Chapter 1

The SOA Challenge

Business processes and information systems have become so tightly intertwined that it is no longer possible to design one without designing the other. Business processes do not simply depend on information systems—they define the services required.

Altering business processes inevitably requires system changes. Conversely, system changes inexorably alter business processes. Herein we find the SOA challenge—designing systems in such a way that accommodating most business process changes simply requires rearranging existing business services. If you can accomplish this, you will not only reduce development costs but will also decrease the time-to-market for these business process changes and thereby improve your enterprise's competitive position.

But business processes involve more than just the functionality of systems. They involve information. Information is central to business processes, and these processes determine what information is required and how it should be managed. Requisitions, orders, claims, and reports, for example, are nothing more than information.

Business processes also depend on people, particularly for decision making and exception handling. From simple work tasks and approvals all the way up to strategic decision making, people provide the flexibility that enables business processes to deal with unexpected emerging opportunities and unanticipated problems. From soothing a disgruntled customer to coping with mergers and acquisitions, people make the difference.

Business processes, people, information, and systems together comprise the symbiotic collaboration that makes the enterprise work. Independently, they accomplish nothing. Together, they bring the enterprise to life. It is this partnership that enables the enterprise to produce its results and achieve its goals.

When you build a service-oriented architecture, you package a significant portion of the systems functionality and related information in the form of services. A *service* is a bundling of information and the functionality required to manage it. An order management service, for example, provides operations for placing, revising, and canceling orders as well as for checking order status. The service manages the order information, including ensuring the durability of the information.

Packaging information in the form of services presents a serious challenge because enterprise goals are not static. They are in constant flux. Changing business pressures and emerging opportunities continually force enterprises to reevaluate and reprioritize goals. When these goals change, enterprises must then refocus this collaboration of business processes, people, information, and systems on the new priorities. This reprioritization needs to be efficient. The ability to respond to new opportunities and changing pressures is, itself, a critical success factor for the enterprise. To quote Thomas Paine (in a phrase later made famous by Lee Iacocca), we can "Lead, follow, or get out of the way." We want to lead. This book will show you how.

The Concept of Total Architecture

Here's the challenge: Organize the collaboration between business processes, people, information, and systems, and focus it on achieving enterprise goals. Design the services in such a way that they facilitate rather than hinder the reorganization and refocusing of the enterprise business processes. Oh, and by the way, do it quickly and efficiently.

To make your enterprises work effectively, business processes, people, information, and systems must be architected together, as a whole. This is *total architecture*. This approach is not a choice. It is a concession to reality. Attempts to organize business processes, people, information, and systems independently result in services that do not fit together particularly well. The modification of business processes becomes inefficient, and the focus on the real business objective gets lost.

When business processes cannot efficiently evolve, enterprises find themselves hamstrung as they try to respond to changing opportunities and pressures. They produce inefficient, fragile, error-prone business processes that seem to defy attempts to improve them. As enterprises grow in scope and complexity, architects embed information systems even more deeply into the business processes to help manage this complexity. The web of people, processes, information, and systems becomes increasingly tangled. So, you have no choice but to address the *total* architecture. The only question is how best to do it.

Total architecture defines the structure and organization of business processes as well as information systems. This is a business responsibility, not an IT responsibility. Because of the interdependency between business processes and systems, this work must be done in concert with the systems architecture. Architecture is no longer just an IT issue—it is an enterprise issue.

Systems Are More Than Services

Systems functionality goes beyond simply providing services. When a business process employs services, some participant in that business process needs to decide when and how each service is employed. We commonly refer to this logic as *service orchestration*. This service orchestration itself is not, necessarily, a service, although many services (known as composite services) will employ service orchestration.

The vast majority of the functionality in your enterprise systems today is not provided in the form of services, and it is unrealistic to think that you can turn all of this functionality into services magically overnight. You must be able to employ this legacy functionality in your business processes just as effectively and efficiently as with your well-designed services. Where appropriate, you also want to evolve this legacy functionality into services.

So while services are the major focus here, there must also be a notion of architecture that is inclusive of service orchestration and nonservice functionality.

Services Involve More Than Business Functionality

When you think of a service, your thoughts probably trend toward the business functionality provided by the service. After all, an order entry service is all about placing and managing orders. But closely related to this business functionality are a bunch of rules regarding the use of the service. Who is authorized to place orders? Who can examine order status? Who is authorized to approve orders?

The rules surrounding the use of services themselves involve additional information and functionality. This information and functionality are also part of the service, and these elements are subject to change as business processes change. In fact, if you look at business process evolution, these rules seem to change more often than the functionality whose use they govern!

While you want business rules to govern access to services, you also want to minimize the technical constraints for accessing them. For the greatest flexibility, you want universal access to your services, rather than having decisions regarding implementation technology or deployment location inadvertently constrain the use of the service. So, for example, you want services implemented in COBOL on mainframes to be as accessible from business process management (BPM) tools and Java-based application servers as they are from within the mainframe itself.

You need location independence in addition to technology independence. For example, a global bank needs to be able to provide services for its customers no matter where they happen to be, for example. A customer from North America should be able to walk into a bank in Europe or Asia and use the banking services as readily as if he or she were at home. From the systems perspective, this means that the local banking system needs to be able to access customer information regardless of the actual location of the service that happens to have information for that specific customer.

This need to alter access rules flexibly and provide ubiquitous access to services in turn drives the architecture of individual services toward a modular design. This modularity separates business functionality from access rules and routing rules, while providing uniform service access from any technology. These are the core service architecture requirements that will enable you to independently alter business functionality, access policies, and routing rules.

Growing Pressures

Business processes have long spanned the multiple silos found within enterprises (Figure 1–1). Then why are you starting to experience more

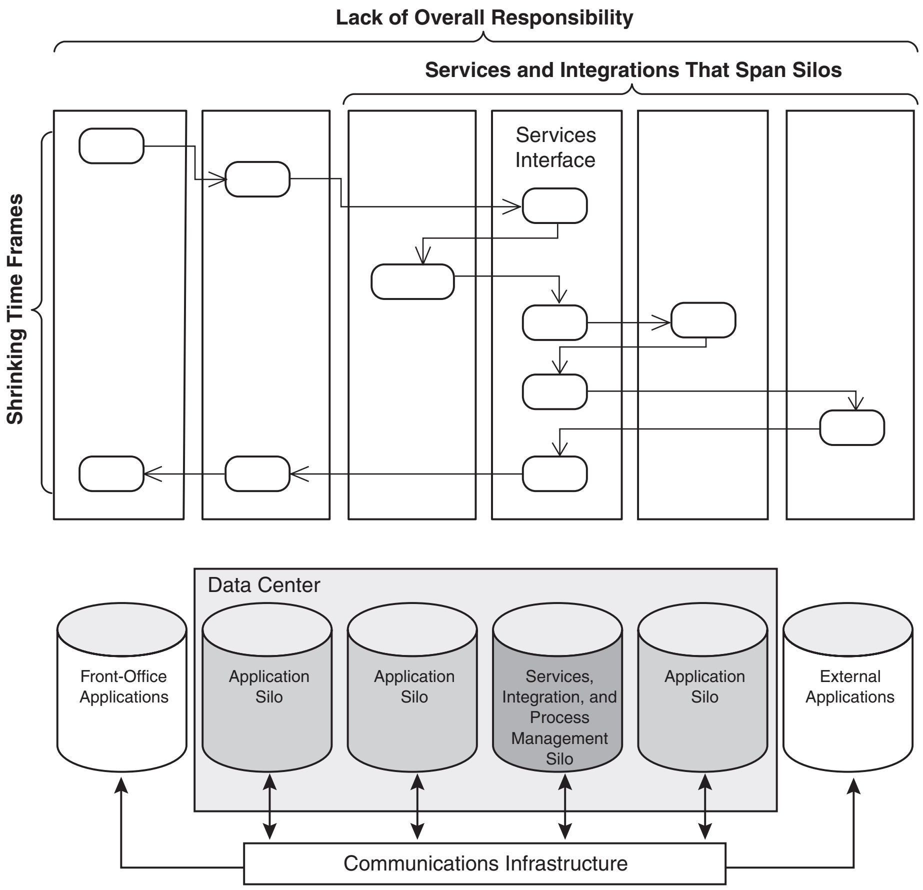
Lack of Overall Responsibility
Services and Integrations That Span Silos
Shrinking Time Frames
Services Interface
Data Center
Front-Office Applications
Application Silo
Application Silo
Services, Integration, and Process Management Silo
Application Silo
External Applications
Communications Infrastructure

Figure 1–1: *Sources of Pressure*

significant problems now? The answer is that changes afoot within enterprises are stressing their existing architectures.

The majority of IT projects have traditionally focused on a single business process activity (or group of activities) residing entirely within a single application silo. The project's focus is generally to improve this activity's functionality, cut its cost, or improve its response time. The entire project generally lives within the silo: business objective, justification, budget, and staff.

Evolving business pressures have begun to force you to take a different view. One of these pressures is to improve the responsiveness of the enterprise as a whole. Customers and business partners want the business to respond more quickly. They want to have ordered goods delivered tomorrow (or today), not next week. They want the status of their order when they ask, not a callback in an hour after someone has investigated. The problem, of course, is that such projects require changes in multiple silos. They do not fit well within existing project structures.

Another major pressure is cost management. The wide adaptation to enterprise resource planning (ERP), for example, has sought to optimize resource utilization by coordinating and managing all enterprise-wide business processes related to planning, procurement, and production. These efforts impact all of the architectural elements, and successful ERP projects address them all, in concert. Projects that treat ERP solely as an IT initiative inexorably, and spectacularly, fail.

There are also increasing pressures to increase the return on investment (ROI) from IT projects and to make systems and business processes more flexible. This has pushed SOA to the forefront as enterprises seek to architect business processes from reusable services. The vision is that the reuse of services will cut IT costs by avoiding the cost of reimplementing existing functionality in future projects. Services also promise the potential for implementing new (or revised) business processes more quickly by simply composing existing services. This not only reduces IT costs but cuts the overall time span of the project as well. It improves an enterprise's ability to respond to outside pressures.

The sticking point here is that business services are pieces of business processes. They involve people and information as well as systems. In defining business services, you are structuring and organizing (i.e., architecting) business processes and business organizations as well as systems. If services are to be reused, they must fit cleanly into multiple business processes and align well with assigned business responsibilities. They require the total architecture perspective and active business involvement.

Business services also pose challenges to existing silo-based project structures. A business service encapsulates functionality provided by one business unit so that it can be used by at least one other business unit. That other business unit is responsible for some other portion of the overall business process. In order for this to work, the interests and needs of these other business units must be factored into the design of the business service, or it will not provide the functionality required.

This multiple-business-unit perspective does not fit well into traditional silo-oriented IT projects. To begin with, the service provider does not accrue any of the benefits of reuse. In fact, the silo implementing the business service will actually incur a higher initial cost for providing its functionality as a service. Nor does the first user accrue any benefit unless the reuse actually occurs within this first project. It is generally the second and subsequent projects that will realize the benefits of services. Existing project structures are not designed to handle projects that span multiple silos and whose benefits will be realized only in the future.

Framing the Challenges

It is clear that responding to these pressures requires projects that span multiple business units. This requirement challenges almost every aspect of the way enterprises conduct projects today. In contrast to traditional projects, these new projects require coordinating the work of multiple business units in order to achieve enterprise goals. This coordination is required both in defining the business processes and in the systems development work needed to get those business processes ready for execution. These projects are doing nothing less than modifying the total architecture of the business.

At the heart of this modification lies the determination of who should be doing what in the revised business process: what work should be done by people, and what work should be done by systems. Since both people and systems live in organizational silos, this determination decides what each silo will be responsible for in the revised process. This, in turn, will determine what development work each silo needs to do in order to implement the revised business process.

Therein lies the problem with current project structures. Who, in silo-oriented projects, has the responsibility for revising the total architecture—both business processes and systems? Who defines the needed services? Who has the responsibility for determining the operational and developmental responsibilities for each silo? Who has the authority to make the silos cooperate in this endeavor? How is the overall budget determined and allocated to the silos?

In most enterprises, the silo-based development processes do not contain explicit tasks for determining who should be doing what—either during development or in the revised business process. They have development processes similar to the one shown in Figure 1–2. You either give the requirements to the silo's development team or tell the team to go determine what the requirements are. If there is any figuring out to be done, the silo's development team does this work. If there is work required in other silos, the development team negotiates with the other silos to get that work done. The silo owns the entire project.

The problem you will encounter with SOA projects is that this streamlined client-server development process doesn't scale. You can't just hand a set of requirements to several development teams, ask each team to figure out what services it ought to be creating, and expect to have an efficient development process that also meets business goals. In fact, with such a fragmented approach, no one is actually accountable for achieving the overall business goals. How can we expect to achieve business goals without such a focus?

An alternative development process looks something like the one shown in Figure 1–3. The project begins with an explicit process charter that provides the vision and focus for the overall effort. This charter sets forth the project goals; establishes the project's cost, schedule, and other constraints; and assigns the key project leadership responsibilities. This process contains an explicit step for determining the services required—the architecture activity. It contains a concession to reality as well: an explicit integration test step. It is impractical to simply turn on a large-scale system with many services for the first time and begin testing. For efficiency, the system needs to be integrated in an organized manner, a few services at a time.

The idea of having a clearly defined project charter, an explicit architecture step, and an integration test step is not new by any means. Their inclusion in the development process is a well-established best practice for software development. But if these steps are no longer present in your development process, you must reintroduce them. In addition, your thinking about the scope of the project and its related

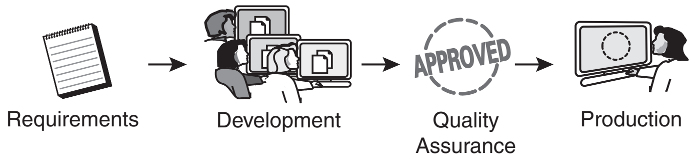

Figure 1–2: *Silo-Based Client-Server Development*

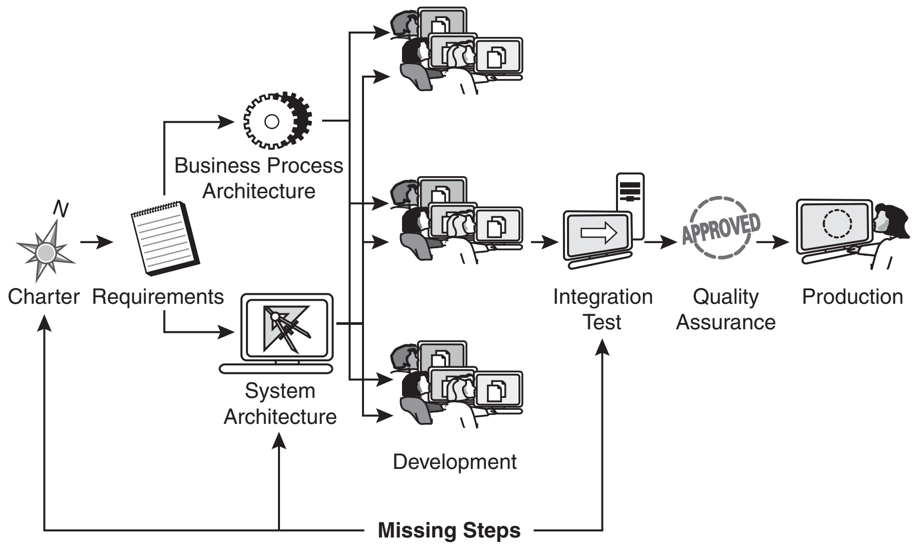

Figure 1–3: *SOA Development*

architecture activity must be extended to encompass business process design as well as system design.

The architecture step is the key to making SOA development work. In this activity, the business process architect determines what services and service orchestrations are needed in the revised business process in order to achieve the business goals. At the same time, the systems architect determines how these services and orchestrations will be provided—and what legacy functionality is required. Together, the architectural responsibilities span both business processes and information systems, both development time and runtime. This step encompasses all of the elements of total architecture.

Closely related to this development process challenge is an organizational challenge (Figure 1–4): Who has overall ownership of a project that spans multiple business units? Who is responsible for the architecture step that determines what each silo should be doing? Where, organizationally, do they report?

This project ownership problem is exacerbated further by a commonplace organizational approach used when introducing new technologies into the enterprise: creating a new IT silo just for the new technology! Unfortunately, this silo, unlike the application silos, does not have a business counterpart (as indicated by the question mark in

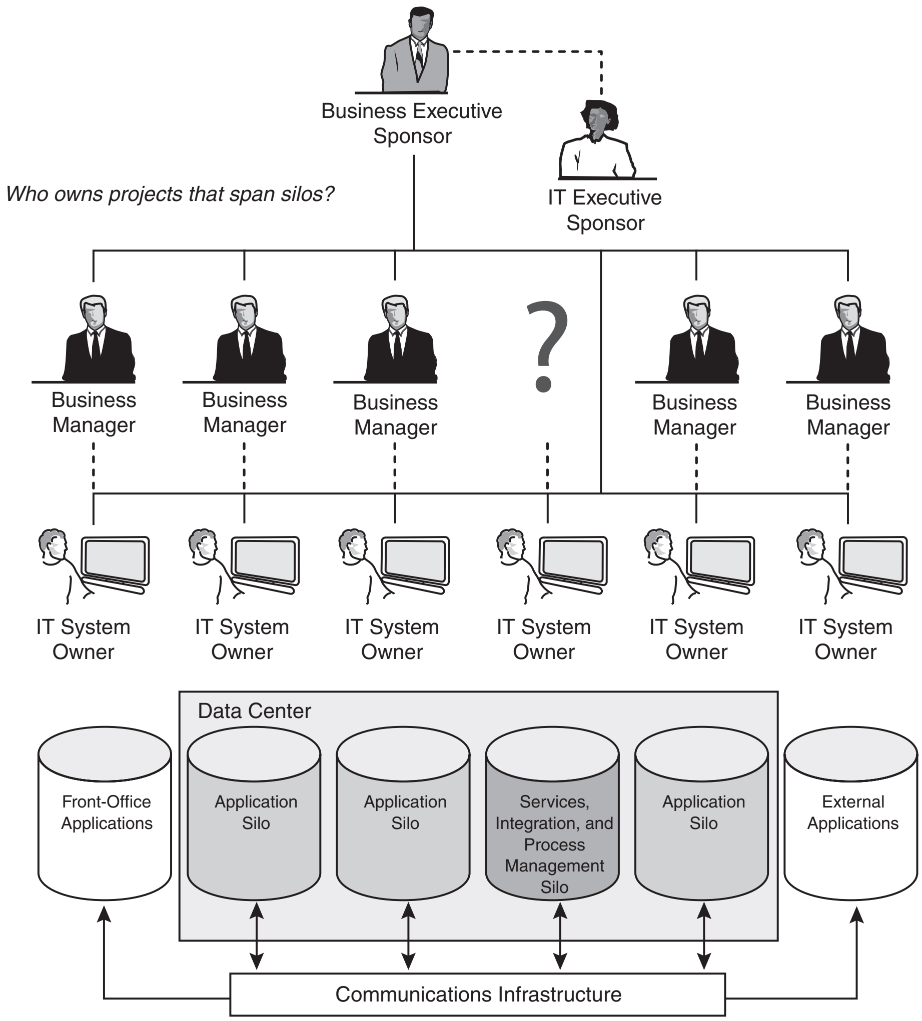

Figure 1–4: *The SOA Organizational Challenge*

Figure 1–4). Yet you expect this silo to use the new technology to integrate the other silos. You make it responsible for services, service orchestration, integration, and process management. Yet unless you give it a business side, with authority over the silos involved, you have not created a recipe for success.

Some enterprises are aware of the project ownership issue and the need for overall business guidance in silo-spanning projects. This is evidenced in the structuring of major initiatives like ERP. The business creates a program office that reports to senior management and is responsible for the overall initiative. Successful ERP programs not only address the total architecture issues but also have the authority to ensure the cooperative participation of all the silos involved.

The problem is that more and more projects span multiple silos. In fact, service-oriented architectures make such projects the norm, not the exception. So you need an organizational home for projects that span multiple silos. This home must acknowledge that these silo-spanning projects are deep collaborations between business and IT. SOA projects are not IT projects—they are business projects that have a major IT component.

The final challenge is posed by the notion of reusable services. While you may develop a service in one project, the intent should be to design that service so that it can be reused in subsequent projects (Figure 1–5). Who can provide this cross-project perspective? Who can determine where else the service might be used? Who can look ahead to future projects and anticipate their needs accurately enough to specify a service that will actually satisfy those needs? Who can ensure that future projects will actually use the available services and not reinvent them?

These diverse challenges are all facets of the enterprise's total architecture. Total architecture spans silos. It spans business and IT. It spans projects, both present and future. In fact, total architecture is the core of the enterprise. It is the structure of organizations, business processes, and systems. It is there, whether you like it or not. So you have a choice. You can choose to turn your back and plod on in ignorance—or you can recognize total architecture for what it is: the very structure of your enterprise. That's what this book is all about.

Staying on Track

There are four keys to staying on track with the total architecture approach to SOA. The first is to ***justify each project on its own business merits***. Each project should make business sense on its own. It should tackle specific business problems with measurable objectives and constraints. It should identify the business processes that need to be modified in order to achieve its objectives. It should maintain focus on modifying those business processes to achieve the business objectives,

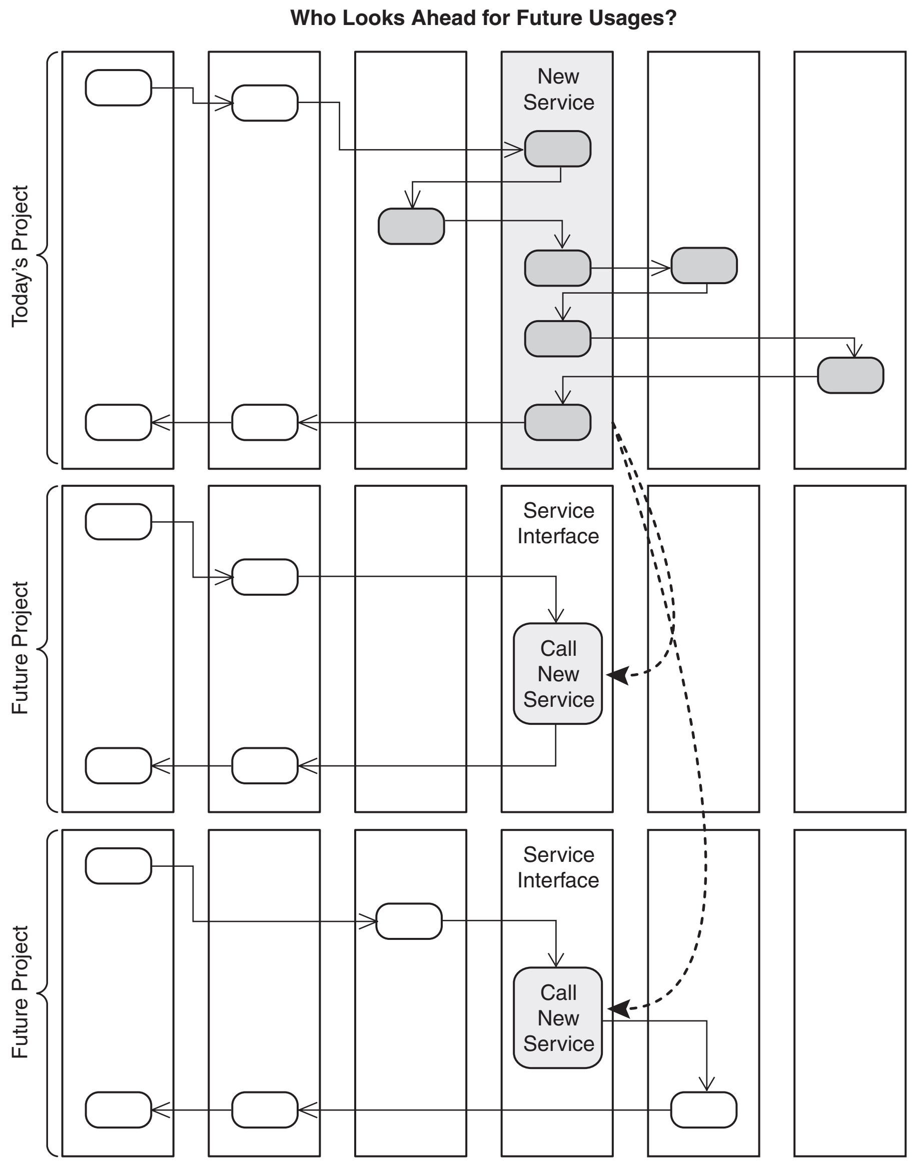

Figure 1–5: *Cross-Project SOA Challenges*

doing so within the project's cost and schedule constraints. Organizing projects around business objectives and business processes ensures that each project yields a recognizable business value and justifies its own cost.

The second key is to ***have an explicit architecture step*** in every SOA project that precedes the actual development work. In this step, you consider ways in which the business process participants might collaborate to get the job done and select the approach you will actually take. Some of these participants will be providing services, while others orchestrate the use of services and access legacy systems. It's a shell game, with the participants (both people and systems) being the shells and their responsibility assignments being the peas. Once you select a satisfactory architecture, you will have a detailed inventory of the required development work. From this, you can determine the cost and then know whether the project can produce the expected business benefit within the given cost and schedule guidelines. This sequence is total architecture at work.

The third key is to ***have an active SOA architecture group***. Total architecture spans all projects and all silos. The results of each project must integrate smoothly with those of others. In fact, this is the prime requirement for service-oriented architectures—services must fit smoothly into future projects. In order to achieve this, someone must have the responsibility to determine how these pieces will fit together and to shape them accordingly, with the authority to ensure project compliance. This is the role of the SOA architecture group.

The SOA architecture group is responsible for the total architecture of the enterprise. It establishes the vision of where the enterprise is going in terms of both business process architecture and systems architecture. It is responsible, directly or indirectly, for the common infrastructure, shared services, best practices, and architectural patterns employed in the enterprise. The SOA architecture group is not an ivory-tower organization. It gets its hands dirty making sure that the work of each project integrates smoothly into the enterprise architecture. It does this through a combination of direct participation, training, mentoring, and governance reviews.

The fourth key is to ***have a living SOA project roadmap***. The roadmap lays out the plan for present and future projects, with a two- to three-year planning horizon. It not only lays out a sequence of projects based on business priorities but also lays out the roadmap for services development: which projects will produce which services, and which future projects are expected to employ those services. This roadmap is a joint effort between the business leadership team, the IT leadership team, and the SOA architecture group.

These four keys provide the strategy for succeeding with SOA. The roadmap plans the services development. The SOA architecture group ensures that services are well conceived. The architecture step ensures that they are used appropriately. Most importantly, justifying each project on its own business merit makes it possible to sustain the investment in services indefinitely. This winning formula has been time-tested in top global and Fortune 500 companies.

How to Use This Book

This book is the first of a pair of closely related books on SOA and total architecture. This book is targeted at the enterprise leadership community. Its purpose is to illustrate why total architecture is critical to enterprise success and the roles that the leadership team must play to make SOA work. The second book is aimed at SOA architects. Its purpose is to arm project and enterprise architects with the knowledge and tools they need to build and manage the enterprise's total architecture.

This book, *Succeeding with SOA*, presents service-oriented architecture from the management perspective. It illustrates what can happen to enterprise business processes and projects when elements of the total architecture are neglected. It explores the deepening reliance of business processes on information systems and the resulting need to keep SOA projects focused on achieving well-defined business objectives within cost and schedule guidelines. It examines the challenges posed by organizational structures and shows how paying attention to five key leadership roles can bring the SOA focus to the enterprise without requiring significant reorganization.

This book also shows how a SOA architecture group can provide continuity and consistency across projects while maintaining an overall focus on enterprise objectives such as reaping the benefits of service-oriented architectures. It explores how the robustness of business processes can be improved by paying attention to the simple structure of the interservice dialog within the process. It illustrates how an understanding of business risk can be used to guide investments in fault tolerance and high availability. It outlines an agile approach to SOA that can efficiently produce a robust architecture, with accurate early determination as to whether the business objectives can be achieved within the cost and schedule guidelines. Finally, it looks at successfully structuring, initiating, and executing SOA projects with the total architecture perspective.

Succeeding with SOA will be followed shortly by a second book, *SOA in Practice: Implementing Total Architecture*. This companion volume is the how-to book for SOA architects. Although the book is organized as a progression of issues that the project architect must address, each chapter also discusses the related activities of the enterprise SOA architecture group. It explores the modeling of business processes and the information they depend on and discusses nonfunctional requirements in the context of the business processes to which they pertain. It covers service-oriented architecture, starting with the high-level structuring problem, and then adds successive levels of detail pertaining to communications, data, coordination, breakdown detection, high availability, fault tolerance, load distribution, security, and monitoring.

The companion volume discusses architecture evaluation, detailing the architecture with specifications, and the role the architect must play in testing. It then delves into some of the more complex aspects of a service-oriented architecture: complex business processes, business process monitoring, business process management and workflow, and large-scale business services. It concludes with a summary discussion of the SOA architecture group, the role it plays, and the challenges it faces.

You can use these books in two very different ways. One way is prescriptive. Together, the two volumes present a structured approach that you can use to organize and conduct both individual projects and an overall service-oriented architecture effort. The other way is as an assessment and review guideline. Each chapter addresses a specific topic and concludes with a list of key questions related to that topic. You can use these questions as a self-evaluation guide for your current projects and SOA efforts. Then you can use the body of the chapter to understand more about the specific issues and the various ways in which they can be addressed. Either approach will improve your enterprise's total architecture.