Service Oriented Architecture (SOA) Solution Reference

Yan Zhao, Ph.D,
Chief Architect, ArchiTech Consulting LLC
(703) 408-1339
yan.zhao@architechllc.com

July, 2006

Table of Content

Executive Summary

In order to achieve business agility and technical flexibility, it is necessary for a business organization to investigate current status of Service Oriented Architecture (SOA) in practice, due to SOA is becoming the next big trend that is leading an evolution in enterprise business and IT. Many companies and government agencies are shifting from the initial try out with SOA projects of limited scales to strategic SOA rollouts in enterprise level with supports from senior management in IT and sometimes business executives. It is clear that the Federal Enterprise Architecture has chosen SOA as the approach.

SOA as an IT strategy has gained a lot of ground in the past years. Its adoptions have happened in various application areas with different scales. It is the time for business to provide higher level guidance and support to enable a cohesive SOA service infrastructure and governance across business functions, and to promote sharing, collaboration, and interoperation toward a common business goal. It is to ensure an enterprise SOA solution for a business, and to integrate enterprise architecture development effort across business functions, and to align business and IT efficiently.

In order to support SOA adoption for our customers, this white paper provides a comprehensive reference in current state of SOA in terms of concepts, technologies, and practice cases. It presents a SOA conceptual model first with introduction to SOA concepts and components, followed by the summary of current technologies and vendors that enable the implementation of the conceptual components. Since SOA is still in its fast growing stage, the landscape for technologies and vendors is evolving rapidly. It is important to separate conceptual model from the implementation technologies. This paper has included some SOA practice case studies with lessons learned and best practices. It has also included content in measurement, maturity models, risk assessment and mitigation, ROI, etc. Finally, it provides some recommendations in SOA adoption steps and approaches.

This white paper is a combination of SOA current state studies and solution recommendations, and can be served as a foundation paper for SOA adoption for a business organization. The further drill down of selected topics and areas can be provided based on requests.

1.0 Introduction

Service Oriented Architecture (SOA) is becoming the next big trend that is leading an evolution in enterprise business and IT. Many companies and government agencies are shifting from the initial try out with SOA projects in limited scales to strategic SOA rollouts in enterprise level with supports from senior management in IT and sometimes business executives. SOA as an IT strategy has gained a lot of ground in the past years. SOA enables a business service layer on top of applications, which makes applications emphasizing more on business function support rather than hardware and software.

The core value of SOA is in delivering business agility and IT flexibility. As presented by IBM, the business benefit of SOA is in service reconfiguration flexibility, with changes done in days by business people, not in weeks by technical specialists. This means that the business and technical architectures must be aligned, which is not the case in most organizations today. Expressing an existing application architecture in SOA terms is not enough. The services must be business-oriented if they are to be orchestrated by business people. SOA helps to streamline IT infrastructure, and helps to align IT investments with business goals, so that can help optimize IT spending. The deployment of SOA in web service allows integration of business with current technologies.

SOA can be evolved based on existing systems and infrastructure rather than requiring a full-scale re-build. Organizations will achieve benefits from SOA by focusing their development effort around the creation of services with using both new and existing components and technologies, combined with the component-based approach to software engineering and the enabling SOA infrastructure. The benefits of SOA include

- **Business agility:** SOA makes easier for business process improvement. It provides the business users with an ideal environment for monitoring business operations. Process modeling is reflected in the business services. Process manipulation and the change of process flow can be achieved by the use of BPM (Business Process Modeling) tools integrated into the SOA infrastructure.
- **Reuse and leverage existing assets:** A business service can be constructed as an aggregation of existing components, using a suitable SOA infrastructure and made available to the enterprise. Legacy systems can be encapsulated and accessed via web service interfaces.
- **Common infrastructure as commodity:** SOA infrastructure is becoming commodity that can be implemented by the use of COTS products. By enforcing the standards, its development and deployment can be consistent across enterprise. Existing components, newly-developed components, and components purchased from vendors can be consolidated within a well-defined SOA infrastructure.
- **Reduce development cost:** The reuse of existing service and components will reduce software development time and cost.

- **Reduced maintenance cost:** Due to the flexibility introduced by SOA for business service enhancement and creation, it's easier to incorporate new business requirement, so that the maintenance cost will be reduced.
- **Risk mitigation:** Reusing existing components reduces the risk of introducing failures in creating new ones. Also, there is a reduced risk in infrastructure support due to its commodity nature.

The concept of SOA is not new, which can be traced back to the Common Object Request Broker Architecture (CORBA) [Pop]. The Java and J2EE platform is actually a simplified version of CORBA. The recent popular component-based and service-oriented architecture has extended its scope to business domain, which is reflected in Federal Enterprise Architecture (FEA). Web Services enable the SOA concept being applied in web environment. However, there are differences between what SOA offers from the past, which are illustrated in Figure 1.1 [Mos].

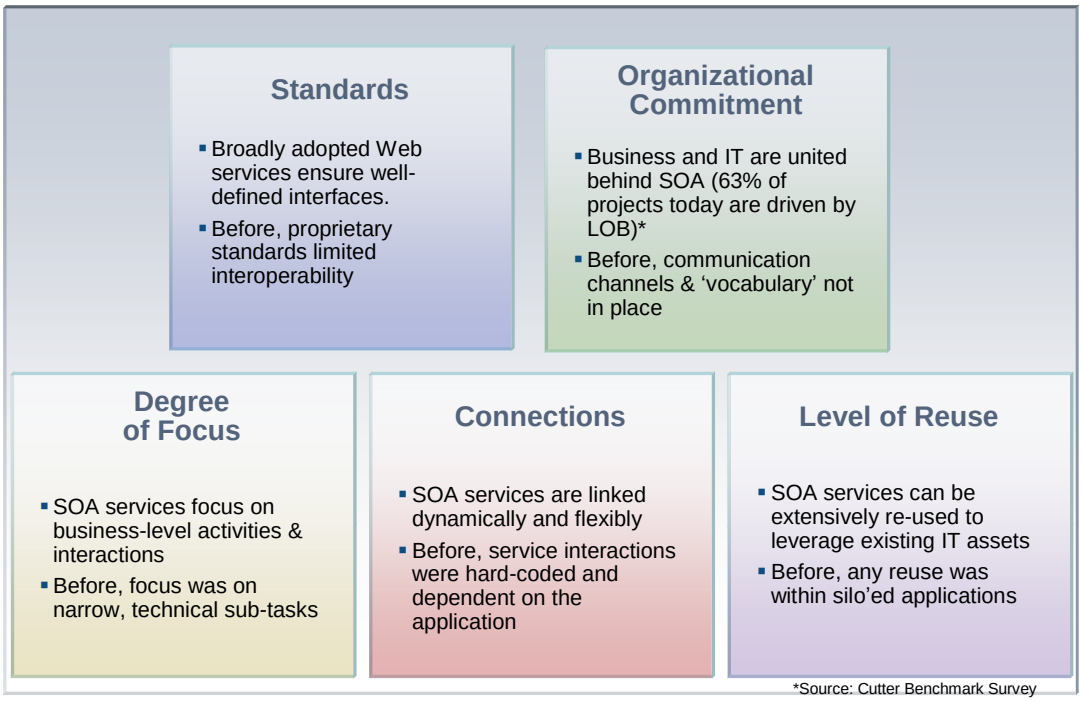

Figure 1.1 What SOA offers are different from the past

SOA can also be considered as a practical modeling discipline for enterprise architecture (EA) development. It can help to bridge EA with solution architecture and implementation by layered service descriptions across business modeling, application modeling, and technology implementation; so that it can help bring EA into reality.

This solution reference paper is prepared for our customers. It is a combination of SOA current state studies and solution recommendations with content includes SOA concepts, technologies, and practice references. The content is a combination of author's original views and the survey of large amount of relevant information in public domain.

2.0 SOA Conceptual Model

SOA is an architectural style that emphasizes well-defined, loosely coupled, coarse-grained, business-centric, reusable and shared services, as well as associated infrastructure. The core of SOA consists of three components, as shown in Figure 2.1:

- **Service Provider:** who publish services to Service Registry
- **Service Consumer:** who find services from Service Registry and use (or "bind" to) them
- **Service Registry:** where contains information for available services.

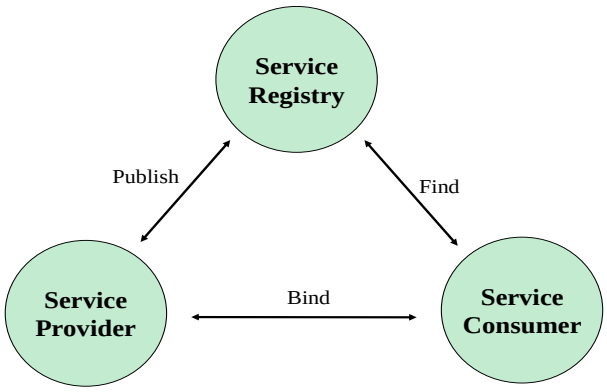

Figure 2.1 The core of conceptual SOA model

2.1 Concepts for SOA

The SOA is an architecture style that emphasizes service-orientation for businesses and their associated applications. It is to deliver business agility and IT flexibility. A flexible SOA model is illustrated in Figure 2.2. It shows how SOA can bring agility to business and flexibility to IT, and the dependency between the two. The evolution of service-orientation is illustrated in Figure 2.3 [Kli], which shows that the applications for a business are evolved from stove-piped silos to shareable, reusable, standard-based business services.

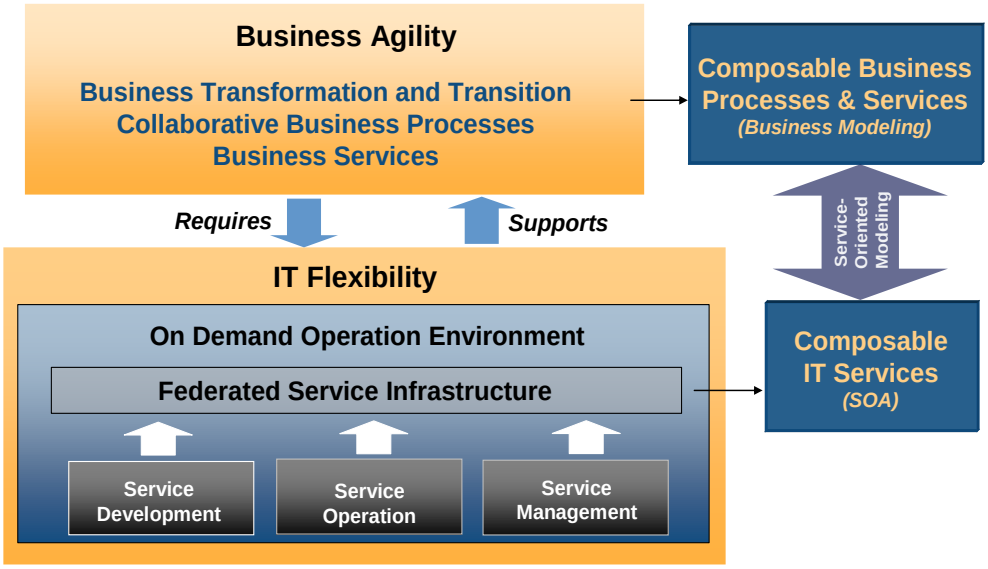

Figure 2.2 A flexible SOA model

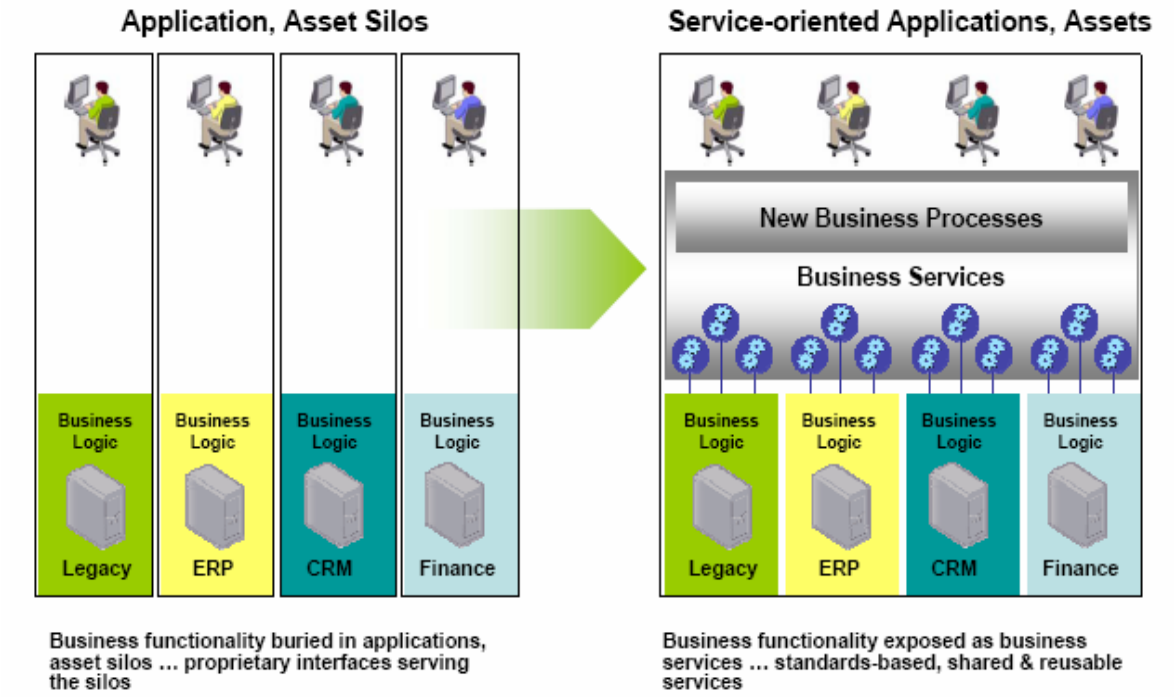

Figure 2.3 The evolution of service-orientation (from Systinet)

A SOA domain model is presented by BEA [14], where it specifies six domains to response to SOA challenges. They are:

1. **Business Strategy and Process:** The response is to have a SOA-enabled business strategy, and business process architecture.
2. **Architecture:** Need to consider reference architecture, and its manageability, availability, scalability, and security.
3. **Building Blocks:** Need to consider infrastructure services, information and access services, shared business services, presentation services, and composite applications.
4. **Projects and Applications:** Need to consider existing applications, key "in-flight" projects, and infrastructure construction plans.
5. **Organization and Governance:** Need to consider organization design, funding, skill sets, roles and responsibilities, standards, operational processes and tools, as well as change management.
6. **Cost and Benefit:** Need to consider construction costs, business and IT benefits, and key measures.

Some major concepts in SOA from Oasis SOA reference model [W5] are summarized below:

Service: In general, people and organizations create capabilities to solve or support the solution of problems they face in the course of their business. SOA is conceived as a way of making those capabilities visible and supporting standard means of access so the existing capabilities can be reused or new capabilities can be readily substituted to improve the solutions. A *service* is a means to access such capabilities.

Service description: To use a service, it is necessary to know it exists, what is accomplished if the service is invoked, how to invoke the service and other characteristics to allow a prospective consumer to decide if the service is suitable for the current needs and if the consumer satisfies any requirements of the service provider to be permitted accessing. Such information constitutes the *service description*.

Service policy and contract: Services are accessed in order to achieve particular effects. However, the nature of SOA is that there is an arm's length relationship between service providers and consumers. As a result, there is a distinction to be drawn between the public interactions with a service and the private actions of the service provider and consumer. An important reason for the scalability and security attributes of SOA is that the distinction promotes independence between service participants. We can focus on the public aspects of using a service by examining the *conditions* of using a service and the *expectations* that arise as a result of using the service. We loosely associate the service conditions with the *service policies* and the expectations with *service contracts*.

Service interaction: Although services are accessed in order to achieve particular desired effects, this is affected by exchanging information between service providers and consumers. Typically this is by exchanging messages using a standardized protocol; however, there are many modalities possible for using services.

Service discoverability: Discoverability refers to the possibility and mechanisms by which service consumers and providers can be brought together. There are many possible mechanisms by which discoverability may be achieved; SOA is not limited to registries or repositories of service descriptions although these are undoubtedly powerful means of achieving it. Discoverability itself is a key concept for SOA.

Service metadata: The service metadata enables machine processable service description. The purpose of the metadata is to facilitate integration across ownership domains. By providing public descriptions, it makes it possible for potential participants to construct applications that use services and even offer compatible services with minimal human-level contact between them. The use of metadata potentially permits automation with computer software. Both service providers and service consumers can benefit from such automation.

2.2 SOA Service Infrastructure

A service infrastructure is necessary to support SOA implementation with service dynamic binding at run-time. It enables loosely coupling of service interfaces from service implementation components, with service interfaces being exposed to service consumers via service registries. The major components for SOA service infrastructure consists of:

- SOA Service Broker
- SOA Service Registry
- SOA Governance Policy
- SOA Event Handling Middleware
- SOA Service Metadata Management

A service broker as a service intermediary that manages the invocation of a set of registered services based on a set of policies and rules. This incorporates routing of the messages and possibly data transformation between the incoming message and the requirements of the brokered service. A broker may be configured to be invoked synchronously or asynchronously. An example is illustrated in Figure 2.3.

A SOA infrastructure can include multiple service brokers that provide federated services, as shown in Figure 2.3. There is a service registry and a metadata repository associated with each service broker, so that federated registry service will be associated with the federated service brokers.

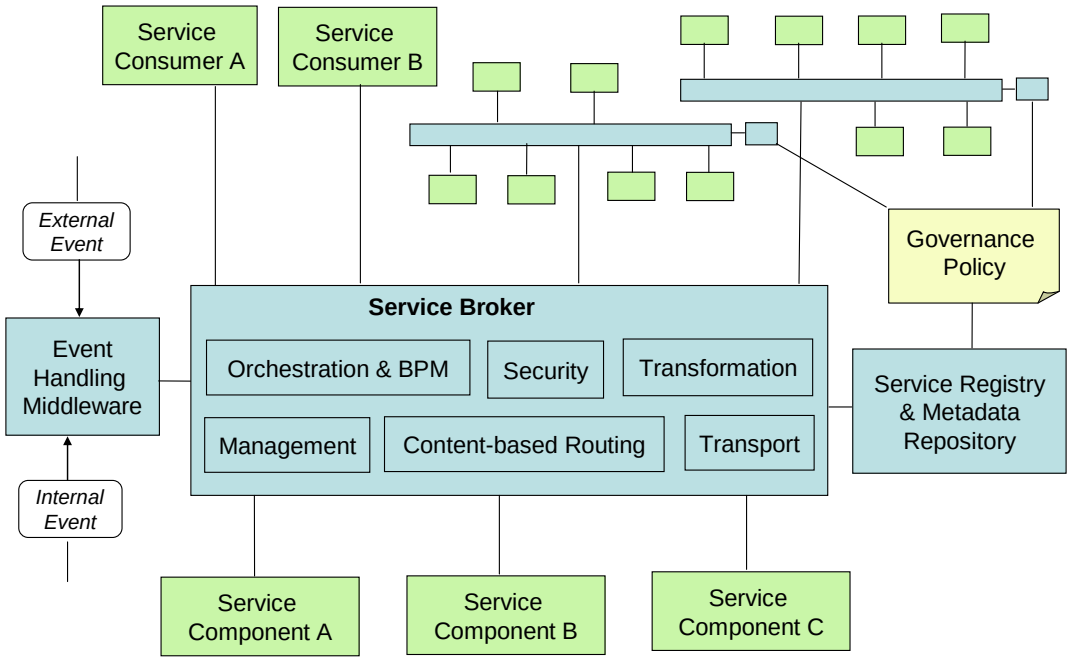

Figure 2.3. A conceptual model for SOA service infrastructure

2.2.1 SOA Service Broker

The service brokers handle service communication, message routing, data transformation and mapping, service management, security, and other common utility services. Some typical services provided by a SOA service broker include:

Content-Based Routing: A content-based routing service makes intelligent message routing decisions based on the content of the message. The routing can be performed based on specific policies. Service brokers may play the role of firewall by not routing certain message, or essentially stopping anything unapproved from continuing its journey through the network. This could be to keep non-authorized service messages from affecting systems, or to make sure that even well-intentioned insiders don't degrade a system's performance by tapping into critical services without approval.

Security: The security service provided in service broker enables security implementation at messaging and transport layers, so that security can be enforced to meet operation, privacy, and regulatory needs. It can provide comprehensive and pluggable authentication, authorization, and encryption services across a service broker, also can provide federated security service across multiple domains using an enterprise level security policies. Also, it is important to leverage standards like WS-Security and SAML to enable interoperation between competing security systems.

Service Management: The management tools can be used to monitor services traffic and alert systems administrators when problems arise. It can help administrators to identify the needs for system scale up or down. Management tools can also record business activities to enable traffic pattern analyses so that to help administrators balance workload and optimize system performance.

Data Transformation: The transformation service can perform XML-based transformation using extensible style sheets (XSLT) to transform the contents of a message body from one XML-based format to another. This feature enables rapid integration between systems with messages in different formats. The transformation service removes the burden of ensuring that a message from the sending application is in the correct format for the recipient. It allows for easy interoperation of services, without requiring message format changing to the applications.

Transport: The transport service provides the flexibility to choose and change communication transport mechanisms as needed for different application services. It eliminates the need for the applications, business processes, and service interactions to be aware of the underlying transport.

Service Orchestration: By exposing applications as services on the bus, it makes it possible to add an orchestration layer on the service broker, allowing services to be orchestrated to directly support the business processes. This orchestration layer provides a flexible and configurable way to automate business processes, promoting reuse of services, and making the overall systems more agile to respond to changing business requirements. Business Process Management component (or tool) can be integrated for this purpose, which is happening today.

The Business Process Execution Language (BPEL) and BPEL4WS can be considered as the *de facto* standard for Web Service-based business process orchestration and workflow description. The SOA composite application development often involves workflow services, which are usually implemented as stateful services with pre-defined start and end states with process definitions. The processes can be defined by BPEL or BPEL4WS.

2.2.2 Service Registry

The registry stores descriptions of services, and enables run-time discovery. It is the control center for orchestration. Currently, its contents may be fairly simple in many implementations [Har], such as just define what a service is and how to describe and organize it. But when an enterprise moves to SOA, it exposes different relationships, such as between producers and consumers, between services and schemas, between business process and the services that it consumes. An enterprise must manage them, or it will not be able to cope with changes. A registry helps to define services, but does not describe relationships. The metadata repository will work with the registry to enable such descriptions.

As shown in Figure 2.3, a service registry contains references to service interfaces, and the metadata repository contains metadata for service dependencies, the identities of service providers and service consumers, service contracts, service usage information, and other operational information. Also, a service repository should contain taxonomies

through which services can be discovered. The UDDI and ebXML provide standard interfaces for service navigation and discovery. A sophisticated registry and metadata repository support enable service control and governance throughout service lifecycle. Registry and repository with applications built on top can better support business values of SOA. The application features might include:

- An easy-to-use configuration tool for configuring metadata such as contracts
- Role-based security and access control for services
- Standards compliance tests such as WS-I Basic Profile Test
- The ability to recognize which consumer invoked what service and how often for capacity planning and load balancing
- Use the repository to store assets and artifacts associated with services
- IT reuse and governance capabilities which extend to non-web services

2.2.3 SOA Governance and Policy

As described in CBDI Journal [Wil], the term govern has numerous meanings including to rule with authority, conduct policy, actions and affairs; constitute a law, rule or principle; and to be predominating influence. Governance is the manner, fact or function of governing. In context with SOA, the purpose of governance is the activity of establishing and implementing policies in order to ensure that the objectives of the SOA are complied with throughout the life of a service. The matters to govern can be the identification, creation and use of shared services together with the standards and practices in a manner that delivers the levels of flexibility and reuse of business and technical services that are appropriate to a given situation.

To ensure service consistency in enterprise level, an enterprise level governance policy should be applied to the distributed service registries based on the community of interest and expected usage for each service broker.

2.2.3.1 SOA Governance

The SOA governance is a very complex issue. It is to control policy and automate workflow of enterprise it governing. It should cover organization structure, process, and policies to control SOA implementation. It requires SOA lifecycle management with robust metadata management and policy support. Ideally, SOA governance should be extend to all corporate governance and the IT governance resources will be available to the corporation as loosely-coupled business services.

In CBDI Journal [Ver], four types of governances are introduced. They are

- Run-time governance
- Design governance
- Asset governance
- Management / program governance

Also, it demonstrates a model for governance life cycles, which is illustrated in Figure 2.4.

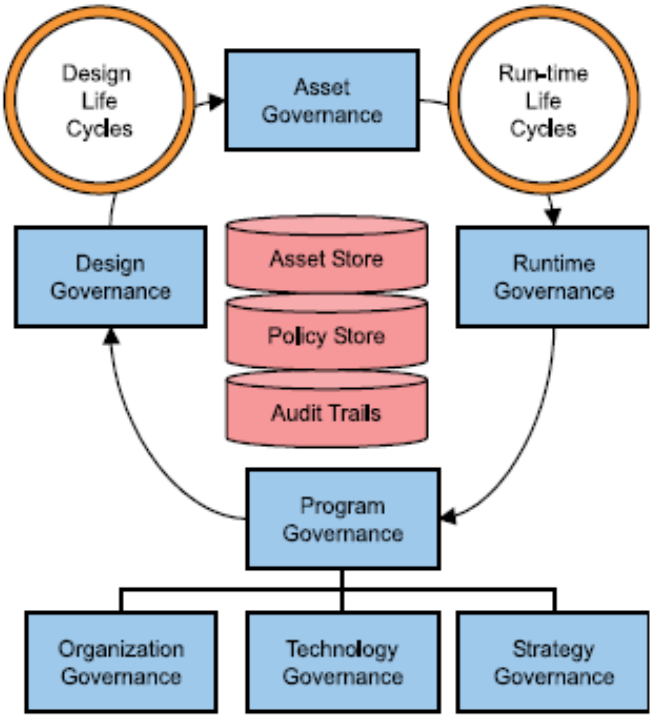

Figure 2.4 Governance Life Cycles (from CBDI Journal)

2.2.3.2 SOA Policy

Policies are the means by which design-time decisions about security, service levels, etc. are enforced in the runtime environment [Har]. For enterprise agility, definition of policies must be separated from their implementation, so that the user does not need to understand the technology. There must be standard policy formats, with composite policies interpreted and enforced by management intermediaries, and the challenge is how to enable management of policies with implementations across multi-vendor and multiple product types.

2.2.3.3 SOA Contract

The service contract is essential for SOA deployment, since the most basic interaction in SOA is between service providers and service consumers, and the two parties must come to an agreement before services can be performed. There are two different senses of contract in play here [Sch3]. First, there is the legal agreement between two business entities, and second, there is the technical relationship between the parties that aim to work with each other. The contracts are the key to loose coupling, which goes to the heart of SOA. Loose coupling mandates that two interacting parties should have as little information as possible necessary to govern their relationship. Furthermore, the entire reason why we want to loosely couple software is so that we can independently create

and control each component in IT environment. The way in which we choose to make loose coupling a reality is by implementing contracted interfaces on systems, and making sure that we enforce those contracted relationships while allowing each party to independently change how they implement the contract.

The definitions of terms in contracts can vary from one company to another, but a policy is a set of conditions that can apply to any number of contracts, from none of them to all of them. For example, a policy might state that all interactions with services must be SSL-secured, and that policy would then apply to all service contracts in an organization [Sch3].

2.2.4 Event Handling Middleware

Event handling middleware connects the service infrastructure with the outside world, providing real-time input and response [Har]. This connection can be made through an event-driven architecture (EDA), which interfaces to software triggers, sensors, and telemetry and provides filtering, aggregation, correlation, and complex event processing. EDA has sometimes been thought of as a competitor to SOA, but in reality the two are complementary. Gartner predicts an increasing role for event processing, with an "era of events" to follow the "era of services".

2.2.5 Service Metadata Management

Service metadata repository is necessary as mentioned in service registry subsection. Both service registry and service metadata repository work together with SOA Service Broker to provide a SOA infrastructure. It stores artifacts to be used in both development time and runtime.

The service contract is the most important metadata for SOA, since the most basic interaction in SOA is between service providers and service consumers, and the two parties must come to an agreement before services can be performed. In addition to facilitate service discovery and interaction, most artifacts in metadata repository are private components which should not be published broadly. These include configuration information, executables, and related metadata, such as routing rules, process definitions, XML schemas, XSLT transformation files, JAR files, etc. An efficient and effective service metadata management mechanism is very important for the operation of this complex metadata repository and for SOA deployment.

2.3 Business Process Support

2.3.1 Business Service Bus

The concept of Business Service Bus (BSB) is accepted by SOA community. It is different from the SOA Service Broker (or Enterprise Service Bus as a current implementation, which will be described in next section) that handles technical details for service operation. The Business Service Bus is the set of business services for a special domain that are available for widespread use across an enterprise, such as the services in

human resource, logistics, billing, etc. These services are published in the Service Registry. One of the reasons for using Business Service Bus is that the common specifications, policies, etc can be made at the bus level, rather than each individual service. A picture from CBDI Journal [Wil2] regarding to enterprise SOA layers is illustrated in Figure 2.5.

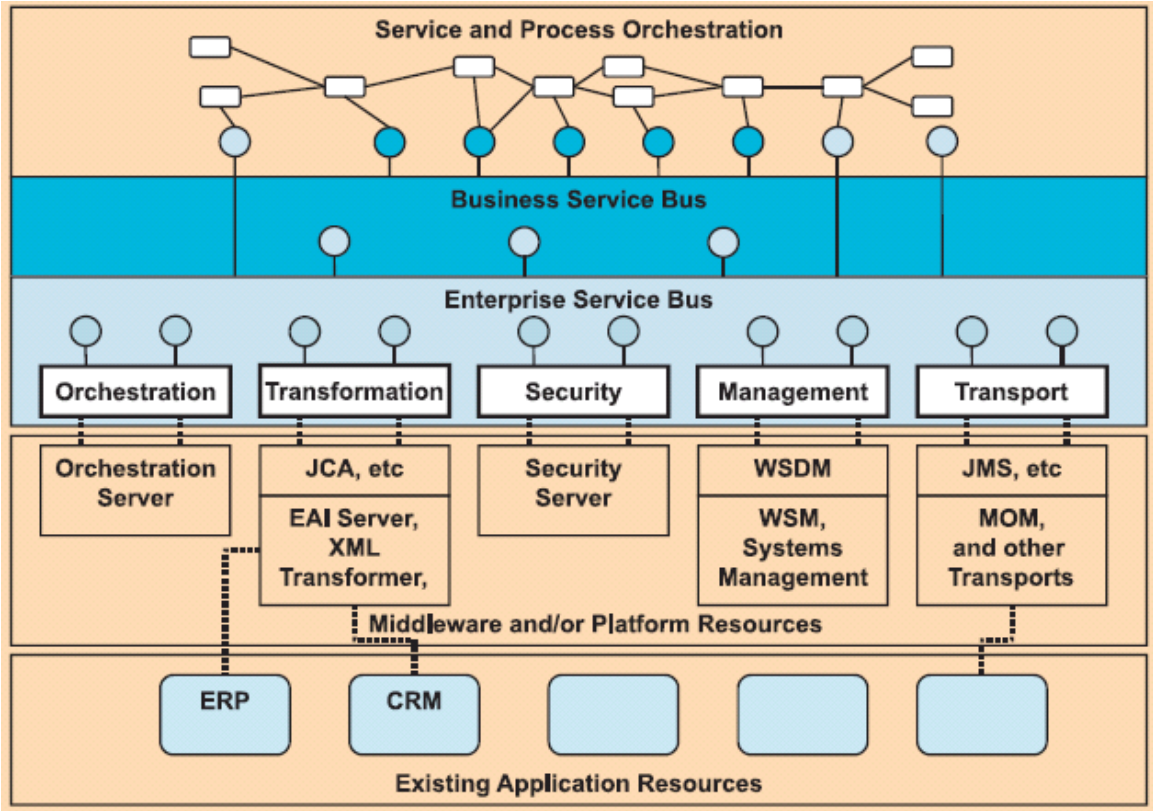

Figure 2.5 Enterprise SOA layers (from CBDI Journal)

The Business Service Bus is based on the premise that there will be multiple implementations of the same business object, either now or in the future, and the purpose of the bus is to make those implementations transparent from service usage. Although the purpose is to establish a single logic bus structure, it may be natural to implement bus structures to mirror organizational boundaries. The successful organization will see the real value in creating cross organizational services that allow the organization evolve independently of technology implementations.

2.3.2 Business Process Management and Service Orchestration

Business process management and service-oriented architecture are a natural match. There are synergies between these two. SOA provides a role for BPM systems in its separating business process management as an independent function, allowing processes to be designed independently of any single application and leveraged as shared business

logic. Also, the true value of SOA comes from the ability to orchestrate service components across executable business processes that BPM systems can provide. In other words, BPM can be leveraged in SOA to build adaptable composite applications capable of supporting today's constantly changing business environments. The process-driven and service-oriented architecture presented by the combination of SOA and BPM provides an ideal environment for building adaptable, model-driven composite applications from existing IT assets and infrastructure.

Figure 2.5 illustrates the position of this Business Service and Process Orchestration layer.

Technically, current SOA is mostly implemented by web services, and most BPM systems on the market today can "consume" web services. In other words, when web services are registered within a standard directory (UDDI) and wrapped with a standard set of descriptors (WSDL and XML) they can be invoked by a BPM system as an automated activity. Many BPM systems also allow web services to be "discovered" from within the process designer and added to the design palette, such that they can be added to the process definition without requiring any other formal integration effort.

2.4 SOA Service Life Cycle Model

SOA service life cycle model has broadened the traditional model for System (or Software) Development Life Cycle (SDLC). Moving to SOA requires a strategic commitment to create more flexible IT systems that can map to business processes and can cope with changes dynamically. Unlike the traditional life cycle, which takes years to roll out a product, SOA services will be modified all the time and there will be ongoing collaboration between design time and run time.

An SOA service life cycle model is illustrated in Figure 2.6. This model identifies the major four stages in SOA service life cycle that across service development and run time. The business goals, objectives, and requirements are served as guidance and inputs to both service construction and operation, so that it enables business-driven services and business agility in practice. The tasks for each stage are also illustrated in Figure 2.6. The SOA governance is applied to the complete life cycle in all of the four stages, which is consistent to the discussion in section 2.3.3 and Figure 2.4.

The understanding of SOA service life cycle can help an organization to make a better plan for SOA adoption. An organization’s governance and policies could have influence to its service life cycle model, while the completed service life cycle model can help the organization to eliciting the more detailed policies for governing the efficient and effective development and usage of software services.

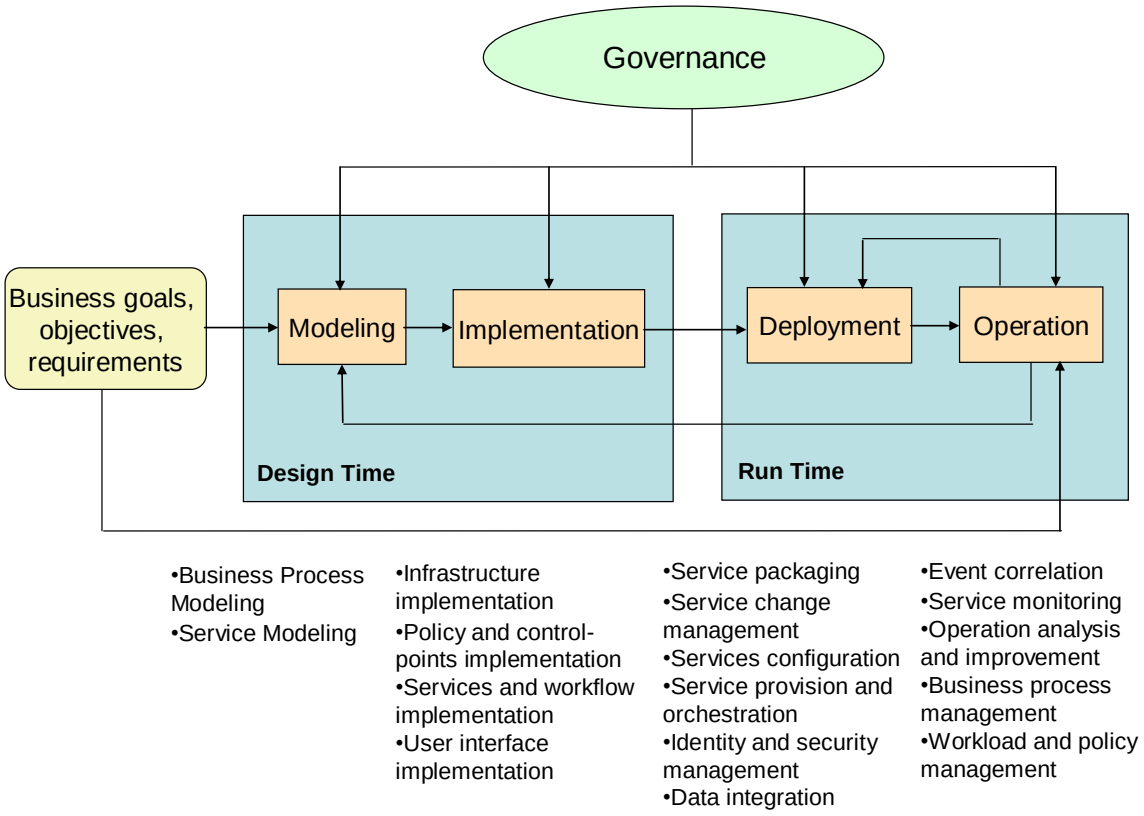

Figure 2.6 A SOA service life cycle model

2.5 SOA Roadmap

The SOA roadmap provides enterprise with executable SOA adoption guidance in moving from current state to the ideal future. An earlier SOA Roadmap framework is presented in CBDI Journal [Spr], which presents a basic roadmap model as a strategy. It also presents detailed roadmap considerations in five streams with four phases. The five streams are:

- Plan and management
- Infrastructure
- Architecture
- Process
- Projects

The four phases are:

- Early learning
- Integration

- Re-engineering
- Maturity

There is a more comprehensive SOA adoption roadmap presented in [Bie], which is shown in Figure 2.7. It states that "An SOA strategy should not be a big-bang replacement of an existing IT environment; rather, it should be a progressive and evolutionary roadmap". "For each of the prioritized business services and components, the roadmap follows the typical phases of IT project development, with inception, elaboration, implementation, and test and production phases, as typified in the Rational Unified Process™. However, each of these phases includes new activities that relate to the service component identification and realization". It provides an overall view regarding to the adoption stages and corresponding activities. It also illustrated in the separated streams the different levels in adoption scopes, services reuse, and architectures, which indicate how much the SOA model penetrates into a business.

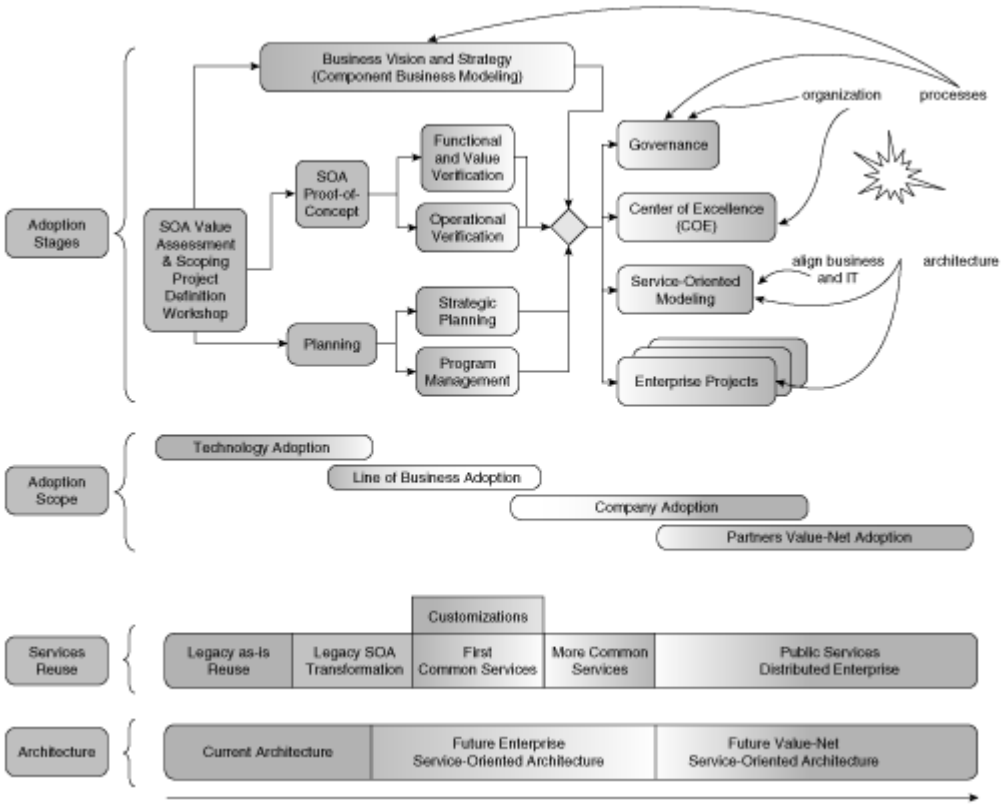

Figure 2.7 SOA adoption roadmap

2.6 Maturity Model

Most SOA Maturity models under development are based on the idea of CMMI (Capability Maturity Model Integration), which is for evaluating and measuring the maturity of an organization's software development and integration processes. Actually, SOA Maturity Models are only loosely related to CMMI [Blo2], because except the notion of a maturity model from CMMI, they don't have much else in common. CMMI measures the maturity of IT *processes*, while SOA Maturity Models should measure the maturity of an organization's *architecture*. It is important for a SOA Maturity Model to measure the maturity of the architecture itself, not just its implementation. As suggested by ZapThink [Blo2], a SOA Maturity Model should contain the following four measures:

1. **Architectural Views:** The SOA Maturity Model should contain, for example, how well a company has fleshed out the various views within their architectural vision. At the lower levels they may only have a technical architectural view of SOA, but as they move increasingly into higher levels of maturity, they should be able to add data architecture, information architecture, and process architecture (among others) into the enterprise SOA as interrelated views.
2. **Service Models:** Another aspect of SOA maturity is how well a company leverages architectural models, in particular the Service Model that should represent both the Services in production as well as the requirements from business for new or modified Services. At lower maturity levels, companies may have no Service Model at all, or at best a sketchy model that serves as a limited design-time artifact. At higher levels, however, organizations leverage the Service Model at both design-time and runtime to represent Services as they continue to evolve.
3. **Scope of SOA application:** The 3rd measure is the scope of the application of SOA At the lower levels, a SOA will often have pilot or departmental scope. As companies increase their SOA maturity, their application of SOA will typically spread across departments, and finally at the highest maturity levels, the application of SOA will be enterprise-wide (or even multi-company).
4. **Architecture Implementation:** A SOA Maturity Model should also contain measures of the maturity for architecture implementation. Clearly, a purely theoretical architecture is not likely to be as mature as one have fully implemented, tested, and put into production. However, a mature implementation is primarily a result of a mature architecture. Without measures of architectural maturity, a maturity model cannot be a true SOA Maturity Model.

There are several SOA maturity models published in public domain. Three of them are included in the reference of this paper. They are representatives from three categories of SOA practitioners: vendors, consulting firms, and industry groups.

A New SOA Maturity Model – Sonic, AmberPoint, BearingPoint, Systinet: As discussed in [W9], this model mainly measures how advanced an organization's services are and how mature their runtime infrastructure might be. Comments from ZapThink [Blo2]: It's a good model for measuring the maturity of the Services an organization develops as part of its SOA initiative, but not for measuring the maturity of the

architecture itself. As a result, this maturity model is really more of a Services maturity model, as it fails to address architectural maturity.

Service Integration Maturity Model and SOA Maturity Model – IBM: The Service Integration Maturity Model (SIMM)[Ars1][Ars2] describes seven levels of maturity measures for service-oriented integration (SOI), and its mapping to CMMI. Also, it indicates that the SOA adoption is a gradual process, and it demonstrates an incremental scope of SOA adoption. The SOA Maturity Model [Mit] defines five levels for SOA maturity measures. It illustrates the characteristics for each maturity level and the impacts associated with.

A Web Service Maturity Model – CBDI: It is presented in CBDI Report [Spr] sponsored by industry leaders. It discusses technology maturity and business maturity separately, and than presents web service maturity model in four phases.

2.7 SOA and Enterprise Architecture

SOA is transitioning from concepts and early stage small scale adoption to mainstream IT strategy. It matches well with the mission of Enterprise Architecture (EA). So that EA is largely successful with SOA adoption. Many enterprise architecture products have been developed with adoption of SOA models and methodologies, someone call these Enterprise Service Oriented Architecture (ESOA).

It is discussed in a Forrester analysis report [Cul] that when the SOA evolves from the first-generation to the second generation, the role of EA should change too. The first-generation SOA is to incorporate SOA concepts in IT, and releases services on the infrastructure level. The second-generation SOA is to embed SOA into IT strategies and processes to sustain SOA growing usages in business. They are characterized in Figure 2.8.

First-generation SOA	Second-generation SOA
Defining basic design principles and models	Basic design principles and models understood
Putting pilot implementations into production	Developed as production services
Services on existing apps are based upon wrapping existing application interfaces	SOA defines app architecture
Focus is on infrastructure services	Increasing focus on business services
"Release 1" — basic services	"Release 2+" — enhanced services
Local strategy	Enterprise strategy
Maintenance and support models undefined	Maintenance and support models defined

Figure 2.8 The characterization of SOA in the first and second generations (from Forrest)

The report points out that EA has successfully championed SOA for the first generation in sponsoring the initial projects and owning the initial services. While it was necessary

for EA to champion SOA as a strategy, this approach is not sustainable as IT moves into SOA's second generation. The current form of EA leadership risks too much ownership by EA and not enough by the organization. All too often, EA becomes too tactical, too operational, and too involved. For the second generation of SOA, EA should transition many of the responsibilities it took for adopting SOA to other parts of IT by:

- Pushing decision-making responsibility toward other parts of IT. Instead of making all services-related decisions, EA has to push for the establishment of the necessary governance mechanisms, such as service steering committees and change control boards, which involve the relevant groups within IT.
- Transferring responsibility for the design of individual services to project teams, though EA has to provide design templates, standards, and guidelines.

3.0 SOA Technologies

The SOA technologies presented in this section provide references in technologies and COTS products for implementation of the SOA concepts discussed in earlier sections.

3.1 Web Service for SOA Implementation

Currently, web service is the de facto standard for SOA implementation, though SOA is independent to any technologies that implement it. In Figure 3.1, a basic model is demonstrated that uses web service to implement the SOA core model illustrated in Figure 2.2. The current standard web service protocols are discussed in [Spr], which are based on the architecture work from W3C Web Service Architecture Working Group. An updated set of protocols are illustrated in Figure 3.2.

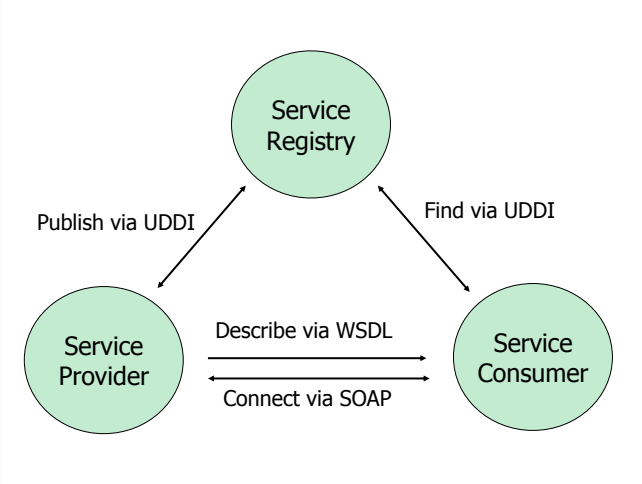

Figure 3.1 A basic model of web service implementation for SOA core model

The *ebXML* standard is from OASIS targeting the creation of the electronic marketplace where enterprise can meet and interoperate. Recently, ebXML has formed an Electronic Business SOA Technical Committee. Some COTS products, such as Infravio uses ebXML to support its Federated Web Service Registry offering in addition to UDDI, which provides more comprehensive service information support.

The web application severs can be applied to facilitate web service implementations. The typical COTS products available in this domain are WebSphere from IBM, WebLogic from BEA, and .Net Framework from Microsoft. More details can be found from corresponding web sites.

Category	Protocols
Management	Distributed Management: WSDM, WS-Manageability
Security	Security: WS-Security, SAML
	Security Policy: WS-SecurityPolicy
	Secure Conversation: WS-SecureConversation
	Trusted Message: WS-Trust
	Federated Identity: WS-Federation
Registry/ Discovery	Discovery: UDDI, ebXML
	Publication: UDDI
	Inspection: WSIL
Description	Portal: WSRP
	Transaction: WS-Transactions, WS-Coordination, WS-CAF
	Orchestration: BPEL4WS, WS-Choreography
	Presentation: WSIA
	Policy: WS-Policy
	Implementation: WSDL
	Interface: WSDL
Transport	Routing/Addressing: WS-Addressing
	Reliable Messaging: WS-ReliableMessaging, WS-Reliability
	Packaging: SOAP, WS-Attachments, DIME
	Transport: HTTP, TCP, SMTP, etc
Integration	Web Service Integration: WS-I

Figure 3.2 The current web service protocols

3.2 Technologies for Service Infrastructure

This section presents technical solution options for SOA service infrastructure, which provide SOA run-time framework for deployment.

3.2.1 Technologies for SOA Service Broker

One type of implementation for a service broker is using Enterprise Service Bus (ESB). The role of ESB is connecting, mediating, and controlling. It is software infrastructure that enables SOA by acting as an intermediary layer of middleware through which a set of reusable business services are made widely available. An ESB reference architecture is presented by Sonic [W17]. Some ESBs are message based, and some others are web service based. There are many ESB variations in market today, and they are mostly driven from vendors existing offerings. Currently, we can find the COTS products in three categories:

Messaging Based ESBs: They provide message based service infrastructure, which can facilitate services in the manners mentioned above and provide integrated services from

platform independent application components. The service interfaces provided by both new and legacy applications can be exposed as web services or message based service interfaces with adoption of standard description language and communication protocols, such as WSDL, SOAP, XML, HTTP(S) etc. Historically, these products were developed to provide reliable messaging and application integration. Often, their core technology is based on a non-web service technology, such as a message service engine. They can be considered as messaging solutions that also do web services, which can facilitate broader integration efforts. Sonic and IBM provide products in this category. Sonic is the most established vendor in this domain so far.

Web Service Based ESB: It is also referred to as SOA Fabric or web service broker, which provides web service based SOA run-time framework. The WebMethods Fabric and The Blue Titan's Enterprise SOA Fabric are in this category. They are designed to interact with web service standards like XML, SOAP, UDDI, SAML, WS-Security, etc. Cape Clear is a new player in this domain, which declares to "simplify this whole middleware muddle" and "can build complex systems in less time and at less cost". It has been recognized as a current leader by both Forrester [Wil2] and Gartner [Pez].

SOA Service Appliances: These are specialized chip-based solutions with optimized software approaches built-in. The goal for appliances is to drive down complexity, accelerate SOA implementation, and achieve better performance (especially for XML messaging). The Cast Iron Systems, Reactivity, and DataPower are recognized players. The trade off for them is not sophisticated and flexible enough yet to support a full-scale SOA solution. The XML Gateway products from Reactivity and DataPower can also be used as front-end processors to enhance XML messaging performance.

It is important to understand the difference between messaging based ESB and traditional EAI integration broker favored previously in the EAI market. As discussed in [Cra], the major difference can be that the ESB is built around standards. This has a number of effects:

- It reduces the need for specialist skills in the integration task, relying instead on knowledge of common standards such as web services.
- It reduces complexity – focusing on supporting only defined standards means that functional options are reduced and simplified.
- Because standards-based offerings reduce functional differences while tending not to exert the same level of vendor lock-in as proprietary EAI approach, pricing is down across the marketplace.
- Because complexity and functional options are less for ESBs, deployment tends to be relatively quick and easy.
- Also, the ESB approach lends itself to supporting highly distributed deployments because of the presence of intelligence at each node rather than the concentration of intelligence in one or more hubs.

However, there is also a potential trade-off in the area of functionality – there will always be complex functionality that is generally restricted to proprietary integration solutions, often due to the lack of coverage of mature standards, and in general the proprietary

solutions will have broader sets of pre-built adapters that have been created over the years to satisfy customer needs. Due to the increasing acceptance of ESBs, now most EAI vendors are offering ESB too.

3.2.2 Technologies for SOA Service Registry

A sophisticated service registry enables service control and governance throughout service lifecycle. This can be achieved by integrating SOA service broker with SOA service registry and governance framework. The current solution options for SOA service registry with governance framework can be the ones provided by Systinet and Infravio, where Systinet is a recognized market leader and Infravio is a new comer. Also, Blue Titan has integrated Service Registry and SOA Police Enforcement Framework into their SOA Fabric products.

In addition, Systinet provides a unified service registry and metadata repository solution. Extending service registry with metadata repository turns SOA into an even better described and governed ecosystem of services [W13].

3.2.3 Technologies for Governance Policy Support

There are several vendors that offer SOA governance capabilities. ZapThink picks WebLayers as current best choice [Blo1]. It said that WebLayers distinguishes itself from the others in several important ways. First, WebLayers' focus is on the SOA design to deployment cycle, providing governance and conformance for architects, developers and system engineers to maximize the benefits and cost savings from the SOA implementation. Second, WebLayers built WebLayers Center from the ground up as a governance application, while other SOA governance vendors have repurposed registries or asset management repositories to serve as governance tools. And third, WebLayers remained in stealth for over two years honing their product, so that when they finally launched WebLayers Center, it came out of the gate as a reasonably mature, enterprise-level product.

Systinet extends its Systinet Registry offer with automatic policy management and life cycle applications and services to support SOA governance. It has announced the Governance Interoperability Framework (GIF) that is intended to provide a common approach for publishing and discovery of service metadata, with interoperability between the registry and other components of the SOA infrastructure such as Service Management, Security, and Integration.

Infravio has also integrated its service registry offering with governance and life cycle management support with Infravio Ensemble, an Extended Web Services Management Suite.

Layer 7 Technologies provides an environment for governing SOA policy across loosely-coupled services that addresses issues of context and identity across security domains. Their *SecureSpan* product suite (includes the *SecureSpan Gateway*, *SecureSpan Bridge*, and *SecureSpan Manager*) enables the establishment of PKI-based trust relationships between portals and Service providers, provide policy authoring and validation, and automatically provision policy across all Service endpoints. Administrators define

policies inside the SecureSpan Manager product and then publish them for runtime enforcement to the SecureSpan Gateway for the portal to execute, or to a UDDI registry for centralized storage [Blo3]. It solves portal challenges in cross domain security requirements, e.g. authentication and authorization, and resolves the identity silo problem by federating identity information. It provides a critical part for a secure SOA infrastructure.

AmberPoint provides policy-based runtime governance software that enables service network discovery and lifecycle management, advanced policy definition and management, runtime versioning management, and service brokering and virtualization.

3.2.4 Technologies for Service Contract

As described in [Sch3], at the core of all Web Service contracts is the content in Web Services Definition Language (WSDL), which forms the basic contract for a Service provider. However WSDL is not sufficient for defining non-functional requirements such as security, process, reliability, and service level requirements for a service, nor does it even come close to defining the semantic requirements for the data a service actually produce. Therefore, service contracts require more than just WSDL.

Ideally, there would be a universal Service contract definition language that would include the description of Service semantics, the definition of Service capabilities and constraints, and an interaction model for both data and behavior. However, no such single language exists yet. Instead, there are a series of additional specifications that augment WSDL now in development that seek to add to the richness of Service contracts, but do so in a piecemeal fashion. At the forefront of these contract-related specifications are policy specifications such as WS-Policy. WS-Policy doesn't provide the actual semantics for defining policies, but rather a generalized container for specifying a range of policy considerations. Therefore, we need additional specifications like WS-Security, Web Services Business Process Execution Language (WS-BPEL), Web Services Choreography Description Language (WS-CDL), Web Service Level Agreements (WSLA), and Web Service Offerings Language (WSOL) to specify non-functional requirements in greater detail. Also on the horizon are specifications like the Ontology Web Language for Services (OWL-S) that aim to provide a more complete, rich language for defining Service contracts.

3.2.5 Technologies for Event-Handling Middleware

The event-handling middleware, as an extension to ESB, enables event-driven architecture co-performing with SOA, which provides business flexibilities. WebMethods and TIBCO have provided such event-handling middleware based on event-driven architecture that works together with and is complementary to service-oriented architecture (and their ESB offers). Both companies combine EDA and SOA into a common platform for addressing business needs that demand the support of both architectures. This has a significant advantage because it allows enterprise to implement their event-driven business processes using SOA. For instance, when a customer places an order on a web site, the event that is generated can be used to initiate the SOA-based processes to handle the order. The so called Web Service Gateway can also be

implemented by these event-handling middleware products to provide controlled access to the ESB for external partners.

3.3 Technologies for BPM Support

As discussed in subsection 2.3.2, Business process management and service-oriented architecture are a natural match. One conceptual technology model provided by SAIC is described in [W12], which is illustrated in Figure 3.3.

It is the current trend that the ESB vendors integrate BPM tools on top of their enterprise service bus. Such as IBM has integrated WebSphere Business Integration Modeler to their ESB offer, and Sun Microsystem SeeBeyond integrated ESB with BPM tool in J2EE platform. Cape Clear integrated their BPM product, Cape Clear Orchestrator, to their ESB that offers a comprehensive Business Process Execution Language (BPEL) runtime, along with extensive graphical design and management capabilities.

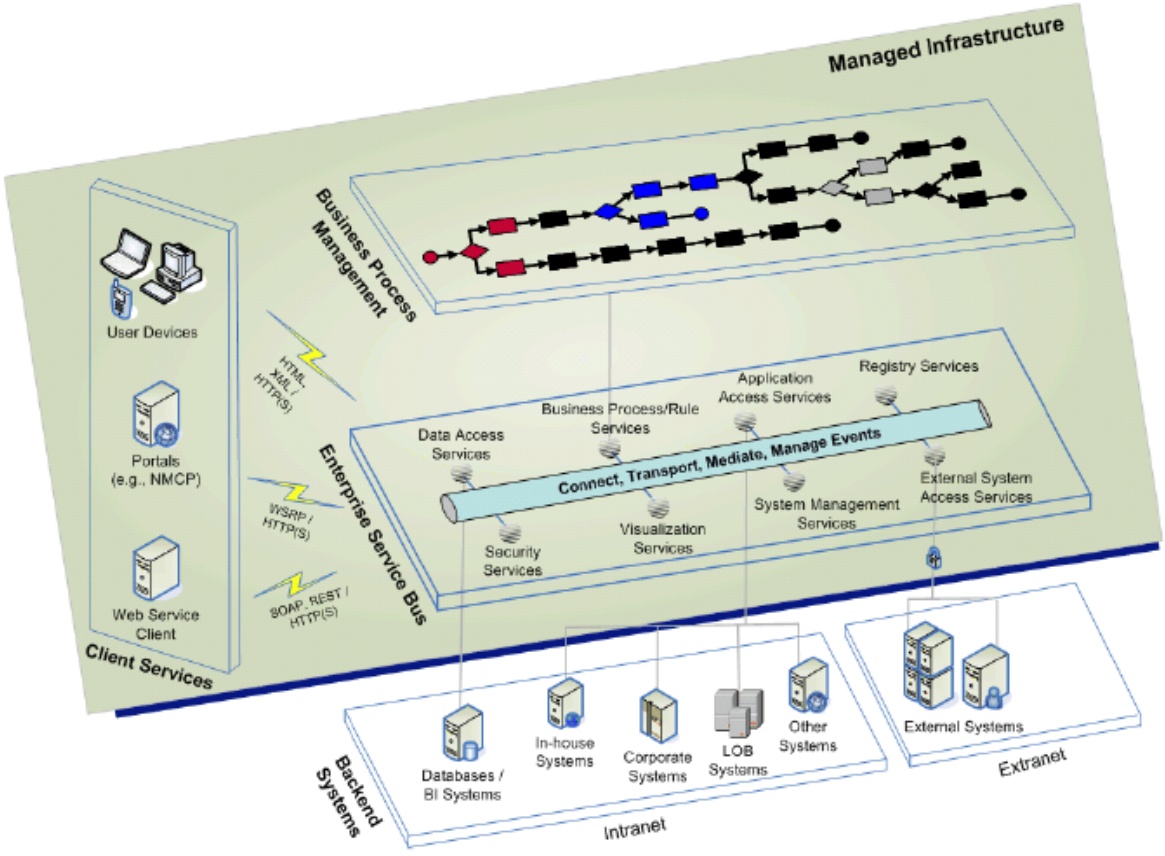

Figure 3.3 A conceptual technology model for SOA infrastructure with BPM and ESB

4.0 The ROI for SOA

The challenge of calculating ROI (return on investment) for SOA is that architecture, by itself, doesn't offer specific features that companies can readily identify with some particular return. After all, architecture is an investment that companies must make in advance of any return, and must continue to make over the lifetime of their SOA implementations. To calculate ROI for SOA, one must understand the full range of SOA value propositions, such as the benefits we addressed in section 1.0. One of the fundamental values that SOA promises is in coping with unpredictable business changes which are hard to predict before it happens.

The ROI for SOA is discussed in a ZapFlash report [Sch2], where the basic benefits for SOA are discussed in four categories:

- Reducing integration expense
- Increasing asset reuse
- Increasing business agility
- Reduction of business risk and exposure.

They are consistent with the benefits we have addressed in section 1.0. The ZapThink's position for ROI is that "Because of the multi-faceted nature of the SOA value proposition, ROI calculations for SOA projects can vary greatly from one project to another. Rather than seeking a single ROI goal for an SOA implementation, companies should take the same iterative, composite approach to ROI that they take for SOA implementation itself. For example, every time they define a Service as part of a company's Service model, they should also define a corresponding ROI objective for that Service. How much will they spend on this Service? What direct and indirect returns can they realize from this Service's implementation? Furthermore, as this particular Service is reused in the company, how will the composition of the Services into processes realize additional ROI for the business? [Sch2]

In many cases, SOA implementations can provide a clear, positive ROI from the first day a Service goes live. However, it is more likely that ROI expectations, like SOA implementations, should be iterative in nature, frequently assessed, and composite. In doing so, practitioners can not only quantify, but also realize the ROI of their SOA implementations.

5.0 SOA Practice Examples

In this section, we demonstrate some SOA practice examples first with lessons learned. Then, we discuss about best practices and worst practices.

5.1 Case Studies

Forrester Research published a report of SOA case studies in September this year [Hef2]. It indicates "To build a comprehensive service-oriented architecture (SOA) platform, enterprises must decide their objectives for SOA, determine how to leverage their current infrastructures, and choose whether to adopt emerging SOA specialist products. Although firms can and do use web services for simple solutions without this deeper level of thought, the seven case studies in this report show how an SOA platform requires a combination of "jump in now" and more comprehensive platform planning." The report describes how seven firms built their SOA platforms, and shows that beyond simply being a better way to do application integration, SOA is an effective tool for transforming business and creating stronger connections between business and IT. The seven firms are:

- **Queensland Transport:** an Australian government agency, used SOA to accomplish multiple government transformations. More than just enabling better application integration, Queensland Transport saw its move to SOA as an opportunity to rethink and restructure its value chain.
- **H&R Block Financial Advisors:** H&R Block Financial Advisors is a small division of H&R Block, the well-known US-based tax firm. It started with SOA based data access and built up to higher-impact business services for distributing sales leads.
- **Unique (Flughafen Zurich AG):** Unique (pronounced YOU-nick) is the firm that operates Zurich Airport. It used SOA to build a portal to provide integrated airport management across several outsourced applications.
- **Thomson Prometric:** Thomson Prometric is a division of Thomson Learning and a member of the Thomson family of companies. It provides certification testing services for organizations ranging from the American Concrete Institute to the University of London. It used SOA to overcome limitations of its core applications and enable partners to provide customized versions of Thomson's Web site.
- **Provide Commerce:** an online retailer of perishable goods. It built a SOA platform with supply chain and eCommerce applications from the ground up to support its new business model.
- **Large Financial Firm:** As a typical large enterprise, this financial firm had a wide diversity of IT projects in progress simultaneously. Rather than developing separate integration solutions for each project, the firm sought to provide a common access architecture to share the business logic of applications that were spread across disparate technology platforms. SOA provided the right design

model for reusable business services, and web services provided an open, standard protocol to connect across platforms. The firm built an infrastructure for service access across all of its major application platforms. The infrastructure was designed for general use across a variety of business solutions and scenarios, and the unified service delivery network is their major focus.

- **Large North American Bank:** This large North American bank also had many IT initiatives in progress at the same time. To reduce development costs and to speed up solution delivery, the bank sought to improve access to key functions that could be shared by multiple applications. The bank adopted SOA as a design strategy to promote reuse, and toke business agility as the real value of SOA. The bank built an infrastructure for service access across all of its major application platforms and for use across a variety of business solutions and scenarios, including partner interfaces. They have implemented a diverse service delivery network.

More details about each case can be found in the report. Other cases can be found in reference are

- **Motorola:** Its SOA adoption path can be found in [Red].
- **British Petroleum:** SAIC has helped British Petroleum implementing a SOA based real-time data operation project, which has been institutionalized in the enterprise level. Its success story in SOA/Web Services is discussed in [Gre].

5.2 Lessons Learned

The lessons learned during SOA platform and infrastructure adoption by the seven companies are [Hef2]:

1. **Build from your existing infrastructure.** Queensland Transport's case — where the initial infrastructure investments were only in an XML authoring tool and a custom, lightweight XML framework — demonstrates that you can make a strong and successful start on SOA by working from your existing infrastructure. The North American bank and H&R Block Financial Advisors each added only one new product to create their SOA platforms.
2. **Start with your Service Delivery Network (SDN).** Because SOA is about making connections across boundaries (technology, security, organization, or application), the SDN was commonly the first SOA platform investment made by the case study firms.
3. **Rely on major vendors for SOA cornerstones.** Most major vendors are pursuing a wide variety of SOA capabilities, and Queensland Transport finds that it is more effective to focus analysis on fewer vendors with broader capabilities. Provide Commerce uses Microsoft and HP as its two cornerstones. The bank relies heavily on IBM. Aside from reducing the number of vendor relationships to manage, this strategy reduces the risks of product incompatibilities and small vendor failures.

4. **Have a service-level agreement (SLA) management strategy from the start.** Provide Commerce's experience highlights the importance of a common business-IT understanding of requirements for service availability, reliability, performance, and scalability to an appropriate management strategy. A management strategy does not require the purchase of a WSM product: Five of the seven case study firms manage service operation using existing management tools.

5. **Be careful when putting business logic in the delivery network.** Unique found that when it put business rules into the SDN, it was hard to maintain the consistency and coherence of business rules. Unique found that it was better to consolidate business rules as much as possible into the implementation of the service, giving control over service business logic to one organization: the service provider. Other case study firms, such as Provide Commerce,

   H&R Block Financial Advisors, and Thomson Prometric, might have orchestration and transformation logic running in the SDN, but they found it important to author such logic in close coordination with the underlying services.

6. **Define governance of data and service semantics.** Three firms — Unique, the financial firm, and the North American bank — emphasized the importance of ensuring that all stakeholders have a common understanding of data and service semantics. Even slight misunderstandings of data usage can cause major issues. The financial firm has gone the farthest in its SOA platform to capture, share, and manage semantics, using Contivo to manage data and interface definitions.

7. **Carefully evaluate SOAP/XML performance.** Clearly understand the response time and latency requirements of your applications, and build prototypes to understand precisely what performance you can expect from your SOA platform. The financial firm found it necessary and appropriate to invest in an XML acceleration appliance. H&R Block Financial Advisors found that even long data access latencies using SOAP were less important than improved data access. Other clients have told Forrester that they tried SOAP, but because of its performance, they are sticking with native protocols for many internal connections.

8. **Don't worry about a repository until you have the discipline to use it.** It is significant to note that at present, not one of the case study firms has a formal service repository. The financial firm comes closest, with its use of Contivo for data and interface definitions. Why don't they have one? They have found that when they are early in their SOA efforts, when the number of interfaces is manageable, or when SOA is used among close-knit teams, they can be reasonably successful by using tactical methods to communicate available services. They have also learned that as their body of services grows, a repository alone is not a complete solution. The investment in a repository will be worthwhile once the firms have matured their processes and disciplines for using one.

9. **Evaluate service orchestration now.** Thomson Prometric and H&R Block Financial Advisors both found orchestration (for example, WS-BPEL process flows) highly valuable. Prometric constructed knowledge worker process and workflow applications across underlying services. When H&R Block Financial Advisors started using orchestration to tie together data access services, business

people were surprised at how fast IT got things done. In addition, BPM and orchestration are on the future evaluation lists of Unique, the bank, and the financial firm.

10. **Develop your own internal standards for Web services usage.** With so many Web services specifications circulating in the market (50-plus) and with so few of them actually set as standards (about 15), there is much room for confusion in using Web services, and confusion leads to a lack of interoperability. To avoid losing control, make explicit organizational decisions about which specifications and standards to use and how to use them. Most of the Web services specifications are implemented as additional items within the header of a SOAP message, and different choices on what is in the header can lead to problems. In addition, you need clarity on what is in the header, because many service management capabilities are based on the data that is available in a message header. As architects at the bank say, "If it's not in the header, you can't manage it." Base your standards on the usage profiles from the Web Services Interoperability Organization (WS-I).
11. **Adopt your own web services interoperability testing regime.** In defining internal usage standards, make sure to prototype and test usage across multiple vendor implementations. The bank found it particularly important to test the interoperability of security standards (e.g., WS-Security). The financial firm highlighted immaturity in web services specifications for reliable delivery.
12. **Craft a vision to guide tactical evolution of your SOA platform.** Even when these firms made tactical SOA platform investments, they were guided by a longer-term vision. For example, neither the financial firm nor the bank has implemented WSM. Instead, they have made a tactical choice to manage their services using existing management tools. However, they are planning for future WSM use by carefully planning and controlling their use of headers in SOAP messages.

5.3 SOA Best Practices

The best practices emerged from the seven case studies can be summarized as the following [Hef1]:

5.3.1 Use SOA to rethink business value chains

The Queensland Transport and Thomson Prometric cases demonstrate the use of SOA to change the nature of interactions with partners and customers. Architects at the case study firms had different ways of emphasizing the value of a strategic SOA focus. They included:

- **Focus on "to be" business process design:** Thomson Prometric's architect agreed that if you approach SOA only as a matter of better application integration, then that is all you will achieve, and that you will achieve much more if you approach it as a way to transform business processes for greater levels of efficiency and

effectiveness. In other words, envision your SOA solutions by examining how your business processes *should* run, not how they *do* run.

- **Focus on strategic business capabilities:** The architect at Provide Commerce described SOA's potential impact by saying that SOA's most important focus is on the strategic operation of the business — SOA's business services embody in software the strategic capabilities of your business. An upfront business-IT partnership to clarify business needs is the critical element to make this happen.
- **Focus on "pluggable business":** The North American bank's architects noted how good service design enables opportunities to plug external parties seamlessly into your business processes. This might range from individual steps in a process (like Queensland Transport's independent vehicle inspectors) up to broadly scoped business process outsourcing.

5.3.2 Use business process design to drive service design

The theme of process-driven service design is directly echoed in six of the seven case studies. Provide Commerce began with a joint business-IT analysis of its core business strategy, mapping the strategy down to the business processes necessary to achieve success. The process definitions served as the foundation for designing its body of business services. Although H&R Block Financial Advisors started its SOA initiative with SOAP-based data access services, it was only when it used process orchestration that its application delivery efforts accelerated to the point where the business was surprised at how fast IT could deliver results. H&R Block's architect emphasized that service design is not object-oriented design — it requires a different mindset focused on processes. Architects at the North American bank also emphasized that understanding the business process context is critical to good service design.

Unique's architects combined its process-focused design approach with a strategy of separating service design and service implementation into two separate discussions. First, it would work with service users to analyze business processes
and design the appropriate service interfaces. Then it would work with implementers to build the services behind the interface definitions. If necessary, it would bounce between the two to.

5.3.3 Ensure all stakeholders have a clear understanding of services

Several of the case study firms stressed the importance of ensuring that services and their operation are clearly understood by both business users and IT implementers. Major areas highlighted by the case study firms were to achieve:

- Clarity on how SOA business services relate to the business.
- Clarity of data and interface semantics
- Clarity of service design principles

5.3.4 Gain buy-in and funding by thinking strategically and acting tactically

With one exception, all of the firms worked from their existing platform and evolved their SOA from there. The financial firm and the bank had the two largest and broadest SOA deployments among the seven case studies, and they had the most to say about buy-in and funding for SOA. Their advice included:

- **Get executive commitment to an overall SOA strategy:** "Get executive buy-in" is not new advice, but the bank and the financial firm had important nuances in how they approached buy-in. First, they did not try to gain executive support for big upfront investments in SOA but rather sought buy-in for 1) dedicating resources to architectural planning of their SOA strategy; and 2) applying the SOA strategy within the context of major application projects. This entailed a combination of demonstrating the general value of SOA and demonstrating practical ways their organizations could evolve to SOA. Even though individual projects must still pay for their SOA investments, this let the bank establish central funding for SOA *strategy*, including the critical step of coordinating the various project-level investments by which the bank built its SOA. Unique and Queensland Transport also emphasized the importance of building SOA as part of real business projects and not as a separate infrastructure project.
- **Leverage executive support to push SOA objectives:** Architects at the financial firm used their CIO's strong SOA buy-in to bring others on board. By citing the CIO's support, they could pressure project teams to give adequate attention to SOA-based options for application delivery. This enabled the firm to get real-world experience with SOA on small projects, after which they graduated to doing a couple of big, highly visible projects. Success with big projects enabled them to further leverage the CIO's support to promote certain SOA investments to enterprise-level funding.
- **To justify SOA, look beyond reuse to business agility:** In the words of the bank's architects, "Reuse is nice, but business agility is the real payback." As the bank gained maturity with its SOA, it found that it could more easily and quickly address the changing needs of the business.

5.3.5 Other Best Practices

Other best practices from the Forrester's report [Hef1] include:

- Be prepared for service support issues.
- Give service ownership to functional teams, not to a central architecture team.

5.4 SOA Worst Practices

In contrary to the SOA best practices, IBM presents some anti-patterns [Ang] as the worst practices for people to be alerted, and not to repeat the same mistake. More details can be found in the reference.

6.0 Critical Success Factors

There are a number of critical success factors can be considered for SOA adoption. Some of them are discussed in the following sub-sections.

6.1 Management of End-to-End Service Life Cycle

One of the critical success factors for SOA adoption is the ability to manage the end-to-end services lifecycle. As pointed out in [W13], it includes:

1. The ability to promote reuse enabled by:
    a. Describing and categorizing services and other components according to their business use;
    b. Establishing a Service Level Agreement (SLA) description for the technical details of services and their use;
    c. Services and deployment models described in enough detail to be understandable to developers, architects, and business analysts;
    d. If a service is available in more than one version, then key differences between versions should be manifested. This includes the business service lifecycle through retirement;
    e. If a service is compliant or not with organizational standards, then this must be available and well known to the communities who use the service;
    f. The security considerations and APIs of a service must be understandable to the communities who are interested in this aspect of the service; and
    g. The information about which business unit owns a service, which development organization is responsible for the engineering for a service, which technology operational unit runs the service, and which operational group manages support for the service must be available.
2. The ability to understand the run-time characteristics and dependencies of the technical environment such as aggregate characteristics of the SLA compliance; the health of a service and its operations from an end-to-end perspective; and the ability to monitor usage.
3. The ability to view and report the above qualities from a variety of viewpoints and perspectives including geographic, product, accounting, line of business, and component dependency characteristics.

6.2 Service Aspects Consideration Cross Multiple Service Spectrum

A discussion on SOA critical success factor with considerations of different service aspects in multiple service spectrums can be found in [Mac]. It addresses SOA and broader IT services in three spectrums: business function services, infrastructure services,

and lifecycle services. Various questions in three categories of aspects are considered, as shown in Figure 6.1.

	Functional contract aspects	Quality-of-Service contract aspects	Commercial contract aspects
Business function services	How can we align flexibility / reusability requirements for business software functions, to business needs?	How can we trade-off flexibility / reusability of business function services with efficiency / openness requirements?	How can we ensure that the right consumers get the right kind of experience from these services, and do so cost-effectively?
Infrastructure services	How should infrastructure elements provide their services to different business functions? How should infrastructure be optimally managed?	How can QoS responsibility be delegated to infrastructure in a way that is easily flexed in response to changing requirements?	How can we minimise the cost and risk of overall process support while creating an business support environment with more moving parts?
Lifecycle services	How should we differentiate lifecycle service levels for different kinds of business function, infrastructure?	How should we define and enforce development, fault-fix and change-request priorities?	How can we demonstrate the overall value of the services that the IT organisation and its resources provide to the business?

Figure 6.1 Example architectural considerations in the context of particular IT services and contract perspectives

6.3 Service Performance

The service performance could be another critical success factor that reflects service quality directly with affects to real-time operations, which should be elaborated. The service performance can be impacted by the following decisions [Bie]:

- Service granularity and placement
- Binding choices
- Message parsing and data volume
- Security model
- Network bandwidth

These considerations are mainly focused on the protocol needed for service invocation, the amount of information that flows across a service interface, and the need for security-related networking interactions. More details can be found in [Bie].

Other considerations regarding to performance include capacity planning for performance prediction and model simulation for performance measurement and verification.

7.0 Risk Assessment and Mitigation

As disscussed in [Bie], when embarking on an SOA roadmap, the first action of the governance body should be to develop an initial *readiness and risk assessment*. The governance body should then periodically update this assessment during the lifecycle. An example of this assessment, as illustrated in Figure 7.1, shows important aspects and criteria that need to be taken into account. The scale values and the specific criteria can be chosen based on the situation of the individual project. The goal of this assessment is to identify the business, organizational, and technical gaps and roadblocks between the current state of the enterprise and a future service-oriented business model.

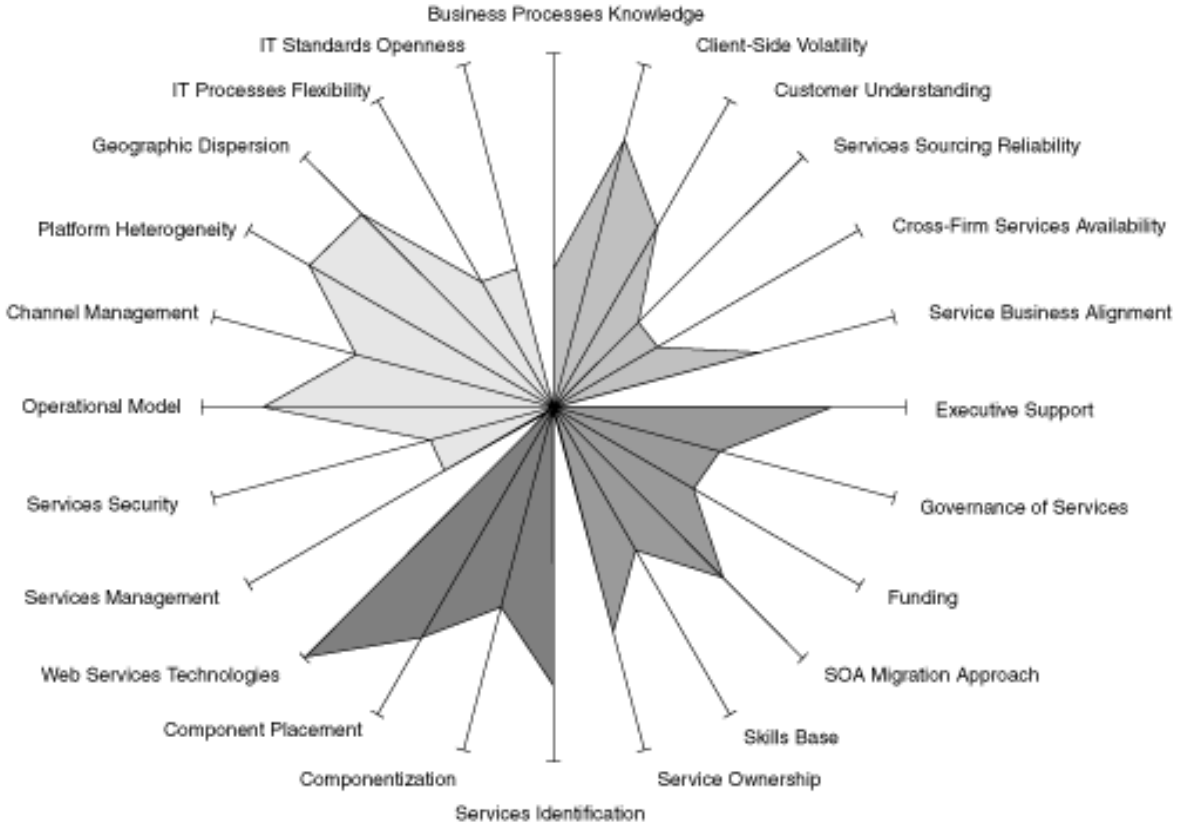

Figure 7.1 An SOA readiness and risk factor assessment, which is adopted from IBM internal SOA assessment practice.

This kind of assessment should balance the vision of the SOA-based solutions with the delivery capabilities of the IT department and should help establish specific business cases for the SOA in the organization. It includes evaluation of both business readiness and IT readiness. It requires understanding of customers and partners, and it should be determined that if the changes of client's or partner's needs can be mapped to existing products or applications in a service-oriented fashion. The assessment then suggests

possible action plans, with focus on improving the less mature aspects of the enterprise relative to the SOA. These improvements to develop the SOA should be executed in well-planned, incremental projects.

The risk and compliance framework proposed by IBM [W11] can be applied to SOA practice as well.

Also, a white paper from Sun Microsystem is included in the reference [W10], which provides Sun's approaches in SOA impact analysis and SOA readiness assessment.

Other risk considerations include the evolution of a large number of standards and vendors and the continuous changing for the landscape of vendors, which make the COTS product selection and standard complying decision difficult. The risk mitigation approaches include separating conceptual models from technical implementations; applying open and component-based architecture to minimize the impacts of technology changes; limiting the number of vendors involved, etc.

8.0 SOA Practice Reference

The SOA provides a practical way for a business to bridge IT advancement with business benefits, and it enables a view of full spectrum value proposition for new investments and performance measurements. A federated SOA model provides a natural way for the implementation of the federated enterprise architecture for a large organization. The deployment of SOA across business boundaries enables sharing, reuse, and the flexibilities in handling composite business services. Enterprise SOA for a business will provide a blueprint for integration of multiple components across the business organizations, such as business programs, strategic planning, enterprise architecture, IT investment, engineering, etc.

8.1 Possible SOA Adoption Steps

SOA adoption could be considered in following steps:

1. **SOA Service Baseline:** A current baseline for SOA adoption should be established first. It includes
   - Assessment of current SOA projects in operating divisions as well as state and local. Identify projects that can be considered as starting points or pilots, and can be promoted for reuse and sharing across agency;
   - Assessment of existing infrastructure for SOA enabling. Create an evolution plan based on current infrastructure, and keep maximum legacy reusability in mind;
   - Create and maintain an agency level common services portfolio.
2. **Strategic Planning:** Incorporate SOA adoption into IT Strategic Plan and IT Tactical Plan, which include:
   - Adopt SOA as an IT strategy to support business goals.
   - Incorporate SOA life cycle model and governance model into IT Strategic Planning Program
   - Include SOA infrastructure and infrastructural service plan in the IT Tactical Plan.
   - Create SOA roadmap and include it into the IT Tactical Plan
3. **Enterprise Service-Oriented Architecture:** Incorporate SOA approaches into Enterprise Architecture to create an Enterprise SOA Blueprint that could consist of a federated SOA model for a large organization. The federated SOA model enables each sub-organization to run its own SOA infrastructure, and to be responsible for its domain services. The infrastructure services provided in top organization level should enable the common governance policy and service sharing across agency in a federated way.

4. **Implementation Coordination:** A federated SOA model enables the collaboration of SOA implementations across organizations. The top organization level should help in
    - Promote SOA solutions based on existing projects, service portfolios, and available infrastructures across the agency.
    - Coordinating SOA initiatives across organizations, and help in consolidating and constructing shareable services across sub-organizations. Provide architectural support and implementation coordination in order to make business and technology service components applicable in enterprise scope and across business boundaries.
5. **Institutionalization:** After being mature and successful in the early phases of SOA adoptions, the SOA approach can be institutionalized across agency in large scale with penetration to all suitable business domains.
6. **External Extension:** Finally, SOA will be applied in external services as well to collaborate with business partners, customers, and general public.

8.2 Possible SOA Solution Approaches

8.2.1 Enterprise View

In order to maximize SOA benefits across a big organization, plan ahead in enterprise level is very important. This includes:

- Incorporate SOA modeling into EA development, and apply federated service planning across organizations.
- Get executive commitment to an overall SOA strategy, and encourage stakeholders' participation, so that to leverage management support to push SOA objectives.
- Leverage best practices, experiences, tools, etc. across organizations.
- Use experienced practitioners to define first set of infrastructure and business services that translates business requirements into service descriptions.

8.2.2 Federated SOA Model and Service Infrastructure

The Federated SOA model and service infrastructure (as shown in Figure 2.3) will be a good approach for a big organization. Each sub-organization can define and operate its own domain services and sub-infrastructure and services. The federated infrastructure, service collaboration, common processes and services, enterprise level governance and policies could be defined and managed via a cross organizational governance body.

8.2.3 Layered Services

The service definitions can be organized into layers, such as in infrastructure service layer, business service layer, application service layer, and technology implementation service layer. The services in each layer can be defined by the experienced practitioners

in that domain. However, this doesn't mean isolated work in each layer. Business team and IT team should work hand-in-hand, which is consistent with the above approach. The core of SOA is about flexible business processes, IT is the means to enable it. Business Process Modeling is the first critical step, and the business services can be created based on business process models.

8.2.4 Component-Based Architecture and Implementation

Although SOA does not require component-based architecture (CBA), the component-based architecture does provide advantages in domain function articulation and modular implementation, which provides manageable scope and makes incremental progress easier. The CBA allows large and complex systems being componentized into self-content smaller components. The components can be componentized further into sub-components recursively for refinement. Defining services based on components was introduced in CORBA [Pop], and now the concept is extended to business domain as well, which is reflected in Federal Enterprise Architecture (FEA), a component-based and service-oriented architecture. The CBA approach works well with the layered services discussed above.

8.2.5 Governance Enforcement

Governance is critical for success. The governance can be enforced by governance structure and policy. It is helpful to have some initial projects that can demonstrate end-to-end implementation process. Also, it well be helpful to

- Build an *SOA Center of Excellence* or something alike to share project experiences across agency
- Use well defined processes and documentation
- Establish Architectural guidelines early
- Establish organizational infrastructure to ensure optimal reuse
- Integrate all aspects of SOA lifecycle including deployment

8.2.6 Iterative Evolution

Considering the size and complexity for an organization, an iterative evolution approach could be appropriate, which enables SOA migration in phases. The SOA adoption requires the business process models and application systems being incrementally modernized with new functionality or system components being incrementally deployed. Also, SOA-based modernization is a continuous process, as described in SOA life cycle models. Some major points to consider including:

- Select a set of well defined business process and application areas that have shared value, to implement an end-to-end SOA solution with a suitable infrastructure construction, which enables the experience and test of the SOA infrastructure and all critical components for SOA adoption.

- Use the enterprise SOA Blueprint to establish the target service oriented architecture and a transition roadmap (can be included in the IT Tactical Plan), so that the path to SOA adoption will be goal-driven and incremental.
- Leverage Existing data, back-end processes, and systems via adaptor technologies and legacy system integration.

8.2.7 Embracing Standard

Adopting a core set of industry standards is critical for realizing the benefits of SOA. These standards enable the interoperability between the components in SOA. In practice, the web service standards as mentioned in section 3.1 continue to be adopted as de facto standards around SOA. Tools are evolving that facilitate the composition of complex workflows and dynamic service invocation, where BPEL has emerged as the leading specification to standardize service orchestration and process automation [W15].

Standard adoption will make life easier along the way, which will enable more opportunities to adopt COTS products and tools and to drive down the cost. Although SOA standards are still evolving, the iterative and incremental approach enables SOA implementation to evolve accordingly after infrastructure and core capabilities are in place.

9.0 Conclusion

This white paper has surveyed large amount of material in public domain as well as incorporating author's views and practice experiences regarding to SOA concepts, technologies, and practices. It discusses SOA adoption with widely covered topics. It intends to provide a comprehensive reference for SOA adoption, especially for a large organization. It has also recommended some possible steps and approaches for practice. We understand that conducting an enterprise SOA practice is a challenging and daunting task. CGI Federal can provide thought leadership and assistance in the following areas:

- **Strategic planning and resource management planning:** Development of enterprise IT strategic plan, tactical plan, and resource management plan in supporting enterprise SOA adoption.
- **Enterprise architecture:** Development of Enterprise SOA blueprint and roadmap along with supporting artifacts such as guiding principles and architectural product selection and evaluation.
- **Governance and life cycle development:** Development and implementation of SOA governance and life cycle management models that are required to support enterprise SOA implementation and deployment.
- **Business process management:** Design of "To-Be" business processes and implementation of business process orchestration in concert with Business Service Bus, ESB, and portal.
- **Enterprise service bus development:** Design and development of an ESB for phased deployment.
- **Infrastructure design and implementation:** Design and implementation of the managed infrastructure required to support delivery of enterprise SOA services.
- **Portal development:** Design and development of a portal that can integrate with ESB and BPM.
- **Legacy system modernization:** Development of modernization strategies, roadmap, and processes to modernize existing systems, so that to enable an iterative evolution path towards complete SOA adoption.

Appendix A. Glossary

Application Server
An application server is a server-side program in a distributed network that is dedicated to hosting the enterprise application's business logic. It provides the middleware infrastructure as part of a multi-tier application.

Authentication
It is to validate and verify the identity of a user, device, or some other computing entity, often as a prerequisite to allowing access to resources in a system.

Authorization
It is a process of granting or denying access to an individual or computing entity. This allows controlled access to various resources based on the entity's identity.

Business Agility
It is the capability of an enterprise to respond with speed of market opportunities, external threats, or customer demands by changing its business processes that are integrated end-to-end across the company and with key partners, suppliers, and customers.

Business Process
It is a set of logically related tasks performed to achieve a defined business outcome. A process is a structured, measured set of activities designed to meet the business objectives.

Business Process Extension Language
An XML-based language designed to enable task-sharing for service oriented architecture environment by orchestrating and choreographing individual web services.

Business Service Bus (BSB)
It is the set of business services for a special domain that are available for widespread use across an enterprise, such as the services in human resource, logistics, billing, etc. These services are published in the Service Registry.

Choreography
It is a mechanism for orchestrating multiple services together by specifying the linkages and coordination between them to create a business process. It also defines the flow of information among the set of services, participants, and activities.

Component
It is a modular unit that is functionality accessible through one or more interfaces.

Content-Based Routing

A content-based routing service makes intelligent message routing decisions based on the content of the message.

Enterprise Application Integration (EAI)
It consists of software and architecture principles to bring together a set of enterprise applications aimed at modernizing, consolidating, and coordinating the enterprise's IT landscape.

Enterprise Service Oriented Architecture (ESOA)
It is enterprise architecture with adoption of SOA as its architecture style for modeling and for architecture development.

Loosely Coupling
It indicates loosely coupled services, even if they use incompatible system technologies, can be joined together on demand to create composite services, or disassembled just as easily into functional components. Participants must establish a shared semantic framework to ensure messages retain a consistent meaning across participating services. It is enabled by web services (or any SOA).

Middleware
It is software that functions as a conversion or translation layer. It is also a consolidator and integrator. Custom-programmed middleware solutions have been developed for decades to enable one application to communicate with another that either runs on a different platform or comes from a different vendor or both.

Public Key Infrastructure (PKI)
It is a system handles digital certificates, certificate authorities (CA), and other registration authorities that verify and authenticate the validity of each party involved in a transaction.

Service
It is based on an application component deployed on network-accessible platforms hosted by the service provider. It is accessible through interfaces that are described by service description, and it can be invoked by or can interact with a service requester.

Service Broker
A service broker as a service intermediary that manages the invocation of a set of registered services based on a set of policies/rules.

Service Consumer
In the context of SOA, a service consumer finds services from Service Registry and use (or "bind" to) them.

Service Delivery Network
It is a networked service infrastructure across organization boundaries. It facilitates SOA service delivery in distributed environment.

Service-Level Agreement (SLA)
It is a contract between a service provider and a service requester that stipulates a specified level of service. It could contain agreements on service options, enforcement or penalty provisions for services not provided, a guaranteed level of service availability, reliability, performance, scalability, etc.

Service Orchestration
It composes or re-configures business processes based on available services. It provides a way to automate business processes, promoting reuse of services, and making the overall systems more agile to respond to changing business requirements.

Service Oriented Architecture (SOA)
SOA is an architectural style that emphasizes well-defined, loosely coupled, coarse-grained, business-centric, reusable shared services.

Service Provider
It publishes services to Service Registry in the context of SOA.

Service Registry
It contains information for available services in the context of SOA, and it enables run-time service discovery.

System (or Software) Development Life Cycle (SDLC)
SDLC is the overall process of developing information systems through a multi-step process from investigation of initial requirements through analysis, design, implementation and maintenance. There are many different models and methodologies, but each generally consists of a series of defined steps or stages. The examples of models include: waterfall, fountain, spiral, build and fix, rapid prototyping, incremental, etc.

Universal Discovery, Description, and Integration (UDDI)
It is an OASIS standard for platform-independent, XML-based registry to publish and discover network-based software components and services.

Web Service
Web service is a software-powered resource or functional component whose capability can be accessed at an internet URI. Standards-based web services use XML (or WSDL more precisely) to interact with each other, which allows them to link up on demand in loosely coupled manner. Microsoft's .NET and Sun's Sun ONE (J2EE) are the major development platforms that natively support these standards.

Appendix B. Acronyms and Abbreviations

BPEL	Business Process Execution Language (BPEL)
BPEL4WS	Business Process Execution Language for web services
BPM	Business Process Modeling
BSB	Business Service Bus
CBA	Component-Based Architecture
CIO	Chief Information Officer
CMM	Capability Maturity Model
CMMI	Capability Maturity Model Integration
CMS	Centers for Medicare and Medicaid Services
CORBA	Common Object Request Broker Architecture
COTS	Commercial Off-The-Shelf
DNI	Director of National Intelligence
EA	Enterprise Architecture
EAI	Enterprise Application Integration
ebXML	Electronic Business using eXtensible Markup Language.
EDA	Event-Driven Architecture
ESB	Enterprise Service Bus
ESOA	Enterprise Service Oriented Architecture
FEA	Federal Enterprise Architecture
HTTP	HyperText Transfer Protocol
IC	Intelligence Community
OASIS	Organization for the Advancement of Structured Information Standards
PKI	Public Key Infrastructure
ROI	Return On Investment.
SAML	Security Assertion Markup Language (from OASIS)
SDLC	System (or Software) Development Life Cycle
SDN	Service delivery network
SLA	Service Level Agreement
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SSL	Secure Socket Layer
UDDI	Universal Discovery, Description, and Integration

WS-CAF	Web Services Composite Application Framework
WSDL	Web services Description Language
WS-I	Web Services Interoperability Organization
WSIA	Web Services Interactive Applications (from OASIS)
WSIL	Web Services Inspection Language
WSM	Web Services Management
WSRP	Web Services for Remote Portlets (from OASIS)
W3C	World Wide Web Consortium
XSLT	Extensible Style Sheet
XML	eXtensible Markup Language

Reference

[Ang] J. Ang, L. Cherbakov, and M. Ibrahim, "SOA Antipatterns", IBM Corp., Nov. 2005.

[Ars1] Ali Arsanjani and Kerrie Holley, "Increase Flexibility with the Service Integration Maturity Model (SIMM)", IBM Corp., Sept. 2005.

[Ars2] Ali Arsanjani and Kerrie Holley, "The Path to SOA", IBM Corp., Oct. 2005.

[Bie] N. Bieberstein, et., *Service-Oriented Architecture Compass: Business Value, Planning, and Enterprise Roadmap*, IBM Press, Nov. 2005.

[Blo1] Jason Bloomberg, "WebLayers Achieving the Benefits of SOA through Governance", ZapFlash, May. 2005.

[Blo2] Jason Bloomberg, "What to Look for in a SOA Maturity Model", ZapFlash, Oct. 2005.

[Blo3] Jason Bloomberg, "Layer 7 Technologies Solving Portal Challenges With Cross-Domain SOA Security", ZapFlash, Oct. 2005.

[Cha1] K. Channabasavaiah, K. Holley, and E. Tuggle, "Migrating to a Service Oriented Architecture, Part 1", IBM Corp., Dec. 2003.

[Cha2] K. Channabasavaiah, K. Holley, and E. Tuggle, "Migrating to a Service Oriented Architecture, Part 2", IBM Corp., Dec. 2003.

[Chap] David A. Chappell, *Enterprise Service Bus*, O'Reilly, June 2004.

[Che] John Cheesman and Georgios Ntinolazos "The SOA Reference Model", CBDI Journal, 2004.

[Cox] D. Cox and H. Kreger, " Management of Service-Oriented Architecture Life Cycle", IBM Systems Journal, Vol 44, No 4, 2005.

[Cra] Steve Craggs, "SOAs And ESBs – A Marriage Made In Heaven?", Saint Consulting Limited, 2005.

[Cul] Alex Cullen, "What Must EA Do to Sustain SOA?" Forrester research, Inc., Sept. 2005.

[Dod] John. Dodd, "The Service Lifecycle", *SOA Life Cycle and Governance*, CBDI Journal, Nov. 2005.

[Dodd] S. Doddavula and S. Karamongikar, "Designing an Enterprise Application Framework for Service-Oriented Architecture", August 2005.

[Gre] R.M. Gregovic, et., "A Common Approach to Accessing Real-Time Operations Data: Introducing Service Oriented Architecture to E&P", SPE Annual Technical Conference and Exhibition, Society of Petroleum Engineers Inc., Oct. 2005.

[Har] Chris Harding, "The Future of SOA", The Open Group, Nov. 2005.

[Hef1] Randy Heffner, "Real-World SOA: SOA Lessons learned", Forrester Research, Inc., Sept. 2005.

[Hef2] Randy Heffner, "Real-World SOA: SOA Platform Case Studies", Forrester Research, Inc., Sept. 2005.

[Hef3] Randy Heffner and K. Dowling, "Large Enterprises Pursue Strategic SOA", Forrester Research, Inc., April 2005.

[Hef4] Randy Heffner and B. Cameron, "Your Strategic SOA Platform Vision", Forrester Research, Inc., March 2005.

[Kli] Sean Kline and Jon Bachman "ESB meets Business Service Registry", Systinet

Corp. and Sonic Software Corp. and, May 18, 2005.
[Leg] Gene Leganza, et. "Human Services Agencies Turn To Enterprise Framework Software", Sept. 2005.
[Mac] Neil Macehiter and Neil Ward-Dutton, "Real SOA: Critical Success Factors", Systinet Corp. June 2005.
[Mit] Kunal Mittal , "Build You SOA, Part 2 The Service-Oriented Architecture Maturity Model", IBM Corp., Nov. 2005.
[Pal] Nathaniel Palmer, "BPM & SOA", AIIM E-DOC Magazine, Oct. 2005.
[Pez] M. Pezzini, "Cape Clear Has Chance to Lead ESB Integration Market", Gartner Research Note T-22-3919, Technology, April 2004.
[Pop] Alan Pope, *The CORBA Reference Guide*, Addsion-Wesley, 1997.
[Tho] Anne Thomas Manes, "Service Oriented Architecture: Best Practices", Systinet Corp. 2004.
[Red] Toby Redshaw, "Motorola SOA", a PowerPoint presentation, Motorola Corp., 2005.
[Rod] J. Rodriguez, "New Rules govern SOA lifecycle", Procullux Media Ltd., July 2005.
[Rog] S. Rogers and S. Hendrick, "Oracle Builds Comprehansive SOA Platform", IDC White Paper, Jan. 2005.
[San] W. Sanchez, et. "Cancer Biomedical Informatics Grid Prototype Project", caGRID White Paper, July 23, 2004.
[Sch1] Ronald Schmelzer, "Understanding the Real Costs of Integration", ZapFlash, Oct. 2002.
[Sch2] Ronald Schmelzer, "The ROI of SOA", ZapFlash, Jan. 2005.
[Sch3] Ronald Schmelzer, "What Belongs in a Contract", ZapFlash, Aug. 2005.
[Sim] Oliver Sims "Developing the Architectural Framework for SOA", *SOA Life Cycle and Governance*, CBDI Journal, Nov. 2005.
[Spr] D. Sprott, L. Wilkes, R. Veryard, J. Stephenson, *Web Services Roadmap, A CBDI Report Series – Guiding the Transition to Web Services and SOA*, CBDI Journal, 2003.
[Ver] Richard Veryard "SOA Governance – from Chaos to Order – The Transformation of Enterprise Architecture", *SOA Life Cycle and Governance*, CBDI Journal, Nov. 2005.
[Wil] Lawrence Wilkes, "SOA Governance in the Lifecycle", *SOA Life Cycle and Governance*, CBDI Journal, Nov. 2005.
[Wil2] Lawrence Wilkes, "Time to Board the Enterprise Service Bus?", CBDI Journal, July/August 2004.

[W1] "Data Integration in a Service-Oriented Architecture", White Paper, Informatica Corp., October, 2005.
[W2] "IBM SOA Foundation: providing what you need to get started with SOA", White Paper, IBM Corp., Sept. 2005.
[W3] "The Need for SOA Infrastructure", Version 1.0, New Rowley Group, Inc. May 2004.
[W4] "The New Enterprise Integration Framework", The Stencil Group, Sponsored by Blue Titan Software, Sept. 2003.

[W5] “Reference Model for Service Oriented Architecture”, Working Draft 08, OASIS, Sept. 2005.
[W6] “Service-Centric .vs. Message-Centric ESBs”, Cape Clear Software Inc., July 2005.
[W7] “SOA Blueprints Concepts”, Draft v0.5, The Middleware Company Research Team, June 2004.
[W8] “SOA Governance: Balancing Flexibility and Control within an SOA”, Systinet White Paper, November 2005.
[W9] “SOA Maturity Model”, Sonic Software Corp., AmberPoint Inc., BearingPoint Inc., Systinet Corp. 2005.
[W10] “Accessing Your SOA Readiness”, Technical White Paper, Sun Microsystem, June 2004.
[W11] “The IBM Risk and Compliance Framework: addressing the challenges of compliance”, IBM White Paper, Jan. 2005.
[W12] “Building Enterprise SOA Application Enabling Services”, SAIC White Paper, Oct. 2005.
[W13] “The Systinet “Blizzard” Platform”, Systinet Corp., June 2005.
[W14] “Domain Model for SOA, Realizing Business Benefit of SOA”, BEA White Paper, July 2005.
[W15] “Putting the SOA Infrastructure Together: Lessons from the SOA Leaders”, SOA Leaders Council Whitepaper, 2005.
[W17] “Sonic ESB: An Architecture and Lifecycle Definition”, Sonic Software Corp., 2005.
[W18] “Report to ITSC on Enterprise Service Oriented Architecture”, FDA SOA Working Group, Dec. 2005.

AmberPoint, Inc., http://www.amberpoint.com/
BEA Systems, Inc., http://edocs.bea.com/
BearingPoint, Inc., http://www.bearingpoint.com/portal/site/bearingpoint
Blue Titan Software, http://bluetitan.com/products/index.html
Cape Clear Software Inc., http://www.capeclear.com/
Cast Iron Systems, Inc., http://www.castironsys.com
DataPower Technology, Inc., http://www.datapower.com/
IBM Corporation, http://www-306.ibm.com/software/websphere/
Infravio, Inc., http://www.infravio.com/comp_analyst.html
Layer7 Technologies, Inc., http://www.layer7tech.com/
Microsoft Corporation, http://www.microsoft.com/net/
Reactivity Systems, Inc., http://www.reactivity.com/
Sonic Software Corporation, http://www.sonicsoftware.com/solutions/index.ssp
Sun Microsystem, Inc. http://www.sun.com/
Systinet Corporation, http://www.systinet.com/
WebLayers, Inc., http://www.weblayers.com/
WebMethods, Inc., http://www.webmethods.com/