Praise for
100 SOA Questions: Asked and Answered

“In this book, Kerrie and Ali truly capture the essence of SOA and its benefits to your Business and IT Deployments. Businesses today are desperate to be more nimble and innovative while reducing costs—a tricky proposition. SOA enables companies to reach these goals by tightly aligning the business and IT around the business processes, breaking those processes into reusable business and IT services, and allowing the underlying business and IT infrastructure to be more nimble in supporting the business goals. Effective SOA deployments also lay the groundwork for Cloud formations that can deliver even greater flexibility and cost saving to the business. This book clearly shows that a visit to the land of SOA and Cloud should be on every CxO’s bucket list for their own business.”

—Daniel A. Powers, VP Amazon Web Services Sales

“100 SOA Questions is a must-read for business and IT users who are interested in improving business innovation and agility. Based on their real-world experiences from hundreds of global customer engagements, Kerrie Holley and Ali Arsanjani do an outstanding job of explaining the multiple facets of SOA and providing a prescriptive approach to help readers incrementally unlock value from rigid business processes and antiquated application silos.”

—Manoj Saxena, IBM Executive and Tech Entrepreneur

“In order for IT systems to be successful in today’s world, they must be correct, complete, and extensible. For a long time we have had systems that are point-in-time complete, partially correct, and minimally extensible. Kerrie and Ali get to the heart of the matter when they speak to us about the DNA of a Service-Oriented Architecture. It has been my experience in working with them that their focus in defining granularity, composability, and loose coupling from the viewpoint and with the help of the business stakeholders is much more than the cliché, ‘aligning IT with the business,’ would lead you to believe.”

—Manny Bonet, Software Architect

“It is the underpinning infrastructure of a system that determines ultimate flexibility and the ability to scale change to keep pace with rapidly changing global markets. Through Kerrie and Ali’s simple, clear, and comprehensive articulation of the Service-Oriented Architecture approach, I can see that systems must pass through the maturing that SOA brings in order to leverage design methodologies of the future. Those who don’t will be destined to revisit SOA. Great work, gentlemen, in showing us, in such a real world way, the differentiator needed to fight the fight in an ever-leveling technology playing field.”

—Robert Mansell, Vice President, Payments & Settlement Systems, PayPal
“This book, written by prominent IT architects Kerrie Holley and Ali Arsanjani, shows a deep and clear understanding of the SOA paradigm for real business starting from the ground up. It will add a strong contribution both for technical and business people in terms of mutual understanding of SOA implementation to solve real life problems of today’s enterprise. This book should be ‘A must’ to have it at each IT architect’s desk.”

—Alexei Chirokikh, Ph.D., CIO, Gazprombank (OJSC)

“In 100 SOA Questions, Kerrie Holley and Ali Arsanjani have presented an up-to-date, vendor-independent explanation of Service-Oriented Architecture suitable for IT executives, as well as the broader audience of business stakeholders and architects. Sufficiently detailed while not being overly technical, 100 SOA Questions is a book I can heartily recommend to participants in the Licensed ZapThink Architect course.”

—Jason Bloomberg, Managing Partner, ZapThink LLC

“Having known the authors for the better part of the past decade, I can think of no better stewards for all that is pure and good with SOA. Through absolute dedication to their craft, Holley and Arsanjani have been at the forefront of SOA IP, crafting best practices, tools, methodology, and thought leadership in ways both theoretical and practical. This book does the impossible by giving you a glimpse of the magnitude of sheer brain power focused on an incredibly relevant and important topic to both Business and IT. Use it wisely.”

—Michael Liebow, Board Member, former CEO and Tech Entrepreneur
Contents at a Glance

Preface ........................................... xvii
Introduction .................................... 1
Chapter 1  SOA Basics  ......................... 5
Chapter 2  Business  ............................. 19
Chapter 3  Organization ....................... 45
Chapter 4  Governance ......................... 71
Chapter 5  Methods  ............................. 95
Chapter 6  Applications ....................... 119
Chapter 7  Architecture ....................... 147
Chapter 8  Information ......................... 173
Chapter 9  Infrastructure ..................... 191
Chapter 10 Future  ............................. 219
Index ............................................. 231
Contents

Preface .................................................. xvii

Introduction .......................................... 1
About This Book ....................................... 2
Intended Audience .................................. 2
How This Book Is Organized ...................... 3

Chapter 1 SOA Basics ................................. 5
SoA Basics: Q&A ....................................... 5
1. What Is SOA? ....................................... 5
2. Is SOA an Architectural Style? ................. 7
3. What Are the Fundamental Constructs
   (the DNA) of SOA? .............................. 9
4. What Is the Difference Between a Web
   Service and an SOA Service? ............... 14
5. What Makes a Project an SOA
   Implementation? ................................. 15
SoA Basics: Key Concepts ......................... 16

Chapter 2 Business ..................................... 19
Business: Q&A ......................................... 21
6. Why Should Business Stakeholders Care
   About SOA? ..................................... 21
7. How Should SOA Be Sold to the Business or
   Business Stakeholder? ....................... 25
8. What Is the Return on Investment (ROI)
   of SOA Adoption? ............................. 28
9. How Should the Business Measure the
   Effectiveness of SOA? ....................... 29
10. What Are the Criteria for Selecting a Project
    for SOA Adoption? ........................... 33
11. What Is Flexibility and How Does SOA
    Deliver on This Promise? ................... 34
Chapter 4  Governance ................................. 71
Governance: Q&A ................................. 72
27. What Is SOA Governance? .......... 72
28. How Does an Organization Get Started with SOA Governance? .......... 75
29. What Is the Role of Change Management? .......... 79
30. Does Implementation of SOA Tools and Infrastructure Equate to SOA Governance? .......... 81
32. Does SOA Require Centers of Excellence, Architecture Boards, or Design Boards? .......... 84
33. Why Do Organizations Need to Focus on SOA Governance When There Is an Effective Enterprise Architecture Activity? .......... 87
34. Is SOA Governance Required for SOA Projects to Be Successful? .......... 89
37. What Are Common Pitfalls of SOA Governance? .......... 92
Governance: Key Concepts ................. 93

Chapter 5  Methods ................................. 95
Methods: Q&A ................................. 96
38. Should an Organization Continue to Use Agile or Object Development Methods for SOA Projects? .......... 96
41. How Should Services Be Identified or Specified to Maximize Reuse? .......... 103
42. How Should the Granularity of a Service Be Determined? .......... 106
Chapter 7  Architecture ........................................ 147

Architecture: Q&A ........................................... 149

61. How Does Architecture Change as a Result of SOA Adoption? ............ 149

62. How Does SOA Differ from Earlier Approaches, such as DCE or CORBA? ..... 156

63. How Do Web Services and SOA Differ? .... 157

64. Is SOA Too Complex and Enterprise-Level Only? ......................... 158

65. How Do Interfaces and Contracts Differ? .................................. 160

66. Should Applications Choose WSDL or REST? ............................. 162

67. What Is the Relationship Between Enterprise Architecture and SOA? ........ 165

68. How Do EAI, SOA, and SOI Differ from One Another? .................. 167

69. What Is the Role of Standards in SOA Implementations? ................... 168

70. How Should Standards Be Applied to Enable Successful SOA Implementations? ..... 169

71. What Are the Common Pitfalls When Adapting an IT Architecture for SOA? .... 170

Architecture: Key Concepts .................................. 172

Chapter 8  Information ....................................... 173

Information: Q&A ........................................... 174


73. What Are Information Services? ........................................ 175

74. How Are Information Services Classified? .................................. 176

75. Do Information Services Differ from Other Services? ....................... 178

76. How Should Information Services Be Identified? .......................... 180

77. When Should Information Services Perform Create, Read, Update, and Delete (CRUD) Operations? ................................. 181
78. Are Enterprise Information Models Required for Effective SOA Implementations? .......... 182
79. What Is a Canonical Message Model? ................. 184
80. How Should a Canonical Message Model Be Created? .................. 186
81. Can SOA Improve Data Quality? ................. 187
82. What Are the Common Pitfalls with Information Architecture and SOA? .......... 188

Information: Key Concepts ........................................ 189

Chapter 9 Infrastructure ........................................ 191

Infrastructure: Q&A ........................................... 193
83. What Are the Building Blocks of an SOA Infrastructure? .......... 193
84. What is an Enterprise Service Bus? ................. 199
85. What Are Best Practices for Creating the SOA Infrastructure? .................. 200
86. What Makes an Enterprise Service Bus Different from Integration Technology? .... 201
87. How Does an ESB and Registry Relate? ................. 203
88. How Does an SOA Infrastructure Support Events? .................. 204
89. How Does the SOA infrastructure Evolve to Realize the Increased Loose Coupling? .... 205
90. How Does SOA Infrastructure Support Policy Management? .................. 209
91. How Is Management of the Infrastructure Affected by SOA? .................. 212
93. What Are the Common Pitfalls in Creating an SOA Infrastructure? ................. 214

Infrastructure: Key Concepts ........................................ 217
Chapter 10  The Future of SOA .......................... 219

Future: Q&A ............................................. 220
  94. Is SOA Dead, Stagnant, or Moving Forward in its Adoption? ............... 220
  95. What Is the Future Trajectory of SOA? .................................. 221
  96. What Are Context-Aware Services? ...................................... 224
  97. What Role Does SOA Play in Embedded or Real-Time Systems? .......... 225
  98. What Is the Relationship Between Event-Driven Architecture and SOA? .. 225
  99. How Does the Slow Maturation of Standards Affect the Future of SOA? .... 227
 100. Do WOA and Web 2.0 Affect the Future of SOA? ........................ 228

Future: Key Concepts ..................................... 229

Index ..................................................... 231
Almost two decades ago, I completed a project to develop and deploy a teller and sales application for a large U.S. bank. Enhanced business capabilities, technology upgrades in the branches, and a pending bank merger were the business drivers. Some months after the production roll-out, as the Chief Architect, I was invited to a meeting with the Vice Chairman of the Retail Banking who wanted to understand my perspective on how the bank should address challenges in meeting future demands that required extending the reach of the teller and sales platform functionality to other parts of the bank.

The Vice Chairman was responsible for all retail functions of the bank and expansion was hot on the agenda. The bank was growing, entering new market places, acquiring banks, opening branches, and rapidly attracting new customers. We sat down and discussed cross selling, expansion goals, and the need for several parts of the Bank such as credit card processing, wholesale banking, and loans, to be able to access and use functionality contained in the teller application we had just built and deployed. Obtaining customer information, account balance inquiries, and address updates were just a few of the basic pieces of functionality needed by these other departments but there were more complex pieces of business functionality required, too.

When wholesale banking or credit card processing needed to access data or functionality in the teller system, they needed to go through a development cycle that necessitated waiting in a queue with others, whereby the IT department could prioritize and satisfy the multiple requests and requirements. The Vice Chairman expressed this current situation as a problem; it impacted the bank’s capability to get more products out the door faster and his ability to meet sales and revenue targets. He asked two questions: How can we do this better and how can the bank provide access to previously built and deployed business functionalities to other parts of the bank without going through IT development queues? Addressing this question
and others by senior business executives has been top of mind for me for two decades.

Over the last decade, I have met with corporate executives from hundreds of companies across the world whose enterprises are characterized by disparate and siloed systems and applications; horizontal integration is the goal as businesses seek greater agility in the global marketplace. Corporate managers are asking how do to make the IT system more flexible so that it is easy to connect across the enterprise and so it is inexpensive in both time and cost. The story of the bank occurred two decades ago, but I find CEOs and other corporate executives asking this same question over and over again, decades later. Everyone is searching for flexibility as competition intensifies. Everyone sees this albatross around their neck getting uglier and negatively impacting goals for growth and limiting the responsiveness and agility required as the cost of maintaining, integrating, and supporting systems is rising. Less capital is available for innovation, changing the business, and delivering new capabilities.

Just a few years ago, I met with a corporate manager responsible for a payments business. His frustration was apparent as we discussed the need to change his three-year-old IT system to accommodate new channels (phones, kiosks, and other mobile devices) and new market segmentations. He was frustrated because although he was not a technologist or software engineer he knew something was not right. He wanted to know why after millions of dollars of investment in a creating a new payment system, built three years earlier, it was not easy to change his payment system to accommodate small and medium businesses or to allow access to payments using handheld devices. He asked this question because his payment system was built with modern software engineering best practices yet flexibility was evasive: adding new channels and new customer segments would take too long and cost too much money as if he were building the system from scratch versus just changing the system. I responded and the short answer is that applying best practices and modern system engineering practices is not sufficient if agility is the goal. I further stated that there is a considerable amount of data that shows this problem is not isolated that most applications become difficult to change within 3 to 5 years after the first production deployment.
Recently, I was in Mexico City working with a large logistics company. It was just finishing an 18-month project to reengineer a core IT system that was no longer responsive to the business. The new system was engineered like the bank system two decades earlier, with the best software engineering practices and tools available. I was asked if this new system would suffer the fate of past systems in its capability to be responsive. That is, would this system become brittle in the future and if so, why? Would this new system be built for change such that flexibility was an attribute of the system and not a platitude? Again I answered no, stating that applying best practices alone will not achieve the goal of agility. I know this is true because his team and teams just like his around the world have been using modern and best practices of software engineering for years with the same results. The result is that three to five years after the system has been deployed it is difficult to change, and is expensive in time and money.

It is not only the commercial world that sees a problem but the public sector. We have met with various public sector organizations over the years and my interactions confirm that they are confronted with the same challenges we see in the private sector. Public and private sector managers see the rising cost of support, integration, and maintenance of the systems as a ball and chain that is a huge drag on cost reduction and as a result, it puts a limit on monies available for creating new capabilities in the theater as the available dollars are limited.

It is these questions and their answers that prompted us to write this book about service-oriented architecture (SOA). This is not a technology book, but a book for technologists and business stakeholders. We hope to demonstrate, that SOA and service-orientation in general, is not solely a technology play but a paradigm and architecture that calls for business and IT collaboration and when understood and applied, it can change the course of your business, where flexibility and lower total cost of ownership become realities.

Total cost of ownership and flexibility are different sides of the same coin. There is less flexibility when funds are not available to spend or when providing new capabilities is constrained because resources are consumed in integration, maintenance, and support.
Flexibility is evident when the business, not IT, has the power to deploy new business features without IT development queues or when new capabilities can be provided in weeks or months instead of years, and when two or more capabilities can be composed at will to create a new, enhanced capability that directly supports business drivers and alleviates painpoints.

If we make the right choices, we will have a chance to escape from the boxes that frustrate us today. The escape will not be easy—we will be constantly challenged to question conventional assumptions and comfortable practices. Many will not even see the opportunity. They will continue to remain closed in the boxes that make every day more frustrating. Some will see the opportunity but will either try to move too quickly or fail to stay the course. They will blame the technology for its failure to produce results. For those few who succeed, the rewards will make the journey well worth the effort.

—John Hagel III in Out of the Box

Our choice as managers, leaders, or architects is to seize the opportunity and release ourselves from self-imposed boxed thinking because “if you don’t change anything, nothing changes.” We can make business flexibility a reality with IT support but it requires a vision, a strategy, execution of the strategy, and most importantly, staying the course. The strategy must be a living plan accompanied with a evolving roadmap that can be implemented, monitored, and measured. It requires you take incremental steps that together bring about change: incremental and quantum leaps over time.

If you want to get out of the boxes that John describes, enabling your IT systems to be engines of innovation, this book will be of value. If you are responsible for strategy in the organization and need to link that strategy to an IT strategy to make your IT systems and infrastructure capable of supporting a rapidly changing landscape or business model, you should read this book. If you are tired of reading about platitudes and seek guidance about how to achieve business flexibility through the adoption of SOA, you will obtain value in reading this book. This book is not about a technology change; it’s about a business journey with IT, where SOA is both the enabler and the catalyst.
This book is different than other books on SOA as content is organized into 100 questions and answers. Feel free to go directly to the chapter that most interests you or go directly to a question for which you would like an answer. Visit www.100Questions.info and submit any questions that remain unanswered.

Kerrie Holley
Acknowledgments

We want to thank the countless number of colleagues and clients who have helped shape our thinking. The problems they’ve presented and perspectives they’ve shared have helped us acquire valuable knowledge.

Although the list is long, we do want to acknowledge specific individuals whose insights and experience have allowed us to learn so much. Thank you to Sarah Garrison, Doug Hunt, and Robert LeBlanc, who got us started on our SOA journey. A thank you to George Galambos and Raghu Varadan, who taught us so much. Thank you to Ray Harishankar, Liang-Jie Zhang, Kishore Channabasavaiah, Shankar Kalyana, Rob High, Rolando Franco, William A. Brown, Abdul Allam, Chris Venable, Siddharth N. Purohit, Brian Paulsen, Jenny Ang, Julian Petriuc, Claudio Cozzi, Richard Disney, Mamdouh Ibrahim, Olaf Zimmermann, Robert Laird, Arun Thakore, Tony Cowan, Rick Robinson, David Janson, Raphael P. Chancey, Manish Vipani, Manoj Saxena, Isabel Van Mele, Ko-Yang Wang, Marc Fiammante, Denise Hatzidakis, Alexei Chirokikh, Don Ferguson, Pamela K. Isom, Manny Bonet, Waseem Roshen, Chuck Gupta, Rachel Reintz, Boris Veroeveren, Jose Loya, Geert-Willem Haasjes, Luba Cherbakov, Ian Turton, and Shuvanker Ghosh, all of whom contributed immensely to our knowledge base.

This book would not be possible without the people at Pearson Education. We thank Ginny Bess Munroe for her encouragement and superb skills as the development editor. Thanks also to the rest of the Production team. Robert Laird and Chris Venable did a remarkable job of reviewing our manuscript and provided valuable insight as to which questions really mattered.

This book would not be possible without the support, patience, project management skills, guidance, perseverance, and leadership of our editor, Katherine Bull. Katherine worked to cultivate the idea for this book, labored to sell the idea, and encouraged us as writers. Katherine showed a tremendous amount of patience, provided endless hours of support, and without her confidence and project management, this book would never have come together.
About the Authors

**Kerrie Holley** has a wealth of experience in application development, software engineering, systems engineering, IT consulting, and enterprise architecture. Mr. Holley has operated as Chief Architect, Strategist, Consultant, and Designer on more than fifty SOA projects. In his current role, he oversees hundreds of SOA projects in their technical direction, strategy, and successful deployment. Mr. Holley's current focus is on the convergence of business rules, business process management, analytics, and SOA in making businesses more agile. Mr. Holley holds several SOA patents and has a BA in mathematics from DePaul University and a Juris Doctorate degree from DePaul School of Law. Mr. Holley has worked in a senior capacity for several companies, including Bank of America, Tandem Computers, Ernst & Young and is currently an IBM Fellow.

**Dr. Arsanjani** is a rare mix of industry hands-on consulting and academic research that he leverages in his Chief Technology Officer role as advisor to high-profile companies. Through his experience as strategist, consultant, and architect, he has helped companies achieve business performance through leveraging and changing IT. His current area of focus is to enable companies to achieve higher levels of business performance and enable them to optimize their business through the agility gained in concert with IT and business operations. Ali Arsanjani has chaired standard bodies such as The Open Group and is responsible for co-leading the SOA Reference Architecture, SOA Maturity Model, and Cloud Computing Architecture standards. In his role as Chief Architect, he and his team specialize in harvesting and developing best-practices for the modeling, analysis, design, and implementation of SOA and Web Services on hundreds of projects.

He is a hands-on, sought-after architect around the world on large SOA projects, and he is the principal author of the industry first Service-Oriented Modeling and Architecture (SOMA) method for SOA. His work on variation-oriented analysis allows companies to
build less software but achieve higher gains, and his patterns for service-oriented software architecture combine SOA with business process management, business rules, and analytics to achieve higher levels of maturity for organizations.
Introduction

You will never stub your toe standing still. The faster you go, the more chance there is of stubbing your toe, but the more chance you have of getting somewhere.

—Charles Kettering

A myth abounds that ostriches hide their head in the sand when frightened, and that same behavior is often attributed to anyone who foolishly ignores problems while hoping those problems magically vanish. The ostrich does many things, but hiding its head in the sand is not one of them. IT departments do many things, and hiding their heads in the sand is unfortunately one of them.

IT departments face many challenges, one of the biggest being that they spend a substantial part of their resources on running the business rather than changing the business. That is, they spend substantially more money on maintenance than on innovation. And this particular problem is getting worse and cannot be improved upon by inertia or standing still. Instead, change is required, and this book covers how to adopt service-oriented architecture (SOA) as a change agent (and deal with the inevitable stubbed toes along the way).

Several forces and events contribute to inefficiencies and higher costs for many IT departments: acquisitions, fiefdoms, technology zealots, infrastructures built over time without a roadmap, financial measurements that incent IT to be cheaper rather than more effective, poor application portfolio management, and ineffective architectural policies. The effects of such inhibit IT departments’ ability to accelerate or improve time to market for new business capabilities. SOA can make a significant and positive difference, but you must
understand that this is a process, a true journey. After all, technology implementation by itself does not guarantee business agility.

**About This Book**

According to your needs and familiarity with SOA, you can use this book as a textbook, quick reference guide, or a tutorial. You do not need to read the book sequentially. In fact, you can start at any chapter and even jump between chapters to learn about the areas that interest you, and you can do so without losing context/continuity.

This book inventories challenging questions from business and IT stakeholders and provides corresponding answers. Where appropriate, the answers are prescriptive. Although, in this book, we attempt to exhaustively anticipate your questions and provide readily understandable answers, we also prove an outside forum for you to ask, in your own words, any questions we might have failed to address. You can access this forum at www.100Questions.info. We invite you to continue our SOA dialogue there.

Questions are numbered sequentially from 1 to 100 throughout the entire book.

**Intended Audience**

This book is intended for executives, managers, IT architects, business architects, business analysts, line-of-business (LOB) managers, and students who want to understand the basic and complex concepts of SOA and the business and technology rationales for developing and implementing SOA.

For example, readers might include the following:

- LOB/product managers who wonder what SOA has to do with the business
- Business executives/stakeholders who want to know how to make new development projects have built-in flexibility and sustained agility
- Business/IT stakeholders who want to know what they need to do differently to make systems more agile
• Architects tasked with a transformation initiative or project and who want to understand how or whether SOA can be applied
• Architects who want to understand how to build a system for change so that the application is not difficult to change three or five years after its initial production deployment
• Enterprise architects who want to be more effective at creating adaptive and usable enterprise architectures
• Students and others who want to know the facts about SOA

How This Book Is Organized

This book is organized in such a way that you can browse and easily find topics of interest. The chapters themselves address specific domains of concern about SOA in the business/IT world, as follows:

Chapter 1, “SOA Basics”—This chapter defines SOA and service orientation. It also examines several myths and misconceptions that prevail in the marketplace about SOA.

Chapter 2, “Business”—This chapter examines the forces that drive businesses in all industries to become more agile, adaptable, responsive, resilient, and profitable. The chapter covers how to address the business value of SOA, sell SOA to business stakeholders, and the return on investment of SOA. This chapter also covers business process management (BPM).

Chapter 3, “Organization”—This chapter discusses the technology and organizational roadblocks that impede forward motion in SOA adoption. The chapter also examines the relationships between business and IT and how they collaborate for SOA.

Chapter 4, “Governance”—This chapter addresses the hot topic of governance, including why it is important and its impact on achieving business results with SOA adoption. The chapter answers questions about governance, adoption steps, how to get started, and how to communicate the SOA journey.

Chapter 5, “Methods”—This chapter addresses questions on methods, both business and system. Service granularity and identification of services are also covered in this chapter.
Chapter 6, “Applications”—This chapter distinguishes between applications and composite applications and identifies what changes about applications as a result of SOA.

Chapter 7, “Architecture”—This chapter considers architecture from various views (for example, application architecture, integration architecture, and enterprise architecture) and discusses the impact the various views and their interrelationships with SOA.

Chapter 8, “Information”—This chapter covers how information, data architecture, and management support SOA. Concepts addressed in this chapter include information as a service, canonical models, and message models.

Chapter 9, “Infrastructure”—This chapter covers the middleware and operating environment required for SOA. Topics addressed include the enterprise service bus, registries, operational impacts of SOA, and the required operational maturity of the infrastructure to support SOA.

Chapter 10, “Future”—The last chapter of this book deals with the future of SOA. Where is it and where is it going? Is SOA dead? Isn’t cloud computing the replacement and our next horizon? What is meant by context-aware services?

At the end of each chapter, we address common pitfalls and how to avoid them. After all, before organizations can take preemptive measures to avoid missteps in SOA adoptions and initiatives, they must understand where others are making mistakes.
SOA Basics

Delusions, errors, and lies are like huge, gaudy vessels, the rafters of which are rotten and worm-eaten, and those who embark in them are fated to be shipwrecked.
—Buddha

Service-oriented architecture (SOA) is defined in a number of ways, but not all definitions are equal, and not all definitions are complete. Instead of just providing another definition of SOA, this chapter describes the basic building blocks of SOA and looks at the value proposition of SOA from key stakeholder perspectives. Besides covering the basic building blocks of SOA, its DNA, and the value propositions of adopting SOA and its ultimate utility, this chapter describes what makes an implementation an SOA deployment. Specifically, this chapter addresses the following questions:

1. What is SOA?
2. Is SOA an architectural style?
3. What are fundamental constructs (the DNA) of SOA?
4. What is the difference between a Web Service and an SOA service?
5. What makes a project an SOA implementation?

SOA Basics: Q&A

1. What Is SOA?

Numerous vendors, application providers, system integrators, architects, authors, analysts firms, and standards bodies provide
definitions of SOA. The definitions of SOA are diverse. Most are complementary and do not conflict with each other. SOA has a variety of definitions because the definition is often tuned to a specific audience, as explaining SOA to a CEO is different from explaining SOA to a programmer. The term *service orientation* is often used synonymously with SOA, but just like SOA it has a wide range of interpretations. Service orientation is broader and represents a way of thinking about services in the context of business and IT. This book makes no distinction between SOA and service orientation and in some cases may use the two terms synonymously.

An agreed-upon definition for SOA eludes the industry. Anyone reading Wikipedia’s definition page for SOA will see the challenges of trying to gain consensus on an SOA definition. Standards bodies, the OASIS group, and the Open Group have provided complementary but different SOA definitions. Presented with a blank sheet of paper, an artist sees a canvas. A poet might fill it with verse. An engineer seize the opportunity to make a paper plane. Kids may see it as a future pile of spit wads. SOA is that blank sheet of paper.

To the *chief information officer* (CIO), SOA is a journey that promises to reduce the lifetime cost of the application portfolio, maximize *return on investment* (ROI) in both application and technology resources, and reduce lead times in delivering solutions to the business.

To the business executive, SOA is a set of services that can be exposed to their customers, partners, and other parts of the organization. Business capabilities, function, and business logic can be combined and recombined to serve the needs of the business now and tomorrow. Applications serve the business because they are composed of services that can be quickly modified or redeployed in new business contexts, allowing the business to quickly respond to changing customer needs, business opportunities, and market conditions.

To the business analyst, SOA is a way of unlocking value, because business processes are no longer locked in application silos. Applications no longer operate as inhibitors to changing business needs.

To the chief architect or enterprise architect, SOA is a means to create dynamic, highly configurable and collaborative applications built for change. SOA reduces IT complexity and rigidity. SOA
becomes the solution to stop the gradual entropy that makes appli-
cations brittle and difficult to change. SOA reduces lead times and costs
because reduced complexity makes modifying and testing applica-
tions easier when they are structured using services.

To the IT architect, SOA is the architectural solution for integrat-
ing diverse systems by providing an architectural style that promotes
loose coupling and reuse. Many IT architects think they have seen
this style before with earlier architectural initiatives such as DCE, the
Distributed Computing Environment, and CORBA, the Common
Object Request Broker Architecture.

To the developer, SOA is a programming model or paradigm
where web services and contracts becomes a dominant design for
interoperability. It is a web service when it uses a Web Service
Description Language (WSDL) or equivalent specification for
describing the service. Web services enable organizations to commu-
nicate information, using messages, without intimate knowledge of
each other’s IT systems.

Delivering on the promises of SOA (improved business agility,
maximized ROI, reduced IT complexity and rigidity, reduced costs,
reduced lead times, reduced risk, new opportunities to deliver value,
increased participation in value networks, and incremental imple-
mentation) requires you take a holistic view of SOA. If we limit the
view of SOA to a single stakeholder (e.g., IT architect, developer, or
business analyst) the benefits will not accrue because SOA then just
becomes one in a long list of overhyped technologies rather than a
novel approach to building flexible business solutions.

2. Is SOA an Architectural Style?

SOA is often seen as an architectural style that has been around
for years. Figure 1.1 shows the architectural style of SOA. In this sce-
nario, a service consumer invokes or uses a service. The service con-
sumer uses the service description to obtain necessary information
about the provider service (e.g., account service) to be consumed.
The service description provides the binding information so the con-
sumer can connect to the service, and the description identifies the
various operations (e.g., open or close account) available from the
provider service. A broker can be used to find the service using a registry that houses information about the service and its location.

In Figure 1.1, it is difficult to determine how the architecture style of SOA enables the strategic benefits of SOA, such as lowering the lifetime cost of an application or bringing faster time to market or making applications resilient to change. SOA as an architectural style often makes an SOA project solely an IT endeavor where the strategic business benefits of SOA no longer become the focus or measured outcomes. Benefits of process flexibility, time-to-market savings, lower costs, and others can be achieved with SOA, but only if we holistically adopt all stakeholder views of SOA and its application and pursue SOA adoption accordingly. When pundits, architects, consultants, or executives define SOA as a pure technology play or as solely an architectural style, they relegate it to the realm of IT science projects, overhyped technologies, and a marketing strategy rather than a novel approach to building flexible business solutions.

An understanding of SOA is enhanced with the next question and answer. By looking at the SOA building blocks of SOA, you can gain a fuller understanding of what SOA is and how to realize its promised benefits.
3. **What Are the Fundamental Constructs (the DNA) of SOA?**

The most basic construct or building block of SOA is a service. Software engineering over the years has evolved from procedural to structured programming to object-oriented programming to component-based development and now to service-oriented. Figure 1.2 illustrates the different levels of abstraction from objects to services. Each evolution of abstraction builds on the previous, and SOA embraces the best practices of object and component development.

![Increasing Levels of Abstraction](image)

**Figure 1.2 Levels of abstraction**

To see architectural style of SOA, refer to Figure 1.1. That illustration shows the fundamental constructs of SOA, such as the service consumer and the service provider and their relationship. The consumer invokes a service, the business functionality, by contract. The provider of the service defines the contract as a service description. An intermediary, such as a broker, uses a registry to find or search for published services. Service consumer, service provider, service description, service broker, and a registry are all part of the DNA of SOA.

A service in SOA is the logical, self-contained business function. Services in SOA have the following attributes:

- **Stateless**: SOA services neither remember the last thing they were asked to do nor do they care what the next is. Services are
not dependent on the context or state of other services, only on their functionality. Talking on the telephone is stateful, whereas posting a letter is stateless. The World Wide Web provides an excellent example, where each request from a user for a web page or URL results in the requested pages being served, but without the web server remembering the request later. Each request or communication is discrete and unrelated to requests that precede or follow it.

- Discoverable: A service must be discoverable by potential consumers of the service. After all, if a service is not known to exist, it is unlikely ever to be used. Services are published or exposed by service providers in the SOA service directory, from which they are discovered and invoked by service consumers.

- Self-describing: The SOA service interface describes, exposes, and provides an entry point to the service. The interface contains all the information a service consumer needs to discover and connect to the service, without ever requiring the consumer to understand (or even see) the technical implementation details.

- Composable: SOA services are, by nature, composite. They can be composed from other services and, in turn, can be combined with other services to compose new business solutions.

- Loose coupling: Loose coupling allows the concerns of application features to be separated into independent pieces. This separation of concern provides a mechanism for one service to call another without being tightly bound to it. Separation of concerns is achieved by establishing boundaries, where a boundary is any logical or physical separation that delineates a given set of responsibilities. For example, an account service has open account, authorization, and audit features representing delineations of responsibilities and three separations of concerns.

- Governed by policy: Services are built by contract. Relationships between services (and between services and service domains) are governed by policies and service-level agreements (SLAs), promoting process consistency and reducing complexity.

- Independent location, language, and protocol: Services are designed to be location transparent and protocol/platform
independent (generally speaking, accessible by any authorized user, on any platform, from any location).

In addition, services in a service-oriented architecture typically have the following characteristics:

- **Coarse-grained**: Services are typically coarse-grained business functions. Granularity is a statement of functional richness for a service—the more coarse-grained a service is, the richer the function offered by the service. Coarse-grained services reduce complexity for system developers by limiting the steps necessary to fulfill a given business function, and they reduce strain on system resources by limiting the “chattiness” of the electronic conversation. Applications by nature are coarse-grained because they encompass a large set of functionality; the components that comprise applications would be fine-grained. Similarly, within an application, a service such as “get account information” (which returns name, account number, and address) could be described as coarse-grained, whereas a service to “get account number” could be described as fine-grained.

- **Asynchronous**: Asynchronous communication is not required of an SOA service, but it does increase system scalability through asynchronous behavior and messaging techniques. Unpredictable network latency and high communications costs can slow response times in an SOA environment, due to the distributed nature of services. Asynchronous behavior and messaging allow a service to issue a service request and then continue processing until the service provider returns a response.

Viewed from the top down, SOA comprises the following constructs, as illustrated in Figure 1.3: consumer, business processes, services, components, information, rules, and policies. Consumers allow invocation or composition of services at the consumer layer through social software, mashups, business processes, or other systems. Business processes represent the flows of activities required to complete a business process; they are compositions of services targeted to achieve business goals. Services are the main structuring element required by a service consumer and are provided by the service provider. Services offer functionality and quality of service,
both of which are externalized within service descriptions and policies. Services can be composed of other services, thus making them composite services. Components realize not only the functionality of the services they expose but also ensure their quality of service. Information flows between the layers (for example, consumer, process, and service) and within a layer. Lastly, rules and policies exist for services, components, and flows.

Although objects are illustrated in Figure 1.3, the word *object* does not imply an implementation of object orientation, because the object can easily be a procedural subroutine implemented in a multitude of languages as easily as it can be implemented in an object-oriented programming language. SOA must have services and components that realize the services. Processes or flows may string services together to fulfill a step or activity of a business process. For example a transfer of funds service may string together both a debit and credit account service.
There is also a technology view of SOA. Technology enables SOA, makes it efficient, and optimizes the implementation, but SOA is not defined by the technologies chosen for implementation. Instead, SOA is defined by providing a uniform means to offer, discover, interact with, and use capabilities (services) to produce desired effects consistent with measurable expectations.

The major technologies associated with SOA include business-focused tools, software construction tools, and middleware technologies. Figure 1.4 illustrates the basic technology building blocks for SOA. Tools are required for SOA addressing design-time and run-time scenarios. Business stakeholders use business-focused tools for modeling and analysis of business processes and flows, and they will also use business activity monitoring technology to gain insights into business performance of processes and workflow. IT practitioners use a set of tools for development of business applications and for managing the operating environment addressing integration, monitoring, and security.

![Figure 1.4 Business and IT tools for SOA](image)

The DNA of SOA will most likely be further investigated and defined by standards groups actively involved in defining an SOA
ontology. For example, see www.opengroup.org/projects/soa-ontology/. Such an ontology will address SOA key concepts, including services, service contracts, service interfaces, composition (orchestration, choreography, and collaboration), processes, service compositions, policies, and events. Each of these makes up the DNA of SOA.

4. What Is the Difference Between a Web Service and an SOA Service?

The distinction between business services or SOA services versus a web service is not often articulated, and many equate the two as being the same. SOA services can be realized as web services, but not all web services are equal to SOA services. Web services represent the use of both a published standard and a set of technologies for invocation and interoperability. SOA services are services that fulfill a key step or activity of a business process and can be described as business services and are often exposed as web services.

Figure 1.3 illustrates both an SOA service and a web service. The picture shows the difference between SOA and web services at runtime (i.e., implementation level) and at design time. The web service is illustrated on the right side of Figure 1.5, specifically the Web Services Description Language (WSDL) and its attributes such as port types and operations. The attribute that makes it a web service is the use of WSDL or equivalent.

In design, we identify and specify a service that provides the design, or we identify and specify interfaces that include method specifications. The combination of the definition of the method and the interface at design time is what we refer to as a service from an SOA perspective. Use cases can be used to capture the functional requirements for a service. Figure 1.5 contrasts the differences between a service in SOA and a web service. Both SOA services and web services are part of the DNA of SOA.

In an SOA, business processes, activities, and workflow are broken down into constituent functional elements called services. They can be accessed and used directly by applications, or they can be mixed and matched with other services to create new business capabilities. Business services or SOA services are reusable business
capabilities. Examples in banking include open account or change address. For transportation, it might be get reservation or hold reservation, and with loan processing, get loan, apply for loan, and update address are examples of business services. Business processes are also key constructs of SOA, part of its DNA.

5. What Makes a Project an SOA Implementation?

The deployment of services makes a project an SOA implementation, where a service is defined in the preceding answer as a web service or an SOA service. The use of the Web Service Description Language (WSDL) or equivalent makes a service a web service. An SOA service must satisfy the criteria described in the Answer 2; namely, an SOA service must be stateless; discoverable; self-describing; composable; loosely coupled; governed; and independent of location, language, or protocol. That is, the use of services alone makes the project or implementation an SOA implementation. However, an SOA implementation may not accrue the desired benefits of SOA around cost savings, reuse, time to market, or flexibility.

Services can have different levels of maturity. For example, services can be ad hoc in their design and implementation where a
WSDL façade is implemented to make function accessible to other systems or applications. Services can also be architected where service modeling and governance are used to maximize service reuse.

The implementation of SOA technologies without a deployment of one or more services could also be defined as an SOA implementation. This would be atypical because middleware and infrastructure implementations (e.g., a registry or enterprise service bus) would be implemented in conjunction with the deployment of services.

Just as services have different levels of maturity, so do SOA adoptions within an organization. Some SOA adoptions require a program of projects to address a journey of increasing maturity to achieve strategic SOA goals of building systems for change, infusing flexibility as an attribute of systems, or reducing the lifetime costs of applications and infrastructure. In this case, the program comprises a series of SOA projects that incrementally raise the maturity of SOA in an organization and along the way enable the realization of the strategic SOA benefits.

Often, because of overselling of SOA, organization leaders, managers, and executives wrongly believe that the benefits of SOA automatically accrue when an SOA implementation occurs. SOA has varied and diverse definitions, and hence its implementations are equally varied. So, organizations seeking to accrue any of the promised benefits of SOA must do more than focus on SOA implementations. That is, each expected benefit of SOA requires a different level of SOA maturity. For example, if the goal is only to reduce the cycle time of a business process that deals with external partners, exposing web services may be the only necessary SOA adoption. However, if the business goal is to reduce time to market for new products, this requires a broader adoption of SOA that addresses reusable services, structuring of applications using services, improving integration using services, and aspects of SOA governance to address service sharing, funding, and ownership.

SOA Basics: Key Concepts

This chapter’s answers emphasized the utility of SOA and how to accrue its strategic and tactical benefits, instead of just providing an agreed-upon definition. However, looking at the definition through the lens of the different stakeholders provides a comprehensive view
of what SOA is highlights the various potentials of SOA. The DNA of SOA comprises service consumers, business processes, services, service descriptions, components, information, rules, policies, web services, technologies (e.g., registries and brokers), and tools that address business and IT domains.

As you learned in this chapter, SOA implementations are as varied as SOA definitions, and the benefits that accrue depend on the maturity of SOA adoption within an organization. Organizations and executives who expect to accrue strategic benefits of SOA will need to treat SOA adoption as a journey realized incrementally by project (not as tactical goals, where a project might be sufficient). The next chapter answers questions that business leaders and executives ask about SOA.
This page intentionally left blank
A

abstraction, levels of, 9
ACID properties, 196
adoption rates in future of SOA, 220-221
agile methods versus object development methods, 96-98
aligned state (business and IT relationships), 48
analytical services, 177
APIs (application programming interfaces), 177
service interfaces versus, 161, 179
application development
accelerating with SOA methods, 112-115
changes in SOA, 128-133
application portfolios, management under SOA, 139-140
applications
BPM (Business Process Management) and, 133-136
for business processes across organizational boundaries, 125
cloud deployment of services, 142-143
composite applications, 122
determining suitability for SOA, 125-128
enterprise focus of SOA, 143
flexibility improvements with SOA, 137-139
flexibility of, 123
legacy applications in SOA, 140-142
lifetime costs, reducing, 114-115
pitfalls to SOA adoption, 144-145
procedural-centric versus transactional-centric systems, 125
in SOA, 121-123
variances, 125
architectural style, SOA as, 7-8
architecture
background of, 147-148
common pitfalls, 170-171
complexity of SOA, 158-160
EAI versus SOA versus SOI, 167-168
layers in, 150-155
reference architectures in building infrastructure, 215-216
service interfaces versus service contracts, 160-162
SOA
differences in, 149-155
and EA (enterprise architecture), 165-167
earlier approaches versus, 156-157
web services versus, 157-158
standards, role of, 168-169
web services versus REST, 162-165
artifacts in application development, 131
asynchronous, defined, 11
atomicity of transactions, 196

B
barriers to SOA adoption, removing, 58-59
batch data channels, 194
best practices, creating SOA infrastructure, 200-201
bootstrapping, 85
BPM (Business Process Management), applications and, 133-136
brittle application code, 114-115
browsers, 194
business architecture, SOA applications for, 40-41
business performance management, 197
business problems, determining suitability for SOA, 123
Business Process Layer (architecture), 151-154
Business Process Management (BPM), applications and, 133-136
business processes, 194
across organizational boundaries, 125
defined, 11
expected changes in, 123
flow determined by runtime policies, 126
variances in, 125
business service catalogs, 97
business service policies, 197
business services, information services as, 180. See also SOA services
business stakeholders
alignment with IT stakeholders, 46
changing processes due to SOA, 37-40
flexibility with SOA, 34-36
importance of SOA to, 21-25
IT processes, relationship with, 50
IT stakeholders, relationship with, 48
measuring effectiveness of SOA, 29-33
pitfalls of SOA, 42-43
reuse with SOA, 36-37
ROI for SOA adoption, 28-29
selecting projects for SOA adoption, 33-34
selling SOA to, 25-27

C
canonical message models
creating, 186-187
explained, 184-185
Centers of Excellence (CoEs), 84-86

centralized service delivery models (organizational structure), 52

change management
common pitfalls, 67-68
organizational change management, role of, 56-58
role of, 79-81

changes in business processes, 123

choreography, 194
classifying services, 65-66
cloud computing in SOA infrastructure, 213-214
cloud deployment of services, 142-143
Cloud Security Alliance (CSA), 143
coarse-grained, defined, 11
COBIT (Control Objectives for Information and related Technology), 73
CoEs (Centers of Excellence), 84-86

Common Object Request Broken Architecture (CORBA), 156
communication processes, 87
complexity of SOA, 158-160
compliance processes, 87
components, defined, 12
composable, defined, 10
composite applications, 122
composite services, 12, 121
consistency of transactions, 196
constructs of SOA, 9-14
consumer access components (SOA infrastructure), 194

Consumer Layer (architecture), 151, 154
consumers, defined, 11
content services, 176
context-aware services, 224-225
contracts, interfaces versus, 160-162

Control Objectives for Information and related Technology (COBIT), 73
CORBA (Common Object Request Broken Architecture), 156
corporate governance, 74
cost, reducing lifetime costs for applications, 112-115. See also funding
CRUD (create, read, update, delete) operations by information services, 181
CSA (Cloud Security Alliance), 143
custom development projects (SOA), 107-108

D

Data Architecture and Business Intelligence Layer (architecture), 152, 155
data integrity in multiple data sources, 126
data models, information models versus, 183
data quality in SOA, 187-188
data services, 176
DBA (database administration), 56
DCE (Distributed Computing Environment), 156
decoupling in SOA infrastructure, 205-208

deployment of services in cloud, 142-143
design-time governance, 82
  versus runtime governance, 91-92
development roles, SOA methods, 109-110
differentiation with IT
  (information technology), 19-25
disposable, defined, 10
Distributed Computing
  Environment (DCE), 156
durability of transactions, 196

e
EA (enterprise activity) versus
  SOA governance, 87-89
EA (enterprise architecture)
  business architecture and, 40
  SOA and, 165-167
EAI (enterprise application
  integration)
  ESB (enterprise service bus)
    versus, 201-202
  SOA and SOI versus, 167-168
EDA (event-driven architecture)
  ESB support for, 204
  SOA versus, 225-227
effectiveness of SOA, measuring, 29-33
efficiency value propositions,
  measuring SOA effectiveness, 31
embedded systems, 225
enterprise, SOA focus on, 143
enterprise activity (EA) versus
  SOA governance, 87-89
enterprise application
  integration (EAI)
  enterprise service bus (ESB)
    versus, 201-202
  SOA and SOI versus, 167-168
enterprise architecture (EA)
  business architecture and, 40
  SOA and, 165-167
Enterprise Architecture
  Framework, 89
enterprise information models,
  182-184
enterprise service bus (ESB), 54,
  57, 198-200
  enterprise application
    integration (EAI) versus,
    201-202
  event-driven architecture
    support, 204
  registry and, 203-204
enterprise-level SOA, 158-160
ESB (enterprise service bus),
  54-57, 80, 198
  EAI (enterprise application
    integration) versus, 201-202
  event-driven architecture
    support, 204
  registry and, 203-204
event-driven architecture (EDA)
  ESB support for, 204
  SOA versus, 225-227
exception and appeals
  processes, 87
Extensible Markup Language
  (XML), 156

F
factory models (organizational
  structure), 52
“fiefdom syndrome,” 45
flexibility
of applications, 123, 137-139
with SOA, 34-36
functional components (SOA infrastructure), 194
functional requirements in application development, 129-131
funding for shared services, 59-62
future of SOA
 adoption rates, 220-221
context-aware services, 224-225
EDA (event-driven architecture) and, 225-227
embedded systems, 225
maturation of standards and, 227-228
real-time systems, 225
trajectory of, 221-223
WOA (web-oriented architecture) and Web 2.0, 228-229

G–H
Glass, Robert L., 103
goal service model, 99
governance
business and IT processes, 49
change management, role of, 79-81
corporate governance, 74
defined, 10
design-time governance, 82
IT governance, 72
overview, 72-74, 93-94
pitfalls of SOA adoption, 42-43
runtime governance, 82
SOA. See SOA governance, 72
Governance Layer (architecture), 152, 155
GPS in context-aware services, 224
granularity of services, determining, 106-107
HTTP bindings, 165
hub-and-spoke integration pattern, 168

I–J
identification of information services, 180-181
IDEs (integrated development environments), 195
IDL (Interface Definition Language), 156
information, as service, 173
information architects, defined, 173
information architecture
 canonical message models
creating, 186-187
explained, 184-185
design-time governance
enterprise information models, 182-184
information services
 CRUD operations by, 181
explained, 175-180
identification of, 180-181
pitfalls with SOA, 188-189
SOA and, 174-175
information flows, defined, 12
information models, data models versus, 183
information services
 classifying, 176
 CRUD operations by, 181
explained, 175-180
identification of, 180-181

**Information Technology**

**Infrastructure Library (ITIL),** 73

information-based projects, 108
infrastructure
effect of SOA management on, 212

**ESB**

*EAI versus, 201-202*
and registry relationship, 203-204

**IT infrastructure, defined,** 192

**SOA infrastructure**

*cloud computing in, 213-214*
components of, 193-199
creating, best practices for, 200-201
decoupling, 205-208
event-driven architecture support, 204
pitfalls in creating, 214-217
policy management support, 209-212

**infrastructure standards, method standards versus,** 169

**integrated development environments (IDEs),** 195
integration centers (organizational structure), 54-56
Integration Layer (architecture), 151, 154
integration services, 176
integration technology. See EAI (enterprise application integration)

**Interface Definition Language (IDL),** 156
interfaces, contracts versus, 160-162

investment management,
business and IT processes, 49
isolation of transactions, 196

**IT (information technology)**
differentiation with, 19-25
governance, 72
infrastructure, defined, 192
in organizational structure, 50-56
problems, determining suitability for SOA, 127-128
processes, relationship with business stakeholders, 50

**stakeholders**
alignment with business stakeholders, 46
relationship with business stakeholders, 48

**ITIL (Information Technology Infrastructure Library),** 73

**K–L**

**KPIs (key performance indicators),** 134, 197

legacy applications in SOA, 140-142
legacy enhancement, 108
legacy transformation, 108
lifetime costs, reducing, 114-115
location transparent, defined, 10
loose coupling
in application development, 131
defined, 10
in SOA infrastructure, 205-208
tight coupling versus, 158

**M**

management (in SOA infrastructure), 196, 212
mashups, 122
master data services, 177
maturity models for SOA, 32
measuring effectiveness
  of SOA, 29-33
  of SOA governance, 90-91
messages. See canonical message models
method standards, infrastructure standards versus, 169
methods, 95
  accelerating application development, 112-115
  agile versus object development methods, 96-98
  development roles, 109-110
  overview, 116-117
  pitfalls in adopting, 115-116
  reducing lifetime costs for applications, 112-115
metrics in application development, 133
middleware, defined, 192
multiple data sources, data integrity of, 126

N–O
object development methods
  versus agile methods, 96-98
objects, 12
Observer design pattern, 227
Open Group Service Integration Maturity Model (OSIMM)
  standard, 32, 222
operational components (SOA infrastructure), 195
Operational Systems Layer (architecture), 150-152
orchestration, 194
organizational boundaries, business processes across, 125
organizational change
  management
    common pitfalls, 67-68
    role of, 56-58
organizations
  alignment between business and IT stakeholders, 46
  business and IT processes, relationship between, 50
  business and IT stakeholders, relationship between, 48
  common SOA adoption pitfalls, 67-68
  funding for shared services, 59-62
  getting started with SOA governance, 72-79
  prioritization for shared services, 63
  removing barriers to SOA adoption, 58-59
  structure of, 50-56
OSIMM (Open Group Service Integration Maturity Model)
  standard, 32, 222
owners of services, explained, 64
ownership of service reuse, 66-67

P
packaged implementation, 108
performance, 197
platform independent,
  defined, 10
point-to-point infrastructures, 216
policies
  business service policies, 197
types of, 209
policy management, 199
  SOA infrastructure support for, 209-212
portfolio management, business and IT processes, 49
prioritization
in application portfolios, 139-140
for shared services, 63
private clouds, 214
procedural-centric systems, 125
processes. See business processes
project prioritization, business and IT processes, 50
projects
information-based projects, 108
selecting for SOA adoption, 33-34
transformation projects, 107
protocol independent, defined, 10
public clouds, 214
publish-subscribe integration pattern, 168

Q–R
quality of data in SOA, 187-188
Quality of Service Layer (architecture), 151, 155
queries by information services, 181
real-time systems, 225
reducing lifetime costs for applications, 113
reference architectures, 171
in building infrastructure, 215-216
registry, 195
ESB (enterprise service bus) and, 203-204
Remote Procedure Call (RPC), 156
removing barriers to SOA adoption, 58-59
Representational State Transfer (REST), 157
web services versus, 162-165
requirements
SOA and, 39
for SOA implementations, 15-16
requirements gathering, business and IT processes, 50
resistance to SOA adoption, overcoming, 58-59
REST (Representational State Transfer), 157
web services versus, 162-165
return on investment (ROI) for SOA adoption, 28-29
reuse
ownership of, 66-67
of services, 103-106
with SOA, 36-37
ROI (return on investment) for SOA adoption, 28-29
role of change management, 79-81
Ross, Jeanne W., 73
RPC (Remote Procedure Call), 156
runtime governance, 82
versus design-time governance, 91-92
runtime policies, determining process flow, 126

S
security, 195
selecting projects for SOA adoption, 33-34
self-describing, defined, 10
service adapter pattern, 207
service centers, centralizing service development, 83
service classification, 65-66
Service Component Layer
(architecture), 150, 153, 178
service components, defined,
178-180
service contracts, service
interfaces versus, 160-162
service descriptions, 195
service development, centralizing
in service centers, 83
service identification. See
identification
Service Integration Maturity
Model (SIMM), 32
service interfaces
APIs versus, 179
service contracts versus, 160-162
service level agreement
(SLA), 212
service modeling (SOA), 101-103
service orientation, defined, 6.
See also SOA (service-oriented
architecture)
service owners, explained, 64
service portfolios, 139-140
service proliferation, 171
service proxy pattern, 207-208
service re-use, ownership of, 66-67
service wrappers, 207
service-oriented architecture. See
SOA (service-oriented
architecture)
service-oriented integration
(SOI), SOA and EAI versus,
167-168
services, 9
in application development,
131-132
attributes of, 9-11
cloud deployment of, 142-143
composite services, 12, 121
defined, 11, 14, 178
granularity, determining,
106-107
information services, 173
classifying, 176
CRUD operations by, 181
explained, 175-180
identification of, 180-181
requirements for SOA
implementations, 15-16
reusing, 103-106
shared services, 105
funding, 59-62
prioritization, 63
SOA services, defined, 15
web services, SOA services
versus, 14
Services Layer (architecture),
150, 153, 178
shared service development
models (organizational
structure), 52-55
shared services, 105
funding, 59-62
prioritization, 63
SIMM (Service Integration
Maturity Model), 32
Simple Object Access Protocol
(SOAP), 156
web services versus, 162-165
skunk work projects, 62
SLA (service level
agreement), 212
SOA (service-oriented
architecture)
applications in, 121-123
development changes in,
128-133
flexibility improvements,
137-139
portfolio management, 139-140
suitability for, 125-128
as architectural style, 7-8
business, 21-27, 37-43
business architecture, 40-41
business problem suitability for, 123
CoEs (Centers of Excellence), 84-86
complexity of, 158-160
constructs of, 9-14
custom development projects, 107-108
data quality and, 187-188
EA (enterprise architecture) and, 165-167
EAI and SOI versus, 167-168
earlier approaches versus, 156-157
EDA (event-driven architecture) versus, 225-227
enterprise focus of, 143
explained, 5-8
flexibility with, 34-36
funding models, 93
future of
adoption rates, 220-221
context-aware services, 224-225
EDA (event-driven architecture) and, 225-227
embedded systems, 225
maturation of standards and, 227-228
real-time systems, 225
trajectory of, 221-223
WOA (web-oriented architecture) and Web 2.0, 228-229
governance
CoEs (Centers of Excellence), 84-86
EA (enterprise activity) versus, 87-89
going started, 72-79
implementation of SOA tools and, 81-82
measuring effectiveness of, 90-91
overview, 72-79
pitfalls of, 92-93
service development, centralizing in service, 83
success of SOA projects, 89-90
implementations, requirements for, 15-16
information architecture and, 174-175
infrastructure
cloud computing in, 213-214
components of, 193-199
creating, best practices for, 200-201
decoupling, 205-208
event-driven architecture support, 204
pitfalls in creating, 214-217
policy management support, 209-212
IT problem suitability for, 127-128
legacy applications in, 140-142
management, effect on infrastructure, 212
measuring, 29-33
pitfalls for applications, 144-145
with information architecture, 188-189
projects
  custom, 107-108
  success of, 89-90
removing barriers to, 58-59
reuse with, 36-37
ROI, determining, 28-29
selecting projects for, 33-34
service modeling, 101-103
services
  defined, 15
  requirements for, 15
  web services versus, 14
stakeholders in, 6-7
standards, role of, 168-169
system development, 98-101
testing methods, 110-112
web services versus, 157-158
SOAP (Simple Object Access Protocol), 156
  web services versus, 162-165
software development, 119. See also applications
SOI (service-oriented integration), SOA and EAI versus, 167-168
stakeholders
  business stakeholders
    changing processes due, 37-40
    flexibility with SOA, 34-36
    importance of SOA to, 21-25
    measuring effectiveness, 29-33
    pitfalls of SOA, 42-43
    reuse with SOA, 36-37
    ROI for SOA adoption, 28-29
    selecting projects for, 33-34
    selling SOA to, 25-27
in SOA, 6-7
standards
  future of SOA and, 227-228
  role of, 168-169
  web services standards, 156-157
stateless, defined, 9
strategic planning, business and IT processes, 49
structure of organizations, 50-56
success of SOA projects (SOA governance), 89-90
synced state (business and IT relationships), 47
system development, 95
  SOA, 98-101
T
technologies in SOA, 13-14
testing methods (SOA), 110-112
tight coupling
  in application development, 130
  loose coupling versus, 158
tools, implementing SOA tools (governance), 81-82
transactional-centric systems, 125
transactions, 196
transformation projects, 107
transportation (SOA infrastructure), 196
U–V
unified state (business and IT relationships), 47-49
use cases in application development, 129-130
value propositions, measuring
  SOA effectiveness, 30-31
variability, 106
variances in applications, 125
virtual service provider pattern, 207-208
vitality processes, 87

W–Z

Web 2.0, future of SOA and, 228-229
web portals, 194
Web Service Definition Language (WSDL), 156, 159, 195
   REST versus, 162-165
web services
   requirements for, 15
   REST versus, 162-165
   SOA services versus, 14
   SOA versus, 157-158
   standards, 156-157
Web Services Interoperability Organization (WS-I), 156
web-oriented architecture (WOA), 228-229
Weill, Peter, 73
WOA (web-oriented architecture), 228-229
WS-I (Web Services Interoperability Organization), 156
WSDL (Web Service Definition Language), 156, 159, 195
   REST versus, 162-165
XML (Extensible Markup Language), 156