ELSEVIER
DIGITAL
PRESS

Kevin Laahs
Emer McKenna
Veli-Matti Vanamo

Microsoft® SharePoint Technologies

Planning, Design and Implementation

COMMUNICATIONS

publishing
partner
hp
invent

Microsoft® SharePoint Technologies

Microsoft® SharePoint Technologies

Planning, Design, and Implementation

Kevin Laahs
Emer McKenna
Veli-Matti Vanamo

ELSEVIER
DIGITAL
PRESS

Amsterdam • Boston • Heidelberg • London • New York • Oxford
Paris • San Diego • San Francisco • Singapore • Sydney • Tokyo

Elsevier Digital Press
30 Corporate Drive, Suite 400, Burlington, MA 01803, USA
Linacre House, Jordan Hill, Oxford OX2 8DP, UK

Library of Congress Cataloging-in-Publication Data

Application submitted.

ISBN 1-55558-301-6

British Library Cataloguing-in-Publication Data

A catalogue record for this book is available from the British Library.

For information on all Elsevier Digital Press publications, visit our Web site at www.books.elsevier.com.

04 05 06 07 08 09 10 9 8 7 6 5 4 3 2 1

Printed in the United States of America

Contents

Part II Planning, Deployment, and Operations

Part III Customizing and Extending SharePoint

This Page Intentionally Left Blank

Foreword

I love software that runs and delivers value out of the box. SharePoint Portal Server (SPS) 2001 certainly delivered great value to customers, because it was easy to deploy and use. SPS 2001 also affected the economics of the portal market, because its pricing made it viable for companies to dip their toes into portal deployment without incurring huge cost. All software evolves over time, and we now have a new version of SPS, plus the addition of Windows SharePoint Services as part of the base Windows operating system. So, now is a good time to take the opportunity to consider the value that companies can extract from SharePoint.

While SharePoint delivers value out of the box, you'll only get limited value if you accept everything as is. For the software to be truly valuable, I believe that you have to consider the needs of the business and then look at how SharePoint can help you address those needs. Bending SharePoint to meet what you think the business wants is probably pointless. Sometimes you'll succeed, but most of the time you'll end up with frustrated and unhappy users. Experience is tremendously important, too, because without experience, you'll never quite know where the limits exist and where you can apply tweaks to satisfy business needs.

This book is late. I have moaned at the authors on this point for a long time, but they quote my own arguments back to me, citing the need to capture their experience of meeting real business needs within HP as we deploy and use SPS and Windows SharePoint Services to help our people be more productive. Experience takes time to accumulate, so I guess I shouldn't be surprised that the book is late. Now that it has appeared, the content in the book shows real experience of technology in action in many places, and that's its real value. Many people can sit down, learn a technology, and describe it in a book. Few can tell the story from the perspective of experience and make their hard-earned knowledge available in an accessible form. Although it took time, I think this book hits the target, and I'm sure it will be valuable to you, especially if you face the challenge of migrating from the original SharePoint architecture as implemented in the 2001 release to the new technology platform exploited by SPS 2003. Migrating data is easy enough; the challenge exists in understanding how to best exploit the new architectural capabilities and what you need to do with the customized Web Parts and other tweaks that administrators and programmers deploy to meet business requirements.

As with any book, you have to take the information presented here and put it into context with your own experience, requirements, and pressures (time and other). Dip into the book, find its delights, and use them to your advantage.

Enjoy!

Tony Redmond
Vice President and Chief Technology Officer
HP Services

Preface

Mergers, mergers, mergers . . .

SharePoint Products and Technologies are the second generation of the SharePoint family, whose roots are to be found in the Version 1 products called SharePoint Team Services and SharePoint Portal Server (SPS) 2001. Since our first book on SPS 2001, we have lived through many forms of mergers (both personal and business, including the largest IT merger in history between HP and Compaq) and have learned that great things can come from successful mergers. Successful mergers tend to leverage the individual strengths of all involved into a single, stronger force that benefits all concerned.

And this is also the case with SharePoint. The very loosely coupled Version 1 products have come together to produce a more tightly integrated and richer platform, upon which successful collaboration and information sharing can occur. As well as finally delivering on the SharePoint brand, the overall offering merges nicely with other major Microsoft directions, such as Windows 2003, the .NET Framework, and Visual Studio .NET. The delivery of the SharePoint platform also offers benefits to others that play in the collaboration space, as seen with the likes of Office, Project, and InfoPath as they consume underlying SharePoint services.

We, the authors of this book, all work for HP Services and are involved with SharePoint at many different levels—through consulting directly with customers, presenting at industry conferences on SharePoint technologies, contributing to industry publications, and architecting and building HP Service's internal knowledge network based upon SharePoint. We hope that the range of our experience helps you achieve success in your encounters with SharePoint.

Book structure

This book does not cover every single aspect of SharePoint, and if we have missed your favorite topic, then we apologize. We do feel, however, that we have tackled the most important areas, as well as those subjects in which our experience complements the standard product documentation and other books that are out there. We urge you to use your favorite search engine if the answer you seek is not contained here!

We understand that your interest in SharePoint may derive from many different sources, and therefore, we have structured the book into three distinct parts. Part I is for those of you who want to find out about what SharePoint can actually do for you in end-user terms. If you are new to SharePoint and want to see if it can do a job for you, then we recommend you read Part I.

Part II is for system architects who want to know how to plan, design, implement, and manage a SharePoint environment. If you are tasked with rolling out SharePoint to support your business, then we recommend that you read Part II.

Part III is for those who want to extend base functionality to meet specific business needs. If you are a developer who is tasked with enhancing a SharePoint environment, then we recommend that you read Part III.

Note that there will be some slight repetition among the three parts in places. We are aware of this, and it was a deliberate choice in order that you could read each part in isolation from the others. The repetition is merely to ensure that you have the prerequisite information to follow along with the current topic.

Where there's a merger, there's a divestiture . . .

Almost as common as mergers are divestitures, and again, we have experienced many over the past years. We're glad to say that none of us has experienced divorce, albeit writing a book could be cited as a justifiable reason in any such action. We just wanted to say that Don Vickers, who coauthored the first book, could not join us for this second journey and we wish him continued success in all that he does. We did manage to find an excellent replacement in Veli-Matti Vanamo, who, aside from being an alphabetically handy replacement, has also matured sufficiently to appreciate (and spell correctly) Laphroaig.

Kevin, Emer, Veli

Acknowledgments

It is with great relief that we find ourselves writing this section of the book; there were times during this process that we didn't think we'd make it, but here we are! While writing a book provides many benefits, including an opportunity for personal growth, it requires an enormous commitment—not only from the authors but from many other sources. The acknowledgments section gives us each a chance to express our thanks to the people who helped to get us here by providing their ongoing advice, encouragement, and support.

We'd like to thank the publishing team—namely Alan Rose, Lauralee Reinke, and Theron Shreve—for their incredible patience, dedication, flexibility, commitment, and support throughout. You truly are a wonderful team and it was a pleasure working with you. Special thanks go to Tony Redmond for his support and providing the Foreword; Mike Topalovich, Gary Adams, and Adam Rosenblatt for taking the time to provide their reviews; and Eric Tipton for his contribution to the review process.

We are fortunate to work inside a company that has a wealth of experience and a sharing culture that allows us to draw on this experience in many ways. So thanks to all our colleagues at HP. We are also fortunate to have a very close working relationship with Microsoft, and we'd like to thank all our friends there for their willingness to help us and answer our many questions—we promise to continue bombarding you guys with e-mails!

Before we move on to our individual acknowledgements, collectively we'd like to thank those of you who ordered this book many months ago and didn't cancel your order—thank you so much for your patience; we hope you find it worth the wait!

Personal Acknowledgments

To Emer and Veli, my family and friends, and to my wife, Wendy, and children, Jenny and Euan—thanks for putting up with me.

—Kevin Laahs

I'd like to thank Kevin and Veli for their knowledge, expertise, and dedication—as always, it was great working with you guys.

Most of all, I would like to thank my wonderful husband, Michael, and my beautiful little girl, Caoimhe, for supporting my contribution to this book, which required countless nights and weekends away from both. I am so relieved that we can spend time as a family once more. I love you both dearly. XOXO

—Emer McKenna

First of all, I would like to thank Kevin and Emer for the opportunity to coauthor the book, and for the countless hours you folks spent pulling me through it. I'd like to thank my wife, Audrey, for her enormous support and even agreeing to go through with the wedding during the lengthy writing process. Finally, many thanks to my family for all the advice and support throughout the years.

—Veli-Matti Vanamo

Part I

Introduction to SharePoint Products and Technologies

This Page Intentionally Left Blank

1

How Did We Get Here?

1.1 From whence we came

This is our second book with the word "SharePoint" in its title. Our first, entitled *Microsoft SharePoint Portal Server: Building Knowledge-Sharing Applications* (2001), dealt primarily with Microsoft SharePoint Portal Server 2001 (SPS 2001). I guess it would be natural to assume that this offering is merely an update covering the next release of this product—Microsoft Office SharePoint Portal Server 2003 (SPS 2003)—but this is a false assumption. This book actually covers Microsoft SharePoint Products and Technologies, an umbrella name for those offerings from Microsoft that facilitate collaboration. Now, while SPS is still a major player in this area, it is only part of a strategy designed to connect people to information across all levels of an organization.

This strategy is significantly different from the one that was around when we wrote our first book. It is therefore important to understand how Microsoft has arrived at such a strategy, as this will help you appreciate the thinking behind the current product offerings. Hopefully, this will then assist you in making informed decisions about how to deploy a SharePoint environment for your specific collaborative needs.

1.1.1 Wasn't Exchange meant to be the collaboration platform?

If you cast your mind back to 2000, you will recall the launch of Exchange Server 2000. A significantly different architecture from that of Exchange 5.5, it was hailed as not just a messaging server, but also as a collaboration server. Additionally, it brought new technologies to the table that would be used by other Microsoft product offerings. Witness here the Web Storage System, which is at the core of SPS 2001.

As a collaboration server Microsoft was clearly trying to offer something that could compete with Lotus Notes. Few would argue that Exchange provided the best messaging platform, while Notes provided the best collaborative application platform, and both products were trying to make ground in their "weaker" areas. Thus, a major focus of Exchange 2000 was providing a platform upon which developers could build collaborative applications in addition to delivering the best-in-class messaging server. As it turned out, Exchange's strength as a messaging server alone was enough to keep the threat of Notes at bay, and this is one reason why Microsoft has subsequently de-emphasized Exchange as a collaboration platform and focused it firmly as a pure messaging server. Indeed, you only have to look at Exchange Server 2003 and note that real-time collaboration options such as Instant Messaging, which were part of Exchange Server 2000, are no longer part of the Exchange offering.

However, the consideration that the messaging battle had been "won" was not the only reason for the de-emphasis. The failure of Exchange Server 2000 to become a ubiquitous collaboration platform was another. True, some applications were built on top of Exchange, but it basically failed to capture the interest of the majority of application developers out there. It also met with resistance from administrators, whose primary purpose in life was to ensure a highly performing messaging server. E-mail is mission critical for most organizations these days, and anything that might jeopardize its availability or service-level agreements is met with scorn!

Understanding the reasons for the failure has helped shape the current strategy. At the time, Microsoft had three major goals for Exchange as a collaborative application development platform:

1. Make Exchange-based data widely available through many protocols and APIs. If the Web Storage System were to be the place to store all sorts of information, then it would have to be ubiquitously available.

2. Provide an extensible platform. In order to meet a wide variety of needs, new services would need to integrate easily into the platform.

3. Deliver a compelling environment for the development of rich, collaborative applications. Encourage developers to use the Web Storage System as their platform of choice.

Let's take a look at each goal and see how it fared. First, Microsoft certainly did provide ubiquitous access to Exchange data. The primary APIs were ActiveX Data Objects (ADO), Collaboration Data Objects for

Exchange (CDOEX), and Web-based Distributed Authoring and Versioning (WebDAV). The support of ADO meant that developers who were used to coding against, say, SQL or Oracle, could code against the Web Storage System in much the same way. This met with Microsoft's goal of making the Web Storage System appeal to the masses. CDOEX provided a rich API that made it easy to handle typical Exchange items such as messages, calendars, contacts, and tasks. WebDAV support made building Web-based clients using the HTTP protocol a reality—Outlook Web Access (OWA) being a prime example here. Indeed, using any mixture of these three APIs, you could quite easily access and manipulate items in user mailboxes and public folders.

Sounds like a pretty good picture for data access, so why wasn't it successful? Well, while ADO and CDOEX were robust and powerful APIs, they could only run on the Exchange Server. This meant that applications themselves would have to be installed and run on the Exchange Server. "Not on MY Messaging Server!" would be the response of most Exchange administrators. You see, administrators tend to be passionate about their messaging service and do not want to risk compromising the service it provides to end users. Therefore, it was very difficult to put forward a good case for installing such applications onto the Exchange Server. The only other alternative would be to supply Exchange Servers that were separate from the core messaging service specifically for these applications—clearly a prohibitive expense!

WebDAV is different from ADO and CDOEX in that it can run remotely. In other words, you can run your custom application on one server and have it access Exchange-based data on another server. Now, while this approach appeased Exchange administrators, writing raw WebDAV applications was rather complex. Thus, developers did not embrace this as a viable solution.

Finally, with the provision of these multiple APIs, there were multiple ways to achieve the same aim. For example, there are eight different ways to create a folder in the Web Storage System, so which is the right one to use? This plethora of options merely confused rather than assisted the developer.

Providing an extensible platform was achieved mainly through the support of events at three different levels—SMTP Protocol, SMTP Transport, and Store. Events allowed you to write your own custom code that could react appropriately as messages traversed your environment and as items were manipulated in the Web Storage System. A couple of examples will help clarify the power of events. First, SPS 2001 actually relied heavily on a Store event in order to ensure the integrity of documents as they were checked in and published (remember SPS 2001 uses the Web Storage System that was initially provided by the Exchange group). As documents are created in an SPS 2001 document library, the Store event ensures that valid profiles are filled in, version numbers are maintained, and so forth.

The second example is that of third-party antivirus vendors, who typically harness the SMTP Transport events in order to scan messages before they are actually delivered to a user's mailbox. This is a great example of how new services could be layered easily on top of the platform.

Now, much like the problems with ADO and CDOEX, any application utilizing events would ultimately have to be installed directly onto the Exchange Server. While it is true that SMTP events could run on boundary servers not hosting user mailboxes, Store events would typically have to be on mailbox and public folder servers if they were to add any significant value. Yet, again, Exchange administrators hollered, "Not on MY Exchange Server!" A secondary issue that relates particularly to SMTP events was that you really needed to be able to code in a language such as C++ in order to exploit their capabilities fully. This alienated a lot of those people who would not consider themselves to be hard-core developers and meant that the feature was not overly utilized, as it did not appeal to the masses.

The third goal of delivering a compelling platform for the development of collaborative applications had its challenges as well. Microsoft wanted to lure developers into using the Web Storage System as their primary platform, not just for the data associated with an application but for all its collateral, including source code, forms, and the like. As such Microsoft envisaged all application code and data to be stored within public folders. To this end it provided supporting technologies and showcase applications such as Web Storage System Forms, FrontPage Extensions for Web Storage System Forms, sample public folder–based applications, a workflow engine/designer, and regular updates to the Exchange Software Developer Kit. Interestingly, the workflow engine actually leveraged Store events and was a great example of how the overall platform could be extended.

Did these features attract the application developer? Unfortunately, the answer was no, due mainly to the fact that the whole paradigm was confusing and inconsistent when compared with other Microsoft initiatives. For example, writing a Web application with Visual Studio 6.0 was fairly straightforward for a developer, but there was no direct support for utilizing the Web Storage System as the storage mechanism for such an application. Furthermore, the whole approach seemed detached from the main development story emanating from Microsoft—that being the .NET Framework, Visual Studio .NET, and the whole Web Services world.

So the bottom line here was that the strategy was confusing, inconsistent, and impractical, and there was a big message from administrators saying, "Don't mess with my messaging server!" So, clearly, another approach had to be taken to find a suitable platform for building applications that would enhance the lives of information workers in the collaboration area.

Note that the effort made to make Exchange a collaboration platform was not all in vain. I say this because there are pieces of the strategy that align with the future and have made Exchange a good citizen in a .NET world—that being the support of WebDAV such that Exchange services can be consumed by remote applications as and when required. As we will see later, this capability is actually harnessed by Windows SharePoint Services in order to allow documents to be e-mailed into a document library.

1.1.2 What else was going on?

Ever heard of a product called SharePoint Team Services (STS)? I'll bet you probably have, but at the time of its release, SPS 2001 was getting all the airplay. In fact, in our first book we dedicated a whole paragraph to STS (on page 19 if you are really interested), which just goes to show that we did not really understand its true purpose in life (hey, at least we are honest!).

In our defense, Microsoft did not actually give STS the exposure it deserved at that time, and its relation to SPS was purely nominal. Part of the Office XP family, STS came free with a FrontPage 2002 license and, from a technical point of view, had its technology roots firmly planted in FrontPage Server Extensions. It was designed to allow the easy creation of Web sites that users could use as a mechanism for sharing information and collaborating.

It is fair to say that Microsoft did not market the capabilities of STS as well as it could have and put more effort into the technically unrelated SPS product that had shipped a few months previously. The confusion over these two products initially caused STS to lose momentum and not attain its potential in terms of customer uptake. In fact, a common misunderstanding was that an STS team site could only handle groups of up to 50 users. Only after Microsoft publicized its internal usage did STS gain credibility as concerns over scalability were alleviated. With additions such as Self Service Creation, STS proved to be a very popular platform for empowering users with the ability to create team sites instantly and start collaborating without involving IT groups for site creation.

SPS and STS both had their strengths, but were architecturally different and used different development metaphors. For example, to build on SPS required knowledge of the Digital DashBoard Architecture and Web Parts, whereas building on STS required skills with FrontPage Remote Procedure Protocol (RPC) and Collaborative Markup Language (CAML).

However, STS filled a crucial role that couldn't be fulfilled by SPS 2001—it provided an environment in which ad hoc teams could come together, work on some collateral, and then disband. For example, a sales team could get together to work on a new proposal for a customer, or three people could cooperate on the creation of a book. SPS 2001 was designed more to support the publishing and discovery of information than the creation of such information and, therefore, did not lend itself well to the needs of team-based collaboration. The lesson learned is that an effective knowledge-sharing environment must support both the dynamic creation and publishing of information.

Collaboration means working together, especially in a joint intellectual effort. It's fair to say that although the three main offerings from Microsoft in the collaboration space each had its individual merits (Exchange Server 2000, SPS 2001, and STS), they did not exactly benefit from collaborative engineering. Clearly, a more cohesive, consistent, and all-encompassing message was required, and, indeed, this is where the much more compelling story of SharePoint Products and Technologies enters the fray. Which brings us to 2003.

1.2 2003—A vintage year for collaboration

Microsoft released a bundle of products in 2003. This is not really surprising, as it releases many products each year, but "working together" and "joint intellectual effort" are clearly emerging product trends.

The Microsoft Office family of products has been significantly extended, with members sharing much more than just the name. The full list of Office 2003 products is as follows:

- Access
- Excel
- FrontPage
- InfoPath
- Live Communications Server
- Live Meeting
- OneNote

- PowerPoint
- Project
- Publisher
- SharePoint Portal Server
- Visio
- Word

The interesting point here is that formerly “stand-alone” server products have now been brought into the family—in particular SPS and Live Communications Server. This indicates a desire to extend the functionality of personal productivity applications through server-based services.

Completing the major 2003 product set is Windows Server 2003, Windows SharePoint Services, and Microsoft Exchange Server 2003. Note that, as discussed earlier, Exchange 2003 has dropped the “Collaboration” tag and is now firmly positioned as a messaging server. Based on the lessons learned with Exchange 2000, SPS 2001, and STS, the collaboration strategy now focuses on an extensible infrastructure supporting services that citizens of multiple collaborative environments can consume, rather than on one that assumes people only need to collaborate from within their messaging system.

The basis of the collaboration story is now rooted inside Windows itself, through Windows SharePoint Services, and through individual product offerings that can easily leverage these services.

1.2.1 The collaboration ecosystem

As people go about their day-to-day jobs, they encounter many opportunities for collaboration. Indeed, the need to work together with one or more people (or with collateral created by other people) can arise at any moment and is seldom restricted solely to when you are working within a specific application. For example, as I author this part of our book, I think of questions that can be answered by my colleagues, I search for other collateral in the same topic area, and I think of tasks that need to be completed before this book can be published. People collaborate on many different levels and from within many environments. Looking at a pre-2003 Microsoft-centric environment, we see that one-to-one communication is performed via Outlook and Instant Messaging; teamwork on ad hoc projects is performed through a mixture of file shares, public folders, and SharePoint Team Services v1.0; conferencing is performed through either NetMeeting or Exchange Conferencing Server, and publishing is performed through SPS 2001.

Microsoft's collaboration strategy is to provide services that can be consumed "naturally" as information workers go about their day-to-day jobs. By this, I mean that the information workers should not have to divert from their current focus when they need to collaborate. If I am authoring a document in Word and something in the document triggers the need to allocate a task to a colleague, I should be able to do so directly from Word itself and not have to go searching for another application. Collaborative services should be baked into the operating environment and called upon as and when it makes sense.

Another part of the strategy builds upon the realization that in order to collaborate effectively on a piece of work, users need to access multiple information sources—and not just document sources. Connecting people and information is very important. For example, if I am researching a topic and come across an interesting piece of work by a particular person, I may want to see other items that have been authored by that same person or may want to converse immediately with that person through an Instant Messaging conversation. Alternatively, I may want to identify people that have a particular skill set and target them for specific questions. Connecting people and information is therefore something addressed by the strategy.

The strategy therefore is to deliver rich collaboration services built into the infrastructure that go far beyond that of traditional file-sharing services, but are just as easy to use. Endusers are empowered to use these services through the tools they most commonly use—Office, Windows Explorer, and Internet Explorer—and other applications can avail themselves of and build on top of these infrastructure services.

1.3 Delivering the strategy

Four server products form the backbone of the collaboration strategy (Figure 1.1).

1.3.1 Exchange Server 2003

Focusing on Exchange Server 2003's strength as a messaging server, we see that previous collaboration services have been removed from the product. Exchange no longer delivers Instant Messaging and the Exchange Conferencing Service. Exchange also provides the best platform for personal information management in terms of calendars, contacts, and tasks and enables

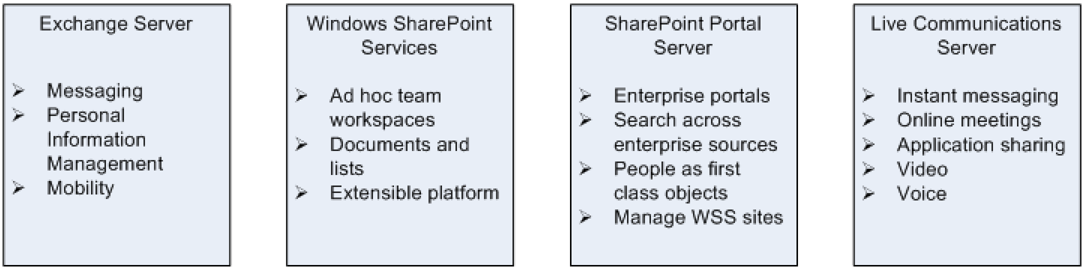

Figure 1.1 *Major products in the collaboration space.*

you to access your mailbox through multiple mobile clients, including Outlook Web Access and Outlook Mobile Access.

1.3.2 Windows SharePoint Services and SPS

Although these products could be seen as upgrades to SharePoint Team Services v1.0 and SPS 2001, it is better to think of them as new products, delivering the core part of Microsoft's collaboration strategy. Together they are known as SharePoint Products and Technologies, and Microsoft had several goals when designing the SharePoint suite.

First, it wanted to provide a scalable base service for collaboration that could be leveraged by many applications. Second, it wanted to ensure that information sharing could happen at all levels of an organization—groups of people, teams of people, departments, and organizations. Third, it wanted to ensure that this environment exploited the .NET Framework and was easily consumed and extended by third-party applications. Finally, it wanted to deliver on the SharePoint brand by ensuring that any SharePoint product utilized the underlying SharePoint services.

Windows SharePoint Services (WSS) is so called because it is part of Windows (available as a free download from Microsoft's Web site), and it provides the base collaborative infrastructure, as we will see in Chapter 2. It is an extensible Web-based environment that lets users quickly provision dynamic Web sites for specific purposes. For example, users can provision a Web site in order to host the collateral of a project or to collaborate on the creation of a particular document. The services offered by WSS are consumable via an object model and Web Services and can therefore be used in any application. Refer to Chapter 3 for details of how Office 2003 consumes the services provided by WSS. WSS provides a scale-out and scale-up three-tier architecture utilizing ASP.NET to deliver Web pages and SQL Server to host data.