

A Practical Guide to SOA for IT Architects

January 2005

A Practical Guide to SOA for IT Architects

A Systinet White Paper

Copyright © 2005 Systinet Corp. All rights reserved. The document is not intended for production and is furnished as is without warranty of any kind. All warranties on this document are hereby disclaimed including the warranties of merchantability and fitness for a particular purpose.

Trademarks

Systinet™ and the Systinet logo are trademarks or registered trademarks of Systinet Corporation. All other company, product and brand names are trademarks of their respective companies.
December 2005

Systinet Corp. One Van de Graaff, 5th Floor Burlington, MA 01803 Phone: 1.781.362.1300
www.systinet.com

A Practical Guide to SOA for IT Architects

Contents

Executive Summary	4	Business Service Lifecycle Planner	10
The Business Value of SOA	4	Business Services Registries Enable Lifecycle Management	11
Key SOA Design Considerations	4	The Evolution To SOA	11
Creating Your SOA Foundation:		Appendix A:	
The Business Service Lifecycle	7	Web Services Enablement Platforms	12
Planning	7	Examples of Web Services Enablement Platforms	12
Enablement	8	Appendix B: SOA Defined	12
Publishing	8	Appendix C: Glossary	13
Discovery	9	About Systinet Corporation	14
Management and Security	9		
Analysis	9		

Executive Summary

As a strategy for creating a flexible and agile IT, service-oriented architecture (SOA) has gained considerable momentum in recent years, largely due to the advent of standards-based Web services. There are some powerful business drivers for implementing SOA today:

▶ IT Agility and Lower Costs

SOAs make IT more responsive to changing business demands and requirements. Reconfiguring business services is simple, fast and low-cost.

▶ Maximizing Enterprise Application Investments

SOAs are not a rip-and-replace strategy—they wrap and reuse business functions from existing enterprise applications and make them available to a significantly broader audience without change. SOA encourages reuse and avoids unnecessary duplication and reinvention.

▶ Facilitating New Applications

Web service-based SOAs smooth the development and management of composite applications. Composite applications improve business performance by pulling together information from multiple applications without complicated and time-consuming IT processes.

▶ Standards—Foundation for the Future

SOAs that use standards-based components and interfaces provide ubiquitous interoperability with all applications and services, making IT more flexible and dramatically simplifying integration.

This paper discusses how to capitalize on the advantages of SOA by adopting an enterprise foundation for SOA governance and business service lifecycle management. The paper presents a management framework for enterprise architects and acts a guide for implementing an SOA. In addition, it reviews the technical, organizational and process issues involved, and offers recommendations on building an SOA infrastructure.

The Business Value of SOA

SOA is an architectural style for maximizing application interoperability, sharing and reuse in a distributed environment. Service orientation is not a new approach to software development—implementations of object-oriented design, message-oriented middleware, and component-based development have all been guided by many of the same principles.

The widespread adoption of Web services standards has reinvigorated the service-oriented approach by providing a universally accepted set of interoperability standards for building, describing, cataloging and managing reusable services. Earlier models required systems to adapt to proprietary software interfaces or to implement complex standards that failed to garner industry support. [Web services technologies, protocols and platforms are discussed in Appendix A.]

As an example, message-oriented middleware provides a basis for an enterprise architecture composed of services designed to interoperate based on well-defined messages transmitted across well-defined message queues. However, this approach requires all systems in the architecture to support the binary message formats and protocols of the messag-

ing provider, and involves major rework to change providers or even to upgrade between versions of a single product.

In a Web services environment, these same services communicate using standard protocols and formats supported by multiple vendors. Messages are XML documents described using standard XML schemas. Service interfaces are defined using WSDL definitions and implemented using SOAP messaging, with additional functionality provided by the layered WS-* standards. Changing technologies or versions is simplified by the lightweight nature of the standards and the overwhelming adoption of Web services standards by technology vendors and corporate developers.

SOAs have some distinct advantages over other architectures. First, interoperability is an innate characteristic of a service-oriented approach. As described in the example above, SOAs built using Web services support service integration based on universally accepted industry standards. When interoperability is an inherent characteristic of all IT systems and purchased software, the problem of application integration becomes moot. Organizations no longer need to invest inordinate amounts of time and resources on projects that connect applications, freeing the IT team to focus on providing new business functionality.

Second, SOAs make enterprise applications more agile and more responsive to changing business demands. By keeping IT focused on projects that provide business value rather than technical interoperability and upgrades, SOAs allow teams to quickly respond to business initiatives.

Gone are the days where no one is available to respond to business change because they are working on the latest middleware upgrade or adapter projects. Coarse-grained interfaces to functionality allow business analysts and stakeholders to better understand the functionality available to them, and loosely coupled services allow them to freely compose services into applications with minimal IT involvement.

Freed from the technical minutiae of your chosen platforms, with an empowered base of business analysts and stakeholders, the IT team can think in terms of business functions and react quickly to changing business demands, requirements or processes.

Key SOA Design Considerations

As your organization begins the SOA planning, development and deployment process, there are a number of things to bear in mind. IT architects should consider the following issues as they approach an OA rollout:

Plan for Incremental Deployment

Unlike many architectural initiatives, an SOA can be deployed incrementally and still show business value. A project that must integrate multiple systems implemented with different technologies is an ideal place to start an SOA initiative and quickly demonstrate value.

For example, if you're developing a new VB.NET desktop application that must make use of business rules that were implemented for an earlier J2EE application, you can use Web-services technologies to enable the new application to directly consume business services provided by the



The Stages in SOA Adoption

Systinet defines three stages of evolution to a service-oriented architecture.

Phase One: Web Services Enablement

This is a tactical implementation of Web services where existing applications have standard Web services interfaces.

Replacing proprietary APIs reduces integration complexity for these applications. These developer-driven implementations have modest returns on investment in the short term.

Phase Two: Business Services Enablement

Success with simple Web services encourages a systematic, enterprise-level approach in the next phase. This architect-driven phase requires added visibility, compliance, governance and manageability to transform Web services into business services with a focus on reuse. The primary benefit of reuse is increased alignment to business, and consequentially, improved agility to facilitate business change. Impact analysis and reporting are key outcomes.

Phase Three: Dynamic Interoperability

The desire to support composite applications and agile or "on-demand" computing drives this phase, which is characterized by a concept Systinet calls Dynamic Business Interoperability. Here the Web services platforms are able to create more intelligent business services (i.e., endpoints) with support for advanced security, reliable messaging, and policy enforcement. Then, by managing business services using high-level business and application definitions called metadata, change to IT or business processes can be dynamically managed, further increasing agility and dramatically lowering operational costs.

J2EE application. If you also need to implement the system with packaged applications such as SAP, you may find that the software provider has already developed Web service interfaces that you can exploit. If you must introduce logic that exists in other legacy systems written in C++ or residing on a mainframe, you can wrap these systems as Web services and exploit their high-level business logic without implementing adapters or relying on low-level APIs.

By wrapping your existing functionality as Web services, you're minimizing your project-level integration effort and contributing to the evolution of your SOA with the same resources required to make the project a success.

Focus on Interoperability

Innate interoperability is a key benefit of SOA. Initially, you should focus on standardizing message payloads with XML schemas, describing operations using WSDL service descriptions, and implementing a Business Services Registry. This will allow you to control the evolution of your SOA by specifying services, operations and data types, using the registry to verify that each implementation follows the standards. It also reduces the reimplementing of common services by making teams aware of what services are being developed elsewhere and how they can be used.

As your SOA encompasses more and more mission-critical processes, you should identify higher-level standards for non-functional requirements such as security, reliability, transactional integrity, monitoring and management, as well as load balancing and service provisioning.

Requirement	Standards
Security	SSL, WS-Security, XML Encryption, XML Signature
Message Routing and Reliability	WS-Addressing, WS-ReliableMessaging
Transactional Integrity	WS-Transaction
Monitoring and Management	SNMP, WS-DistributedManagement
Load Balancing and Service Provisioning	WS-Addressing, SPML, WS-Provisioning

Some of these standards will already exist in your architecture. Some of the newer standards can be implemented in hardware, but most will be provided in software running at your Web services endpoint. Be wary of centralized solutions for implementing Web services standards, as they can limit your system agility and require allocating resources to maintain these single points of failure.

As you identify standards and define policies for these additional requirements, your Business Services Registry provides the infrastructure for a robust governance model to ensure that these standards are employed in the services as part of your SOA. By defining your policies using the WS-Policy framework, you can ensure at the service endpoint that they are followed.

A Practical Guide to SOA for IT Architects

Focus on Business Agility

Business and IT agility should always be the primary and overarching goal of your SOA strategy. As your SOA evolves, IT systems begin to mirror business processes, making it easier to map business change to system change.

SOA infrastructure makes it easier to implement IT change because systems are composed of loosely-coupled business services that depend only on well-defined discrete interfaces, preventing side-effects and internal dependencies that hinder component reuse. Changes to services should not cause connections between services to fail, allowing processes to be quickly reconfigured without changes to service implementations.

For example, if your payroll processing system is dependent upon a business service defined by your 401(k) provider, it should interact with that provider only through clearly defined service boundaries with no dependencies between the service implementations. This will allow your business to switch plan providers without changes cascading deep into your payroll logic. If your 401(k) provider uses industry-standard XML schema and service interfaces, you may be able to change your provider with no impact on the underlying payroll system.

The business services that compose an SOA represent a coarse-grained view of IT or application assets—i.e., with services defined around high-level business concepts rather than low-level technical details. This allows business analysts to easily understand and work with business services to implement change without turning to IT. With analysts and business unit developers making use of these higher-level services to adjust to business changes, the IT team is freed to work on building more and more services that deliver direct business value to multiple stakeholders.

Recognize the Registry's Critical Role in SOA

The basic SOA model is defined by interoperable service providers and service consumers teamed with a Business Services Registry.

Providers and Consumers use services in the SOA according to predefined business policies. The Business Services Registry provides a system-of-record for these policies, and is used to define and enforce business policies. The registry is employed proactively to define the interfaces and policies for individual services before they are implemented, and reactively to verify that project teams are publishing services to the registry that adhere to the policies you have established. At the project level, teams discover the policies required to interoperate with a service before they work with it. This prevents the confusion and schedule slips that are common when the requirements for interoperability are poorly understood.

Beyond these baseline considerations, other issues will become important as the SOA is incrementally deployed across the enterprise. These include the following.

Define and Enforce Application Interoperability Policies

An organization must define an interoperability architecture and policy to manage all integration efforts. With Web services-based SOAs, the interoperability protocols of SOAP, WSDL and UDDI provide the core of this architecture. The architecture and its policies must further define the protocols to meet higher-level interoperability requirements. To prevent a point-to-point integration model from overtaking your SOA efforts, you should define a reference architecture that supports services interoperating in a flexible and reusable fashion.

Transform Your IT Development Processes and Policies

SOA represents a way to drastically improve IT processes, especially in application development. With solid architectural guidance, development teams can create modular, component-based applications using Web services design standards. You can enforce compliance to WS-I and internal standards at design and runtime using available Web services management (WSM) and SOA enforcement tools. Your Business Services Registry is a key component for design-time enforcement, allowing development teams and enterprise architects to clearly state and review compliance for each service. With composite applications, the registry can streamline compliance reviews by providing a picture of the standards compliance for each constituent service.

Define and Enforce Your Business Interoperability Policies

As your organization expands its services portfolio, you will increasingly interact with other business services from customers, business partners and other organizations within your enterprise. The resulting business integration challenge will be easier to manage when your SOA defines a policy for B2B interoperability and a strategy for leveraging industry XML standards.

Monitor, Measure and Analyze Your SOA Service Network

As mentioned previously, SOA metrics should be defined early in your SOA strategy and planning cycle. These metrics should help you determine the overall effectiveness of your SOA by answering the following kinds of questions:

- ▶ Are the services leveraging one another in a symbiotic, networked fashion?
- ▶ Are we getting the maximum interoperability and services reuse from our SOA? If not, why not? Are policies being enforced at design and runtime?
- ▶ Do we have a truly interoperability-based SOA, or do we have islands of business and Web services? How can we unite these services?
- ▶ Are our business services secure and reliable to the level required by the business processes using them?
- ▶ Finally, have initial SOA business goals been met? How can performance be improved? If goals were not met, why not? What must be done?



SOA Governance

SOA governance consists of the corporate, business and IT processes and rules required to control the business success of an SOA and Web services. SOA governance defines and enforces the Web services policies needed to manage SOA applications and data for business success.

Examples of SOA Governance

The SOA governance model will result in policies for services reuse, IT compliance and security.

Reuse: A reuse policy would describe technical and business aspects of services reuse such as WSDL design conventions and WS-I compliance for interoperability, as well as a review process to ensure that existing services are used prior to developing new ones.

Compliance: An SOA compliance policy would be a governance policy describing what internal and industry standards will be followed for all services, whether internal or from external providers.

Security: A security policy might specify what security standards and credentialing processes will be enforced during services design and consumption, such as SAML, WS-Security, XML signature, and others.

Creating Your SOA

Properly implemented, SOA is an excellent approach for managing change—change of service owners, service implementations, provider-consumer relationships and even entire business processes. The key to managing change in an SOA is managing the lifecycle of constituent services—what we call Business Service Lifecycle Management.

The business service lifecycle is a defined set of activities and procedures that continuously support SOA business service development, deployment and management. The cycle complements your software development lifecycle and extends it for services runtime deployment and management. It can also be tailored to meet the specifics of your approach.

For example, for some traditional development lifecycles, the architect may define specific services and the policies around them during the plan phase of the service lifecycle. In modern agile software development lifecycles, a project team will define some basic policies for services during the plan phase, and service interfaces and specific policies are defined as part of the enable phase.

Consistent with SOA principles, the business service lifecycle recognizes the need to achieve defined business outcomes. It acknowledges that there are new and unique requirements to help an organization successfully employ an SOA and ultimately manage dynamic business interoperability.

Many will recognize these steps as the way business is done and IT systems are developed today. With previous architectures focusing on reusable service libraries, a major issue is the failure to define simple and standard mechanisms for informing the application development community of the available services and their interfaces. The robust publishing and discovery model elevates the importance of sharing reusable services and assuring that business analysts and developers can find the services they need to implement business functionality. The publishing and discovery steps are mandatory for successful SOA implementation and handle the integration of design and runtime environments inherent in an SOA.

Planning

As in any development process, planning is critical—in this case, planning the transition from Web services to reusable business services. Here are some considerations:

Business Services and SOA Governance

Business services are Web services operating in an SOA, augmented with the necessary compliance, governance, security and manageability policies that enable enterprise-wide use and reuse.

Effective governance involves mapping corporate, business and IT policies to specific SOA business services, and then ensuring policy enforcement. Storing policy compliance information in your Business Services Registry precludes compliance issues by informing service producers and consumers of their obligations before services are implemented. Describing policies using WS-Policy assertions allows service endpoints and management tools to further ensure compliance at runtime by disallowing non-compliant uses of services.

A Practical Guide to SOA for IT Architects

A clearly defined services lifecycle ensures that governance policies are established and followed at each phase of service development. The services lifecycle infrastructure ensures proper implementation and operational support during service enablement, publishing, discovery and management.

Architects determine policies for security, interoperability (such as WS-I), data formats (such as FpML or ACORD), and other internal and external processes and procedures.

SOA Metadata and Classification Management

The application of SOA metadata to fully describe services is essential to ensure that you get the most visibility, productivity and reuse from your SOA. Classifications and taxonomy modeling are a critical step in the development of a robust SOA. You must identify the key metadata surrounding your services and define taxonomies that represent the valid classifications of services according to each metadata category.

A solid SOA defines clear taxonomies for services across multiple dimensions. These dimensions can encompass technical constraints such as standards compliance, organizational metadata such as business unit or line of business, and regulatory compliance criteria such as HIPAA or Sarbanes-Oxley standards. The Business Services Registry provides a mechanism to define taxonomies, publish services with the proper classification, and discover services based on how services are classified.

Taxonomy modeling is similar to other typical data modeling, and you can take an incremental approach to defining your taxonomies. Your taxonomy model can also take advantage of the effort that has gone into developing your existing data, organizational and process models.

New SOA Component Requirements

An SOA requires new infrastructure, including a Web service enablement of endpoints, a registry, Web service management, and security/identity management. A key aspect of this infrastructure is the Business Services Registry, which provides a centralized location for managing all the descriptions of services and related SOA information. In fact, the Business Services Registry provides the central point of SOA governance to manage policies and drive enforcement at the services level, essentially becoming the "system of record" for the SOA.

Enablement

SOA enablement is two-fold: the installation of new SOA infrastructure and the process of developing standards-based business services. At its most basic, the enabling infrastructure can be simplified as follows:

- ▶ Web services enablement of endpoints for service providers and consumers
- ▶ Standards-based Business Services Registry supporting governance and life cycle management
- ▶ Supporting infrastructure such as Web services management, identity management and service-oriented messaging services

Web Services Enablement of Providers and Consumers

For a service provider, enablement involves the core Web service enablement, along with publishing business service definitions, descriptions and policies to the Business Services Registry. It also involves some aspects of service management. These include instrumenting source code to gather metrics, updating service descriptions to assert policies, and tracking uses of the service to facilitate dependency analysis.

Like service provider enablement, service consumer enablement goes beyond core Web service enablement to include approaches to service discovery and policy negotiation. A service consumer must be able to discover providers using standard mechanisms such as UDDI.

Service consumers may also need to negotiate with service providers at either the business or technical level, according to predefined, acceptable policies. These policies may be described directly in the WSDL service description using WS-Policy, or by querying metadata from within the registry.

Service consumers must be able to accommodate changes from service providers, reliably and on demand. It is inevitable that the service provider will need to be updated and maintained independently of the service consumer.

Business Services Registry

The Business Services Registry is the centerpiece of an SOA infrastructure, providing the single thread of visibility and control of all service interoperability activities and related information across the lifecycle. The registry should adhere to Web services standards such as UDDI. Given its centrality in an SOA, the registry should provide functionality above and beyond the UDDI standard to support advanced classification management, security features, and mapping of SOA information such as WSDL and XML schemas according to best practices. Since no two businesses are exactly alike, the registry should provide the ability to configure and customize its view of your SOA to support your policies and taxonomies.

Publishing

Publishing services in an SOA is a policy-driven process that in some ways mirrors your software release process. The publishing process must answer questions such as:

- ▶ Who is allowed to publish a service to the registry?
- ▶ What release procedures must be followed?
- ▶ How will various designs, standards and security policies be approved, certified and enforced in the SOA?

These are all issues that are typically managed and controlled by a Business Services Registry. Publishing is critical in the lifecycle because Web services are not truly "business" services until they can be discovered and shared by all SOA participants.

The registry must support diverse user communities, and your modeling must take this diversity into account when developing classifications that contextualize services for various audiences—developers, business analysts, IT managers and line-of-business management. For example, devel-

A Practical Guide to SOA for IT Architects

opers might search for services based on the operations they provide or the protocols they require. Business analysts may look for services based on the document types they consume or business keywords that define the purpose of the service.

The references and taxonomies used by one audience may not provide the proper context and meaning for another. The registry's ability to provide meaning and context to services in an SOA is critical to increasing usage, supporting service discovery, and maintaining control of services in the SOA.

The publishing process should also support clear workflows to ensure that services are published only if they are compliant with your SOA policies. The Business Services Registry should facilitate these workflows and allow architects and managers to review services before they are made available for broader reuse. Each workflow should include a step to ensure that the service implements the standards asserted in the service registration.

Discovery

An SOA is predicated on the concept of discovery. The notion of reuse for the purpose of optimizing agility and alignment of IT to business is fundamental to an SOA.

Standards-based registries use the UDDI protocol to support the sharing and discovery of services. The UDDI specification provides a flexible approach to mapping all your business service classifications to the registry, ensuring standards-based discovery. A standards-based registry permits your organization to leverage existing services and support the composition of multiple services to meet new business requirements. It facilitates service reuse while providing a foundation to accelerate time-to-market for new business and IT functionality.

Management and Security

SOA management, or Web services management (WSM) solutions, provide control for an SOA. This includes:

► Management

Monitoring (e.g., metric computation, service-level objective evaluation)
Automation (e.g., deploy, un-deploy and upgrade) Auditing and utilities (e.g., alert notification and logging)

► Security

Message Integrity Message Confidentiality Single Message Authentication

► Enforcement of policies

To be effective, an SOA deployment must include these management and security capabilities. Some may be provided by your existing management infrastructure. For example, if your SOA enablement infrastructure supports SNMP, you can use your existing management console. If you're IT infrastructure supports SSL with client-side certificates, you may be able to use SSL to secure your messages.

However, the nature of SOA drives further management requirements. SOA management is about managing and understanding the relationships between interoperating business services, regardless of platform. The textual nature of XML on-the-wire within your SOA suggests that management tools allow visibility into message traffic, and the use of standard messaging interfaces allows message traffic to be easily rerouted through managed service-endpoints and on to their original service destination. These managed endpoints can provide runtime policy enforcement, dependency tracking and exception management. Additionally, through monitoring services, interdependencies can be determined automatically, providing visibility and allowing change management/impact analysis to be performed quickly and easily.

The textual nature of XML introduces new concerns for message security. Messages can easily be intercepted from any network, but a text based network makes the message payloads easier to interpret. To secure message payloads, you can explore security both at the transport level, e.g. by using SSL, and also at the message level using WS-Security and the related XML Encryption and XML Signature. With these standards, you can secure messages directly within the XML payload using public-key encryption and digital signatures.

After you place initial management and security infrastructure into your SOA, you need to consider the evolution of service producers and consumers. Since service producers and consumers evolve independently, exception management is given a new importance. Exceptions generated by changes in service versions or policies may be recoverable with minimal intervention and without causing the business process to fail. With the proper tools, business analysts can inspect the XML documents that trigger exceptions and adjust their business processes without turning to IT.

The dynamic nature of SOA interoperability allows for on-demand reconfiguration of various aspects of a deployment such as location, transport, security and policy parameters beyond the exception-handling case. With dynamic discovery, these parameters can be changed in the Business Services Registry without making any changes to the services or the business processes orchestrated with them.

As an architect, your goals for managing your SOA will differ from those of business analysts and system operators. You are looking to assure adherence to standards and policies, while the business analyst is looking for business performance metrics, and the operations manager strives to maintain service availability.

To provide the best total control of an SOA, WSM solutions must work with a Business Services Registry. The registry also manages versioning and new service releases by coordinating across the full lifecycle.

Analysis

The business service lifecycle includes the analysis, monitoring and feedback mechanisms that help optimize the SOA, and ultimately the business processes it enables. These mechanisms provide a constant feedback loop that allows you to refine your architecture based on information about the status, performance, use and success of business services. Your analysis tools can also provide impact and dependency analysis of individual services or groupings of services, and reporting on a wide variety of issues such as reuse, policy violations or compliance reporting.

Business Service Lifecycle Planner

The table below summarizes the business service lifecycle, requirements, enabling infrastructure and relevant standards, as well as potential vendors of various solutions.

Phase	Requirements / Activities	SOA Infrastructure	SOA Standards	Example Vendors
Planning	SOA governance & management	Registry	Corporate standards & policies	Microsoft
	Policy enforcement	Security	WSDL	System Integrators
	SOA metrics	Management	WSDM	Systinet
	Quality of service/ reliability/latency policies	Governance	WS-Policy	WebLayers
	Security policies Taxonomy design			
Enablement	Business modeling	SOAP-WS Server and runtimes	BPEL4WS	AmberPoint
	Corporate, business, IT governance	Application servers	WS-Addressing	BEA
	SOA infrastructure development	Management and/or SOA network	WS-I Basic Profile	IBM
	Web services development	Security proxies	WS-Notification	Microsoft
	Infrastructure development	WSDL, taxonomy, XML and other modeling tools	WS-Eventing	Oracle
		ESB, SOA networks, EAI/message brokers	WS-RM	Salesforce
		WS-Security	SAP	
		XML, SOAP, WSDL	Security vendors	
			Systinet	
			TIBCO	
Publishing	Business service approval	Registry, taxonomy creation, content management	UDDI	IBM
	Certification process	Policy design		Microsoft
	Change management	Process design		Systinet
	Registration process & management			
	Categorize services and create taxonomies from service interface			
	Enrich service interfaces with policy-related metadata			
Discovery	Find and invoke business services	Registry	UDDI	IBM
	Walk taxonomy trees			Microsoft
	Design time usage			Systinet
	Runtime usage			
	Configuration and change			
	Operational management Introspect metadata			
Management And Security	Operate and manage business services	Registry	UDDI	Actional
	Create, monitor and enforce SLAs and other policy	Identity server, hardware/ software firewalls	WS-DM	AmberPoint
	Enforce security and identity	Management proxies and instrumentation tools	WS-Security	CA
	Control service provider access	Alerting systems		HP
	Track and manage provider-consumer relationships	Discovery tools		IBM
	Create parameters to monitor and provision monitoring tools			Microsoft
	Create and change taxonomies			Systinet
	Create and change service providers			Reactivity
				Netegrity
				Sun
Analysis	Analyze performance	Registry	UDDI	Actional
	SOA metrics management	Management console	WS-Policy assertion	AmberPoint
	SOA performance analysis	Data mining/ analysis Visibility solutions		Service Integrity

Business Services Registries Enable Lifecycle Management

The SOA business service lifecycle requires visibility, control and management across all stages. This level of visibility and control can only be provided by a registry-based approach to SOA.

A Business Services Registry is the core infrastructure for the SOA, managing the lifecycle of SOA services and paving the way to faster ROI for Web services. The registry not only speeds the transition from Web services to reusable business services; it is the hub that connects all OA participants—providers, consumers, developers, customers, partners, business executives, IT executives and more. The registry provides the pathway to a business-driven SOA by improving the speed and control of:

- ▶ SOA deployment
- ▶ business services rollout and deployment
- ▶ internal business and IT SOA usage
- ▶ partner SOA usage

The Business Services Registry supports the entire service lifecycle and enables the transition from initial Web services to business services, to Dynamic Business Interoperability via an SOA.

The Evolution To SOA

The evolution to an SOA requires new thinking about service oriented application design, the creation and reuse of business services that leverage existing enterprise applications, the lifecycle management of business services, and the IT deployment roadmap for new SOA infrastructure. SOA provides the design approach for a new generation of modular, standardized business services. The business service lifecycle provides both the control and a logical way to map business and technology requirements into an SOA model. A business-driven SOA strategy will help focus on the goal of Dynamic Business Interoperability.

The SOA business service lifecycle helps clarify the service oriented pathway; the Business Services Registry provides the visibility, management and control of the SOA information. Both lead to Dynamic Business Interoperability, dramatic business results and an agile IT.

Appendix A: Web Services Enablement Platforms

Examples of Web Services Enablement Platforms

Web services standards and technologies enjoy a unique level of industry support and adoption. Web service enablement platforms are available for most computing environments today, whether the goals are exposing mainframe CICS as services or enabling SAP using NetWeaver or third-party tools.

The speed and ease of enabling Web services are among the greatest benefits of this approach. Below are some examples of Web services enablement tools for various computing environments.

Packaged Applications:

- ▶ SAP NetWeaver allows SAP users to expose SAP functionality as services as part of their end-user's SOA strategy.
- ▶ Salesforce.com's Sforce allows end-users to customize, integrate and extend Salesforce.com's CRM solution to meet their business needs.
- ▶ Numerous other leading independent software vendors including BMC Software, Cognos, Progress Software and FileNet have integrated Web services functionality into the latest versions of their products.

Legacy Applications:

- ▶ Many third-party Web services enablement solutions, including products from Systinet, support legacy enablement for:
 - ▶ Mainframe CICS, IMS
 - ▶ Proprietary applications
 - ▶ C/C++ systems

Strategic Business Applications:

- ▶ Amazon.com Merchant Network allows Amazon to extend its sales processes to affiliates and others using its Web services enablement framework. This is an example of "syndicating a process" to trading partners.
- ▶ T-Mobile allows third-party content providers to offers new services to handset users using a simple Web services integration framework

Application Platforms:

- ▶ Microsoft .NET provides many tools and solutions for rapidly enabling Web services within the .NET environment.
- ▶ The J2EE platform is supported by over 30 tools for Web services enablement.

Middleware Applications:

- ▶ Message oriented middleware (MOM) vendors, including IBM, TIBCO, WebMethods, and SeeBeyond, have added Web services tools to their products.
- ▶ Third party vendors, including Systinet, offer low-cost, reliable, standards-based solutions for extending existing MOM infrastructure to support Web services.

Appendix B: SOA Defined

Service orientation is an approach to designing software systems. A service-oriented architecture (SOA) is a system consisting of modular software components with standardized component-access and usage interfaces that are independent of any specific platform or implementation technology. More importantly, an SOA enables software components to become standard services that can be invoked on demand, rather than repeatedly designed and programmed. In SOA, a "service" is typically a group of software components that together carry out a high-level function or business process, such as placing an order or making a credit approval on a purchase. At its most basic, an SOA is simply a collection of standardized services on a network that communicate with one another in the context of a business process. This approach dramatically eases integration in heterogeneous environments and provides a major enhancement in agility.

All services share some common characteristics:

- ▶ Services have interfaces that are platform or implementation-technology independent. Services are exposed using standards-based, identical interfaces that make them easy to use and reuse, and guarantee dynamic interoperability.
- ▶ Services are "loosely coupled." Services can be created without any forethought as to how or who will consume them. In addition, changes made to the service implementation will have no ripple effect on the consumers.
- ▶ Services are "coarse grained." Services focus on high-level business processes using standard interfaces, and thus mask the underlying technical and operational complexities of how a service is implemented.
- ▶ Services are modular. A service represents a discrete unit of business, application or system functionality. Multiple services can be combined to deliver more valuable services. This modular approach gives organizations great flexibility in system design. By reassembling services into a new configuration, a business can create a new business service to support a different business objective.

Appendix C: Glossary

BPEL—Business Process Execution Language: an XML-based language designed to enable task-sharing for a distributed computing or grid computing environment, even across multiple organizations, using a combination of Web services.

Business Services—Web services operating in an SOA with the necessary governance, policies and business taxonomies that enable business customers, IT applications and data, business partners, and internal enterprise users to access them. Business services provide the visibility, reusability, adaptability, and manageability for true business interoperability using an SOA. Business services can be service producers, service consumers or (most typically) both.

CICS—Customer Information Control System: an online transaction-processing (OLTP) program from IBM that, together with the COBOL programming language, has formed the most common set of tools for building customer transaction applications in the world of large enterprise mainframe computing over the past several decades.

CRM—Customer Relationship Management: all aspects of interaction between a company and its customer, whether sales or service related.

IMS—Information Management Software: IBM's premier transactional and hierarchical database management system for critical online operational and e-business applications and data.

MOM—Message-Oriented Middleware: a specific class of middleware that supports the exchange of general-purpose messages in a distributed application environment.

OASIS—Organization for the Advancement of Structured Information Standards: a not-for-profit, international consortium that drives the development, convergence and adoption of e-business standards.

SAML—Security Assertion Markup Language: an XML-based framework for exchanging security information.

Service Consumers—business services that consume service providers. Typically, they discover, retrieve and introspect service information that they obtain via a WSDL description obtained from a known URL or a Business Services Registry. They are quite different from service providers in many respects, including security and system usage requirements. Typical examples include Web services, Web users and applications, PC users and applications, and special devices (e.g., cell phones).

Service Instance—a concrete realization of a business service. An instance is sometimes also called an "endpoint," which denotes a runtime instantiation of a logical Web service, accessible via a particular technical protocol and transport.

Service Owner—a system entity that provides a collection of services. A service provider usually represents a business model and related process(es), generally claiming ownership and management responsibilities over its services. A service owner may host one or more Web services, which may be hosted on one or many physical machines.

Service Providers—business services that publish business service definitions, descriptions, information, and access control and authentication rules. Typically, service providers will be categorized along a range of business, functional and technical taxonomies based on a business model. Examples include data marts/warehouses, business processes and commercial off-the-shelf packaged applications.

SOA—Service Oriented Architecture: a simple software design approach and system in which all software functions are modeled as modular components and are implemented as platform and implementation technology independent services that can be consumed over a network using standards-based interfaces.

SOA Governance—the organization and processes required to guide the business success of an SOA and Web services. SOA governance defines and enforces the Web services policies needed to manage SOA applications and data for business success.

SOA Lifecycle Management—an approach to continuously managing the development, deployment and management of SOA business services through a set of defined activities and processes including planning, enablement, publishing, discovery, management and analysis of services.

UDDI—Universal Description, Discovery and Integration: an OASIS standard for Web services publishing and discovery using a service registry.

Web Services—a set of standard interoperability specifications for loosely coupled, self-describing software functions accessed programmatically across a network.

WSDL—Web Service Description Language: an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information.

WS-I—Web Services Interoperability: an open, industry organization chartered to promote Web services interoperability across platforms, operating systems and programming languages. WS-Security

XML—Extensible Markup Language: a simple, very flexible text format derived from SGML (ISO 8879).

About Systinet Corporation

Systinet provides the leading foundation for SOA governance and lifecycle management. Founded in 2000, Systinet's award-winning, proven, and standards-based products enable IT organizations to rapidly leverage existing technology investments, provide interoperability between heterogeneous systems, and better align business processes with IT. Customers receive the benefits of a simpler, faster, standards-based way to dramatically improve IT responsiveness and technology asset reuse, while maximizing the ROI for SOA. Systinet's customer base of over 150 Global 2000 clients includes Amazon.com, BMC Software, Interwoven, JPMorgan, Motorola, Defense Information Systems Agency, and SAIC. Headquartered in Burlington, Massachusetts, Systinet is a privately held company with over 100 employees.

To find out how Systinet can help your business, visit <http://www.systinet.com>, call 1.781.362.1300, or email us at sales@systinet.com.