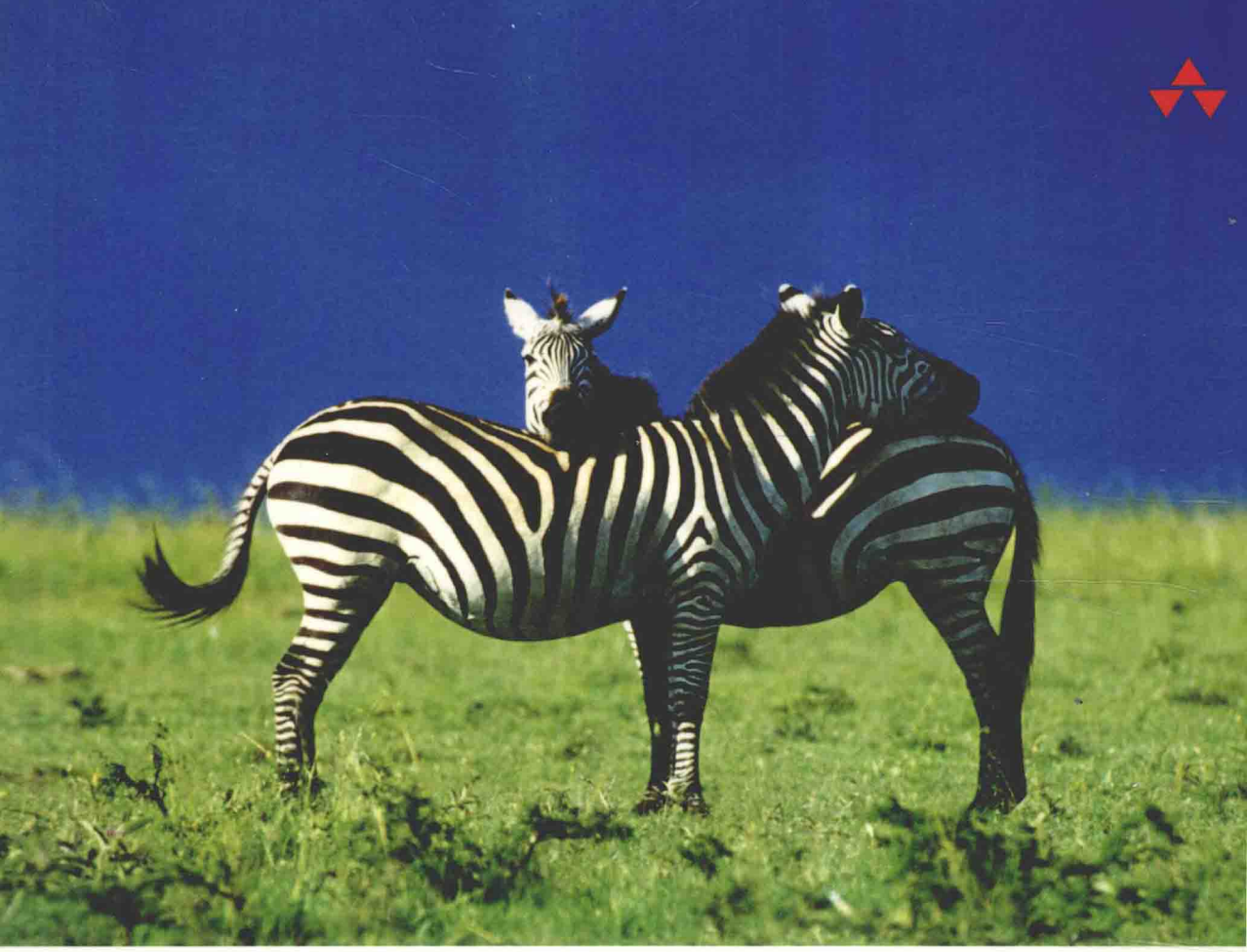

Understanding SOA with Web Services

Eric Newcomer
Greg Lomow

Independent Technology Guides
David Chappell, Series Editor

Understanding SOA with Web Services

Eric Newcomer and Greg Lomow

Addison-Wesley

Upper Saddle River, NJ ■ Boston ■ Indianapolis
San Francisco ■ New York ■ Toronto ■ Montreal
London ■ Munich ■ Paris ■ Madrid ■ Capetown
Sydney ■ Tokyo ■ Singapore ■ Mexico City

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The authors and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U. S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the U. S., please contact:

International Sales
international@pearsoned.com

Visit us on the Web: www.awprofessional.com

Library of Congress Catalog Number: 2004112673

Copyright © 2005 Pearson Education, Inc.

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, write to:

Pearson Education, Inc.
Rights and Contracts Department
One Lake Street
Upper Saddle River, NJ 07458

ISBN: 0-321-18086-0
Text printed in the United States on recycled paper at Phoenix Color Corp. in Hagerstown, Maryland.
First printing: December 2004

Understanding SOA with Web Services

Independent Technology Guides

David Chappell, Series Editor

The **Independent Technology Guides** offer serious technical descriptions of important new software technologies of interest to enterprise developers and technical managers. These books focus on how that technology works and what it can be used for, taking an independent perspective rather than reflecting the position of any particular vendor. These are ideal first books for developers with a wide range of backgrounds, the perfect place to begin mastering a new area and laying a solid foundation for further study. They also go into enough depth to enable technical managers to make good decisions without delving too deeply into implementation details.

The books in this series cover a broad range of topics, from networking protocols to development platforms, and are written by experts in the field. They have a fresh design created to make learning a new technology easier. All titles in the series are guided by the principle that, in order to use a technology well, you must first understand how and why that technology works.

Titles in the Series

Brian Arkills, *LDAP Directories Explained: An Introduction and Analysis*, 0-201-78792-X

David Chappell, *Understanding .NET: A Tutorial and Analysis*, 0-201-74162-8

Eric Newcomer, *Understanding Web Services: XML, WSDL, SOAP, and UDDI*, 0-201-75081-3

For more information, check out www.awprofessional.com.

此为试读,需要完整PDF请访问: www.ertongbook.com

To Jane, Erica, and Alex.

—Eric Newcomer

For Barb and Princess Fluffy Muffin, and in loving memory of

Lonnie, Mac, and Moses.

—Greg Lomow

Preface

The widely adopted and implemented core Web services standards (SOAP and WSDL) have achieved unprecedented interoperability across highly disparate software systems. As a result, new Web services standards have been proposed for extended features such as security, reliability, transactions, metadata management, and orchestration that extend Web services for use in a broad range of new applications.

The service-oriented architecture (SOA) has also become widely recognized for its important role in information technology projects. An SOA is a style of design that guides an organization during all aspects of creating and using business services (including conception, modeling, design, development, deployment, management, versioning, and retirement).

Despite some limitations (which we document), an SOA with Web services is the ideal combination of architecture and technology for consistently delivering robust, reusable services that support today's business needs and that can be easily adapted to satisfy changing business requirements.

Think about an SOA as an assembly line in a factory. It's an investment in the future operation of your business, so a significant amount of planning, design, and development may have to go into it before it starts to really pay off. The first car off a production line is more expensive than the thousandth. Similarly,

the first service deployed in an SOA is more expensive than the hundredth. The major benefits of SOA arrive over time, although as we will see, it is possible to start small and incrementally build up to a full-fledged SOA.

SOA with Web services is important because it aligns information technology (IT) with business requirements and because it reduces the costs of IT systems and applications. An SOA gives you the ability to more easily integrate IT systems, provide multi-channel access to your systems, and to automate business processes.

Rather than relying entirely upon the skill and knowledge of certain specific individuals to implement business requirements in technology, SOA provides a foundation for rapidly assembling and composing new applications out of a library of reusable services that anyone can understand. When an SOA is in place and services are developed, developers can easily reuse existing services in their new applications and automated business processes.

Like any new investment in technology and infrastructure, it's important to understand the right way to do it and what you can and can't do. SOA and Web services are great, but they can't do everything. We hope that this book will help you achieve the benefits of SOA with Web services while avoiding the pitfalls.

Acknowledgments

The authors would like to sincerely thank series editor David Chappell for his invaluable assistance in reviewing several early drafts of the manuscript and providing unwavering clarity and vision during major rewrites to guide the book toward its current form. We would also like to thank Rich Bonneau for his help during the initial planning stages.

The authors would also like to thank the many reviewers who provided us with specific comments, suggestions, and corrections that significantly improved the quality of the book. In particular, Anne Thomas Manes for providing the most thorough review, catching many technical errors and highlighting sections whose clarity needed improvement—and doing both with concrete and helpful edits. We'd also like to thank Jason Bloomberg, Daniel Edgar, and Ron Schmelzer for reviewing the entire manuscript twice, and offering many detailed comments and overall suggestions for improving the text. And finally, we'd like to thank Steve Vinoksi for calling our attention to significant problems with the initial draft, and Max Loukianov for his many helpful comments on the final version of the manuscript. Any remaining errors, inconsistencies, and unclear text are our responsibility alone.

We'd like to thank the extremely professional and helpful editorial staff at Addison-Wesley for their fine work in producing this book, including Mary O'Brien who supported us from the beginning of the project and never lost faith, Brenda Mulligan who helped twice in arranging reviews so essential to a high-quality manuscript, and the production staff, including Sarah Kearns and Ben Lawson, who helped polish things up and ensure as much consistency as possible between the various parts that we each wrote.

I would like to thank Sean Baker for his encouragement and assistance at the beginning of the project, and the members of the senior management team at IONA for their understanding during the tough writing stages near the completion of the book. I would also like to thank Rebecca Bergersen for her assistance with the policy specifications and Wolfgang Schulze for his help with SOA concepts. And I would especially like to thank the members of my family for their patience with what they all now hope will become known as the final book project—in particular, my wife Jane and kids Erica and Alex, who have all been extremely supportive in the face of much frustration over weekends and vacation time consumed with working on the book.

—Eric Newcomer

I would like to give special thanks to Brian Unger, Marshall Cline, Joe Schwartz, Peter Cousins, and Jim Watson—you are great friends and mentors, and you continue to inspire and motivate me. I would also like to thank Ivan Casanova, Dirk Baezner, Jim Quigley, Dmitry Grinberg, Cemil Betanov, Sam Somech, Steve Marini, Ted Venema, Larry Mellon, William Henry, John Dodd, Kevin Maunz, Jeff Mitchell, and all my other friends and colleagues from Jade Simulations, Level 8 Systems, IONA Technologies, and BearingPoint—it has been a pleasure and honor working with each and every one of you. I would also like to thank my wife, Barb—thank you for all of your patience and tolerance during the writing and production of this book.

—Greg Lomow

About the Authors

In the role of Chief Technology Officer at IONA, **Eric Newcomer** is responsible for IONA's technology roadmap and direction as relates to standards adoption, architecture, and product design. Eric joined IONA in November 1999 as transaction architect, and most recently served as Vice President of Engineering, Web Services Integration Products. Eric has 26 years experience in the computer industry, including more than 15 years at Digital Equipment Corporation/Compaq Computer, where he held a variety of technical and management positions before receiving a corporate-level technical appointment. Eric received his BA in American Studies from Antioch College, with a minor in computer science.

In addition to *Understanding Web Services*, published in 2002, Eric is co-author of *Principles of Transaction Processing*, published in 1997 by Morgan Kaufman, and co-author of a chapter called "The Keys to the Highway" in *The Future of Software*, published in 1995 by MIT Press. Eric is also the author of numerous white papers and articles, co-author and editor of the Structured Transaction Definition Language specification published by X/Open (now The Open Group) in 1994, former member of the Transaction Internet Protocol working group at IETF, former member of the X/Open Distributed Transaction Processing committee that created the XA specification, former chair of the OTS RTF at OMG, and chair of the team that developed the XML Valuetype

此为试读,需要完整PDF请访问: www.ertongbook.com

specification at OMG to map XML to CORBA. He was a charter member of the XML Protocols Working Group at W3C, where he served as an editor of the requirements document that led to SOAP 1.2. He served for nearly two years as an editor of the W3C Web Services Architecture Specification, and most recently served as co-chair and editor of the Web Services Composite Application Framework set of specifications at OASIS.

Greg Lomow, Ph.D., is a senior manager and consultant for BearingPoint, Inc. Greg has 12 years of experience as a consultant and enterprise architect working in the financial services, telecom, and federal government sectors designing business applications using service-oriented architecture, developing simulation applications using distributed object technology, and training developers in object-oriented design and programming techniques. He also worked for eight years as a product manager at Jade Simulations, Level 8 Systems, and IONA Technologies responsible for integration, web services, and middleware products. Greg co-authored *C++ Frequently Asked Questions* published by Addison-Wesley in October 1999 (1st ed.) and again in January 1999 (2nd ed.). He completed his Ph.D. in computer science at the University of Calgary, Canada, in 1988. Greg is an active member of the Web Services Interoperability (WS-I) Organization.

Introduction

In the early days of business computing, no one paid much attention to sharing application logic and data across multiple machines. The big question was how to develop systems to automate previously manual operations such as billing, accounting, payroll, and order management. Solving any one of these individual problems was challenging enough, not to mention the additional challenge of basing all systems on a common, reusable architecture, and few organizations were in a position to tackle it.

In the modern era, most operational business functions have been automated, and now the big question is how to improve the ability of these systems to meet new requirements. Adding a new user interface, combining multiple data sources into a single view, integrating mobile devices, or replacing an old application with a better one are common reasons for investing in new projects.

The paradigm of service-oriented development, although not new, is catching on as the information technology (IT) world shifts from developing new systems to getting more out of earlier investments. Developing services and deploying them using a service-oriented architecture (SOA) is the best way to utilize IT systems to meet new challenges.

A service differs from an object or a procedure because it's defined by the messages that it exchanges with other services. A service's loose coupling to the applications that host it gives it the ability to more easily share data across the department, enterprise, or Internet. An SOA defines the way in which services are deployed and managed. Using an SOA increases reuse, lowers overall costs, and improves the ability to rapidly change and evolve IT systems, whether old or new.

Supporting the shift toward service-oriented development and SOA is a large cast of characters called Web services technologies. These represent the most widely adopted distributed computing standards in the industry's history. As we'll see, Web services are an ideal platform for SOA.

The modern answer to application integration, therefore, is an SOA with Web services. An SOA maps easily and directly to a business's operational processes and supports a better division of labor between technical and business staff. Furthermore, an SOA uses a description model capable of unifying existing and new systems.

When Web services descriptions are available throughout a department or an enterprise, service-oriented integration projects can focus on composing Web services instead of dealing with the complexity of incompatible applications on multiple computers, programming languages, and application packages. Whether you use Java, C++, Visual Studio, CORBA, WebSphere MQ, or any other development platform or software system, such as J2EE or the .NET Framework, SOA provides the architecture, and Web services provide the unifying glue.

As businesses and governments continue to struggle to align IT expenditures with the bottom line to achieve strategic market objectives, software productivity gains are elusive, especially compared to gains in hardware price and performance. With the widespread adoption and implementation of Web services, products and technologies are finally in position for enterprises to realize these benefits.

Compared to the standards of the past, Web services are much easier to learn and use. The broad adoption of Web services standards makes it easy to imagine a world in which all applications have service interfaces. And from that vision, it's easy to imagine IT staff performing the bulk of their activities using Web services interfaces.

It's true, however, that someone will have to deal with the plethora of legacy technologies in order to service enable them. But the beauty of services and SOA is that the services are developed to achieve interoperability and to hide the details of the execution environments behind them. In particular for Web services, this means the ability to emit and consume data represented as XML, regardless of development platform, middleware, operating system, or hardware type.

The most important application of SOA is connecting the various operational systems that automate an enterprise's business processes. A company's operational environment can be as unique and varied as its culture because the way a company does business can be part of its competitive edge. Classic examples include Wal-Mart's inventory management system, American Airline's pioneering automated reservation system (SABRE), and Dell's on-demand manufacturing supply chain management system. Companies need IT systems with the flexibility to implement specialized operations, to change as easily as business operations change, to respond quickly to internal as well as external condition changes, and to gain or maintain a competitive edge.

In general, the evolution of the software industry has been about improving the ease with which people interact with computers, continually raising the abstraction level of languages to make them accessible to more people with less scientific training. The purpose of this evolutionary change is to adapt software systems more easily and directly to the needs of people and institutions.

Making it easier for people to tell computers what to do drives the evolution of the software industry because the easier it is to tell computers what to do, the easier it is to get more projects done quickly and cheaply. The abstraction benefit of Web services (having the ability to work with any type of implementation software) helps reduce IT's single largest cost: labor. For instance, although it

might be easy enough for an office worker to create reports and spreadsheets, or for a consumer to surf the Web and order books, it is still not easy enough for the back office staff to tell those big machines (or farms of machines) what to do to service the millions of transactions a week that many businesses require to stay in operation.

When thinking about IT projects, one of the most difficult subjects is determining who will be able to bridge the business-technology gap—that is, which project member will have the right skills to ensure that the business issues are well enough understood to be implemented in the technology and that the technology issues are well enough understood to meet the business requirements. One of the great potential advantages of solutions created using an SOA with Web services is that they can help resolve this perennial problem by providing a better separation of concerns between the business analysts and service developers. The service developers can take responsibility for implementing services that meet business requirements, while the analysts can take responsibility for defining how services fit together to implement a business process.

Web services marry the ease of use of a document markup language to distributed computing concepts and apply the result to solve IT integration problems cheaply and easily. The standards are firmly in place for Web services basics and are quickly emerging for enterprise qualities of service, metadata management, and orchestration, meaning that everything is ready for businesses to start realizing the true benefits of an SOA.

Integrating existing and new applications using an SOA involves defining the basic Web services interoperability layer and the enterprise quality of service layer to bridge features and functions used in current applications such as security, reliability, and transactions. It also involves the ability to define automated business process execution flows across the Web services after the SOA is in place.

An SOA with Web services enables the development of services that encapsulate business functions and that are easily accessible from any other service. Composite services, furthermore, allow a wide range of options for combining

Web services and creating new application functionality. Web services technologies can be used to solve a broad range of IT problems, especially when used to quickly and easily join disparate pieces of software. Web services provide basic interoperability solutions and can easily be extended for use in enterprise integration architectures that lower cost and increase IT value, and they are especially valuable when used to implement SOA-enabled solutions.

What's in the Book

This book describes the best approach to designing and developing an SOA-based integration solution using Web services technologies and covers how an SOA provides a foundation for addressing other IT requirements, such as multi-channel client access, application interoperability, and business process management.

This book also provides tutorials on the advanced Web services technologies that help realize SOA solutions for enterprises, including metadata management, security, reliability, and transactions.

This book is intended for developers, technical managers, and architects interested in understanding the major principles and concepts behind SOA, how to implement an SOA using Web services, and how to achieve the business benefits of SOA. SOA is explained in the context of how it is most often used, such as enabling multi-channel access, composing applications, and automating business process management.

Organization of the Book

The book is divided into two major parts, with an overall introductory chapter. The two major parts are focused on:

- Service-oriented architecture, business process management, and how they are implemented using Web services technologies.
- Tutorials on Web services specifications for metadata management, security, advanced messaging (including reliability and notification), and transactions.

此为试读,需要完整PDF请访问: www.ertongbook.com