Service-Oriented Computing: Semantics, Processes, Agents

Munindar P. Singh Michael N. Huhns

July 30, 2004

Preface

The current World-Wide Web was intended to be used by people, but most experts, including the founder of the WWW, Tim Berners-Lee, agree that the future WWW will have to evolve to include usage by computer systems. Moreover, the impact of computer usage will exceed that of human usage. The evolution is expected to occur through the design and deployment of Web services. The term *Web services* sometimes refers to services that employ a particular set of basic standards. Since these standards are all but incidental to the key concepts of services and services apply even in settings strictly different from the WWW, it is helpful to think of service-oriented computing as a more general topic.

The objective of this book is to explain the principles and practice of service-oriented computing, although most of its concepts are developed in the context of Web services. The book presents the concepts, architectures, theories, techniques, standards, and infrastructure necessary for employing services. It includes a comprehensive overview of the state of the art in Web services and related disciplines.

Services are a means for building distributed applications more effectively than with previous software approaches. For this reason, it is not useful to talk about services without talking about service-based applications, how they are built from services, and how services should be designed so they can feature as parts of serious applications. Indeed, the *raison d'etre* for services is that they be used for multiple purposes. And, services are used by putting them together or *composing* them—the rare case where a service is used without any contact with another service can be thought of as a trivial composition. For this reason, every aspect of services is designed to help them be composed: specifically, so they can be described, selected, engaged, collaborated with, and evaluated.

Many of the key techniques now being applied in building services and service-based applications were developed in the areas of databases, distributed computing, artificial intelligence, and multiagent systems. These are generally established bodies of work that can be readily adapted for service composition. Some additional techniques, although inspired by these areas, are being developed from scratch by practitioners and researchers in the context of service-oriented computing. These new techniques address the essential openness and scale of Web applications that previous work did not need to address. Both classes of key techniques should be incorporated into our best practices. In most cases, they can be applied on top of existing approaches.

Philosophy and Approach

The philosophical position taken by this book is that it is not possible for computer scientists to develop an effective understanding of service-oriented computing by merely studying the basic standards for Web services. Unless we take an impoverished view of the subject and are content merely to run some canned examples, we must perforce examine several areas of computer science that come together in connection with services, and from which services derive their intellectual underpinnings.

All science, as the Nobel Laureate Ernest Rutherford once famously asserted, is either physics or stamp collecting. By physics Rutherford meant clean, succinct principles that apply to diverse phenomena and by stamp collecting he meant the cataloging and organization of large sets and varieties of observations.

To develop complex services and service-based applications requires both physics and stamp collecting. Almost reluctantly, the professional community has adopted some software engineering approaches for services. Some of these are based on organizing knowledge as "ontologies" and then agreeing upon the ontologies. In the end, they will require a lot of organizing, but the right abstractions will go a long way in streamlining the task of organizing.

Most importantly, however, traditional software approaches are fundamentally not sensitized to the challenges of an open environment. The physics, as it were, is designed for a closed environment. The effect is analogous to applying traditional mechanics to quantum mechanical problems. So, while we obviously need engineering methodologies to take care of the stamp collecting, we also need elegant principles that would give us the new physics.

Audience

We have sought to make this book complete as a college textbook. However, because it also includes many illustrations and examples, the book is eminently suitable for use by students and practitioners to learn about service-oriented computing on their own. This book serves the following main kinds of readers in the following ways.

Practitioners, who can supplement their knowledge of the details with an understanding of the concepts that underlie the standards and tools that they use, and the best practices based on those concepts. The book can help them leverage their practical experience to build stronger service-oriented systems and applications.

Technologists, including advanced developers and architects, who need to get their arms around service-oriented computing. They can better understand the key technologies and their key strengths and limitations, so they can conceive and execute their new major projects.

IT Strategists, who are concerned by the notorious business versus IT divide of modern practice. Service-oriented computing, as understood in this book, provides a conceptual framework to bridge that divide.

Researchers, who recognize the value of service-oriented computing as a source of major research problems of practical import. They might be deeply knowledgeable in some aspect of the subject, but might nevertheless wish to get a crash course on the remaining aspects.

Students, both graduate and senior undergraduate, who need to know about service-oriented computing simply to be adequately prepared for the expanding applications of services. And especially if their university curriculum is out of date and does not include service-oriented computing, they need to learn it on their own. The book can help them prepare to participate in the workforce in any of the above roles.

We have given numerous tutorials at a number of leading conferences based on the materials presented here. The attendees at those tutorials represented all of the above categories of reader. Further, the manuscript has been used as a text for two graduate courses and an advanced undergraduate course on Web services, and sections have been used within other graduate courses in multiagent systems and cooperative information systems.

The book includes numerous theoretical and programming exercises and some project ideas that all readers can use to solidify their understanding. (Homework is not just for the students!) Slides are available for use by instructors.

This book is self-contained: it includes the essential background for anyone planning to learn and develop the principles and applications of service-oriented computing. It has only a few prerequisites: some experience with Web programming or the willingness to learn it.

The Contents

This book is divided into the following major parts.

Basics. Part I describes the key trends and architectures in modern computing, which motivate why and how services are emerging. It also gives a crash course on current Web service technologies so that readers can quickly begin to experiment with these technologies, possibly beginning on a small project.

The key idea of an architecture based on Web services is that it identifies three main components: a service provider, a service consumer, and a registry. Providers publish their services on registries, and consumers find the service providers from registries and then invoke them. Current standards and techniques support these steps and enable many important use cases. However, to our way of thinking, they are unnecessarily limited in some respects. Yet the Web service architecture provides a nice framework, which can be fleshed out with more powerful representations and techniques. These are established computer science approaches and serious practitioners are already using them, although they are omitted from most expositions of Web services. The rest of the book shows what these are and how they can be employed.

Description. Part II addresses techniques and methodologies for describing services. These techniques include ideas from conceptual modeling of database schemas and domain

knowledge, and cover both representation and reasoning approaches. They lead naturally into some of the XML-based technologies gaining currency as the *Semantic Web*. The idea is that when services are described with sufficient richness, then it is easier for providers to state what they offer and for consumers to specify what they need, leading to meaning-based interactions.

Engagement. Part III deals with how services may be engaged or executed so as to facilitate the simpler kinds of composition. Often, when services are described, there is an emphasis on invoking services as if they were no more than methods to be called remotely. We prefer the term *engagement*, because it more accurately reflects the power of the service metaphor for computing. Imagine going to a carpenter, a human service provider, to have some bookshelves built in your home. You will typically not instruct the carpenter in how to build the shelves. Instead, you will carry out a dialog with the carpenter in which you will explain your wishes and your expectations; the carpenter might take some measurements, check with some suppliers of hard-to-get materials (maybe some exotic woods), and suggest a few options; based on your preferences and your budgetary constraints, you might choose an option or get a second estimate from another carpenter.

Likewise, in computing, instead of merely invoking a particular method, you would engage a service. Here the relevant computational themes are peer-to-peer computing, messaging, transactions (traditional, nested, and extended or relaxed), workflow, business processes, and exception handling. A number of standards for services are emerging in these areas.

Collaboration. Part IV discusses advanced concepts that arise from a computational standpoint in composing services, where it is helpful to think of the services as collaborating with each other. Some of the key technologies that apply for collaboration include protocols, agents, contracts, service agreements, and negotiation techniques. The engagement techniques in Part III give us the basis for engaging services while considering various transactional properties. The techniques of collaboration in this part go several steps further in characterizing the interactions among the consumers and the providers of services, dealing with how they plan and enact service episodes, how they maintain consistency, negotiate, enter into and execute contracts and agreements, and carry out specified protocols. This part includes a discussion of monitoring compliance with contracts and service agreements.

Selection. Part V introduces concepts of service discovery and selection, and distributed trust. Service discovery in its simplest form involves registries where services can be registered and looked up. However, selecting desirable services in practice also involves accommodating notions of trust, endorsement, and reputation.

This part of the book also includes a discussion of how services can be evaluated by the parties using them. This is essential to complete the cycle of locating services, engaging the services, and then evaluating the services to determine if they were successful.

Fair and accurate evaluation can enable the various parties to find, select, and engage the services that are superior in some way.

Engineering. Part VI focuses on the engineering of service-based applications. It discusses methodologies and techniques for building services in the context of some important classes of applications, especially knowledge management and e-business. This part also discusses the best practices for the main kinds of techniques described in the previous parts.

Directions. Part VII discusses some of the key trends in services and in architectures. It considers architectural policies, privacy, and personalization from the perspective of how services fit into the larger world. It also discusses more advanced philosophical notions, such as ethics and social mores, with a view to inspiring services that function in a manner that improves the network at large, not just optimizing the results for themselves.

Appendices. Part VIII has appendices on important background topics such as XML technologies and Web standards and protocols. The appendix on XML, in particular, is as extensive as a chapter and includes description of all the key XML technologies that you need to know in order to read this book. It also includes exercises for readers who wish to test their knowledge.

The organization of the book is designed to encourage the building of a series of projects beginning with the most basic applications of the most established standards, and ending with areas where technologies are still gelling.

Note to the Reader

This book brings together a lot of interesting concepts that apply in service-oriented computing. Where possible, we have sought to describe the techniques that these concepts support—in other words, to make the concepts actionable. However, the concepts in many cases are subtle; you must master the concepts before you can be effective with the techniques.

We recommend that you read the text carefully and work through several of the theoretical exercises. If this book is successful, it will have piqued your interest about several topics. If you have a theoretical bent, you will want to pursue deeper results. Virtually all topics discussed here have a lot of depth, and a number of Ph.D. problems lurk within. The book cites the key work for each topic, which will give you excellent starting points.

For those with an interest in practical implementations (and we recommend that even theoreticians work on a few, just to keep grounded), the book provides a fair amount of detail about how various techniques can be realized. It discusses virtually all the key emerging standards for service-oriented computing, and concludes with a discussion of engineering challenges and the emerging methodologies and best practices to address them. In this sense, it is a lot more practical than the typical advanced textbook. If you are reasonably experienced

as a programmer, you can find the few necessary details to get started by downloading the latest versions of the tools from the Internet.

However, as you can well imagine, each standard and tool has a lot of nitty-gritty details. All too often these are not based on any theoretical concepts, but are accidents of history or practical concerns of implementations. This book does not describe such idiosyncrasies. In our own system development, we would learn such details by Internet searches or by trial and error, and promptly forget them as soon as we could! For the bleeding edge standards and tools, there may be no other way; for the older ones you can perhaps find detailed books. In either case, this book will help you master the concepts that will last a long time, rather than details that are lost in the next release.

Lastly, please note that writing the first big book on a wide-ranging new topic is a daunting task. In our combined four decades of post doctorate experience, this is the largest intellectual venture that we have attempted. There will undoubtedly be lots of room for improvement. We welcome your suggestions. But please be kind!

Acknowledgments

A book such as this inevitably describes the work of others. Besides the authors cited within, we are indebted to our colleagues, students, teaching and research assistants, and the audiences of our tutorials at various conferences. In particular, we would like to thank Soydan Bilgin, Michael Maximilien, Yathi Udupi for help with some of the programming assignments, José Vidal for contributing several exercises and Paul Buhler for evaluating an early version of the text in his class. Amit Chopra gave extensive comments on some chapters. Several students, in particular Leena Wagle and Sameer Korrapati, gave useful suggestions on previous drafts. We thank Mona Singh and Hilary Huhns for drawing several figures.

We gratefully acknowledge support from the government and corporate sponsors of our research, which enabled us to develop the understanding of service-oriented computing that is described in this book. Mike has greatly benefited from his association with Ray Emami, Alok Nigam, and their research team at Global Infotek, Inc. Munindar's research sponsors include Cisco, DARPA, Ericsson, IBM, and the National Science Foundation. Mike's research sponsors include the National Science Foundation, DARPA, the U.S. Department of Agriculture, and NASA.

As usual, we are deeply indebted to our families for their patience despite the onerous demands made by our writing.

Munindar P. Singh
Raleigh, North Carolina

Michael N. Huhns
Columbia, South Carolina