również wsparcie dla tworzenia różnego typu bibliotek dokumentów. Poza standardowymi typami bibliotek dokumentów istnieją również bardziej specjalistyczne typy, takie jak biblioteka obrazów czy stron typu wiki.



Rysunek 1-9 Strona Utwórz umożliwia użytkownikom tworzenie na żądanie nowych elementów witryny.

Strona Utwórz jest interesująca nie tylko z punktu widzenia użytkowników z uprawnieniami administratora, ale i programistów. Wyświetlany na stronie Utwórz zestaw typów elementów, które mogą zostać utworzone, może być swobodnie rozszerzany. W niniejszej książce zaprezentowanych zostanie wiele różnych technik dodawania niestandardowych typów elementów, np. własnych typów list i bibliotek dokumentów, zaprojektowanych z myślą o konkretnym zastosowaniu biznesowym. Dodatkową zaletą jest to, że niestandardowe typy elementów pojawiają się obok standardowych typów elementów WSS. Dzięki temu użytkownicy mogą w spójny sposób wzbogacać swoje witryny o różne, nowe elementy.

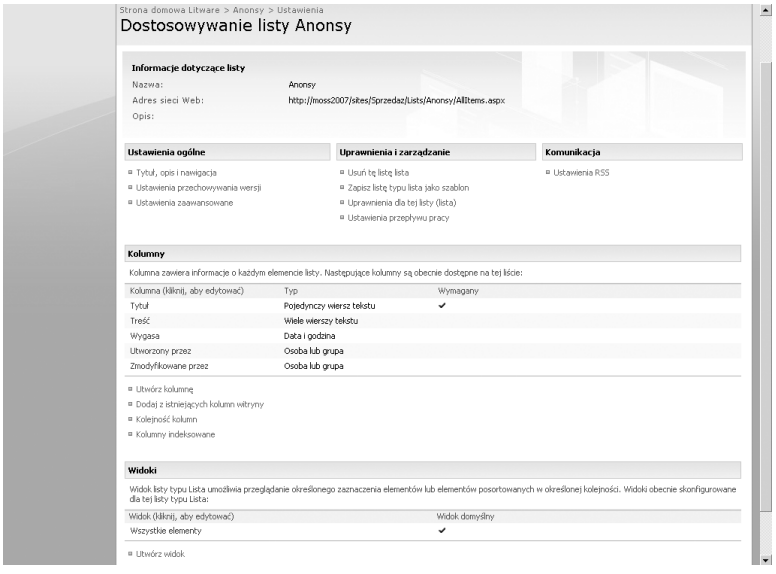
Tworzenie list i bibliotek dokumentów

Usługi WSS znacznie ułatwiają użytkownikom z odpowiednimi uprawnieniami dostosowywanie witryn poprzez tworzenie list i bibliotek dokumentów. Użytkownik po prostu klika łącze dla określonego typu listy na stronie Utwórz, a następnie zostaje przeniesiony na inną stronę i monitowany o nazwę oraz opis tworzonej listy. Należy mieć na uwadze, że nazwy list i bibliotek dokumentów w witrynie muszą być unikalne.

Po stworzeniu lista lub biblioteka dokumentów jest gotowa do użycia. Każda lista i biblioteka dokumentów oferuje zestaw stron, które umożliwiają użytkownikom dodawanie, wyświetlanie i usuwanie elementów oraz dokumentów. Aby móc aktywnie śledzić zagadnienia omawiane w kolejnej sekcji, należy teraz stworzyć w nowej, pustej witrynie przynajmniej jedną listę.

Dostosowywanie list przy użyciu polecenia Ustawienia Listy

Usługi WSS są bardzo elastyczne i pozwalają użytkownikom dostosowywać wiele aspektów istniejących list oraz bibliotek dokumentów. Każda lista zawiera pasek narzędziowy z menu Ustawienia. Menu Ustawienia zawiera polecenie Ustawienia Listy, które powoduje przeniesienie użytkownika do specjalnej strony, zaprezentowanej na Rysunku 1-10. Wyświetlana strona zawiera tytuł, który składa się z wyrażenia *Dostosowywanie listy* oraz nazwy określonej listy. Na przykład tytuł wygenerowany dla ustawień listy Anonsy brzmi *Dostosowywanie listy Anonsy*.



Rysunek 1-10 Polecenie Ustawienia Listy powoduje wyświetlenie strony umożliwiającej użytkownikom dostosowywanie listy.

Polecenie Ustawienia Listy umożliwia użytkownikom dostosowywanie właściwości listy, takich jak nazwa oraz wyświetlanie listy w postaci łącza w pasku Szybkie Uruchamianie. Przy użyciu tej strony ustawień użytkownik może skonfigurować inne, istotne aspekty dotyczące listy m.in. ustawień zabezpieczeń i przechowywania wersji. Strona Ustawienia Listy zwiększa również elastyczność dostosowywania list lub bibliotek dokumentów, dzięki możliwości dodawania, usuwania i modyfikowania kolumn i widoków.

Dostosowywanie i personalizacja witryn przy użyciu składników Web Part

Wsparcie dla technologii Web Part stanowi jeden z aspektów usług WSS stwarzających największe możliwości. Strona składników Web Part wzbogaca witrynę o bazujący na języku HTML, rozszerzalny interfejs użytkownika.

Zatrzymajmy się na chwilę i przyjrzyjmy się stronie składników Web Part z perspektywy osoby wykorzystującej technologię Web Part. Na stronie głównej administrator witryny może wybrać z menu Akcje Witryny polecenie Edytuj Stronę, aby wejść w tryb edytowania, w którym można dodawać, modyfikować i usuwać składniki Web Part. Modyfikacje składników Web Part dokonane w tym trybie nazywane są dostosowaniami, ponieważ są widoczne dla wszystkich użytkowników witryny.

Ponadto usługi WSS zapewniają użytkownikom o mniejszych uprawnieniach możliwość personalizowania stron składników Web Part poprzez dodawanie i modyfikowanie składników Web Part. Należy mieć na uwadze, że wykorzystywane w technologii WSS terminy *dostosowywanie* i *personalizacja* mają inne znaczenia. Dostosowywanie niesie za sobą zmiany widoczne dla wszystkich użytkowników. Natomiast w przypadku personalizacji modyfikacje są widoczne tylko dla użytkownika, który ich dokonana. Usługi WSS

potrafią w sposób inteligentny przechowywać wspólne dane dostosowań i prywatne dane personalizacji osobno w bazie danych zawartości. W kodzie mechanizm ten będzie nazywany składowaniem wspólnym (ang. shared storage) oraz składowaniem osobistym (ang. personal storage). Układ strony składników Web Part tworzą strefy składników Web Part. Składnik Web Part jest dodawany do strony składników Web Part poprzez umieszczenie go w strefie składników Web Part. Usługi WSS umożliwiają użytkownikom tworzenie nowych stron składników Web Part przy użyciu wbudowanego zestawu szablonów. W przeglądarce można użyć jednego z tych szablonów stron składników Web Part do stworzenia nowej strony o predefiniowanym układzie stref. Gdy użytkownik za pośrednictwem przeglądarki tworzy nową stronę składników Web Part, wynikowa strona składników Web Part musi być tworzona w kontekście biblioteki dokumentów.

WSS jako platforma programistyczna

W pewnym sensie usługi WSS działają na niekorzyść profesjonalnych twórców oprogramowania, ponieważ umożliwiają użytkownikom samodzielne tworzenie i dostosowywanie witryn. W ciągu zaledwie kilkunastu minut użytkownik może stworzyć witrynę WSS, dodać kilka list i bibliotek dokumentów czy dostosować wygląd witryny tak, aby odpowiadał on wymogom określonej sytuacji biznesowej. Stworzenie identycznego rozwiązania za pomocą technologii ASP.NET zajęłoby programistom kilka tygodni a nawet miesięcy.

Jednak patrząc z innej strony, usługi WSS otwierają przed profesjonalnymi programistami nowe, atrakcyjne możliwości. Podobnie jak każda inna struktura (*ang. framework*), domyślna funkcjonalność usług WSS wystarcza tylko do pewnego momentu. Później zwykle okazuje się, że konieczne jest stworzenie niestandardowych typów list i napisanie kodu dla niestandardowych składników WSS, takich jak składniki Web Part, obsługa zdarzeń i przepływy pracy. A ogromną zaletą usług WSS jako platformy programistycznej jest to, że zostały one od początku zaprojektowane z myślą o rozszerzaniu programistycznym. Platforma programistyczna WSS zawiera model obiektowy, strukturę Web Part Framework, usługi sieci Web i model inicjowania obsługi administracyjnej witryn, co czyni ją uniwersalnym szkieletem dla skalowalnych aplikacji pełniących rolę portali.

Dostosowywanie konta programowanie

Przed przystąpieniem do projektowania oprogramowania dla platformy WSS trzeba poznać granicę pomiędzy mechanizmem dostosowywania a programowaniem. Usługi WSS są bardzo elastyczne i zostały zaprojektowane tak, aby wspierać wysokopoziomowe dostosowania. Jak już wspominaliśmy, aby budować złożone witryny o bogatej funkcjonalności, wcale nie trzeba być programistą.

Zaawansowany użytkownik może tworzyć i dostosowywać listy i biblioteki dokumentów. Użytkownicy mogą również dostosowywać i personalizować wygląd stron przy użyciu składników Web Part. Doświadczeni użytkownicy mogą nawet tworzyć efektywny mechanizm stosowania znaków firmowych, dostosowując powiązane z witrynami strony wzorcowe oraz kaskadowe arkusze stylów przy użyciu programu SharePoint Designer.

Należy pamiętać, że usługi WSS zapisują wszystkie dostosowania witryny, modyfikując dane w bazie danych zawartości. Ma to miejsce za każdym razem, gdy tworzona jest nowa lista lub istniejąca lista jest dostosowywana poprzez dodawanie nowych kolumn i widoków.

Zasada ta ma zastosowanie również w przypadku wszelkiego typu dostosowań witryny dokonywanych przy użyciu programu Microsoft Office SharePoint Designer.

Fakt, że wszystkie dostosowania są zapisywane jako modyfikacje w bazie danych zawartości, stanowi zarówno zaletę, jak i wadę usług WSS. Zaletą jest duża elastyczność, jaką ten mechanizm gwarantuje użytkownikom i administratorom witryn w zakresie dokonywania dostosowań ad hoc. Mechanizm ten stanowi natomiast wadę z punktu widzenia profesjonalnych programistów, ponieważ utrudnia kontrolowanie wersji modyfikacji i replikowanie ich w wielu witrynach.

Wyobraźmy sobie standardowy projekt programistyczny ASP.NET, w którym wszystkie wykorzystywane pliki źródłowe znajdują się w jednym katalogu na naszej maszynie. Po zakończeniu fazy wstępnego projektowania i implementacji witryny można dodać wszystkie pliki źródłowe witryny do systemu zarządzania kodem źródłowym, takiego jak Microsoft Visual SourceSafe. Rozwiązanie takie umożliwia zastosowanie bardzo zdyscyplinowanego podejścia do problemu instalacji i aktualizacji witryny po przeniesieniu jej do środowiska produkcyjnego. Można także zdecydować się na przeniesienie zmian do środowiska publikowania, gdzie strony i kod witryny mogą zostać wnikliwie przetestowane przed umieszczeniem ich w środowisku produkcyjnym.

Jako programiści powinniśmy zadać sobie następujące pytania. W jaki sposób realizować będziemy zarządzanie kodem źródłowym zmian dostosowań? W jaki sposób dokonywać będziemy zmian dostosowań w definicjach list lub instancjach stron, a następnie jak będziemy przenosić te zmiany ze środowiska rozwoju do środowiska publikowania, a w końcu do środowiska produkcyjnego? W jaki sposób dokonywać będziemy modyfikacji dostosowań w witrynie, a następnie zastosujemy tę modyfikację w stu innych witrynach? Niestety, niełatwo jest odpowiedzieć na te pytania i zazwyczaj okazuje się, że możliwe rozwiązania są zbyt kosztowne w realizacji.

Na szczęście programiści mogą pracować na poziomie niższym niż infrastruktura dostosowań WSS. A dokładniej, programista może wykorzystywać niskopoziomowe pliki źródłowe do tworzenia wewnętrznych definicji obiektów, takich jak listy i szablony stron. Te niskopoziomowe pliki źródłowe nie są przechowywane w bazie danych zawartości, a znajdują się w systemie plików na serwerze frontonu sieci Web. Praca na tym poziomie jest bardziej skomplikowana i wymaga bardziej zaawansowanych umiejętności. Jednocześnie jednak pozwala na scentralizowanie zarządzania kodem źródłowym i stanowi bardziej metodologiczne rozwiązanie, zapewniające mechanizm podpisywania podczas przenoszenia stron i kodu ze środowiska programistycznego do środowiska publikowania, a stamtąd do środowiska produkcyjnego. Takie podejście zwiększa również możliwości zarządzania przechowywaniem wersji i ponownym wykorzystywaniem kodu w obrębie wielu witryn, aplikacji sieci Web oraz farm.

W pozostałej części tej książki wprowadzone zostało rozróżnienie pomiędzy dostosowywaniem a programowaniem zgodne z następującymi kryteriami. Dostosowania WSS stanowią modyfikacje witryny osiągane poprzez dokonywanie zmian w bazie danych zawartości, z reguły za pośrednictwem przeglądarki sieci Web lub programu SharePoint Designer. Dostosowanie witryny nigdy nie odnosi się bezpośrednio do serwera frontonu sieci Web. Natomiast programowanie WSS wiąże się z przetwarzaniem plików, które muszą zostać umieszczone w systemie plików serwera frontonu sieci Web. Programowanie WSS polega na tworzeniu szablonów stron i definicji list, jak również komponentów instalowanych w skompilowanych zestawach, takich jak niestandardowe składniki Web Part, programy

obsługi zdarzeń i szablony przepływów pracy. Programowanie WSS na tym poziomie nazywane jest również *rozwijaniem składników obsługi administracyjnej*.

Należy pamiętać, że programowanie wymaga pewnego stopnia kontroli nad serwerem frontonu sieci Web, ponieważ wiąże się w umieszczeniem na nim plików, takich jak szablony stron i zestawy. Programista pracujący w środowisku WSS, w którym nie ma uprawnień do dokonywania żadnych zmian na serwerach frontonu sieci Web, nie będzie mógł zainstalować owoców swojej pracy. W takim środowisku można zastosować jedynie techniki dostosowywania, które są realizowane poprzez dokonywanie zmian w bazie danych zawartości.

Chociaż mechanizm dostosowywania zapewnia użytkownikom sporą elastyczność, programowanie stwarza dużo większe możliwości. Stanowi ono dużo lepszą metodę w przypadku tworzenia rozwiązań WSS wielokrotnego użytku, jak również najlepszą metodę optymalizowania czasów odpowiedzi i przepustowości w witrynach o wysokim natężeniu ruchu. Programowanie składników obsługi administracyjnej przy użyciu usług WSS stanowi podstawowy temat niniejszej książki, natomiast omówienie dostosowań i związanych z nimi narzędzi, takich jak program SharePoint Designer, to temat drugorzędny.

Możliwości rozwoju

Programista może rozszerzać usługi WSS w wielu różnych obszarach. W kolejnych dwóch rozdziałach omówimy metody tworzenia niestandardowych stron aplikacji i szablonów stron w celu zwiększenia funkcjonalności rozwiązań biznesowych. Jak będzie się można przekonać, pewne typy stron w witrynie WSS pozwalają na pisanie schowanego bezpośrednio za nimi kodu zarządzanego, a inne nie. Jednak nawet, gdy typy stron nie wspierają kodu w sposób bezpośredni, nadal można pisać kod w składnikach Web Part i formantach użytkownika ASP.NET, a następnie wykorzystywać te komponenty na stronach.

Rozszerzenia wprowadzone w wersji WSS 3.0 wiążą się z dodaniem integracji ze stronami wzorcowymi ASP.NET i kaskadowymi arkuszami stylów (CSS). Integracja ta pozwala lepiej kontrolować mechanizm stosowania w witrynie znaków firmowych, które będą widoczne dla klientów oraz zarządu. Zapewniona została również ścisła integracja ze strukturą dostawcy nawigacji ASP.NET, dzięki której można tworzyć wszelkiego typu niestandardowe struktury nawigacyjne, znane ze standardowych aplikacji ASP.NET 2.0.

Możliwe jest również rozwijanie różnego typu niestandardowych komponentów wykorzystywanych w witrynach WSS. Do komponentów tych zaliczają się standardowe formanty serwerów ASP.NET, składniki Web Part, programy obsługi zdarzeń, niestandardowe typy pól i niestandardowe zasady. Przyczyny i metody rozwijania tego typu komponentów będą stopniowo omawiane w dalszej części książki.

Programista może również kontrolować sposób przechowywania zawartości w niestandardowej witrynie. Usługi WSS pozwalają na definiowanie niestandardowych typów na potrzeby list i bibliotek dokumentów. Dzięki temu można kontrolować, jakie kolumny i jakie widoki są dostępne dla użytkowników.

Usługi WSS 3.0 wprowadziły pewną użyteczną innowację, zwaną *kolumną witryny*. Kolumna witryny stanowi definicję kolumny wielokrotnego użytku, która może zostać zastosowana w wielu listach. Kolumna witryny definiuje nazwę kolumny, typ pola i inne cechy charakterystyczne, takie jak wartość domyślna, formatowanie i sprawdzanie poprawności. Po zdefiniowaniu kolumny witryny można wykorzystywać ją podczas definiowania

schematu niestandardowych list i bibliotek dokumentów. Płynie z tego oczywista korzyść polegająca na tym, że wystarczy zaktualizować kolumnę witryny w jednym miejscu, a zmiana ta zostanie odzwierciedlona we wszystkich listach, w których wykorzystana została ta kolumna witryny.

W usługach WSS 3.0 wprowadzono również inną, funkcjonalną innowację związaną z przechowywaniem zawartości. Nosi ona nazwę *typ zawartości*. Typ zawartości stanowi elastyczną definicję typu WSS wielokrotnego użytku, który opisuje kolumnę i zachowanie elementów list lub dokumentów w bibliotekach dokumentów. Przykładowo, można stworzyć na potrzeby dokumentów zawierających prezentacje typ zawartości z unikalnym zestawem kolumn, programem obsługi zdarzeń i własnym szablonem dokumentu. Następnie można stworzyć dla dokumentów ofert drugi typ zawartości z innym zestawem kolumn, przepływem pracy i innym szablonem. W kolejnym kroku można zdefiniować nową bibliotekę dokumentów i skonfigurować ją tak, aby wspierała ona oba typy zawartości. Wprowadzenie w wersji WSS 3.0 typów zawartości jest bardzo znaczącą zmianą, ponieważ umożliwia obsługę różnych typów zawartości w tej samej liście lub bibliotece dokumentów, co w wersji WSS 2.0 było niemożliwe.

Kolejny obszar, który może być programistycznie rozwijany, wiąże się z wykorzystaniem nowego formatu plików Office – Open XML. Ta nowa technologia wprowadzona wraz z wersją Microsoft Office 2007 pozwala na generowanie i/lub przetwarzanie dokumentów Microsoft Office Word oraz Microsoft Office Excel za pośrednictwem kodu po stronie serwera umieszczonego w niestandardowych komponentach, takich jak programy obsługi zdarzeń czy przepływy pracy. Zaletą tej technologii jest to, że format plików Office Open XML eliminuje konieczność instalowania i uruchamiania na serwerze wersji stacjonarnej aplikacji Microsoft Office, takiej jak Office Word czy Office Excel. Wszystko można osiągnąć poprzez programowanie przy użyciu serwerowych bibliotek, co zapewnia wysoką skalowalność i stabilność.

Jednym z najciekawszych obszarów WSS 3.0 podlegających rozszerzaniu są przepływy pracy. Usługi WSS wykorzystują możliwości nowej platformy Microsoft Windows Workflow Foundation, która stanowi część .NET Framework 3.0 i wzbogacają platformę o dodatkowe elementy, które umożliwiają dołączanie logiki biznesowej do elementów list i dokumentów w witrynie WSS. Usługi WSS rozszerzają podstawowy model Windows Workflow Foundation, powiązując listy zadań i historii z każdym przepływem pracy. To powoduje odwzorowanie pewnego zakresu odpowiedzialności w przepływach pracy, które są z natury zorientowane na symulacje rzeczywistych procesów biznesowych realizowanych przez człowieka, takich jak recenzowanie lub zatwierdzanie dokumentów.

Zarówno usługi WSS, jak i program MOSS zawierają wbudowane przepływy pracy, które są domyślnie instalowane i gotowe do użycia. Usługi WSS zawierają prosty przepływ pracy obiegu wspierający takie aspekty jak filtrowanie i zarządzenie treścią oraz jej zatwierdzanie. Program MOSS wspiera bardziej złożone przepływy pracy wykorzystywane w takich operacjach, jak proces zatwierdzania realizowany na poziomie zarządzania zawartością sieci Web.

Dodawanie niestandardowych przepływów pracy to jeden z typowych mechanizmów wykorzystywanych przez programistów tworzących rozwiązania biznesowe przy użyciu usług WSS oraz programu MOSS. Poza standardowym wsparciem dla rozszerzeń Visual Studio Extensions for Windows Workflow Foundation, zespół Office stworzył specjalnie z myślą o usługach WSS pakiet Workflow Software Development Kit (SDK) wraz z pakietem pomocniczym Workflow Developer Starter Kit zawierającym szablony projektów Visual Studio do tworzenia niestandardowych przepływów pracy dla witryn WSS.

Zabezpieczenia i członkostwo w witrynach to kolejne dwa obszary, które pozwalają programistom na rozszerzanie standardowej infrastruktury WSS. Poprzednie wersje usług WSS były ściśle powiązane z kontami Windows i Microsoft Active Directory, co utrudniało wykorzystanie technologii SharePoint w witrynach udostępnianych w Internecie. W wersji WSS 3.0 jest inaczej. Nowa struktura uwierzytelniania użytkowników została zaprojektowana ponownie na bazie infrastruktury dostawcy uwierzytelniania wprowadzonej w ASP.NET 2.0.

Gdy nie chce się utrzymywać kont dla użytkowników WSS w katalogu Active Directory, wystarczy po prostu zbudować lub nabyć dostawcę uwierzytelniania ASP.NET, który został zaprojektowany tak, aby przechowywać i zarządzać kontami użytkowników w alternatywnym repozytorium tożsamości, takim jak np. baza danych SQL Server. Dzięki takiemu rozwiązaniu architektonicznemu, usługi WSS 3.0 zyskały dużo szersze możliwości zastosowania w roli platformy służącej do budowania witryn internetowych.

Technologia ASP.NET 2.0 zawiera dostawcę uwierzytelniania, dzięki któremu można wykorzystać konta użytkowników przechowywane w bazie danych SQL Server i uwierzytelnianie wykorzystujące formularze. Można skonfigurować ten typ dostawcy tak, aby dostarczał usługi uwierzytelniania na potrzeby witryny WSS 3.0. Już niewielka ilość pracy wystarczy, aby umieścić w Internecie witrynę WSS, która umożliwia nieznanym użytkownikom rejestrowanie się w roli członków. Technologia ASP.NET 2.0 oferuje wygodne w użyciu wsparcie dla tworzenia i przechowywania kont użytkowników, a nawet umożliwia użytkownikom modyfikowanie swoich haseł. Wykorzystując uwierzytelnianie przy użyciu formularzy w WSS, można zastosować te same techniki zarządzania kontami użytkowników co w rozwiązaniach ASP.NET 2.0.

Wprowadzenie do funkcji

Chociaż niestandardowemu rozszerzaniu podlega wiele obszarów usług WSS, warto rozpocząć naukę od *funkcji*. Funkcje stanowią innowację wprowadzoną w wersji WSS 3.0 specjalnie z myślą o programistach. Funkcje dostarczają mechanizm definiowania elementów witryny i dodawania ich do docelowej witryny lub zbioru witryn przy użyciu procesu nazywanego *aktywowaniem funkcji*. Do typów elementów, które mogą być definiowane za pomocą funkcji, zaliczają się: polecenia menu, łącza, szablony stron, wystąpienia stron, definicje list, wystąpienia list, programy obsługi zdarzeń i przepływy pracy.

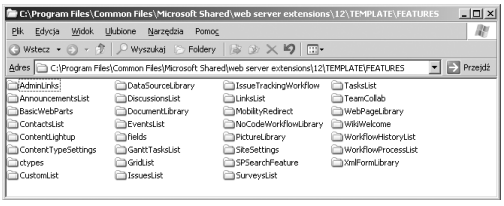
Pod względem fizycznym funkcja składa się z podkatalogu stworzonego w specjalnym folderze WSS znajdującym się w systemie plików na każdym serwerze frontonu sieci Web. Katalog odpowiadający funkcji składa się z jednego lub więcej bazujących na technologii XML plików, które zawierają kod w języku Collaborative Application Markup Language (CAML). Zgodnie z narzuconą konwencją, każdy katalog funkcji zawiera plik manifestu o nazwie feature.xml, który definiuje wysokopoziomowe atrybuty funkcji, takie jak unikalny identyfikator i przyjazny dla użytkownika tytuł.

Poza plikiem feature.xml funkcję opisuje zazwyczaj jeden lub więcej dodatkowych plików XML (np. elements.xml), które definiują jej poszczególne elementy. Katalog danej funkcji może także zawierać kilka innych typów plików zawierających takie elementy jak definicje list czy szablony stron, jak również inne rodzaje zasobów, takie jak pliki graficzne, kaskadowe arkusze stylów czy pliki JavaScript.

Jedną z metod pomocnych w zrozumieniu funkcji, jest zapoznanie się ze standardowym zestawem funkcji dostępnych w podstawowej instalacji WSS. Wygląd przykładowego

katalogu FEATURES tuż po instalacji usług WSS został zaprezentowany na Rysunku 1-11. Jak widać, każda funkcja ma swój własny katalog. Warto także zauważyć, że strona Funkcje Witryny będzie wyglądała zupełnie inaczej, gdy zainstalowany jest program MOSS, ponieważ dostępnych będzie ponad 100 funkcji.

W końcowej części rozdziału będzie można dowiedzieć się, jak stworzyć i zainstalować swoją pierwszą niestandardową funkcję. Po stworzeniu funkcji trzeba zainstalować ją w usługach WSS na poziomie farmy. Po zainstalowaniu funkcja może zostać aktywowana w kontekście witryny lub zbioru witryn przy użyciu strony administracyjnej dostępnej za pośrednictwem strony Ustawienia Witryny.



Rysunek 1-11 Wiele szablonowych elementów WSS, jak np. listy współpracy, jest zaimplementowanych w formie funkcji.

W dalszej części książki wprowadzane będą nowe funkcje służące do rozwiązywania określonych problemów, takich jak stosowanie w witrynie znaków firmowych przy użyciu niestandardowej strony wzorcowej i kaskadowych arkuszy stylów. Inne funkcje pozwolą zademonstrować tworzenie nowych instancji stron i udostępnianie w witrynie komponentów, takich jak składniki Web Part bądź niestandardowe przepływy pracy.

Niestandardowa funkcja umożliwia programiście definiowanie jednego lub kilku elementów, które mogą być aktywowane w kontekście witryny lub zbioru witryn. Silnik WSS pozwala również na definiowanie całego projektu witryny poprzez stworzenie niestandardowej *definicji witryny*. Mechanizm rozwijania niestandardowych definicji witryn daje programistom kontrolę nad prawie każdym aspektem nowej witryny, m.in. nad zastosowanymi znakami firmowymi, wstępnym zestawem list i wykorzystywanych funkcji. Rozdział 9 „Rozwiązania i instalacja" poświęcony jest pracy z definicjami witryn.

Programowanie przy użyciu modelu obiektowego WSS

Innym istotnym aspektem rozszerzania usług WSS jest programowanie przy użyciu modelu obiektowego WSS. Podstawowe typy dostarczane przez model programistyczny WSS są udostępniane poprzez standardowy zestaw WSS o nazwie Microsoft.SharePoint.dll.

Przyjrzyjmy się prostemu przykładowi. Wyobraźmy sobie, że stworzyliśmy aplikację konsolową i że dodaliśmy do niej odwołanie do Microsoft.SharePoint.dll. Model obiektowy WSS udostępnia klasę SPSite, która służy jako punkt wejścia do modelu obiektowego WSS na poziomie zbioru witryn. Każda witryna w zbiorze witryn jest reprezentowana jako obiekt SPWeb. Każda lista w witrynie jest reprezentowana jako obiekt SPList. Oto prosty przykład wykorzystujący model obiektowy WSS do uzyskania dostępu do witryny najwyższego poziomu w docelowym zbiorze witryn i sprawdzenia wszystkich znajdujących się w niej list.

```
using Microsoft.SharePoint;

namespace Hello_WSS_OM {
  class Program {
    static void Main() {
      string sitePath = "http://litwareinc.com";
      // wejście do modelu obiektowego poprzez zbiór witryn.
      SPSite siteCollection = new SPSite(sitePath);
      // pobranie referencji do witryny najwyższego poziomu.
      SPWeb site = siteCollection.RootWeb;
      // enumeracja po listach w witrynie
      foreach (SPList list in site.Lists) {
        Console.WriteLine(list.Title);
      }
      // sprzątanie zasobów poprzez wywołanie funkcji Dispose.
      site.Dispose();
      siteCollection.Dispose();
    }
  }
}
```

Na końcu tego przykładowego kodu powinny pojawić się dwa wywołania metody Dispose. Część typów obiektowych w modelu obiektowym WSS (m.in. SPSite i SPWeb) wykorzystuje niezarządzane zasoby i musi zostać szybko zutylizowana. W przeciwnym wypadku, gdy w zakresie zarządzania pamięcią polegać będziemy jedynie na mechanizmie Garbage Collector platformy .NET, kod może stać się źródłem problemów, wykorzystując więcej pamięci niż to konieczne. Ogólna zasada jest taka, że gdy tworzymy utylizowalny obiekt, odpowiadamy również za wywołanie na nim metody Dispose. Jednakże referencje do obiektów pozyskane za pośrednictwem obiektu SPContext aplikacji sieci Web WSS nie powinny być utylizowane.

Tworzenie pierwszej funkcji

Nie ma lepszego zakończenia dla pierwszego rozdziału niż skok na głęboką wodę i stworzenie od podstaw prostej funkcji Hello World. Ćwiczenie to przeprowadzi nas przez podstawowe aspekty tworzenia, instalowania i testowania funkcji. Aby dodatkowo uatrakcyjnić ćwiczenie, na zakończenie dodamy programy obsługi zdarzeń wywoływane w związku z aktywowaniem lub dezaktywowaniem funkcji. Kod znajdujący się w programach obsługi zdarzeń wykorzystywać będzie model obiektowy WSS do zmiany pewnych cech charakterystycznych docelowej witryny.

W przykładzie wykorzystamy Microsoft Visual Studio 2005 do stworzenia nowego projektu programistycznego dla funkcji. Visual Studio 2005 zapewni odpowiednie formatowanie znaczników XML i ASP.NET oraz poprzez mechanizm IntelliSense zwiększy wygodę pracy z plikami XML koniecznymi do zdefiniowania funkcji.

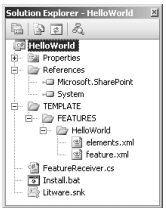
Rozpocznijmy od stworzenia nowego projektu Class Library o nazwie **HelloWorld**. W tym przykładzie stworzymy projekt C#, ale nic nie stoi na przeszkodzie, aby czytelnik stworzył projekt Visual Basic .NET. Na końcu dodamy kod, który będzie kompilowany do wyjściowej biblioteki DLL wykorzystywanej przez programy obsługi zdarzeń danej funkcji. Jednak wcześniej stworzymy plik feature.xml.

Przed rozpoczęciem procesu tworzenia pliku feature.xml należy rozważyć fakt, że funkcja musi zostać umieszczona w dedykowanym katalogu w folderze systemowym WSS o nazwie FEATURES. Katalog FEATURES znajduje się w innym katalogu systemowym WSS o nazwie TEMPLATE.

```
c:\Program Files\Common Files\Microsoft Shared\web server extensions\12\TEMPLATE\
FEATURES
```

Biorąc pod uwagę wymagania stawiane przez instalację funkcji, dobrym rozwiązaniem może okazać się stworzenie jednakowej hierarchii folderów w projekcie Visual Studio wykorzystywanym do rozwijania funkcji WSS. To ułatwi kopiowanie plików funkcji do odpowiednich lokalizacji i testowanie ich podczas prac programistycznych.

Na początku należy dodać folder o nazwie TEMPLATE do katalogu głównego aktualnego projektu. Po stworzeniu katalogu TEMPLATE należy stworzyć podkatalog o nazwie FEATURES. Na zakończenie należy stworzyć w katalogu FEATURES kolejny katalog o nazwie takiej samej jak nazwa projektu funkcji. W tym przypadku nazwa tego katalogu to HelloWorld, jak pokazano na Rysunku 1-12.



Rysunek 1-12 Projekt Visual Studio służący do rozwijania funkcji WSS.

Następnie należy stworzyć plik XML o nazwie feature.xml wewnątrz katalogu HelloWorld. Do tego katalogu dodamy informacje XML, które definiują wysokopoziomowe atrybuty funkcji. Należy wstawić następujący kod XML do pliku feature.xml, aby dodać element Feature najwyższego poziomu wraz z atrybutami opisującymi samą funkcję.

```
<Feature
  Id="B2CB42E2-4F0A-4380-AABA-1EF9CD526F20"
  Title="Rozdział 1: Funkcja Hello World "
  Description="To jest pierwsza niestandardowa funkcja "
  Scope="Web"
  Hidden="FALSE"
  ImageUrl="menuprofile.gif"
  xmlns="http://schemas.microsoft.com/sharepoint/">

  <ElementManifests>
    <ElementManifest Location="elements.xml" />
  </ElementManifests>

</Feature>
```

Jak widać, funkcja jest definiowana przy użyciu elementu Feature zawierającego atrybuty, takie jak Id (Unikalny identyfikator), Title (Tytuł), Description (Opis), Version (Wersja), Scope (Zakres), Hidden (Ukryta) i ImageUrl (Adres URL obrazu). Trzeba stworzyć nowy identyfikator GUID dla atrybutu Id, aby funkcja mogła być unikalnie identyfikowana.

Atrybuty funkcji Title oraz Description powinny być przyjaznym dla użytkownika opisem tekstowym nowej funkcji. Atrybuty te będą widoczne dla użytkowników za pośrednictwem stron administracyjnych WSS służących do aktywowania i dezaktywowania funkcji.

Atrybut Scope definiuje kontekst, w którym funkcja może być aktywowana i dezaktywowana. Tworzona funkcja ma zakres o wartości Web, co oznacza, że może ona zostać aktywowana i dezaktywowana w kontekście witryny. Gdybyśmy przypisali atrybutowi Scope wartość Site, funkcja byłaby aktywowana i dezaktywowana w zakresie zbioru witryn. Dwie inne możliwe wartości, które można nadać atrybutowi zakresu, to zakres WebApplication i Farm.

Jak widać, w powyższym kodzie atrybut Hidden zawiera wartość FALSE. Oznacza to, że po zainstalowaniu w farmie funkcja będzie widoczna dla użytkowników, którzy mogą chcieć ją aktywować. Można także stworzyć funkcję z atrybutem Hidden o wartości TRUE. Skutkiem będzie ukrycie funkcji na liście dostępnych funkcji prezentowanej użytkownikom. Ukryte funkcje muszą być aktywowane za pośrednictwem wiersza polecenia, kodu niestandardowego lub poprzez zależności z aktywacją innej funkcji. Zależności aktywacji zostaną omówione w sposób bardziej szczegółowy w Rozdziale 9.

Można również zauważyć, że atrybut ImageUrl ma wartość, która wskazuje jeden z graficznych obrazów stanowiących część podstawowej instalacji WSS. Obraz ten zostanie wyświetlony na stronach interfejsu użytkownika obok nazwy funkcji.

Ostatnią część zaprezentowanego pliku feature.xml stanowi element ElementManifests. Element ten zawiera wewnętrzne elementy ElementManifest, które odwołują się do innych plików XML, w których definiowane są elementy składające się na funkcję. W naszym przypadku istnieje tylko jeden element ElementManifest, który wykorzystuje atrybut Location do wskazania pliku o nazwie element.xml.

Włączanie mechanizmu IntelliSense w Visual Studio

Wewnątrz katalogu TEMPLATE znajduje się katalog o nazwie XML, który zawiera różne pliki XML Schema, w tym jeden o nazwie wss.xsd. Po powiązaniu pliku XML Schema z plikami funkcji, takimi jak feature.xml i elements.xml, Visual Studio będzie oferowało wsparcie IntelliSense podczas ich edycji, co znacznie ułatwi tworzenie niestandardowych funkcji. Można również skopiować pliki XSD do katalogu C:\Program Files\Microsoft Visual Studio 8\Xml\Schemas\.

Teraz nadszedł czas, aby stworzyć plik element.xml i zdefiniować pojedynczy element CustomAction, który zostanie wykorzystany do dodania prostego polecenia do menu Akcje Witryny. Należy dodać do pliku elements.xml następujący kod XML, który definiuje element CustomAction.

```
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
   <CustomAction
    Id="SiteActionsToolbar"
    GroupId="SiteActions"
    Location="Microsoft.SharePoint.StandardMenu"
    Sequence="100"
    Title="Hello World"
    Description="Wartość menu dodana przez użytkownika za pomocą funkcji"
```

```
    ImageUrl="_layouts/images/menuprofile.gif" >
      <UrlAction Url="http://msdn.microsoft.com"/>

  </CustomAction>
</Elements>
```

Ten element CustomAction został zaprojektowany tak, aby spowodować dodanie polecenia do menu Akcje Witryny. Dostarcza on przyjazny dla użytkownika tytuł (Title) oraz opis (Description) oraz adres URL, który będzie wykorzystywany do przekierowania użytkownika, gdy wybierze on to polecenie menu. Przedstawiona przykładowa funkcja prezentuje zaledwie niewielki zakres dostępnych możliwości, niemniej stanowi prosty wstęp do podróży po procesie instalacji i testowania funkcji.

Teraz, gdy już stworzyliśmy pliki feature.xml oraz elements.xml definiujące funkcję HelloWorld, pozostało nam jeszcze do wykonania kilka czynności związanych z jej zainstalowaniem i przetestowaniem. Po pierwsze, należy skopiować katalog funkcji HelloWorld do systemowego katalogu WSS o nazwie FEATURES. Po drugie, należy wywołać komendę STSADM.EXE, w celu zainstalowania funkcji w usługach WSS. A w końcu należy aktywować funkcję w kontekście witryny WSS. Dwa pierwsze kroki można zautomatyzować, tworząc plik wsadowy o nazwie install.bat w katalogu głównym projektu HelloWorld i dodając następujące instrukcje wiersza polecenia.

```
REM – Pamiętaj aby usunąć znaki końca wiersza z dwóch pierwszych linii
@SET TEMPLATEDIR="c:\program files\common files\microsoft shared\
                  web server extensions\12\Template"
@SET STSADM="c:\program files\common files\microsoft shared\
             web server extensions\12\bin\stsadm"


Echo Kopiowanie plików
xcopy /e /y TEMPLATE\* %TEMPLATEDIR%

Echo Instalowanie funkcji
%STSADM% -o InstallFeature -filename  HelloWorld\feature.xml -force

Echo Ponowne uruchomienie procesu roboczego IIS
IISRESET
```

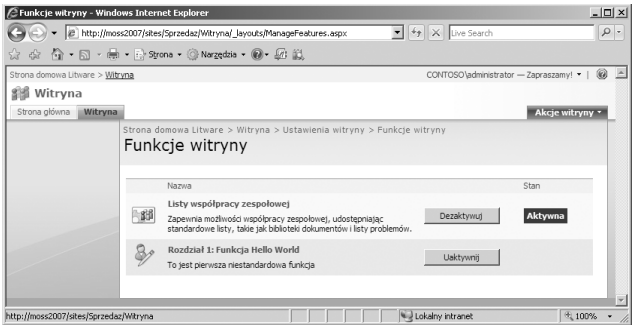
Zasadniczo można również zautomatyzować ostatnią czynność aktywowania funkcji w określonej witrynie, uruchamiając operację ActivateFeature za pomocą narzędzia STSADM. Jednak w prezentowanym przykładzie technika ta nie została wykorzystana, aby czytelnik samodzielnie przeszedł przez proces aktywowania funkcji podobnie jak to czynią inni użytkownicy – za pośrednictwem interfejsu użytkownika WSS.

Po dodaniu pliku install.bat można skonfigurować Visual Studio tak, aby plik ten był uruchamiany za każdym razem, gdy budowany jest projekt HelloWorld. W tym celu należy przejść do karty Build Events we właściwościach projektu i dodać następujące instrukcje w polu Post-build Event Command Line.

```
cd $(ProjectDir)
Install.bat
```

Pierwsza linia *cd $(ProjectDir)* jest konieczna do zmiany aktualnego katalogu na katalog projektu. W drugiej linii uruchamiany jest plik wsadowy, który kopiuje pliki funkcji do odpowiednich lokalizacji i instaluje funkcję przy użyciu operacji InstallFeature, wykorzystując narzędzie wiersza polecenia STSADM.EXE.

Po prawidłowym zainstalowaniu funkcja może zostać aktywowana w kontekście witryny. W witrynie najwyższego poziomu w stworzonym we wcześniejszej części niniejszego rozdziału zbiorze witryn należy przejść do strony Ustawienia Witryny. W sekcji Administracja Witryną należy kliknąć łącze zatytułowane Funkcje Witryny. To powinno spowodować pojawienie się strony podobnej do zaprezentowanej na Rysunku 1-13. Warto mieć na uwadze, że w farmie, w której zainstalowany został program MOSS, zaprezentowanych zostanie dużo więcej funkcji niż w farmie z samymi usługami WSS.



Rysunek 1-13 Po zainstalowaniu funkcja może być aktywowana i dezaktywowana przez użytkowników.

Strona Funkcje Witryny powinna zawierać funkcję HelloWorld. Po jej odnalezieniu można kliknąć przycisk Uaktywnij, aby aktywować funkcję. Po wykonaniu tej czynności w rozwijanym menu Akcje Witryny powinien pojawić się niestandardowy element, co zaprezentowano na Rysunku 1-14. Po wybraniu tego niestandardowego elementu menu, użytkownik zostanie przekierowany pod adres URL, który został zdefiniowany w atrybucie Url elementu UrlAction w pliku elements.xml.

Rysunek 1-14 Element CustomAction może być wykorzystywany do dodawania niestandardowych poleceń do menu akcji witryny.

Po pomyślnym aktywowaniu funkcji i przetestowaniu niestandardowego polecenia menu, warto dodatkowo poeksperymentować, powracając do strony Funkcje Witryny i dezaktywując funkcję. Po dezaktywowaniu funkcji HelloWorld powinno się okazać, że niestandardowy element został usunięty z menu Akcje Witryny.

Poznaliśmy fundamentalną zasadę dotyczącą funkcji. Programiści tworzą różnego typu elementy witryn, które mogą być dodawane lub usuwane z witryny za pośrednictwem procesu aktywacji i dezaktywacji.

Dodawanie do funkcji procedury obsługi zdarzenia

Teraz nadszedł czas, aby dodatkowo rozbudować przykład funkcji HelloWorld, dodając procedury obsługi zdarzeń i wykorzystując w kodzie model obiektowy WSS. Należy rozpocząć od dodania do projektu referencji do biblioteki Microsoft.SharePoint.dll. Następnie należy zlokalizować plik źródłowy nazwany Class1.cs i zmienić jego nazwę na FeatureReceiver.cs. Później należy dodać następujący kod.

```
using System;
using Microsoft.SharePoint;

namespace HelloWorld{
  public class FeatureReceiver : SPFeatureReceiver {
    public override void FeatureInstalled(
                          SPFeatureReceiverProperties properties){}
    public override void FeatureUninstalling(
                          SPFeatureReceiverProperties properties) { }

    public override void FeatureActivated(
                          SPFeatureReceiverProperties properties){
      SPWeb site = (SPWeb)properties.Feature.Parent;
      // zapisujemy oryginalną wartość parametru Title danej witryny,
      // używając zbioru właściwości SPWeb
      site.Properties["OriginalTitle"] = site.Title;
      site.Properties.Update();
      // zmieniamy tytuł witryny
      site.Title = "Hello World";
      site.Update();
    }

    public override void FeatureDeactivating(
                          SPFeatureReceiverProperties properties) {
      // przywracamy oryginalny tytuł danej witryny
      SPWeb site = (SPWeb)properties.Feature.Parent;
      site.Title = site.Properties["OriginalTitle"];
      site.Update();
    }
  }
}
```

Pierwszą rzeczą, na którą należy zwrócić uwagę, jest sposób tworzenia procedury obsługi zdarzenia, która jest uruchamiana, gdy funkcja jest aktywowana lub dezaktywowana. Sposób ten polega na stworzeniu klasy, która dziedziczy z klasy SPFeatureReceiver. Jak widać, można obsługiwać zdarzenia poprzez nadpisywanie wirtualnych metod klasy bazowej, takich jak FeatureActivated i FeatureDeactivating. Istnieją także dwie dodatkowe metody obsługi zdarzeń, które będą uruchamiane podczas instalowania i odinstalowywania funkcji, ale w tym podstawowym przykładzie nie będą one wykorzystywane.

Metoda FeatureActivated została napisana tak, aby modyfikować tytuł aktualnej witryny przy użyciu modelu obiektowego WSS. Warto zwrócić uwagę na technikę pozyskiwania referencji do bieżącej witryny – parametr Properties (Właściwości) służy do pobrania referencji do obiektu SPWeb. Parametr Properties wywodzi się z klasy SPFeatureReceiverProperties, która udostępnia właściwość Feature, która z kolei udostępnia właściwość Parent