

The SOA Magazine

Feature Article



SOA in the Enterprise: A Survey of the Technical Landscape

by Cyrille Thilloy

Published: August 28, 2006 (SOA Magazine Issue I: September/October 2006, Copyright © 2006)

[Download this article as a PDF document.](#)

Abstract: This article is the first in a series exploring the various technologies and options available to an enterprise when establishing a service-oriented architecture (SOA) roadmap. It introduces the notion of an SOA infrastructure and enriches it with the models and frameworks currently available to SOA implementers. By describing architectural components and frameworks common to SOA, it defines the baseline for the service-oriented enterprise. Subsequent articles will further explore a possible methodology for SOA governance based on the formal description of the purpose of each service. This methodology provides diagrams and template examples of inputs and outputs on the decision paths needed for SOA adoption and roadmaps [REF-1].

Introduction

In order to keep or gain a competitive advantage, organizations need to deliver richer business solutions, faster and at lower costs. Historically, enterprises have implemented silos of solutions embedding informal expertise that served specific business domains with very little reuse and almost no integration capabilities. The limitations of those architectures are now more fully understood. Service-oriented architecture (SOA) is gaining more and more momentum in the end-user community by promising to overcome those limitations in order to achieve some of the more elusive strategic benefits, such as reduced automation cost, higher organizational agility, and increased overall flexibility for the enterprise.

Although current SOA implementations strongly leverage Web services technology, SOA is at first a concept and an orientation. One of the greatest challenges to achieving a successful SOA implementation is ensuring that the high-level business objectives (continuity of business, security exposure, business requirements, liabilities, etc.) are addressed in the implementation of each individually delivered service. Attaining these strategic goals requires, on a fundamental level, that the concept of "service" be first defined *outside* of technology and specific implementation concerns. This is an issue that will be addressed in the next article of this series. For now, we will establish a common background for the service-oriented computing landscape.

Hypothesis: SOA is based on Web Services technologies

SOA represents an evolution in distributed processing approaches that distinguishes itself by emphasizing a strict separation of responsibilities and concerns using standard contracts (external interfaces). While the underlying principles are not tied to any specific technology, SOA is commonly implemented with Web services because this technology framework inherently support some of the major principles (like the separation of concerns and the definition of standard contracts) of service-orientation as defined in [REF-2].

It is often recommended to start an enterprise SOA initiative with a small project, but it must be understood that SOA itself is not a project but rather a journey. The enterprise must understand that the impact of SOA is huge and goes far beyond traditional IT boundaries and simple point-to-point integration. When planning this journey, especially from an enterprise perspective, it is helpful to have an idea of your final destination (this long-term vision will also make your

CIO and CFO feel better). The SOA infrastructure presented in this article may appear to be too comprehensive. Remember that the SOA infrastructure is the ultimate destination that your enterprise will eventually reach.

History of Enterprise System Integration

Prior to mapping out our ultimate destination, a bit of history is necessary to better appreciate the current state of systems integration within the enterprise. Figure 1 depicts how historically systems integration in the enterprise (or between enterprises) has been handled through proprietary point-to-point integration architectures. This integration approach was the result of a lack of standard technologies and protocols, and the project-oriented nature of many past integration initiatives.

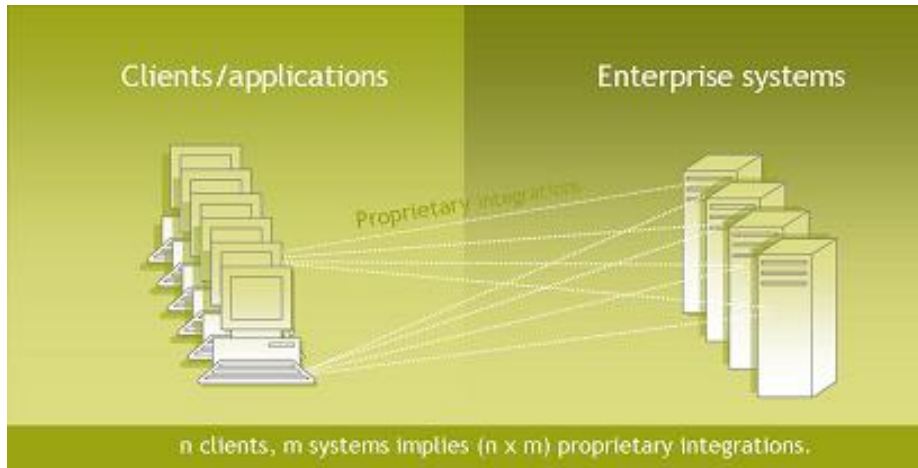


Figure 1: Traditional enterprise integration architecture.

Point-to-point integration architectures have major drawbacks. First, the cost of integration (including testing and maintenance) is a factor directly proportional to the number of systems in use. Secondly, integration channels are almost always specific and cannot be generalized for reuse. When this type of strategy is adopted, the enterprise can quickly become overwhelmed by the resulting maintenance costs. This resulted in the opportunity for a new niche integration market termed "Enterprise Application Integration" (EAI). EAI solutions helped reduce the burden of integration requirements by limiting the growth of point-to-point interfaces through a reduced set of normalized control points.

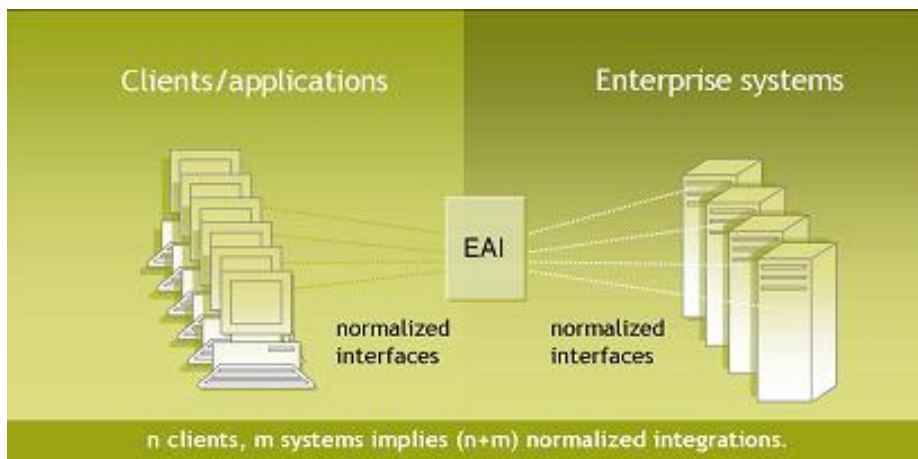


Figure 2: Typical enterprise application integration (EAI) architecture.

As shown in Figure 2, the main selling point of EAI is the reduction in the quantity of required interfaces. EAI interfaces are qualified as being normalized since (even if most are XML-based) they are still specific to the vendor platform. While effectively reducing the number of integration points EAI solutions did not really liberate the enterprise from proprietary integration architectures.

In an increasingly diverse market, organizations are no longer willing to rely on just a single EAI vendor. For example, in the banking industry, more and more ASP providers offer marketplaces for non-core banking services like mortgage or loan origination. With that model, a particular bank cannot force its ASP partner to endorse its internal EAI platform.

This situation simply goes against the business rational of the ASP model and would force the ASP to support each EAI platform of each client.

Competition forces the enterprise to adopt new business practices (like outsourcing, M&A, etc.), while regulation forces the enterprise to inject new controls into its current business practices. In both cases, the enterprise answer is openness. This includes the adoption of standardized interfaces simply to transfer the burden of integration out of traditional enterprise boundaries.

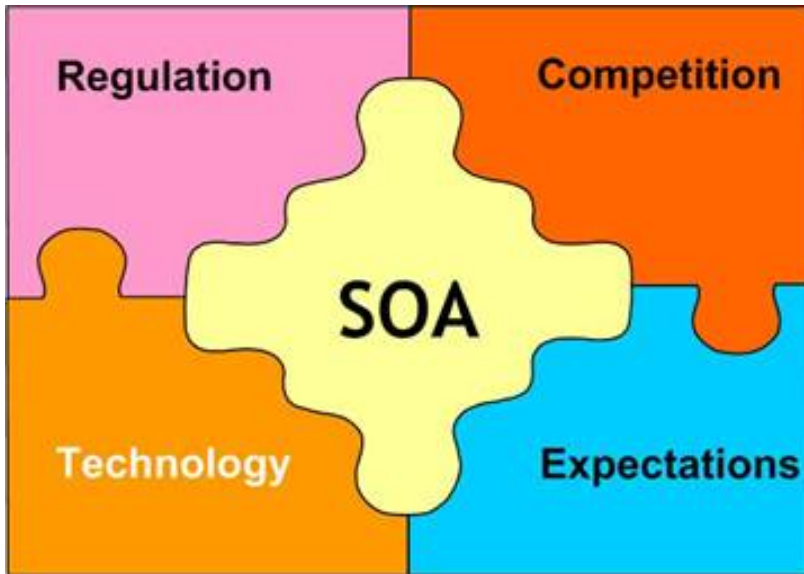


Figure 3: Key enterprise drivers leading toward SOA adoption.

With openness in mind, SOA is seen as the right answer for the enterprise. In pursuit of an open, service-oriented enterprise, most organizations are adopting a pragmatic approach to Web services enablement. This enablement has led to the development of a new market in the integration landscape: the XML gateways.

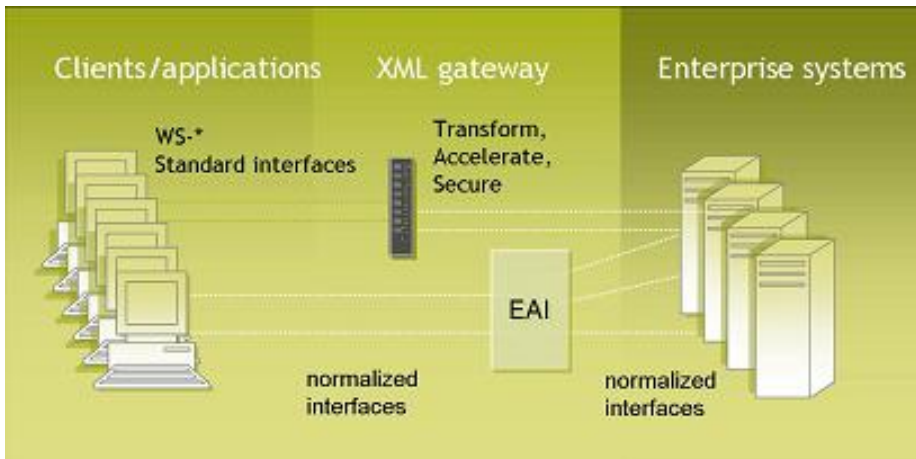


Figure 4: Pragmatic Web services enablement approach for the enterprise.

XML gateways are used to hide through mediation the specifics of enterprise system interfaces and expose them as standard Web services. Usually gateways can do more than just mediation. Especially when they are hardware-based, they can accelerate XML processing performance and handle security-related tasks. Deploying gateways to mediate legacy environments is often considered part of a first stage in Web service enablement and a significant step towards achieving a level of SOA implementation maturity.

Figure 4 provides more of a conceptual view by showing how XML gateways related to other parts of the enterprise. Several alternatives are currently available for Web services enablement, including:

- Waiting for the back-end (enterprise) vendors to directly support Web services.
- Waiting for the EAI vendors to directly support Web services.

- Adopting XML gateways (hardware or software) to accelerate integration.

Currently, very few enterprises have gone beyond XML gateway deployment to achieve a meaningful level of Web services enablement for their legacy systems.

Figure 4 only hints at what the next generation of integration architecture will look like. While all vendors (the service producers) are embracing the SOA bandwagon, new technologies and capabilities are also becoming available for the development of service consumer programs.

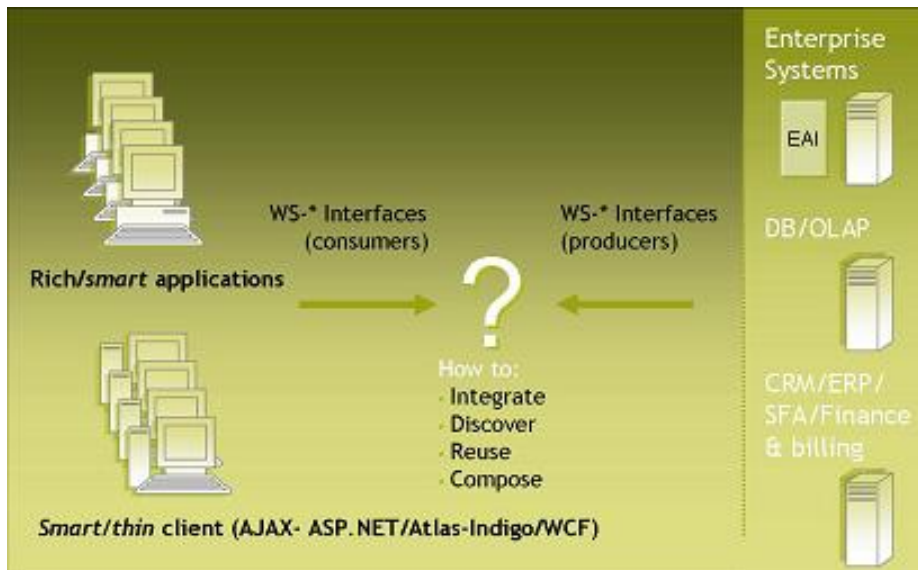


Figure 5: The next wave of integration architectures.

The challenge for many organizations will be how to handle this new trend. For example, an enterprise should be able to link services produced either from off-the-shelf applications (CRM, ERP, etc.) and from internal solutions with a variety of service consumers.

New user-interface paradigms, like Ajax, ASP.Net/Altas, Indigo (now named Windows Communication Foundation) or WSRP (see below) are creating a new collection of service consumption options, both inside and outside the enterprise.

This goes far beyond point-to-point or project-based integration. A supporting infrastructure will be needed, which, in fact, is the demarcation point between Web services enablement and SOA enablement. Without the proper infrastructure, a mesh (and a mess) of ungovernable services interactions will result.

It is therefore important to understand that SOA is not a trend. In the enterprise, SOA is a necessary path for achieving systems integration. If the enterprise wants to leverage its informational assets, it will need the proper governance and also an infrastructure to support them.

Understanding SOA Infrastructure

In order to promote service reusability, efficiency and thus cost reduction, a typical enterprise SOA will encompass a complete infrastructure composed of various software and hardware components. Understanding the requirements behind assembling such an infrastructure is crucial to successfully map an organization's business needs to the utilization of individual technology components.

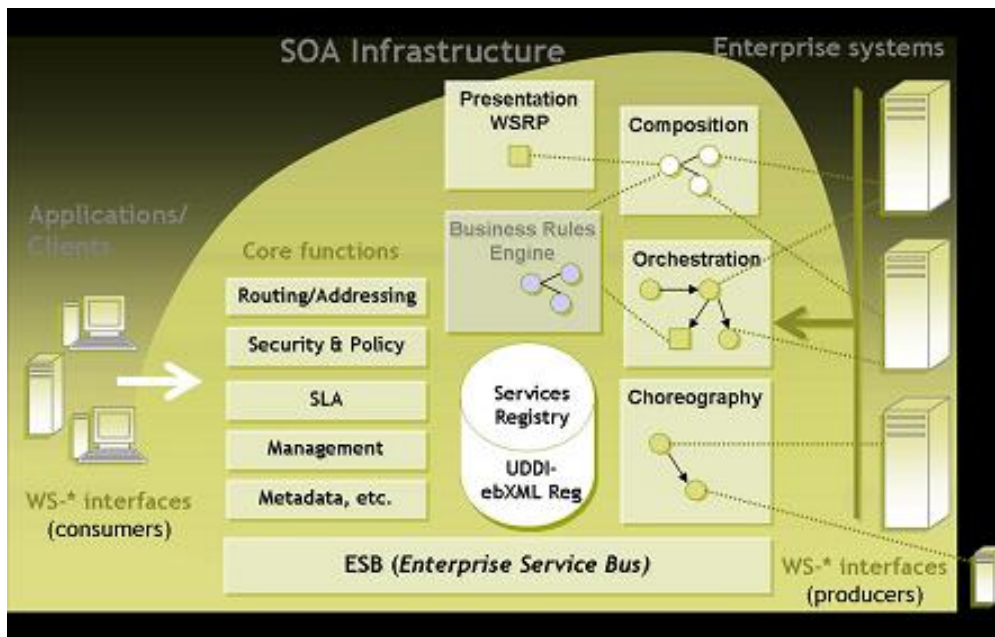


Figure 6: The emerging enterprise SOA infrastructure.

At first glance, an SOA infrastructure may appear complex. It is noteworthy to mention that:

- All the components presented are not always needed, especially in the initial adoption stages.
- Components are presented in a logical view. A single solution or server can therefore encompass several parts of the infrastructure.
- Several of the described components can be custom-made or purchased as off-the-shelf products.

Every organization's technical environment is unique, and depending on how a service-oriented architecture is being designed and positioned as part of a long-term evolutionary cycle, not all components may actually be required as part of the final infrastructure. However, it is important to appreciate the potential scope and magnitude of a true SOA enterprise infrastructure in order to ensure that all required planning considerations are taken into account.

Service composition technologies

Service composition is the act of aggregating several Web services into one. For the Web service consumer the internal services are hidden. Composition is based on message mediation and could be eased (and automated) by the semantic description of the composing services. (Semantic service descriptions will be covered in a future series article.)

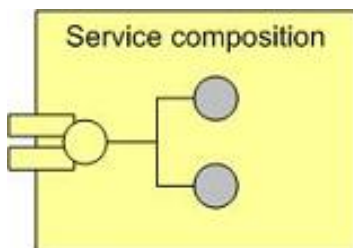


Figure 7: Basic service composition.

The *Web Services Composition Framework (WS-CF)* is an OASIS work-in-progress specification that defines logical components (participants, groups of participants and registrars) and associated primitives for the composition of Web services. WS-CF relies on the use of the WS-Context specification.

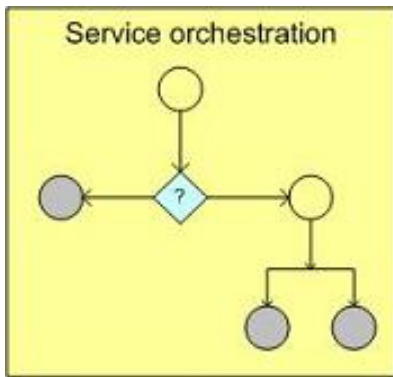


Figure 8: Service orchestration.

Orchestration (also considered a form of service composition) is the act of aggregating services (and often human interactions) as part of formally defined workflow logic. Orchestration essentially describes a process flow between activities (and sometimes services) controlled by a single entity. It supports sessions and transactions (short or long running) and is often implemented as a combination of Business Process Management (BPM) using the OASIS Web Services Business Process Execution Language (WS-BPEL).

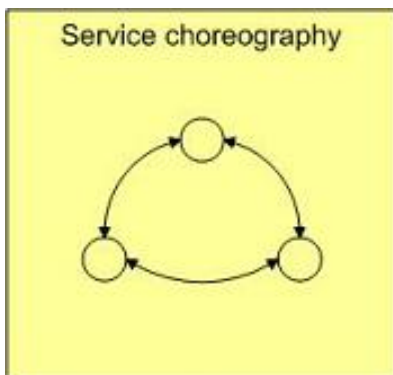


Figure 9: Service choreography.

Service choreography can be seen as an extension of orchestration that increases the scope of service composition to multi-party participants. With choreography, the focus is not on the process flow but also cross-organization collaboration through message exchanges. Processing control is delegated among all participants and thus choreography only describes the external behavior of the service activity.

The *Web Services Choreography Description Language (WS-CDL)* is a W3C work-in-progress that provides a language for describing peer-to-peer collaboration through the formal definition of common and complementary observable behaviors and ordered message exchanges. WS-CDL is not an execution language like WS-BPEL and mostly focuses on the external behavior representation. It provides constructs that describe participants, roles, and relationships, and also provides a means of conveying semantic descriptions.

Other supporting technologies

Before we continue, let's briefly describe some of the remaining architectural components from Figure 6:

- The *Enterprise Service Bus (ESB)* is a message brokering facility that uses an event-driven approach to processing. Initially there was some controversy around the combined use of service-oriented and event-driven architectures. However, it is now generally accepted that the two processing models (event versus procedural) are complementary rather than competitive. Some SOA deployments solely rely on the use of an ESB, while others just utilize an ESB to offload non-core processing tasks (for example by pushing audit and log information onto the bus for delayed processing).
- *Service registries* are repositories for SOA artifacts. Some registry specifications, like UDDI, focus more on the service description and can be compared to a service inventory phone book. Other specifications, such as ebXML, support added artifacts and are more comparable to the yellow pages (in that they can provide more meta data for each registry record). Service registries are generally considered a core part of any enterprise-wide SOA infrastructure.

- *Business Rule Engines (BRE)* occupy a specialized area where no common standards have yet been adopted. This is mostly due to the nature of the logical processing embedded in a typical BRE, which can be based on either process or event-driven models.

Note that the other components qualified as "core functions" in Figure 6 are not explained in this section since they are more related to common functions needed for service enablement. Instead, the following section provides descriptions of Web services technologies that can be used to enable those functions as part of an enterprise SOA infrastructure.

Understanding WS-* Standards and Frameworks

The second-generation Web services standards and technologies (generally referred to as WS-*) provide the foundation for the future Web services-based service-oriented enterprise. All major vendors are currently adopting portions of this second-generation platform, albeit at different levels of maturity.

Figure 10 provides a simple map that highlights some of the more prominent WS-* technologies we will be describing.



Figure 10: WS-* standards and technologies.

While not complete, Figure 10 provides us with a sense of the technological mapping that will be required to establish an SOA roadmap for the typical enterprise. Maps such as this can also be used as a documentation tool to help validate which WS-* standard or technology should be used for specific purposes and to illustrate the possible impact of semantic usage (as represented by the dashed-line).

Adopting WS-* specifications and standards generally results in extra effort and cost (training, performance especially for XML-based protocols, etc.) and therefore, it is recommended that organizations assess each such technology before committing to it. Not all service-oriented solutions need to adhere to WS-* technologies right away. It can sometimes be most beneficial to observe the vendor industry and then decide to formally adopt a WS-* specification once sufficiently mature implementations are provided by multiple vendor platforms.

Contract-based Frameworks

Let's now turn our attention to the various SOA-related frameworks currently under development. When planning an

SOA transition, organizations will have the option to build their own technology frameworks (hopefully adhering to some of the standards introduced previously) or to use or based their framework on some of the ones currently being developed.

Among the various initiatives underway, the use of standard interfaces is a key service-orientation principle. It should therefore not be a surprise to see emerging frameworks centered around strong agreements (service contracts) as the basis for both electronic business and the representation of business domains.

When studied closely, one can see that the contracts used in the frameworks embed and convey semantic information, such as roles and responsibilities, logical statements, data structures, functional dependencies, and so on. Although they have much more complete semantic formalisms than what we are able to define for service orchestration or choreography, their semantic data is generally targeted at expressing very specific capabilities.

Below are brief descriptions of some of the on-going frameworks:

ebSOA (OASIS work-in-progress)

The Electronic Business Service-Oriented Architecture technical committee is working to define a specification for eBusiness within the context of SOA. While not yet finalized, the proposed specification is currently based on the "Federated Enterprise Reference Architecture" [REF-3] and is composed of three main elements:

- An Information Model (ebSOA IM) containing all the objects needed in a federated collaborative environment. The inter-relationships between objects are also described; implying that the semantic data is embedded into the model.
- Collaboration Semantics (ebSOA CS) using a "Collaboration Process Information Document" that defines the collaboration between participants of the federation.
- A runtime environment defining the various software components needed within a federation.

While it can also support WS-CDL and WSDL, it seems that ebSOA has inherently a more intuitive support for the Collaboration Protocol Profile and Agreement (CPPA) and ebXML Business Process Specification Schema (BPSS) specifications. ebSOA is well suited for B2B exchanges between federated enterprises, but both the IM and the CS provide data and functions that could be used outside the common runtime.

ebXML BP (also referred as ebBP/BPSS; OASIS work in progress)

The OASIS ebXML Business Process specification provides a technology representation and model compatible with an underlying generic metamodel for business processes, activities, and collaboration. This representation and model attempt to establish a set of guidelines that define business process-rules, semantics and syntax for both binary and multi-party collaboration. They are designed to work within the ebXML architecture and support standards-based development and exchange of business process definitions.

Although WS-BPEL may appear to have a similar focus, ebXML BP is quite different since it enforces the notion of contracts between parties by making extensive use of the Collaboration Protocol Profile and Agreement (CPPA). ebXML BP formalizes collaboration into processes and exchanges using business documents and "signals." As you might expect, ebXML BP is primarily targeted at supporting B2B collaboration and can architecturally be extended through the use of ebSOA.

SEE (OASIS work-in-progress)

The goal of the Semantic Execution Environment specification is to provide guidelines, justifications, and implementation directions for an execution environment for semantic Web services. In that sense SEE sits in between a framework and a semantic Web language. Based on previous works from the Digital Enterprise Research Institute (<http://www.deri.org>), SEE is composed of three related specifications:

- The Web Service Modeling Ontology (WSMO), an OWL-based ontology that describes aspects related to semantic Web services.
- The Web Service Modeling Language (WSML), a language that offers a formal syntax and semantics for WSMO.

- The Web Service Execution Environment (WSMX), which represents an execution environment for the dynamic discovery, selection, mediation, invocation and inter-operation of semantic Web services.

More Resources

Below is a list of additional OASIS works-in-progress that may also be of interest to you:

- *Business-Centric Methodology (BCM)* - This committee is defining a roadmap for the development and implementation of procedures that produce effective, efficient, and sustainable interoperability mechanisms. BCM establishes four fundamental layers: conceptual, business, extension, and implementation. Transition between layers is described in templates and accomplished through the use of "Choice Points." BCM leverages existing ebXML schemas like CPPA and promotes the OASIS "Content Assembly Mechanism" (CAM) for dynamic integration. The CAM template defines the structural formatting and the business rules for the transaction content as well as providing hooks to semantic representations. Interestingly, BCM re-employs existing artifacts developed primary for electronic business purposes.
- *Framework for Web Services Implementation (FWSI)* - This specification facilitates the implementation of Web services by defining practical and extensible common functional elements that practitioners can adopt to create high quality Web services solutions without re-inventing them for each implementation. This committee also defines guidelines and an accompanying implementation methodology.
- *SOA Reference Model (SOA-RM)* This committee is chartered to develop a reference model for service-Oriented architecture. This model is being developed to encourage the continued growth of different and specialized SOA implementations while preserving a common layer of understanding about what SOA is.
- *SOA Adoption Blueprints* - An initiative aimed at developing, publishing and maintaining archetypal blueprint sets of requirements and functions to serve as generic, vendor-neutral instances of service-oriented solutions for real business requirements. The blueprints are a set of business and functional requirements can be fulfilled by widely acceptable SOA technologies and are intended to serve as functional descriptions and working examples that form the basis for a collection of best practices. An adoption blueprint provides a business problem statement, a set of business requirements, and a normative set of functions to be fulfilled, all on a vendor-neutral basis.

Each of these efforts is in a different state of completion and the extent to which any one of the resulting specifications will actually be adopted by the SOA and Web services communities remains to be seen. However, it is always worth having an awareness of what standards organizations are doing in the SOA space.

ROI of a SOA Initiative

So far, we've described enterprise SOA from a high-level IT perspective. Even if the depicted infrastructure seems complex and crowded, it is important to understand what a successful SOA adoption can mean for an organization. Specifically, an organization needs to have a solid understanding of why it is adopting SOA. Establishing the Why will help the define the ROI.

The Why can encompass:

- The continuity of business (limiting the impact on legacy systems).
- The reduction the operational costs (reducing the burden of integration, operation, and maintenance through open industry standards).
- Increased responsiveness and agility (bringing new solutions to the market in a very timely fashion).
- Competitive influences and customer expectations (efficiently enhancing or extending existing solutions).
- The conformity and regulation management (establishing a federated, standardized enterprise based on open technologies).

Justifying the ROI of an SOA initiative is always a concern. Financial returns can be difficult to assess due to the fact

that many benefits are long-term and intangible (such as preventing the loss of business opportunities, increasing customer expectations, etc.).

Nevertheless other metrics can be quantified, such as the cost of deploying SOA intermediaries versus rebuilding legacy system interfaces. If the enterprise outsources its IT integration and support, it is quite easy to ask the outsourcer for a bid and measure the resulting cost reduction.

Either way, it is considered a good practice to build a business case around a planned SOA initiative. This document will almost always need to be customized to accommodate the unique considerations and requirements of an SOA transition project.

Conclusion

This article aimed to provide a broad perspective of systems integration in the enterprise context. It has only addressed the technical aspects of Web services and SOA components and has not even touched upon the subject of SOA governance. This is the first in a series articles for The SOA Magazine in which I will explore a methodology for road-mapping SOA into the enterprise. The outputs of the methodology define the service perspective within the overall scope of enterprise architecture.

References

[REF-1] Enterprise Service Oriented Methodology by Multiforce Technologies, Inc. (Version 5.00, June 2006) available at <http://www.multiforce.com>

[REF-2] "Service-Oriented Architecture: Concepts, Technology, and Design", Thomas Erl; Prentice Hall, 2005; ISBN-0-13-185858-0, available at <http://www.soabooks.com>

[REF-3] Closing the Process/Technology Gap, FERA: Federated Enterprise Architecture, available at http://www.cpd-associates.com/index.cfm?content=subpage&file=include_RPPage.cfm&ID=72404138&DOC=177813046

THE PRENTICE HALL SERVICE-ORIENTED COMPUTING SERIES FROM THOMAS ERL

