
Preface

Motivation for the Book

When confronted with technological advances, we often overestimate their potential in the short term and underestimate what will actually happen in the long term. Short-term hype results from the desires of industry, academia, funding agencies, and financial institutions to make the most of new ideas in as short a time span as possible. Long-term unpredictability reflects the chaotic nature of the process whereby society adopts new technologies.

Web services are no exception to this rule—neither in the current hype nor in the lack of a clear understanding of how they will evolve. This makes it extremely difficult to get a coherent picture of what Web services are, what they contribute, and where they will be applied. Nevertheless, Web services are likely to play a crucial role both in the short and in the long term. In the short term, Web services have the potential to solve many of the interoperability problems that have plagued application integration efforts, as they can be seen as an attempt to standardize the interfaces of middleware platforms. By itself, this would already be a significant contribution. In the long term, Web services could become the basis for a seamless and almost completely automated infrastructure for electronic commerce and wide area, cross-enterprise application integration. Many developers, researchers, and users have been attracted to the notion of Web services precisely because of this highly ambitious goal.

The question is then how to distinguish between what can be done today and what Web services may become in the more or less distant future. This is by no means an easy question to answer. For instance, the rapidly growing body of literature and specifications is no guarantee that the functionality they describe will be supported by current or future products. In many cases, existing specifications ignore the complexities of implementing the functionality they describe and build upon other specifications that have not yet been standardized. More often than not, existing Web services specifications simply state how properties of Web services could be expressed rather than how they

can be implemented, or even whether it is feasible or reasonable to implement them. This makes it difficult to determine the exact state of the art in Web services and to distinguish between reality and long-term research and development objectives.

But if the state of the art is unclear, the future of Web services is even fuzzier, as the uncoordinated proliferation of specifications raises doubts about Web services having a clear direction. Recently, concerns have been raised about the fact that Web services standards are being proposed by different groups that may not have an interest in yielding influence to other organizations, e.g., OASIS (Organization for the Advancement of Structured Standards), W3C (World Wide Web Consortium), and WS-I (Web Services Interoperability Organization). In several instances, such bodies are developing competing standards addressing the same problem. Furthermore, many existing specifications covering different aspects of Web services have not been proposed as open standards. The fact that the authors of those specifications have indicated that they will pursue a *RAND* (reasonable and non discriminatory) licensing policy, thereby implying they will charge fees for use of the standard, can have a dramatic impact on the actual adoption of those specifications. *RAND* licenses will certainly make the adoption process slower and will inevitably trigger the emergence of competing standards that will further fraction the efforts around Web services. Such a process will make it difficult for Web service technology to converge toward a common goal in the long term.

Goals of the Book

This book is an attempt at taking a step back to regain some perspective on Web services as well as to clarify the state of the art and the potential of Web services. This is not an easy task. Web services are still at a very early stage of development and hence there is very little evidence of what their true value will be. To avoid being snared in the low-level and continually evolving details of current specifications, the book focuses on the problems that Web services try to solve rather than on how Web services can be implemented using this or that programming language.

Currently, Web services are no more than the latest attempt to master the complexity of enterprise application integration. Some would even say that Web services are simply a standardization effort that builds upon decades of experience in developing and deploying middleware systems. Indeed, the main line of argument followed in this book is that Web services are, by design, evolutionary rather than revolutionary. This is why the first part of the book discusses conventional middleware and enterprise application integration. Only by understanding how Web services extend current middleware platforms can we hope to understand the forces that are defining and shaping the current efforts around Web services.

Of course, it is very well possible that Web services will trigger a radical change in the way we think about middleware and application integration, or in the way we use the Internet. In that case, Web services will evolve into something yet unforeseen. Before that stage is reached, however, Web services need to provide solutions to the many complex problems that have plagued application integration efforts in the past. In the second part of the book we take a close look at these problems and discuss in great detail if and how Web services can help to tackle them.

Our Approach

Our main concern in this book has been to deliver a coherent, concise, and accurate picture of the state of the art and future evolution of Web services. In pursuing this goal, a key decision was to avoid turning the book into a lengthy commentary on existing specifications spiced up with examples of XML encoding. Since this approach differs from much of the available literature on the topic, the reader is entitled to an explanation on why we have chosen to adopt this particular format.

Our reasons are both pedagogical and practical. First, on the practical side and in terms of how to deal with the available specifications, we have tried to be comprehensive and reflect what is available at the time of publication. Yet, our goal is not to simply describe these specifications. We aim at giving a critical overview of the specifications: what is provided, why it is provided, what is the model behind the specification, what is missing, why it is missing, and which are the available alternatives. Hence, we are not so much interested in illustrating how to implement a “Hello world!” Web service, but rather in understanding the concepts underlying the technology. When looking at a particular specification, both the basic as well as the most advanced ones, our main objective has been to clarify the approach taken by Web services and to examine to what extent this approach can succeed in view of past efforts. Web services as they are now may disappear in a few years, but the problems that Web services are trying to tackle will not go away. Whether the solution is called Web services or something else, what matters is to have a solid understanding of the problems that must be solved and of the constraints associated with realistic solutions. Following this idea, we have organized the book along the problems that need to be solved rather than around the specifications available today.

On the pedagogical side, we have opted for including what we thought would be most useful to a reader who is interested in going beyond XML encoding of existing specifications. Accordingly, we use XML very sparingly and prefer abstract descriptions rather than lengthy XML examples. In our opinion, including pages and pages of XML does not contribute to the understanding of Web services and leaves little room for a critical appraisal of the technology. Moreover, it might even soon become a pointless exercise. If

Web services succeed, their XML representation will be just a low-level detail hidden to all but a few system developers occupied with the Web service infrastructure. At that stage, it will be far more important to have a deep understanding of how and when to use Web services than to be able to write and parse XML by hand.

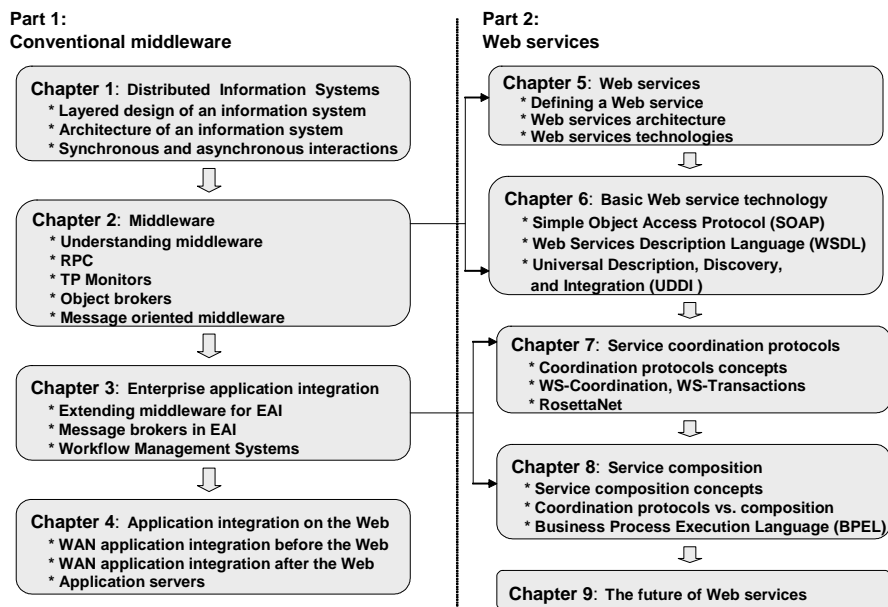
Organization of the Book

Following this approach, the book is organized in two parts. The first part covers conventional middleware with an emphasis on the problems that need to be solved and the approaches followed by different forms of middleware. Specifically, it covers the architecture of distributed information systems (Chapter 1), a variety of middleware platforms (Chapter 2), an overview of the problem of enterprise application integration (Chapter 3), and concludes with a discussion of the basic technologies that enable applications to be integrated over the “Web” (Chapter 4). The second part covers Web services. It starts with a general introduction to Web services (Chapter 5). With the basic aspects covered, the book addresses increasingly more complex aspects of application integration using Web services. Chapter 6 provides a critical overview of basic Web service technologies and standards (such as SOAP, WSDL, UDDI). Chapter 7 discusses coordination protocols, including an in-depth coverage of WS-Coordination, WS-transactions, and RosettaNet. Chapter 8 describes the problem of service composition and some of the existing specifications in this area (focusing in particular on BPEL). The final chapter (Chapter 9) puts into perspective all the ideas discussed in previous chapters and speculates on how Web services may evolve in the future.

Intended Audience

The book is intended for those interested in a critical overview of what Web services are and what they might become. First and foremost, this book was written for those involved in deciding when and how to resort to Web services. We make a distinct effort to relate Web services to the problems and ideas typical of middleware and enterprise application settings. These settings are the most obvious candidates for using Web services, but there is a clear need for separating the hype from the facts. By presenting Web services as what they are (i.e., a natural step in the evolution of enterprise application integration platforms), and by closely relating Web services to existing middleware platforms, we expect that practitioners will find this book a valuable reference when evaluating the potential of Web service technology.

For this same reason, the book can be of help in establishing research priorities in this area. We certainly hope that it will bring some clarity in terms of what can be done today with Web services and where more work



is needed in order to have the necessary basis for pursuing more ambitious endeavors. We believe that Web services have, to a certain extent, become an outlet for ideas that proved impractical in the past. It is not at all clear that Web services will make these ideas any more feasible than they were before. Hence, although proposals around topics such as the Semantic Web, dynamic marketplaces, and automatic contract negotiation and enforcement are all very appealing, we do not discuss them in great detail. The reason is that all these ideas heavily depend on Web services not only being widely adopted but also evolving far beyond the current state of the art. Clarifying the current state of the art is a difficult enough task as it is without also engaging in speculative exercises. We hope that by clearly pointing out the limitations of Web services technology we have experienced in industrial settings, we will motivate researchers to tackle some of the many open problems that still remain in this area. That will create a solid foundation for further work exploring more visionary issues.

Last but not least, we have made a considerable effort to organize the book in such a way so as to facilitate it being used in an advanced course on distributed information systems. It is certainly not meant to compete with books that cover specific forms of middleware. Its purpose is instead to provide a unified view that confers meaning to what is typically presented to students as a conglomerate of separated and independent issues. The book can be used as the main reading source for a course on middleware and enterprise application integration, with more specific details provided directly by looking at the wealth of product information and white papers available on almost every

aspect of middleware. Based on our experience, students learn much more and understand these complex issues much better when the different topics are presented as an evolutionary process rather than as a random collection of technologies and implementation techniques.

Guidelines for Teaching

The book has been designed so as to provide the basic reading material that can serve as the backbone of an advanced course on distributed information systems, covering from conventional middleware platforms to Web services. Based on the experience gathered at ETH Zürich in the past five years, where such a course has been offered on a yearly basis by one of the authors, the main difficulty when covering this topic is the constant change in the technology. A very important aspect of the book is the evolutionary perspective taken when discussing middleware, enterprise application integration, and Web services. This perspective is often lost among the details of how each individual middleware platform works. However, without understanding this evolution it is very difficult to make sense out of the developments in this area. The advantage of this book is that rather than presenting topics in isolation, it always frames each technology in relation to previous systems and with a view to motivate the systems that followed. By focusing on the problems to solve and the constraints imposed on the potential solutions to the problems, students can gain a deeper understanding of middleware and be much better prepared for the rapid changes that will continue to occur in this area.

To achieve this effect, we recommend organizing such a course as follows. Assuming a 14 week course, with 2 hours of teaching and 2 hours of exercises a week, the material should be divided into two parts corresponding to the two parts of the book. Depending on what aspects the course wants to emphasize, the assignment of time to each topic can vary. A good first approximation is to devote seven weeks to the first part of the book: middleware and application integration. The first two weeks should cover Chapter 1 of the book. The next three weeks should devote two hours each to selected middleware platforms, covering their purpose, architecture, underlying technology, precursor systems, limitations, and leading up to the next natural step in improving such platforms. A good selection is TP and object monitors, workflow systems, and message oriented middleware (Chapters 2 and 3). With this background, students can then move on to the challenges imposed by the Internet and the need to interconnect heterogeneous middleware platforms. This material should be covered in another two weeks using Chapter 4. The remaining seven weeks should be devoted to analyzing Web services in detail, starting by discussing SOAP, UDDI, and WSDL (one week each), and then moving on to more advanced topics such as coordination protocols and composition (four weeks in total to cover the basic problems and existing solutions).

In this program, the book provides the glue that keeps the course together and relates the different parts of the lecture. It includes enough material to cover all these topics without having to resort to other textbooks. A good working methodology is to ask students to read the corresponding chapter before the lecture, and use the lecture to go into the details of each particular topic. Lectures and the chapters of the book can be easily complemented with one or two articles covering concrete products or technologies in more depth, so as to give students a solid link to actual systems. The material in the book is sufficiently self contained to let students read it on their own while devoting the lectures to pursuing certain topics in more depth (e.g., distributed transaction processing or concrete programming examples in one of the middleware platforms). The book also includes a companion Web site (<http://www.inf.ethz.ch/~alonso/WebServicesBook>) where students and professors can find additional reading material and presentations on selected topics.

In terms of exercises, in a 14-week course, a useful and very rewarding approach for the students is to develop, step by step, a complex application integration project. As an example of what could be done, at ETH Zürich we organize the exercises as a series of three projects that build upon the previous one. The first project involves developing clients for well defined services. Depending on the emphasis of the course, one can use conventional middleware services like stored procedures or RPC services; it is also possible, however, to start directly by building clients for well-established Web services. This gives students a feeling for how services are used and what basic problems need to be addressed. It also allows for the introduction of important notions such as response time, throughput, and benchmarking. The second project asks students to implement their own service by either developing it from scratch or by wrapping an existing service to make it conform to a given interface type (e.g., wrap a set of Web pages as a Web service). This gives students a taste of the problems related to interface mismatches, session and state information, persistence, connection pools, and a wide range of other important issues. The project can be used to illustrate techniques that vary from the most prosaic ones such as screen-scraping to advanced research topics such as adaptive middleware and run-time modification of information infrastructures. The third and final project asks students to use a complex middleware platform (e.g., a workflow tool) to compose clients and services into a sophisticated business process (which again can be done using conventional services or Web services, depending on what aspects the course wants to emphasize). Based on the business process developed, students can be asked to perform exhaustive execution analysis, suggest optimizations, and critically evaluate the tools at their disposal for developing the process. If services to be composed are dynamically selected from a UDDI registry, then students can also appreciate the opportunities and the difficulties of performing dynamic binding, even in simple settings. These exercises, when combined with the use of the book in the lecture, give students a very good basis for appreciating the

advantages of using middleware and the implications of increasing complexity as they need to rely on additional layers of middleware to implement the projects.

Acknowledgements

Books are born as a result of a coordinated and focused effort by the authors. However, the knowledge behind the ideas presented is often the result of a long learning process whereby ideas discussed with many people mature to the point in which it is possible to give them concrete shape. This has certainly been the case for this book. In this regard, we would like to thank the former and current graduate students of the Information and Communication Systems Research group at ETH Zürich for many useful discussions on the topics covered in the book and for useful input on how to improve the presentation of some of the material. We are also grateful to our colleagues in HP Labs for the many useful and enlightening discussions on Web services. They helped us to unveil which are the key issues behind Web services as well as how to present them. We also thank colleagues in the HP business units for sharing their insights about industrial applications of Web service technology and about the problems that Web services can and cannot solve in such scenarios.

Special mention should be given to the editors of this book series, Stefano Ceri and Mike Carey, as well as Barbara Pernici, who acted as a reviewer. Their comments were extremely helpful in making this a better book. We would also like to thank Ralf Gerstner of Springer-Verlag, who showed infinite patience on the face of many delays on our side and was always very helpful during the editing process. We thank Mike Lemon, Rena Takahashi, and David Pietromonaco for their gracious help with the indexing and proof-correcting process. Finally, we thank Heidi Schümperlin and Dan Muntz for proofreading several chapters of this book.

Zürich, Switzerland,
Palo Alto, California,
Palo Alto, California,
Palo Alto, California,

Gustavo Alonso
Fabio Casati
Harumi Kuno
Vijay Machiraju
July, 2003