THE SOA PLATFORM GUIDE:
EVALUATE, EXTEND, EMBRACE

White Paper
February 2006

Table of Contents

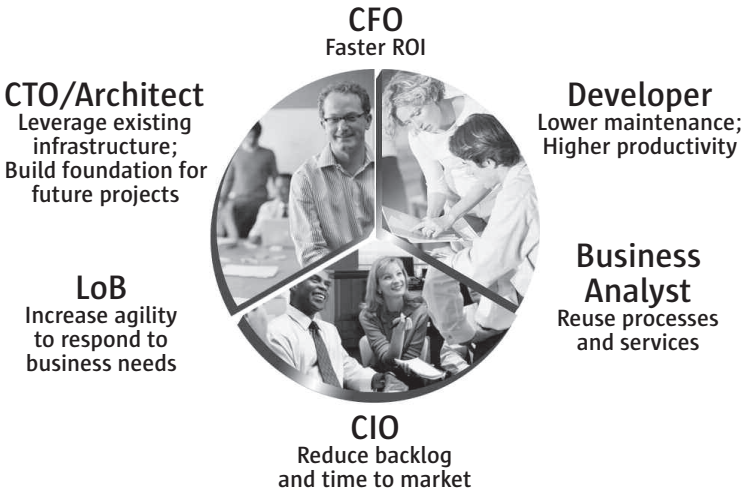
Chapter 1

Executive Summary

In today's world of shortened product cycles and heightened competition, IT's task is to create a flexible environment; to ensure that the enterprise is strategically positioned to foster innovation; to respond to changing needs faster than ever before by reducing time to market for new services; and to drive down the cost of integration and total cost of ownership (TCO). This flexibility is made possible through the use of a Service-Oriented Architecture (SOA) — a design paradigm based on a loosely coupled collection of reusable services. The SOA enables agility through aligning the business and IT, providing business processes that embody core capabilities to employees, customers, suppliers, and partners. The set of infrastructure tools employed by IT to build, configure, deploy, monitor, and manage services is called the SOA platform. As technologies that facilitate service communication and orchestration are standardized, enterprises are able to fully realize the benefits of SOA without fear of vendor lock-in.

Transitioning to an SOA cannot be accomplished in a single step, due to the complexity of business processes and accompanying IT infrastructure, which embodies the business logic. To realize the full potential of an SOA, the enterprise must focus on four key components: people, process, practice, and platform. Sun offers solutions and best practices through customized programs and initiatives that can guide employees through this cultural change. And Sun's robust product offering, the Java™ Enterprise System, gives enterprises the ability to build an SOA incrementally, while leveraging existing assets.

Figure 1. SOA Benefits



Chapter 2

Service Oriented Architecture — Introduction

The objective of this white paper is to help customers understand the SOA tools and infrastructure that will enable them to utilize SOA. The business rationale for SOA is that it reduces the cost and complexity of implementing the business processes that embody core capabilities provided to employees, customers, suppliers, and partners.

Prior to the SOA, many businesses found this objective almost unattainable, because technical roadblocks made it difficult to offer a business process as a service that could be universally shared by its target community of users. The Web has demonstrated that universal access is not only possible but is now a fact of business life, and has proven that a combination of open protocols, tools, and infrastructure can create great value for the business community. The SOA extends this value to cover the creation and sharing of business processes, utilizing Web protocols, tools, and infrastructure to meet this new objective.

Chapter 3

Definition of SOA

A recent Forrester report[1] included the following definition of SOA.

A style of design, deployment, and management of both applications and software infrastructure in which:

- Applications are organized into business units of work (business services) that are (typically) network accessible.
- Service interface definitions are first-class development artifacts, receiving the same degree of design attention (and more) as databases and applications.
- Quality of service (QoS) characteristics (security, transactions, performance, style of service interaction, and so on) are explicitly identified and specified for each service.
- Software infrastructure takes active responsibility for managing service access, execution, and QoS.
- Services and their metadata are catalogued in a repository and discoverable by development tools and management tools.
- Protocols within the architecture are predominantly, but not exclusively, based on industry standards (such as the emerging stack of standards around Simple Object Access Protocol or SOAP).

Forrester[1] notes that, "SOA is as much about software infrastructure design as it is about application design. Beyond simply packaging business logic as services, SOA infrastructure takes on responsibility for running services, leaving application developers more free to focus on business logic. Also, SOAP-based Web services are only one manner of service access — a very important one, but only one nonetheless. Other access paths include message-oriented middleware (MOM), distributed object protocols (such as CORBA-IIOP), and even nontraditional application connection paths such as e-mail."

As this paper describes, there is a lot more to an SOA service than an application that communicates via a MOM (or other access protocol noted above). SOA is about creating platform-independent applications by seamlessly integrating services deployed in a variety of environments.

1. Randy Heffner, *'Your Strategic SOA Platform Vision,'* Forrester Report, March 29, 2005

Chapter 4
SOA Characteristics

The job of building a business' Web presence has much in common with the effort and investment required to build an SOA presence. In both cases, the nature of the technologies allows the investment to be done incrementally, as justified by individual projects with long- range architectural strategies. SOA is not architecture for architecture's sake. It is tactics and strategy, combining technical and business platforms to exploit the opportunities offered by global business process integration. Just as there is no magic bullet for a business building out its Web infrastructure, an SOA also requires long-term vision, balancing investment against the business value it delivers.

Standards

First, rigorous conformance to standards is required. Standards are the building blocks of interoperability. In addition, standards allow a business to control the core architectural dependencies within its own environment.

Loose Coupling

A business must bring together an ever-evolving set of hardware, OS, tools, middleware, applications, and more that together function as its infrastructure platform. The degree to which an enterprise controls dependencies between all these parts determines how much flexibility it enjoys when optimizing the IT environment.

Accessibility and Reuse

To provide a business process as a service, the client view of the process must be radically simplified. Accessibility and radical simplification are common characteristics of successful SOA projects. Once a business process is made available via an SOA, it becomes a resource that can be more easily linked to or embedded within services. This shared services approach can deliver a degree of integration and flexibility not achievable by prior business integration architectures.

SOA Governance

It is impossible to embody a business function as a service without a clear understanding of how the information required by the service relates to its larger business scope. As services make business processes more accessible, confusion will result if the services' information and function are not effectively governed. If this meant that every nook and cranny of a business had to be normalized before an SOA could be used, it would be a major barrier to success. Instead, it means that each service must carefully consider what information demands it makes on its users, and how these radically simplified requirements should be governed. SOA governance provides architectural oversight of the business functions a service is allowed to offer and the services it is allowed to use.

Service Descriptions

To understand the SOA, it is important to understand core aspects of service descriptions, because these aspects uniquely enable an SOA to support continuous business optimization.

Service Interface

A service interface formally describes the function of a service and how it is accessed over the network. This formal service description is captured in the Web Services Description Language (WSDL). WSDL is the core metadata of an SOA, in much the same way Structured Query Language (SQL) table descriptions are the core metadata of a relational database management system (RDBMS).

Messages and Message Exchange

Service interfaces define the messages and message exchanges offered by a service. Messages are documents that flow between services. They emphasize the fact that an SOA is a data flow architecture rather than a procedural architecture. Services do not call each other — they flow messages which contain data that drives business processes forward, describe queries the business must respond to, and disseminate information required by business functions.

Conversational Services

Often a business action spans several message exchanges — some are initiated by the service's user, and some by the service's provider. This allows business processes to converse with each other, and offers a better digital model for the way those processes interact.

Synchronous versus Asynchronous

Technically, message exchanges can be synchronous or asynchronous. Services are allowed to respond more or less immediately — for instance, to provide a product's stock-on-hand information, or deliver a shipping notice for an order after the event has occured. It is worth noting that most business processes do not complete immediately, so their service interfaces typically involve a mixture of immediate acknowledgment-of-receipt responses, coupled with asynchronous responses describing the actions that happen later.

Chaper 5
An SOA Platform

An SOA strategy includes identifying the core platform collaboration standards it will use, and insuring that SOA platform vendors conform and collaborate via open, industry standards at both the protocol and services levels. Proprietary collaboration dependencies between SOA elements weaken the SOA platform by limiting flexibility and introducing risk.

While an SOA can include homogeneous or heterogeneous components, it is important to maintain an underlying modularity that delivers flexibility to mix and match components when designing an IT infrastructure tailored to an enterprise's requirements. This means that, while the path to an SOA may consist of incremental steps, it is essential to identify in advance the key areas needed to create a successful SOA platform.

An SOA platform provides the tools and underlying software infrastructure that enable a business to utilize the SOA. It is made up of platform services that utilize the same SOA principles and standards as the SOA business processes built with it, where each service is a point of platform collaboration. Since an SOA is an extension of Web computing, it has no central point of control — no single facility can impose its will over the platform. Instead, a diverse set of system elements collaborate with each other via shared platform services. Each element — Web server, Java Platform, Enterprise Edition (Java EE) server, registry, portal — must be able to collaborate with SOA platform services such as deployment, identity, management, monitoring, and more.

SOA Platform Design Centers

To clearly understand the structure of an SOA platform, it is useful to visualize it as being composed of five design centers:

- **Service Composition**
  Development, evolution, and governance
- **Service Control**
  Policy, management, and monitoring
- **Service Delivery**
  Mediate communication
- **Service Access**
  End usage of services
- **Composite Application Platform**
  Abstraction of business logic from multiple applications into an agile solution

The following sections explain the major functions provided within each design center.

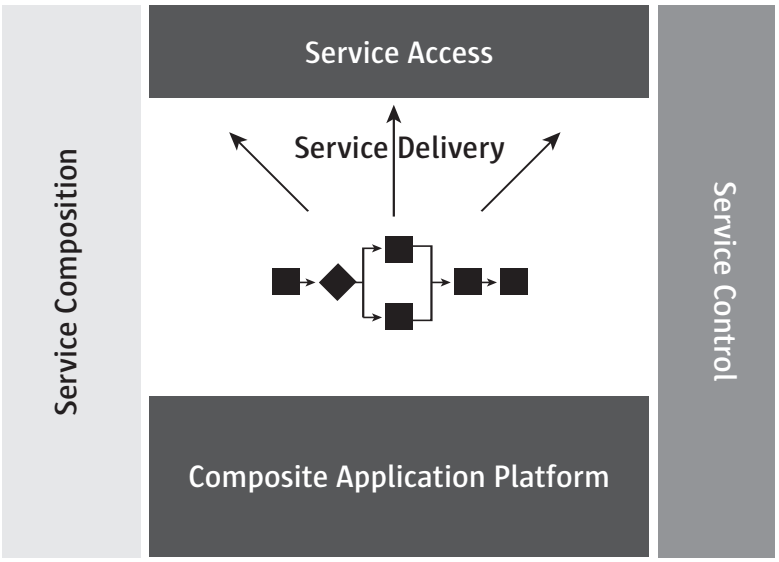


Figure 2. SOA Platform — Design Centers

Service Composition

The role of service composition is the design, development, and evolution of composite applications that embody an enterprise's business processes. An SOA platform's service composition design center is composed primarily of development and governance tools. It also contains platform services that allow these tools to collaborate, such as service metadata repositories, source code control systems, test automation systems, and so on.

While this sounds similar to how application development was achieved prior to the SOA, there are some important differences. First, the overall objective is the creation of a new class of application — the composite application — that is better suited to the job of implementing business processes than earlier application architectures. Second, service composition uses SOA design principles within its own architecture to collaborate with SOA platform services based on service composition standards.

The major functions of service composition are:

Service Design

Each service deployed to an SOA reflects one aspect of an organization's business processes. This means that service design, especially service interface design, must be closely aligned with and driven by business processes. Business process modeling establishes the requirements for service design.

Service interfaces are the main element of an SOA structure. While service interface design is more or less independent of service implementation, practical realities of IT projects lead to designing both in concert. Messages — their content, creation, transformation, and exchange — affect service design from top to bottom.

Designing the first version of a service is a challenge; however, it is only the beginning. Services are likely to be long lived, and must support both existing and new users as their capabilities evolve and expand. A service is typically a 24x7 commitment, and meeting this requirement requires tools that allow the business to understand, anticipate, and govern the impact of change.

Service Implementation

Programmed business logic remains an important method of service development. In addition, a number of service technologies can dramatically improve developer productivity, such as data transformation, business process, query, and business rule tools. These are far more than wizards that readily generate canned programming templates. They are full-fledged technologies on their own. While a service can be implemented with one of these, creating a service often requires them to be used together. A key function of service composition is the creation of these composite applications.

Composite Application Design

The core requirement of service interfaces is that they offer clear functional views to users, without unnecessary implementation detail. The core requirement of a service implementation is that it be delivered as a practical deployment unit — a composite application.

Composite application design includes identifying and integrating the service interfaces it depends on; selecting the service technologies it uses; organizing internal message processing into practical units; applying patterns and best practices; and so on. A benefit of the SOA is that it can be applied within a composite application, in much the same way it is applied at its service interface boundary.

Change Management

The benefit of services is that they are more easily shared across access channels and business processes. This increases efficiency, makes change easier, and improves control; however, it also broadens the impact of a service failure. This means that service stability is a critical deliverable of service composition, making the management of service change even more important.

In an SOA, there are numerous reasons to support running multiple versions of services, and it is not practical to assume that consumers (applications that consume these services) will upgrade in lock step with service change. In other words, service change should occur without breaking functionality provided by existing consumers. The objective is for a service to be designed from the outset to support a mix of older and newer consumers, so change can happen in place. This model of change-in-place is a core SOA principle. Without it, IT would be locked into managing a chain of services simply to support a legacy of consumers.

Service Registry Repository

To effectively support and govern service design and development, service composition requires a repository for organizing and accessing its metadata and data. The repository is a core SOA platform service used throughout service composition as the prime integration point between it and the other SOA platform design centers.

Service Control

The function of service control is to deploy, manage, and monitor SOA services, as well as the composite applications that implement them. It also oversees the SOA platform's underlying service delivery and composite application platform infrastructure. Service control includes business activity monitoring for visualizing the flow of business events emitted by business processes.

Important functions of service control are:

Business Activity Monitoring

Service control leverages the uniformity and enhanced accessibility that services bring to IT architecture as a means to deliver better visibility into the dynamic state of the business and business events as they occur. To be of value, this raw information must be continuously aggregated and reported to business management. Internally, services and access channels are specialized for this use, and include high-level tools for customization.

Identity and Access Management

Identity management is a function that touches everything within IT. Service control focuses on a slice of identity management that is applied to services. Service interface standards are designed to include many existing forms of identity; however, these work only within limited trust boundaries and may restrict service integration. An identity standard, the Security Assertion Markup Language (SAML), provides interoperable, federated identity, making it a key service control technology.

Controlling access to services is crucial. Access management relies on service interface descriptions that accurately relate their data and actions to the business processes they implement.

Together, identity and access management underlie IT's ability to meet legislative and regulatory compliance requirements. Each requires tailored auditing and reporting functionality. This is an area of increasing focus for IT.

Service Deployment

Business functions are typically deployed as project units. These may contain several composite applications, and a project may span several application platforms. The job of service control is to coordinate the deployment of IT projects so the business change they deliver is reliably installed. The uniformity of service interface and platform standards helps standardize IT project deployment.

Service Monitoring

IT must monitor multiple levels of elements within an SOA to ensure that throughput and availability objectives are met. The uniformity, access, and metadata provided by service interfaces, as well as instrumentation provided by service delivery, help service control provide a better functional view of its services. By linking this service view to the underlying composite application view, service control provides the information IT needs to react to changes in loads, bottlenecks, exceptions, and so on that occur within an SOA.

Service Level Management

Service interfaces make it easier to instrument and manage service levels. It is possible to instrument throughput and availability at the business process level while managing operations to this gauge.

Service Registry Repository

Service Control includes a service registry service that provides the service metadata describing how to use a deployed instance of a service. It often provides the ability to categorize service instances in various ways relevant to the business. This metadata can be employed by access channels to dynamically find and invoke a service instance. It can also be used for governance functions.

Policy Enforcement

A policy defines a course of conduct for a class of target, and policy enforcement insures that all instances of the class follow it. For example, a policy could set computing resource limits for a class of composite applications, security for a class of service interfaces, the role an identity must have to use a class of service, and so on. Service control provides the vocabulary for describing these policies, as well as tools for managing and enforcing them.

Service Delivery

Service delivery is the main architectural element that differentiates the SOA from previous generations of application integration architectures. It is the center of connection and communication in an SOA platform, and all communication between SOA services is mediated by this center. Service delivery implements protocols employed by services to ship messages to one another. The operational view of service delivery is provided by service control.

Important aspects of service delivery are:

Service Delivery Protocols

As noted in the SOA definition section, service delivery may support a wide range of protocols. A key service delivery requirement is SOAP, which provides the broadest service interoperability. Since SOAP is based on the eXtensible Markup Language (XML), it can be extended, and the industry is actively adding to it. The Web Services Interoperability Organization (WS-I) describes a conservative profile of SOAP that is stable and interoperable.

Service Delivery and MOMs

Many IT organizations have existing Message Oriented Middleware (MOM) systems. Service delivery typically supports their use in restricted ways that leverage its reliable queueing functionality to increase SOA messaging Quality of Service (QoS). Since a MOM uses proprietary protocols, interoperability is limited. Service delivery may also employ a MOM behind a more interoperable service interface channel to add queueing at one end or the other. As SOAP evolves to provide improved reliability, service delivery will directly provide queueing and the use of MOMs will decrease.

Security

A core function of service delivery is security. Security of interface channels, messages, and data elements within messages is a major focus of Web service standards and service delivery. While the details are too complex to discuss here, it is the general consensus that all service channels should be secured by default. But this is not enough, because as restricted data passes through services, the identities involved (both programs and individuals) must be formally verifiable and auditable, and end-to-end access control must be maintained.

Message Processing Optimization

XML messages can become quite large because XML is biased towards clarity rather than compactness. For some services, message size can be a performance limitation. The industry solution is the Fast Infoset standard created by the ISO and International Telecommunication Union (ITU). Fast Infoset compresses XML while optimizing the balance between processing and size overhead. Although this is not yet a service delivery requirement, it will likely become one.

Customization of Service Delivery Channels

Often a service and its consumers reside within the same SOA platform. In this case, service delivery spans both ends of these local channels and therefore, it can further optimize them. While service delivery does not execute services, it does execute some data transformations, rules, policies, and others that are best applied within the context of a service delivery channel.

Service Access

A fundamental requirement of an SOA, service access ensures that services can readily be consumed by the human or system actor. It consists of a citizen's portal, Web application, mobile client, Radio Frequency Identification (RFID) channel, and more, which consume services made available via service delivery channels in an SOA platform. The nature of the service access channel is closely tied to the business environment where it is actually used. Important aspects of service access are:

Service Clients

Services can be consumed by a variety of clients such as portals, Web applications, mobile devices, RFID channels, and more. These clients are linked to the service delivery channel via open standards and protocols. For example, portals traditionally played the role of important service access channels by providing end-user access to services in an aggregated, personalized, secure, and reliable manner, supporting a wide variety of user agents and devices, including mobile devices.

Service Views

If a service is available for interaction with an end user, it typically manifests itself as a service view. A given service may be exposed via multiple views for finer granularity. On the flip side, it is possible to aggregate multiple views across different services to form a composite view. This composite view may even enable coordination among its individual views, so they look cohesive to the end user. These service views may be assembled with the rest of the composite application during service composition.

Service views can be facilitated as industry-standard portlets within the portal framework. Or, they could even be part of a traditional Web application.

Within the context of portals, standards such as Web Services for Remote Portlets (WSRP) can enable service views to be distributed onto other environments in a plug-and-play fashion. To further the composite view concept, distributed views from diverse sources could also be aggregated.

Overall portals provide infrastructure that effectively and efficiently makes these service views available, either locally (as portlets) or distributed (as remote portlets).

Composite Application Platform

One way to minimize risk when transitioning to an SOA is by leveraging existing applications — rather than utilizing the rip and replace approach — concurrently implementing new applications within the new SOA. A key component of this low-risk approach is the inclusion of composite applications. They are a collection of existing and independently developed applications, information stores, and abstracted business logic orchestrated into a single solution. The abstraction of business logic out of legacy or packaged applications and into an easily modifiable service composition layer is a key reason why composite applications are an integral way to introduce agility into an SOA. For this reason, SOA platforms must support composite applications.

Important aspects of the composite application platform are:

Application Platforms

Today's IT applications use a variety of application platforms — integration server (sometimes called Enterprise Service Bus or ESB), application server, Web server, portal server, platform embedded within a packaged application, RDBMS, and so on. It is the job of service composition, service delivery, service access, and service control to paper over the discontinuities between these application platforms.

Complexity is reduced if IT can deliver a project as one — or at most, a few — composite applications. When a project's needs cannot be accomplished with a single application platform, it must be split into multiple applications that are composed across their platforms. Typically this is done by making each application a service and then composing them via the SOA.

While service composition is a core benefit of the SOA, the last thing IT wants is to create extra services simply to span technology gaps within an IT project. The closer an application platform comes to being a full composite application platform, the fewer IT resources are wasted on bridging the lack of integration between application platforms.

Adaptors to Existing Applications

Service delivery should not be confused with the various adaptor frameworks supplied with application platforms, which do whatever it takes to communicate with a wide variety of existing IT resources through whatever channel they supply. These adaptors still play an important role in an SOA platform, as they allow a service to interact via a raw channel with IT resources. One of the steps that marks IT's transition from Enterprise Application Integration (EAI) to SOA is the transition from raw channels to service interfaces. Often, this is accomplished by using a service to wrap a raw channel. In other cases, vendors provide this wrapping. While an objective of SOA is to reduce IT's dependence on raw channels, their use will likely persist for some time.

Java Business Integration

The recent Java Business Integration (JBI) standard is an important step taken by the Java Community to architect a composite application platform that extends beyond today's application platforms. Many integration vendors and application server vendors in the Java Community plan to support JBI. It will allow today's application platforms to technically mix and match their service technologies, so IT can minimize the complexity of the services it develops. JBI also speeds the addition of new functionality by leveraging standard and open interfaces and the JBI partner ecosystem, promoting desired functionality without the risk of vendor lock-in.

Interoperable Java Technology and .Net

Today, SOA services implemented on the Java Platform can interoperate with services on the .Net platform via Web services standards. The Java Platform is committed to delivering comprehensive Web services interoperability with all platforms that do the same. In addition, Sun is committed to delivering comprehensive .Net interoperability that extends comprehensive support for Web services standards with .Net-specific interoperability.

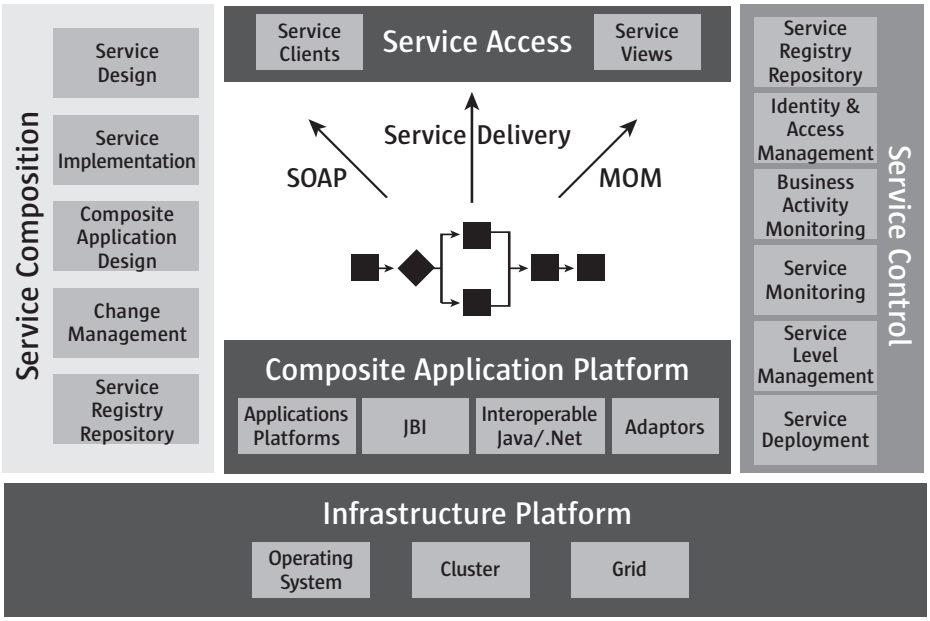


Figure 3. SOA Platform Components

Chapter 7

Sun SOA

Given the complexity of business processes and the IT architecture that implements them, IT's transition to SOA cannot be accomplished in a single step. The success of SOA is due to its ability to understand the core of each business, and create an IT infrastructure that adapts to dynamic enterprises. This requires an extensive analysis of business processes to define the right set of services and how they fit your business. It also requires providing employees with the skills they need to ensure clear communication between the business and IT, demystifying the business process, and translating it into a service oriented design. Depending on the immediate needs of your enterprise, the path to SOA might begin with small, incremental steps based on best practices and solution frameworks. The true SOA solution adapts to your needs and leverages existing IT investments. To do so, the SOA platform must be integratable, requiring multiplatform support for both hardware and software.

Sun is uniquely positioned to facilitate your transition to a service-oriented enterprise by sharing knowledge gained through years of experience, as well as best practices that evolved through helping companies meet their integration needs. Sun's standards-based product offering, the Java Enterprise System, is a leading SOA platform that leverages standards to create a heterogeneous architecture that accommodates your existing IT investments, while helping launch next-generation services.

The Sun Java Enterprise System consists of eight suites: the Sun Java Composite Application Platform Suite, Sun Java Identity Management Suite, Sun Java Application Platform Suite, Sun Java Web Infrastructure Suite, Sun Java B2B Suite, Sun Java Enterprise Service Bus (ESB) Suite, Sun Java Availability Suite, and Sun Java Communication Suite. These satisfy the needs of enterprise customers looking for core integrated features delivered as modular suites of integrated products. The Java Enterprise System helps enterprises overcome the technical complexity of business integration by offering an industrial-strength platform strategy that ensures superior QoS through multiplatform support. It also delivers a high degree of consistency among its component offerings.

For example, the Java Application Platform Suite supports interoperability between Java technology and .Net; runs on the Solaris™ Operating System, Linux, HP-UX, and Microsoft Windows; and supports the Universal Description, Discovery and Integration (UDDI) and Electronic Business XML (ebXML) standards. Service registry and repository (offered as part of the Application Platform Suite) enables secure, federated information management. The Identity Management Suite helps enterprises address SOA governance and larger IT compliance objectives through end-to-end, information life cycle management.

The Java Composite Application Platform Suite is comprised of four suites, including the new Java B2B Suite and Java ESB Suite, in addition to the existing Java Application Platform Suite and Java Web Infrastructure Suite. This complete solution delivers a proven SOA foundation that is designed, developed, managed, and monitored through a common tool set. It can be deployed into a common runtime environment or exposed as individual Web services.

The four Java Composite Application Platform Suite components include:

- *Java B2B Suite*
  Enables efficient management of diverse business-to-business (B2B) requirements across customer supply chains, distribution networks and clearinghouses.
- *Sun Java ESB Suite*
  Advanced distributed enterprise service bus through open standards support for Web services, XML, transformation, and intelligent routing that helps simplify complex and costly messaging infrastructures.
- *Sun Java Application Platform Suite*
  Addresses the challenges of manageability, performance, security, and availability of business-critical applications and processes. It enables process-driven portals delivering role-based composite applications, providing authorized information through a single user experience.
- *Sun Java Web Infrastructure Suite*
  Improves Web security, strengthens online customer and partner relationships, and reduces Web operations costs.

Sun's SOA platform also provides support for a wide range of hardware platforms, including SPARC®, x64, and x86 technology. Important benefits of SOA are its ability to connect disparate systems in a seamless manner, and plug in components to the SOA platform that are in line with the enterprise IT strategy. To this end, Sun leverages its rich partner ecosystem to help our customers build truly heterogeneous platforms.

SOA and Data Center Architecture

Sun has extended the principles of the SOA into data center architecture. Time to market, business ability, reduced cost, predictable service levels, and an evolutionary approach to transforming IT are all concepts being realized within Sun's vision for the data center.

In Sun's SOA data centers, application building blocks are being deployed using Java Enterprise System components as network service utilities. QoS is managed with technologies such as Sun Cluster and Sun Enterprise™ servers which deliver high Reliability, Availability, and Serviceability (RAS) capabilities. Sun methodologies, such as the Service Delivery Network, securely partition and assemble infrastructure components into business services that are consistent with patterns prevalent in SOA design. And through virtualization technologies such as Solaris 10 Containers and Sun's Application Switch, abstraction is created in both the operating environment and network architecture. Using the Sun N1™ provisioning system brings a dynamic response to the data center, enabling it to rapidly respond to changing business conditions.

Chapter 8

Summary

Leadership — even survival — in today's highly competitive business world depends on agility born out of concept innovation as well as the enterprise's ability to adapt to the ever-changing needs of its customers. When companies began to set up Web sites, they made it easier for customers to do business with them. An SOA enables enterprises to make their business processes more transparent to customers, and deliver additional services in shorter timeframes. Providing a better set of services enhances the customer experience, creating additional revenue without adding costs to the bottom line.

Quite simply, the SOA is the future of IT services. The SOA is a mature framework that is gaining in popularity, in large part due to the availability and stability of the standards, technologies, and products that support it. A Web services-based SOA enables integration of virtually all IT resources, including isolated data silos and previously incompatible legacy, .net, and Java technology applications. Thanks to the depth of Sun's knowledge and service offerings that leverage years of experience, as well as best practices developed by helping companies meet their integration needs, Sun is uniquely positioned to facilitate the transition to a service oriented enterprise, and can supply everything your enterprise needs to get started on the transition today.

Product Information

Products are listed by design centers.

Service Composition

- Java Studio Enterprise
- Java Studio Creator
- Sun Studio
- Sun SeeBeyond eInsight™ Business Process Manager
- Sun SeeBeyond eVision™ Studio
- Sun SeeBeyond ePortal™ Composer
- Sun SeeBeyond eView™ Studio
- Service Registry

Service Control

- Service Registry
- Java System Identity Manager
- Java System Access Manager
- Sun SeeBeyond eBAM™ Studio

Service Access

- Java System Web Server
- Java System Web Proxy Server
- Java System Portal Server
- Java System Portal Server Secure Remote Access
- Java System Portal Server Mobile Access

Composite Application Platform

- Java System Application Server Enterprise Edition
- Sun SeeBeyond eGate™ Integrator
- Sun SeeBeyond eTL™ Integrator
- Sun SeeBeyond eXchange™ Integrator
- Sun SeeBeyond eWay™ Intelligent Adapters
- Java System Directory Server Enterprise Edition
- Sun Cluster
- Cluster Agents
- Sun Cluster Geographic Edition

For More Information

- Service Oriented Architecture, www.sun.com/soa
- Sun's Service Registry, www.sun.com/products/soa/registry/
- "*Java Business Integration Vision*" white paper, www.sun.com/products/soa/resources.jsp
- "*Effective SOA Deployment Using an SOA Registry Repository*" white paper, www.sun.com/products/soa/resources/jsp

Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 USA **Phone** 1-650-960-1300 or 1-800-555-9SUN **Web** sun.com

02/06