

A Westbrook Technologies White Paper

# SOA

Introduction to Service-Oriented Architectures  
for Electronic Document Management Systems

September 2005

Authored by

Bud Porter-Roth

Porter-Roth Associates



## Table of Contents

Introduction.....	1
The Need for Interoperability by Design.....	1
What is an SOA?.....	3
Business Processes and SOA.....	10
Interoperability .....	13
Security and Web services .....	13
Standards .....	14
Messaging Specifications.....	15
Security Specifications .....	16
Reliable Messaging Specifications.....	16
Web services Interfaces .....	16
Web services Discovery .....	16
Metadata Specifications .....	16
XML Specifications .....	17
Management Specifications.....	17
Business Process Specifications .....	17
What is the Value Proposition for SOA based System? .....	17
Are SOA Systems Ready for Primetime? .....	19
Conclusion.....	20
Westbrook Technologies Fortis SOA.....	22
Porter-Roth Associates.....	22

## **Introduction**

Document management systems have come a long way from their original beginnings. The first systems were centered on scanning and storing paper documents as digital images. While “effective” for that time and the technology available, many users were simply not satisfied with these simple systems and wanted to do more – such as pass the document around a network, manage electronic documents like a Microsoft Word file, and exchange data from the scanned images with a legacy application.

It seems that users of electronic document management (EDM) technology have always wanted more out of the technology and have been pushing vendors to constantly improve and innovate. And, after roughly 20 years of improvement and innovation, electronic document management is no longer a simple document storage solution used to keep digital documents after the work has been completed. Today’s systems have become an integrated component of the daily work whether that is accounts payable, claims processing, or managing records for compliance purposes.

However, as EDM becomes integrated with more mainstream applications, the stumbling block has become the need to hardwire each EDM application to each business or legacy application. The programming effort for this is high, the programming is not reusable, and once made, the connections are fragile. The resulting complex network still does not provide the interoperability needed to share data efficiently and often companies are forced to re-key data from one system to another. Data sharing is often limited by the inability to connect applications due to programming complexity, costs, and time.

The fundamental problem with software and hardware applications is that they are still built as stand alone systems that do one thing well but do not easily interoperate with other systems. Phrases such as “islands of automation” and “data silos” while more than 20 years old, are still valid concepts and still describe the state of the software industry today.

With the advent of current web-based technologies and the drive towards open standards-based networks, the software industry is beginning to develop and write applications that freely communicate with other applications, reduce the complexity of new software applications, and allow existing systems to share data. The Internet and its related technologies are driving software vendors to build applications that are inherently open and capable of interoperating with other applications as opposed to the traditionally closed and monolithic systems that exist throughout industry today.

## **The Need for Interoperability by Design**

Driven by the universal adoption of Web services and convergence of key standards, service-oriented architectures (SOA) have emerged as the key enabling technology for interoperability among disparate pieces of software regardless of the network architecture, platform, or programming language. SOA is not a software product, but is a conceptual framework for designing and building networks and Web services that allow for data to be exchanged where and when it is needed. SOA facilitates interoperability among heterogeneous systems due to its standards-based interfaces.

An SOA may also be extended from within an organization to other SOA platforms that are external to a corporate network – effectively linking many corporate networks to facilitate the exchange of data, business processes, and work. SOA and Web services allow software vendors and corporate IT departments to concentrate on the business model and user needs –the concept of “interoperability by design” is replacing “islands of automation” and “data silos” by allowing applications to be designed and implemented without regard to who owns the data, what platform it exists on, where is it geographically located, or what programming language is needed to access the application.

Underlying the concept of an SOA is the idea that each software application within a business operation can be a “service” that can either provide data to a requestor or receive data that has been requested. Each service on the network is neutral with regard to programming languages and can cross applications or platforms as needed to send or receive data. A service is self-contained and does not depend on the context or state of other services. Even legacy or packaged applications with no inherent Web services built in can be Web service-enabled and function on the network as any other Web service application by requesting and receiving data. Simply put, interoperability between diverse applications, application platforms, and programming languages can be realized using Web services and SOAs.

An SOA offers the ability to build new applications using existing systems without sacrificing network and application reliability, data integrity, and current business processes. Essentially, an SOA allows a company to:

1. Reuse existing software programs. Instead of implementing new software applications, replacing older applications, or writing expensive single use programs to interface with existing applications, SOAs allow you to build a network that uses existing software assets, provides access to data within existing applications, and allows you to implement applications that pull or push data from multiple legacy systems.
2. Reduce integration requirements and expenses for disparate systems. Currently, dissimilar systems need low-level programming to be able to talk to each other. This programming is typically a one-off effort for a particular interface, is not generally extensible, and is not reusable. SOA allows systems to operate on the Inter/Intranet and be available for any Web service applications.
3. Reduces overall risk. Because applications within an SOA network act independently of each other, deployment of new applications does not affect any other application on the network. The risk of crashing the network or corrupting a

A Web service is a self-describing, self-contained, modular software application that provides some business functionality to other applications through an Internet connection. Applications access Web services via ubiquitous web protocols and data formats, such as HTTP and XML, with no need to worry about how each Web service is implemented. Web services can be mixed and matched with other Web services to execute a larger workflow or business transaction.

Web services are not SOAs. Web services provide the standards and tools to build an SOA.

data set is typically not associated with implementing a new Web service application.

4. Increase business flexibility and efficiency. Because new services can be a combination of new applications and old applications, using existing legacy data when available, new business processes can be rapidly built and integrated to meet business demands.
5. Rapid modeling of new services and business processes. Because services act independently of each other, new business services and processes can be modeled and tested without affecting existing applications. Rapid modeling allows business users to participate more directly in the process which, conversely, provides IT with better and clearer requirements. SOAs allow the development of business-centric applications instead of technology-driven applications.
6. Supply to, or accept data from, outside the corporation. SOAs are not limited to the internal network of a corporation and may communicate with other corporations, government agencies, and customers across the Internet.

An SOA is not a software product by itself or even a standard set of software products but is the natural progression of standards, technologies, and concepts. SOAs allow for a stable yet flexible network based on need and application services can be built incrementally as required or permitted by time and budget. Once an SOA is established, it becomes an open and extensible network that can grow or shrink as business demands. Perhaps most importantly, there is no need to replace existing systems or to continue to be locked into a specific vendor platform or architecture. SOA allows you to build best of breed applications by making the network and application connectivity a non-issue.

### **What is an SOA?**

An SOA is a design framework for building applications that can share data. An SOA is an open standards based architecture that enables new or existing software assets to exchange data on demand without a persistent connection to each other. The data exchange may be a simple exchange of data such as the verification of a credit card number, or it may be a complex event in which two or more applications are combined to coordinate an activity such as a business process workflow that checks a credit card number, checks inventory, and approves the transaction.

Because it is not a product, like a database or a CRM application, SOAs are being pushed into the marketplace as a group of related specifications and industry standards. Many IT departments are beginning to implement SOAs without any vendor assistance by using existing IT resources. Also, there is not a single way to implement an SOA, and there are multiple specifications that can be used to create an SOA application, which further distances SOAs from being sold as a product or set of products or a proprietary methodology. There are, however, vendors who have begun to specialize in integrating diverse applications and data via Web services and SOAs and have developed tools that simplify the building of an SOA while adding functionality.

SOAs are derived from the convergence of key technologies such as XML and Web services and the universal adoption of the Internet as a primary communications method.

The goal of an SOA is to provide a neutral architecture in which multiple software applications (services) can send or receive data on demand. New services can be added to the SOA without interrupting existing services and these new services can be both a requestor and a provider of data.

One of the basic outcomes of developing Web service applications for an SOA is that more emphasis can be placed on the business need – where and when data is needed rather than looking for a purchased application that can solve the business need or building complete new applications from the ground up to solve the problem. SOAs allow developers to concentrate on how users work, what data they work with, and how that data request may be provided by existing applications that previously did not communicate.

An example of a Web service application may be the ability of a “New Account Manager” at a local bank to be provided with a credit report from one of the credit services as part of the credit application process. The credit report may be automatically “requested” by the “new application service” as one of the tasks that are completed by the bank as part of the on-line loan application. The New Account Manager would be presented with the credit report and save it as part of your application.

Web services provide the key technologies that enable SOAs to be built by providing the following key capabilities:

1. Applications can communicate over the Web without regard to programming languages or platform architectures. Web services are platform neutral.
2. Web services are self describing so that other services can find them and use them as needed.
3. Web service capabilities are published to a registry so that other services can find them – this is also called a “broker” Web service.
4. Web services are loosely coupled to many other applications and communicate on-demand instead of being hard wired to a single application.

Loosely Coupled vs. Tightly Coupled. One of the key attributes of an SOA is that it is a loosely coupled architecture. But what does that mean? The traditional method of linking two applications was to “tightly couple” them such that they communicated exclusively on a one-to-one basis. The applications are virtually locked together through programming and changing one will directly affect the other. This is also sometimes called “hard wiring” the applications together.

Loosely coupled applications, on the other hand are not dedicated to each other and changing one will not have a direct affect on any others. In a loosely coupled network, many applications can be on the network accessing each other when needed. They are not hard wired together and find each other when needed through a registry

The above key capabilities are made possible due to four open standards-based specifications. While there are many standards that play a role in building an SOA, the following are the basic building blocks of an SOA.

**Extensible Markup Language (XML).** XML is a cross-platform, extensible, and text-based standard language used for communications and data exchange across the Internet. XML is a formal specification of the World Wide Web Consortium (W3C) and has become the universal format for structured documents and data on the Web. XML is intended to represent the content and structure of common data formats, and provide a flexible way to share both the format and the data on the World Wide Web, intranets, and elsewhere.

**Web Services Description Language (WSDL).** WSDL allows a Web service to be self-describing to potential users of that service, what operations it will perform as a requestor or provider, and how to access it. WSDL is language and protocol neutral and used in combination with SOAP and UDDI to provide Web services over the Internet.

**Simple Object Access Protocol (SOAP).** SOAP allows Web services to communicate with each other using XML messages. SOAP messages are independent of any operating system or protocol and may be transported using a variety of Internet protocols, including SMTP and HTTP. SOAP also provides a way to access services, objects, and servers on remote systems without regard to the remote system's location, operating system, or platform. (Note: SOAP is now a standalone term and is no longer an acronym.)

**Universal Description, Discovery, and Integration (UDDI).** UDDI serves as an XML-based registry for Web services and allows one service to discover and access another service. UDDI is platform independent and provides information about local services as well as services outside of the enterprise. The registry defines standards-based descriptions of the Web services available in addition to defining how the services need to interact. UDDI is often called the "Yellow Pages" of Web services.

A key to making an SOA work is the idea that a service may be both a requestor of services and a provider of a service to other services:

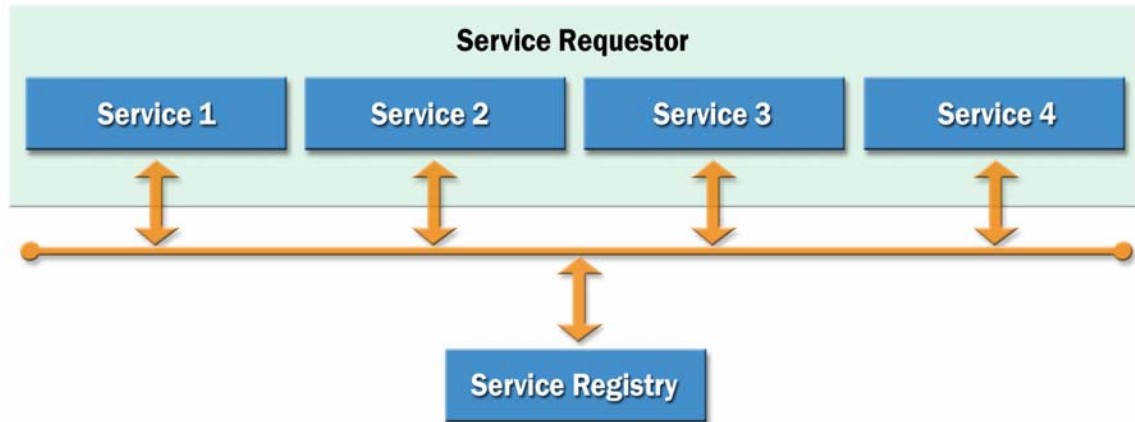
1. Requestor. A requestor application requests data from a service provider in order to satisfy a request for data or to accomplish a task. A service request uses Web services to locate the needed service (UDDI), communicate with the service provider (SOAP), and describe the information or action that is needed (WSDL).
2. Provider. A provider application provides responses to a requestor using Web services as described above.

A simple diagram of a Web service may look like this.



Another key component to an SOA is the UDDI function that allows services to be registered so that other services can locate them on a network. This is called a "registry" and the registry function is essentially a listing of Web services. The registry may be considered a meta-data service because it provides data and information about all of the

services available within an SOA. A simple diagram showing the services/registry interaction may look like this:



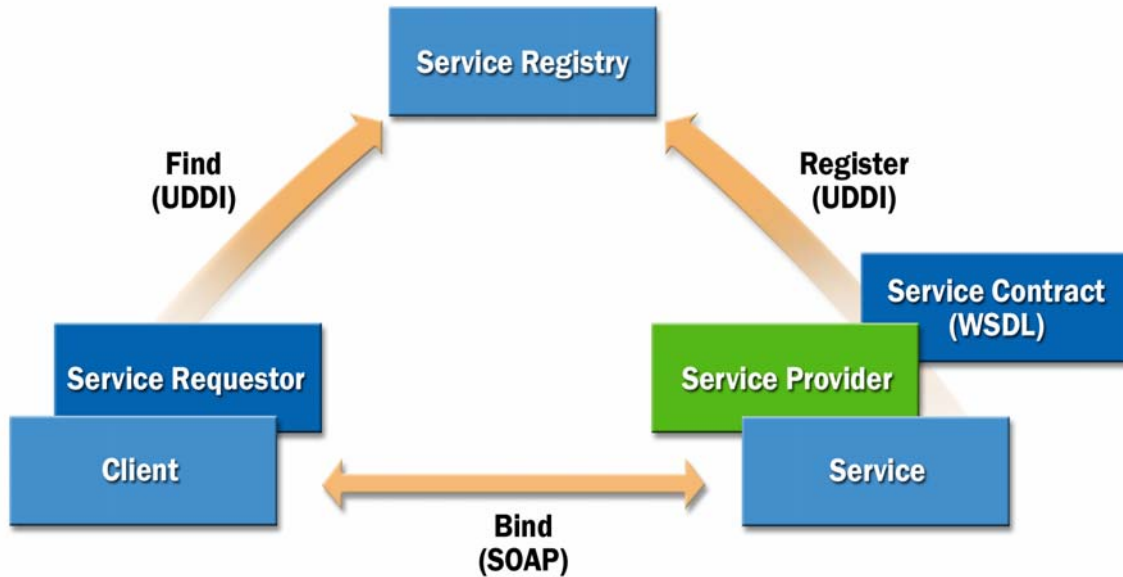
Service 1 may query the registry looking for a credit card approval application and find that Service 4 provides that application. Once the service is identified, Service 1 and Service 4 interact independently of the Registry.

There are also several other key concepts to enable a basic SOA.

*Service Contract.* The service contract is an agreement between the Requestor and the Provider as what services are being provided and how they are provided. The service contract must be open and be able to respond to requests from any Web service such as a .Net or J2EE. Once a contact is made, the Service Contract specifies such things as what services is being provided, the security requirements, the format for the data being delivered, and what happens when an error occurs.

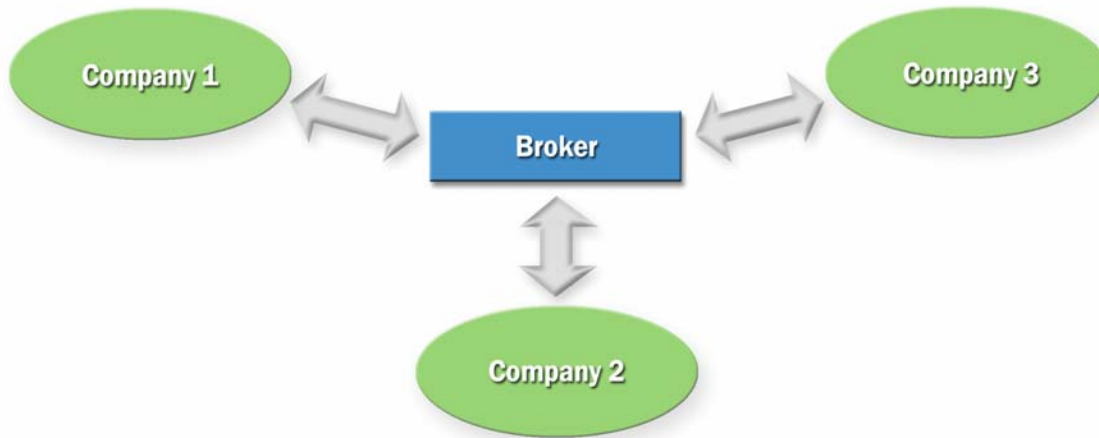
*Service Binding.* As part of the service contract, the requestor and provider are bound together, or agree to exchange data, for the duration of the contract. Since Web services are autonomous, the bindings of services are formed or dropped as needed – an example of loosely coupled services.

The illustration below shows a complete SOA with its basic component parts.

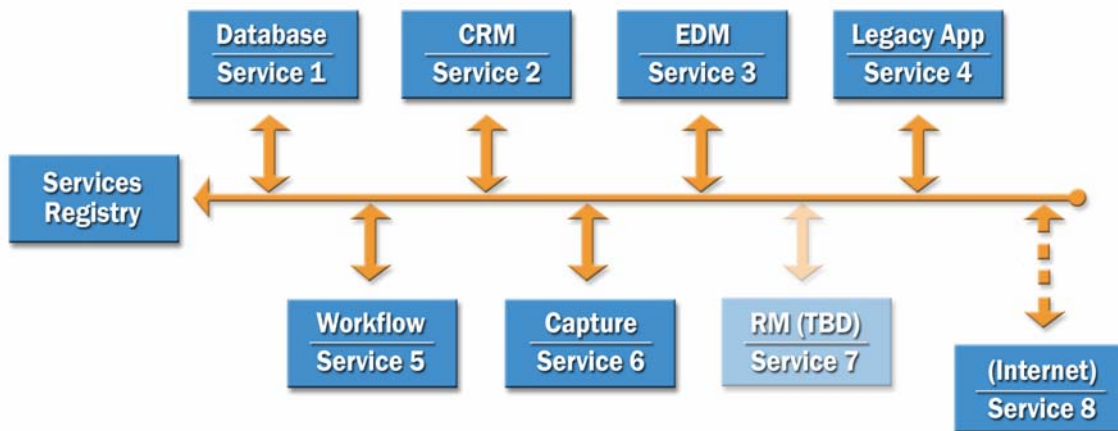


As an example application using the above components, let's say you ordered a book from Amazon and about 7 days later you received a message saying your book had not shipped and is being delayed. The message is a result of a "timeout condition" being generated when your book didn't ship on time. The "Client" service needs to send a message to the customer and looks to the Registry to find a messaging service that is customer facing. It finds the "Service," forms a contract with it, and the Service sends you a message indicating your book has not shipped and you can take the following actions: do nothing and wait for the book to ship or cancel the order. The same two "services" may be invoked to send you a message 2 days later to announce that your book has shipped.

An SOA can be extended beyond the corporate network such that services are requesting or receiving data from external services located anywhere on the Internet but found through the broker service, i.e., to be found, Web services must be registered with a broker.



Putting it all together, we may have an example corporate SOA architecture like the following:



In the above architecture, each application is built as a service and each service is registered with the service registry. A records management application (Service 7) may be added to the architecture without disrupting the other services or having to reprogram them to pass data between, for example, the EDM service and the RM service. Each service may request data from another service or provide data to satisfy a request. For example, the EDM Service 3 may request a name lookup from the Legacy Service 4 to complete the indexing fields for a new document being added to the EDM system.

### SOA and Web Services Example Application

You are a customer service representative (CSR) at a bank branch office and are taking a paper loan application from a customer. When the application is complete and signed, you walk it over to a multi-function device (MFD) and scan the application. The MFD is part the SOA network and the following Web-services (highlighted in yellow) are invoked:

#### Submit Loan Application Service

The scanned application is sent to a new loan application service, which communicates with the following services:

**OCR Service** (Networked PC running Windows XP)

1. The scanned image is OCR'd, which picks up the following:
  - a. Forms recognition (Line of Credit Application)
  - b. Dollar amount
  - c. The applicant info – Name, address, SSN, etc
2. Data is sent to the create new account service
3. Document image is sent to the DMS system for storage

**Commit Image Record Service** (DMS Application running Windows Server)

1. The scanned image is filed into the proper location using OCR'd indexing data
2. The scanned image is classified by the records management service using metadata

**Create New Account Service** (Legacy Application running UNIX)

1. The OCR'd data is used to create a new account in the legacy system
2. Account type and amount determines how loan is qualified and next steps
3. An event clock is started to ensure that the application is serviced according to the SLA (service level agreement) (“We approve loans within 48 hours....”)
4. Original application document (scanned image) is linked to new customer file
5. Account data is sent to a loan qualification service

**Qualify Loan Applicant Service** (3<sup>rd</sup> party services being invoked over Internet)

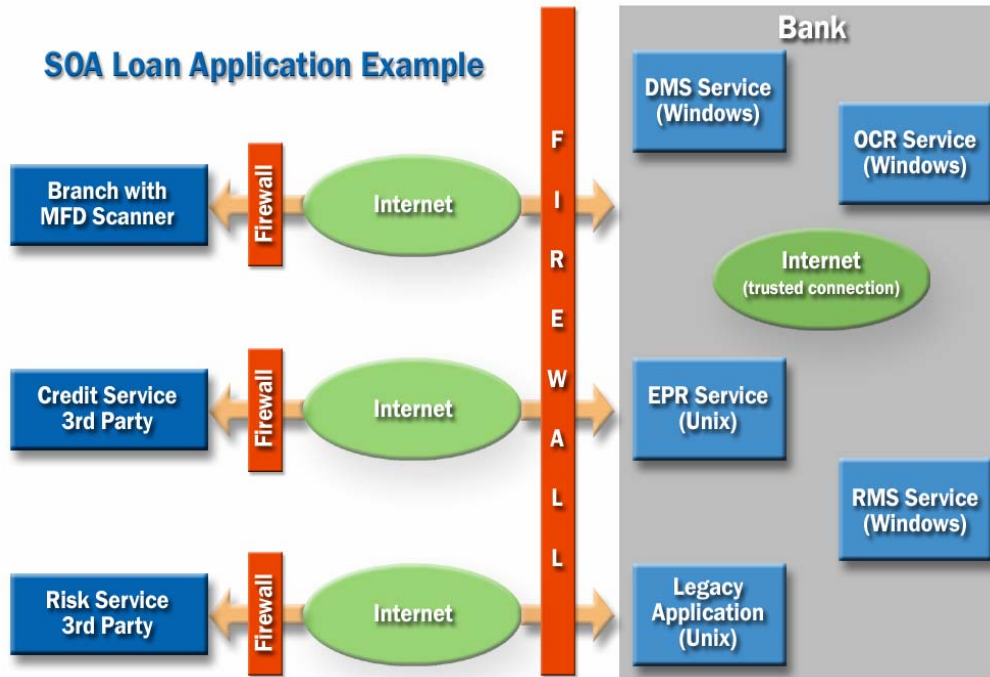
1. Account data is sent to an outside **risk management service** (dollar amount >\$20,000)
2. Account data is sent to an outside **credit services** company for review and processing
  - a. Credit score <500 points, loan denied
  - b. Credit score >500 points, loan approved
3. Send risk management report and credit score to loan approval service

**Loan Approved/Denied Application Service** (ERP Application running UNIX)

1. The internal risk management and credit report are returned and added to the account
2. The approving loan officer is notified via e-mail that the account is ready for review
3. The loan officer approves the account
4. An e-mail notice is sent to branch customer service clerk (who calls customer)
5. An approval letter is generated and sent to the customer
6. The approval letter is automatically filed with the original loan application

All of this happens by Web services being invoked to perform their unique action – requesting data, starting processes, and receiving data. OCR and forms recognition is handled by an OCR Web-service, the type of loan application and dollar amount requires the application to be sent to both the risk management and credit services instead of just a credit review, the approval/disapproval of the loan kicks off the e-mail to the branch clerk

so he or she can personally notify the customer and automatically generates the appropriate form letter to the customer. Imaged paper and electronic documents are indexed and stored automatically with correct RM classifications. Each of the services operates on a separate platform, running different packaged applications, and running different Operating Systems.



## Business Processes and SOA

So far, we have seen that the primary benefits of an SOA are that incompatible corporate systems, applications, and other assets can be pulled together through Web services into a single umbrella architecture that enables them to communicate with each other. This allows a corporation to reuse existing applications, quickly include new applications and systems, and provide a degree of flexibility that is not present in a standard client-server or monolithic architecture.

Web Services – Business Process Execution Language (WS-BPEL) enables IT and users to describe business process activities as Web services and define how they can be connected to accomplish specific tasks. WS-BPEL, within an SOA framework, allows a company to create truly distributed work processes that work on-demand, are interoperable, and are essentially without network boundaries

WS-BPEL allows a company to model and create a work process that includes such things as upon receipt of a PO, checks inventories, orders products, generates invoices, and accepts payments. This is very similar to existing “workflow” processes except that workflow was limited by the interoperability of the existing systems and applications – that is, if Application A didn’t communicate with Application B, a workflow process between the two applications would not work – workflow needs to have process that communicate with each other. Current workflows are sometimes called “human-based

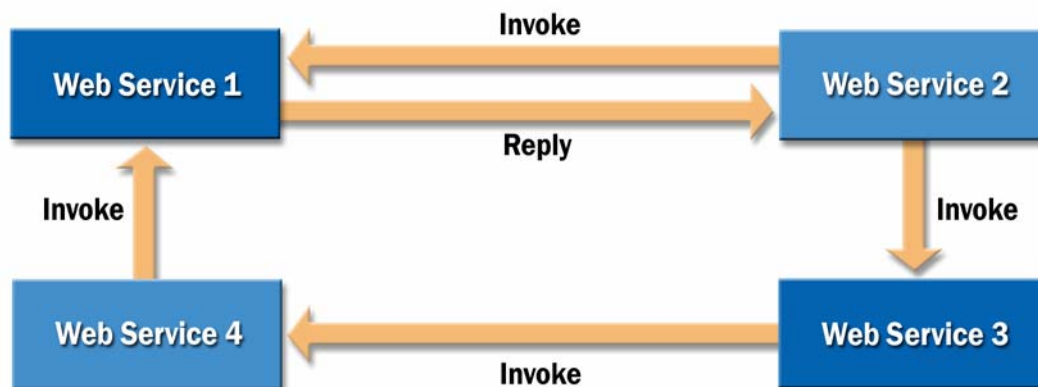
workflows” because a person is always involved as a process participant and in many cases, a decision maker. WS-BPEL workflow can be totally automated based on the initial input and there can be virtually no human participation.

WS-BPEL allows Web services to be combined to produce workflows that are internal or external to the corporation. The first is called “Orchestration.” Orchestration allows a central Web service to orchestrate a workflow process within a corporation such as processing a shipping receiver, adding the items to an inventory, and causing an invoice to be generated. Orchestration looks like:



### Orchestration

The second type of workflow process that can be made with Web services and WS-BPEL is called “Choreography” and is different in that there is not a central Web Service that orchestrates other services but more of a loose grouping of services that operate independently of each other but in a coordinated manner. Choreography looks like:



### Choreography

Choreography, like orchestration, can be a workflow within a corporation, but is more typically used to extend Web services outside of the corporate network. For example, a credit application is received by customer service but the credit report is a Web service request to Dun and Bradstreet, who supplies the report, which is sent to a risk assessment Web service and then on to the person processing the credit request.

With Web services and an SOA, interoperability between departmental business applications and legacy systems is no longer an issue. Workflow applications can now be built that actually define the work to be completed, implement the work process, and WS-BPEL is the specification that allows this to succeed.

An example of a WS-BPEL application is a purchasing application in which the goods purchased arrives at a remote location and is checked into a warehouse.

1. The “Receiver” paper document is scanned.
2. Scanning kicks off an OCR service that reads the PO Number, quantity received and this data is uploaded to the purchasing database. The OCR service may not be located locally but invoked at a central server location.
3. The purchasing database creates a match for the receiver.
4. Once the match is confirmed the ERP system updates inventory but notes that only 5 of 10 items have been received. Inventory is updated to reflect the addition of the item but the shortage is noted and a message is sent to the AP service and to customer service. The message created and sent to AP and Customer Service is the result of an “error” condition in which the expected results (receive 10 items) were not received.
5. AP invokes a messaging service to send a message to the supplier noting the shortage and then generates partial payment.
6. A customer service process kicks off an inquiry to the supplier to determine when the remaining items can be shipped. Based on the supplier response, the service may query other suppliers to determine who has the remaining items available.
7. This information is then provided to a CSR who can make the determination to do nothing or to cancel the remaining items on the order and reorder from a second supplier.

Workflow and business process management (BPM) are firmly in place in today’s corporations. However, existing workflows and BPM structures have the same problem that earlier SOA-like architectures had, which is that they were hard-wired to the applications that enabled the work process to be completed. SOA and WS-BPEL frees the process from hard-wired dependencies and the inherent fragility of the connections.

By allowing the processes to be self-describing and loosely-coupled, new processes (workflow tasks) can be added quickly as needs demand. In addition, WS-BPEL processes can be completely automated computer-to-computer transactions without involving human interaction for approval or problem resolution.

Perhaps the single most important feature of WS-BPEL is its capability to combine several Web services and processes into a new composite service that presents, for example, a composite view of data to a requester from multiple data sources or corporate assets. Essentially, WS-BPEL allows the development of a new “application” or Web service by utilizing existing corporate applications, databases, and combining that information content into a composite application that can be rapidly built, tested, and delivered to the business user.

## **Interoperability**

The primary purpose of Web services and SOAs is to allow and promote interoperability between disparate systems. Knowing that these systems are composed of different operating systems, architecture platforms, middleware, and programming languages, IBM and Microsoft, together with other industry leaders, formed the Web Services Interoperability Organization (WS-I), which is "committed to promoting interoperability among Web services based on common, industry-accepted definitions and related XML standards support. WS-I brings the work of multiple standards development organizations together for the purpose of providing clarity and conformance around Web Services...." Currently, membership in WS-I is well over 100 companies.

On a business level, interoperability allows applications to talk to each other within the corporation but to also allow the corporate network to be extended beyond the firewall to other designated Web services. For a document management system (DMS), for example, users would have the ability to commit and index a document to the DMS without actually being logged into the system. Based on metadata available within the document, the DMS may also, for example, kick-off a workflow pattern to complete a process, store the record in the correct storage device, and archive the record to the records management system.

Interoperability is not limited, in a sense, to software applications but would allow companies to better utilize existing hardware-based assets such as scanners, multifunction devices, and hardware storage devices such as RAID, NAS, SAN, optical storage, and tape storage. Other dedicated software applications, like optical character recognition, handwriting character recognition, and mark sense recognition could become Web services available to all users on the network who have the ability to request a document be converted from a scanned image (or fax) to machine readable type. In the not too distant future, companies may no longer purchase dedicated applications, like OCR, but use them as Web services over an extended SOA being charge for usage, not ownership.

## **Security and Web services**

As Web services potentially opens the network by making applications more accessible from both within and outside for the company, security is of vital concern and importance. In order to address the security issues, the WS-Security specification was created to support, integrate, and unify current security models and to incorporate additional WS-\* security specifications.

Basic security protection mechanisms are built around encryption, authentication, and authorization. In addition, because WS-\* and SOA are built around XML, which is human readable text messages, XML based security specification and standards are also incorporated into the security mix. A significant amount of work has been done in designing security standards in a modular, complementary, and evolutionary manner. As the domain of security standards is becoming mature, Web services platform vendors have started adopting these technologies.

See the Standards and Security Specifications below for list of the current security specifications available.

## Standards

Since SOA is not a product, it owes its existence to an agreed upon set of specifications and standards. These standards allow the interoperability between systems such as .NET, J2EE, various mainframe-based architectures, programming languages, and even allow the inclusion of CORBA and DCOM-based networks into the SOA.

Standards are in some senses diametrically opposed to vendors and their products in that vendors want to have the only product with a certain feature or capability and vendors are not open to sharing their unique capabilities. But with the realization that corporations do not buy only from a single vendor and in fact actively look to

### Open Standards

Open standards are publicly available specifications for achieving a specific task. By allowing anyone to use the standard, they increase compatibility between various hardware and software components since anyone with the technical know-how and the necessary equipment to implement solutions can build something that works together with those of other vendors.

protect their IT investments by not being dependent on a single company or technology, most product vendors now fully embrace the need for open standards that allows basic interoperability between products. For many companies today, the marketing message is, “We can connect to and be compatible with any software architecture and any legacy application.”

Open standards help to consolidate competing standards, increasing the aggregate pool of resources available for using them while avoiding proprietary “standards” and single vendor de facto standards. By subscribing to and using open standards, vendor independence is increased as is ability to integrate best of breed components. By defining standards for component interfaces, interoperability between systems is increased, which results in simpler reusable components and faster, less expensive integration.

This fresh new attitude has been, in part, responsible for the growth and acceptance of new Web services standards that allow communications between diverse applications and the resulting interoperability between vendors and their products.

Specifications used in the development of an SOA are governed by the following standards organizations and independent groups:

**WS-I.** WS-I is an open industry organization chartered to promote Web services interoperability across platforms, operating systems and programming languages. The organization’s diverse community of Web services leaders helps customers to develop interoperable Web services by providing guidance, recommended practices, and supporting resources. WS-I creates, promotes and supports generic protocols for the interoperable exchange of messages between Web services. In this context, “generic protocols” are protocols that are independent of any action indicated by a message, other than those actions necessary for its secure, reliable and efficient delivery, and “interoperable” means suitable for multiple operating systems and multiple programming languages.

**W3C.** The World Wide Web Consortium (W3C) develops interoperable technologies (specifications, guidelines, software, and tools) to lead the Web to its full potential. W3C is a forum for information, commerce, communication, and collective understanding. W3C is responsible for SOAP, WSDL, WS-Choreography, WS-Addressing, WS-Policy, XML Encryption, and XML Signatures. W3C was also responsible for furthering web standards such as HTTP, HTML, and XML.

**OASIS.** Organization for the Advancement of Structured Information Standards (OASIS) is an international consortium that drives the development, convergence, and adoption of e-business standards. The consortium produces Web services standards along with standards for security, e-business, and standardization efforts in the public sector and for application-specific markets. OASIS is responsible for UDDI, WS-Security, WS-BPEL, WS-Composite Application Framework, WS-Notification, WS-Reliability, and others.

**IETF.** The Internet Engineering Task Force (IETF) is a large open international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet. IETF is responsible for defining and maintaining specifications such as TCP/IP.

**OMG.** The Object Management Group (OMG) is an open membership consortium that produces and maintains computer industry specifications for interoperable enterprise applications. OMG is home to specifications that define WSDL language mappings to C++ and CORBA to WSDL mappings.

Web services specifications originate in the various standards bodies but may also originate in private companies like Microsoft and IBM. Private companies developing specifications may be supported by other companies such as BEA, Intel, SAP, and Tibco or they may create a consortium of companies with a common goal. Initial work on a specification may start with a private company but to gain widespread acceptance, the specification is submitted to one of the standards groups. Submission to a standards group ensures that the specification is free of any proprietary controls and becomes part of the open standards collection. For example, Adobe .pdf files were originally “proprietary” in that Adobe controlled the format but today, .pdf is in the public domain and has become an open standard.

As a consolidated representative list, but by no means a definitive list, the following represent the basic specifications available to SOA designers. These specifications are collectively known as WS-\*: This list also shows the depth of work surrounding Web services as some of the specifications below are still being promulgated in private committees and some are specification in progress by the various standards bodies.

## **Messaging Specifications**

SOAP

ASAP (Asynchronous Web services)

WS-Addressing (now superseded by MTOM)

MTOM (Message Transmission Optimization Mechanism)

WS-Enumeration

WS-Eventing

WS-Transfer  
SOAP over UDP (User Datagram Protocol)  
WS-Choreography

## **Security Specifications**

WS-Security : SOAP Message Security  
WS-Security: Username Token Profile  
WS-Security: X.509 Certificate Token Profile  
WS-Security Kerberos Binding  
WS-Security Minimalist Profile  
WS-SecureConversation  
WS-Security Policy  
WS-Encryption  
WS-Signature  
WS-Trust  
WS-Federation  
WS-Authorization  
OASIS SAML (Security Assertion Markup Language)

## **Reliable Messaging Specifications**

WS-Reliability  
WS-ReliableMessaging  
WS-Acknowledgment  
WS-Coordination  
WS-Atomic Transaction  
WS-BusinessActivity  
WS-TransactionManagement

## **Web services Interfaces**

WSDL  
WS-Remote Portlet

## **Web services Discovery**

UDDI  
ebXML Registry  
WS-Discovery  
WS-Inspection Language (WSIL)  
WS-Notification

## **Metadata Specifications**

WS-Policy  
WS-PolicyAssertions

WS-PolicyAttachment  
WS-MetadataExchange  
WS-MessageData

## **XML Specifications**

XML  
Namespaces in XML  
XML information Set

## **Management Specifications**

WS-Management  
WS-Management Catalog  
WS-Manageability  
WS-Distributed Management (WSDM)  
WS-Provisioning

## **Business Process Specifications**

WS- Business Process Execution Language (WS-BPEL)

The above lists also lead us to conclude that there are many different paths that can be taken when designing and building an SOA. Many of the WS-\* specifications above are works in progress and may change over the course of time or may be dropped as new and more innovative specifications are developed.

As is often said, “The good thing about standards is that there is so many of them...” should be a reason for caution. As with any standard, for example SQL, it is possible to modify it such that it no longer interacts in a standard way and is not interoperable with other SQL databases. The many “modified” flavors of SQL are an example of this.

While this may be a danger signal to many, it is also a sign of the robust and flexible nature of SOAs and the ability to create them with a variety of Web services. Previous attempts to enforce standards had a fatal flaw in that they were contained within the industry and did not really apply to everyday people or commercial products.

Ask any person today if they could live without the Internet: without ordering books from Amazon, without looking up who the cast of Casablanca was, or not being able to send e-mail, and the answer would be, “Give me more Internet services.” Web services are being propelled by everyone who uses the Internet – not just by Company A trying to electronically invoice Company B. Web services are being driven, in part, by consumers and the resulting advances are being incorporated by industry.

## **What is the Value Proposition for SOA based System?**

The development of an SOA provides a company with many potential benefits. Below are some of the leading examples:

**Leverage Existing Assets.** This is one of the most important concerns for many corporate IT managers today and one of the most expensive considerations. Should we scrap the current legacy system and buy a new one, try to patch it with some add-on applications, or live with it until the decision is made for us? SOAs and Web services allow companies to reinvigorate legacy systems and extend their life indefinitely or until a more reasonable choice can be made.

**Composite Applications.** With an SOA and all business applications Web services-enabled, it is possible to create completely new business applications using existing assets and data. The composite application is not limited to applications within the existing network but may extend beyond the corporate network to incorporate one to many applications into a new business process.

**Faster Implementation of Projects.** Over time, a company will build libraries of Web services and be able to reuse them, which should significantly reduce the time to design, build, test, and implement new project.

**Risk Mitigation.** Fewer update issues/coding problems/rebuilds/data corruptions – SOAs preclude the current need for lockstep change as when one application changes you must change all others at the same time. In addition, by reusing existing software applications/packages, the risk of introducing new errors into the system is reduced since the application itself is not disturbed.

**Distributed Deployment.** As systems are tied together, there is less dependency on single applications as services are spread across existing applications and applications outside of the network.

**Lower Overall Maintenance Costs.** Instead of purchasing new software and adding to the overall maintenance budget, SOAs may significantly lower the overall projected cost of maintenance over a long period of time.

**A Flexible and Resilient System Architecture.** A well organized and built SOA should provide a company with a platform that allows rapid changes to meet market demands while reducing system downtime due to coding changes and potential system problems.

**Better Business Processes.** SOAs allow developers to concentrate on the business application instead of trying to fit the business process to the software application. Thus, business users and IT can select best of breed applications to meet their needs instead of trying to customize an application that doesn't quite fit.

**Outsourcing Discrete Applications.** As businesses begin to use more Web services and lessen their need on legacy systems, it will be possible to leverage applications outside of the corporate network – in essence outsourcing the application to commercial vendors. This is already being done but as more companies become consumers of Web services, the number of service applications available will increase.

**Tactical versus Strategic Thinking.** Without a doubt budget is and will continue to be a consideration when it comes to long-term strategic planning. Is it better to solve the immediate problems on a department by department basis or begin to implement the strategic vision for a new architecture? Many strategic visions get waylaid by tactical needs and often the budget is exhausted by buying applications that are temporary but necessary – similar to the metaphor of treating the symptoms not the disease. Because

Web services and SOAs can be built with existing assets, and the application build can be incrementally accomplished, corporations can begin implementing their strategic vision as time and budget allow.

**Software Application Choice.** With an SOA in place, a company is no longer locked into any particular software application and may be freer to choose applications based on price, functionality, and features, which produces a best of breed product set. Corporate standards for products were originally mandated to keep products compatibility and maintenance issues within the same vendor. That reasoning is no longer valid with an SOA.

### **Are SOA Systems Ready for Primetime?**

While there is much attention being paid to SOAs within the business and technical world, SOAs are not without some problems and issues. But it should be understood that the problems and issues are with the many specifications and products that together form an SOA. Thus, in some sense, the concept for an SOA will not go away, but the current specifications, standards, and products may change underneath it. The question to be answered will be if the change will be evolutionary, allowing currently operating SOAs to evolve with the changes or will the change be revolutionary producing an incompatible second generation?

It is not likely that there will be a revolutionary change that will obsolete an entire generation of Web service standards and products. Since the basic premise of an SOA is interoperability, it is more likely that SOAs will continue to evolve with Web Services and related standards. In addition, many companies, such as IBM, Oracle, Microsoft, and others, are firmly behind the current standards and implementing SOAs at customer sites. There is little incentive, for vendors, to radically change the playing field again and ask customers to spend additional time and resources.

Be that as it may, there are several issues worth reviewing and being aware of when considering whether to build an SOA.

- ✓ Security is number one on the list of concerns. Since SOAs are open networks they can exist within a corporation and traverse corporate boundaries, being able to secure the network is of prime importance.
- ✓ Performance. Since we are building a network in which one service can respond to many services, how do you ensure performance is maintained and how do you dynamically add or reduce resources when needed?
- ✓ Complexity. SOAs will become inherently more and more complex with time as new services are added. Being able to fully monitor an increasingly complex system will become difficult with existing tools. Tracing an error condition could be very time-consuming when each service is maintained by different IT groups or companies.
- ✓ XML messages. The original XML messages that drove SOAs were relatively small and transparent in terms of performance. Today, XML messages are becoming increasingly larger as the tasks become more complex. It is possible that the basic XML message will begin to hinder the performance of the network if the size continues to grow. (Think of the early days of document imaging when document

images quickly swamped a network and performance became a system adoption issue.)

- ✓ Legacy systems will have to have Web services tacked on to them as they may not inherently support Web services – especially purpose built stand-alone applications. Many purpose built legacy systems may have their life extended by the addition of a Web service front-end, but there is little chance that an older legacy system will be rewritten to provide native Web services.
- ✓ Legacy code and its documentation could be hard to tap into for systems in which the code is undocumented and may be difficult to work with.
- ✓ Standards/specifications. To the uninitiated there are an overwhelming number of standards/specifications making groups and too many standards/specifications to understand and keep current with. SOAs may require a higher initial learning curve and may represent more work to a company that is just beginning to work with Web services.
- ✓ Standards are moving rapidly and are hard to pin down. Many companies may take a wait and see attitude hoping that single standards and directions will emerge.

## Conclusion

With new Web services being developed every day and a host of new Web-enabled products being built, there is an increasingly important need for flexible connections and application interoperability. Perhaps the most important aspect of SOAs is that it finally allows a company to build a best of breed architecture using software applications that *best fit the business needs*, not the needs or limits of existing software applications. Instead of having the architecture driven by programming interfaces, programming languages, application incompatibilities, and long-established vendor relationships, SOAs actually allow the business needs to drive the architecture, how it is built, and what applications are used to build it. Interoperability by design becomes a key to building SOAs that reflect the needs of the business.

In many senses, SOAs are contributing to the decomposition of legacy systems and monolithic software applications. As companies build more and more one-off service applications using Web services, create libraries of truly reusable services, and lessen their dependence on single software systems, the overall corporate architecture becomes comprised of many small services and applications all working together in an on-demand environment.

Web services have produced a fundamental change in both the software application industry and the corporate IT department because it actually works as promised and works on a broad scale cutting across software platforms, packaged applications, and inherently closed business processes. Web services have gained critical mass throughout industry and are continuing to gain momentum because they are successful and it can be shown in a variety of industries and a variety of companies that range in size.

Finally, because an SOA is not a product controlled by a single corporation and is the result of many different corporations working together to form standards to support SOAs, it has gained the recognition and acceptance needed for continued growth. And,

because there is not a single “correct” way to build an SOA, SOAs can be built by both small and large companies using different products but achieving the same result – which is the ability to quickly build business processes that make a company competitive among its peers.

## **Westbrook Technologies Fortis SOA**

Interoperability with business applications and legacy applications has been the traditional weakness for electronic document management systems. While many systems are sold as “enterprise” capable systems, most systems operate within a single department, such as HR, and data sharing or information sharing is limited to uploading data to the ERP system. Even when an electronic document management system is implemented in different departments, those departments remain isolated applications that function within the department but have little ability to share its data and resources.

Today’s business environment is all about interoperability, connectivity, and sharing work processes across work groups, departments, and enterprises. Web services provides corporate business users and IT departments a tool set that encourages an entrepreneurial spirit allowing new business models and products to be built and tested quickly.

Westbrook Technologies Fortis SOA is the first electronic document management system to be built from the ground up using Web services and is interoperable by design. Fortis SOA is built to provide its functionality as services that are available to any other SOA-enabled application on the network.

These functional services include capture, output, electronic document management, workflow, records management, enterprise report management, and others. Fortis SOA is platform neutral becoming an active and integral part of the business enterprise making data, workflow, and business process management functionality easily accessible to business and legacy applications. Fortis SOA is the next evolutionary step for electronic document management systems.

## **Porter-Roth Associates**

Porter-Roth Associates is an electronic document management (EDM) consulting company. Our consulting services include initial EDM strategy sessions, project analysis and planning, EDM requirements analysis, RFP development, and project oversight. We provide objective and unbiased consulting and are not affiliated with any vendors, resellers, or systems integrators of EDM products.