

SETLabs Briefings

VOL 5 NO 1
Jan - Mar 2007

BUSINESS INNOVATION through TECHNOLOGY

SERVICE ORIENTED ARCHITECTURE



Infosys®

POWERED BY INTELLECT
DRIVEN BY VALUES

Managing the Hype

SETLabs Briefings Advisory Board

Aveejeet Palit

*Principal Solutions Manager,
System Integration Practice*

Deependra Moitra

*Associate Vice President,
Software Engineering &
Technology Labs*

Gaurav Rastogi

*Associate Vice President,
Global Sales Effectiveness*

George Eby Mathew

*Principal Researcher,
Software Engineering &
Technology Labs*

Kochikar V P

*Associate Vice President,
Education & Research Unit*

Raj Joshi

*Managing Director,
Infosys Consulting Inc.*

Ranganath M

*Vice President & Head,
Domain Competency Group*

Srinivas Uppaluri

*Associate Vice President & Head,
Global Marketing*

Subu Goparaju

*Vice President & Head,
Software Engineering &
Technology Labs*



A recent Forrester study stated that Service-oriented architecture (SOA) adoption continues to be strong, especially for large enterprises. Nearly 70 per cent of SOA users say they will increase their use of SOA, and 46 per cent of large enterprise users of SOA use it for strategic business transformation. SOA is 'in' but some companies are taking risky shortcuts in implementing SOA, forgetting that SOA requires robust governance, software development methodologies and staffing.

First, those that say they have a robust SOA governance essentially mean that they have defined the processes and not that stakeholders have bought in and are willing to enforce

them. Second, given that SOA can be applied in varied contexts ranging from plain data and information integration to service oriented application integration to more strategic initiatives like enterprise architecture transformation including infrastructural virtualization, those that are grappling with methodologies are stuck without standards or a deeper understanding of the scope of deployment of SOA for specific contexts. Third, architects and developers who have expertise in contract formation, in terms of richness of service specific descriptions are scarce.

Over the last 15 years Enterprise IT has seen fads come in and go. So critics are fair in asking whether SOA is the flavour of the season typified by initial excitement and hyped demand under girded by a disregard for complexity and maturity. Many SOA questions still remain unanswered. For example, who pays for SOA. Will SOA be able to bridge, if not obliterate the gap between business and IT?

The temptation to treat SOA as integration plus opportunity withstanding, we believe that SOA requires a strategic outlook to maximize re-use and flexibility, with models for sharing costs and benefits across the organization. A pragmatic approach to gaining business process flexibility should be the key driver for SOA adoption. By far the best SOA strategies use a portfolio approach, lightweight SOA visions, continuous improvement and are tactically grounded.

Keeping this in view, this issue of SETLabs Briefings leverages Infosys experience in implementing SOA to look at various dimensions of SOA for enterprise transformation. At one level we look at the value dimensions of SOA, simplistic SOA interventions such as creating shared standards-based data integration, legacy modernization, and the role of enterprise service bus in the integration of services for enterprises. On the other, we look at how enterprises leverage value out of SOA by asking the question whether SOA be able to bridge the business-IT divide? We also look at a consumer-driven replenishment system that represents a huge multi-billion dollar opportunity in the CPG industry. The view point on how SOA should embrace grid and virtualization technologies as part of the enterprise fabric is a good read.

We are grateful to our authors, editors and contributors for making this issue an excellent primer on SOA realities. My gratitude to Ronald Schmelzer, founder of ZapThink, for highlighting that SOA is an architecture meant to handle change. As always, we love hearing from you.

A handwritten signature in black ink, appearing to read 'GEM'.

George Eby Mathew
george_mathew@infosys.com
Editor



Tutorial: Leveraging shared data services in data integration

By Krishnendu Kunti, Mohit Chawla and Vikram Sitaram

Shared data services can go a long way in providing a seamless approach for implementing standards based data integration projects.

3

Research: Value Dimensions of Service-Oriented Architecture

By Jai Ganesh, PhD

Business processes, reuse and flexibility are the key dimensions to measure the value created by SOA for enterprises.

9

Viewpoint: ESB: More than Promises

By Bijoy Majumdar, Terance Dias and Ujval Mysore

Notwithstanding the confusion surrounding the concept of Enterprise Service Bus, it is emerging as a vital component for enabling SOA in enterprises.

13

Forward Thinking: Service Oriented Infrastructure

By Shubhashis Sengupta, PhD., Hariprasad Nellitheertha and Srikanth Sundarajan

True benefits of SOA can only be achieved with effective deployment of virtualization and grid technologies at the infrastructure tier in the enterprise fabric.

21

Viewpoint: SOA and BPM: Complementary concepts for Agility

By Parameswaran Seshan

Effective utilization of the complementarities existing in BPM and SOA can help derive substantial business value.

29

Framework: SOA Technology Competency Center

By Shreyas Kamat

The SOA TCC concept is becoming prevalent in the industry rapidly as organizations are learning the need to manage SOA transformation effectively.

37

Perspective: Evolutionary Approach to realizing SOA: A Microsoft Platform example

By Ananthalakshmi Vallapuzha and Manish Srivastava

The Microsoft suite of products provide a platform for realizing SOA in an evolutionary and modular fashion, which in turn, help realize incremental benefits.

43

Case Study: SOA: Saving Grace for Legacy Applications

By Anubhuti Bharill, Biji Nair and Binooj Purayath

This implementation explores the business value of Legacy Modernization, through Service Oriented Integration.

53

Third Angle: "SOA is an architecture for change"

Ronald Schmelzer, founder of ZapThink asserts that the key value of SOA lies in its ability to address change without re-coding and reconfiguration effort.

61

Research: Improve Integration Efficiency with Semantic SOA

By Bhavin Raichura and Shaurabh Bharti

Enhancing SOA with semantic discovery capabilities can enhance integration efficiency, dynamic discovery and adaptive capabilities.

67

Practitioner's Perspective: Rejuvenate, Collaborate and Innovate: CPG industry context

By Vaidyanatha Siva and Vishal Puri

SOA and Event Driven Architecture can catalyze the consumer-driven replenishment system representing a huge multi-billion dollar opportunity.

75

Viewpoint: SOA - Impact on Enterprise Architecture Service View

By Peter Jarman

The concept of a service existed long before SOA came to prominence. How does SOA impact the service concept?

81

Implementation: Enterprise Application Framework for SOA Realization

By Shyam Kumar Doddavula and Sandeep Karamongikar

Enterprise Application Frameworks can catalyze SOA implementations in translating concepts to execution.

87

Index

98

“SOA allows changes in business processes, policies, rules, and composition of different IT assets without significant redevelopment or recoding effort.”

Ron Schmelzer

Founder

ZapThink, LLC.

“An organization would have a slim chance of success in SOA adoption without an effective TCC, analogous to a basketball or football team trying to win the championship without a competent coach.”

Shreyas Kamat

Principal Architect

Infosys System Integration

Technology Consulting Group

Leveraging shared data services in data integration

By Krishnendu Kunti, Mohit Chawla and Vikram Sitaram

A seamless approach for implementing standards-based data integration projects

Web services technology has come a long way from its inception to maturation of core web service standards i.e., WSDL, SOAP and UDDI. Today, most enterprises have come beyond proof of concept stage to full fledged web services project. Also, in today's enterprise a large chunk of the web services projects are aimed at tactical assignments like platform independent integration and data access. Projects aimed at data integration using web services are often implemented using custom data access (both retrieval and update) code with a web services wrapper on it. This is resource and time intensive and also this approach does not allow enterprise wide data definition and restricts reuse of data access interfaces towards creation of other data access interfaces. We discuss the merits of using shared data services platform for implementation of web services data integration projects in this article.

SOA AND WEB SERVICES

Traditional enterprise information systems have developed over time to cater to individual needs

for a particular line of business. As business grew over time and business processes were outsourced to multiple trading partners, the need to integrate systems arose. However, due to the inherent tight coupling between existing systems it was not possible to integrate business processes easily and creating custom integration bridges are not only time and effort intensive but also require creation of a new bridge every time a new application needs to be integrated. Also there was considerable replication of functionality across different lines of business and trading partners. The solution of the above mentioned issues lies in the principles of Service Oriented Architecture (SOA). Service Oriented Architecture stands on the twin pillars of reusability and interoperability. Any function which has a potential of re-usage is a candidate for SOA. A reusable interface which is interoperable can be consumed across multiple platforms. SOA is based on standards based definition of re-usable and interoperable interfaces. Web services technology is an implementation of SOA principles. It comprises of a set of core and

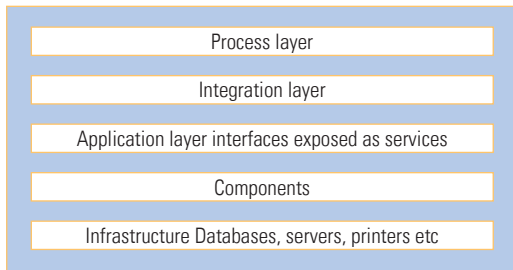


Figure 1: SOA at multiple levels of enterprise IT

Source: Infosys Research

supporting standards. The core standards allow definition of standard based interfaces that can be consumed across multiple platforms.

SOA can be realized across different levels of enterprise IT infrastructure. Infact, the real value proposition of SOA is realized by ensuring re-usage across multiple levels of enterprise IT.

As seen from Figure 1, SOA can be realized at the following layers:

Infrastructure layer: Shared resources such as databases, servers etc., can be exposed as services. Once a shared resource is exposed as service the underlying implementation is abstracted from the consuming application. Also, the platform exposing multiple shared resources as services takes care of issues like connection, distributed transaction etc.

Components layer: At this level business components are defined that can be consumed across multiple application interfaces.

Application layer: Application interfaces are exposed as services. These services might cater to business level functionality or commonly used function such as interest calculation. Services at application layer can be implemented either in platform dependent language or using standards based interfaces like web services technology.

Integration layer: At this layer one or more

application interfaces are combined to create business services. Integration layer also adds value added functions like translation, routing security etc.

Process layer: Business processes are created in this layer using business services from the underlying layer.

In the sections below we discuss how SOA can be used at database layer and application layer towards implementing a web services data access project.

WEB SERVICES FOR DATA ACCESS

A large chunk of web services projects cater to data access. In these projects, back end data is exposed as web services interfaces. Once the data access interfaces are exposed as web services, they can be consumed by any platform. However, in most cases the data access code is manually written and as a result every single data access component along with support functions like database connection code, connection pooling, distributed transaction management, caching, role based etc., needs to be created for every new interface.

Most of the data access code in such a project is not created keeping in mind the business domain model in question. As a result every individual interface development occurs in isolation of the other, without a domain model in place. This approach of development does not facilitate re-usage of an existing data access function while creation of subsequent data access functionality.

Also, these kind of projects lack tools to aid business users towards definition of common business entity and its relation to other business entities. In order to access the retrieved data from remote location using standards-based interfaces, web services wrapper layer is created. Ideally the web services wrapper code should not be manually written.

Often a number of web services in a project might be related i.e., fields retrieved in a web service might be superset of fields retrieved in another web service. In a JDBC-centric coding without any concern for domain business model separate code might be written for even related web services. However, if due consideration is given to definition of organization-wide common entity at service level, one or more web services can be created from an underlying business component model.

IMPLICATION OF SOA IN DATA ACCESS LAYER

Application of SOA principles at data access layer aims at creation of standards based reusable data access interfaces. The advantages that accrue from applying the principles of SOA to the data access layer include reusability of data access interfaces, platform-independent data retrieval and update, loosely coupled interfaces and standards-based interfaces.

The use of shared services for accessing data also leads to another significant advantage in terms of the usefulness and the relevance of data obtained from the data source. In a situation where the consuming application has to directly access data from the data source, the onus of refining and polishing the data obtained to make it relevant to the consuming application is on the client application or a broker that sits in between the consumer and the data source. Using an SDS platform allows us to define fine-grained business services that are built on top of the data access services. This provides a more standards-based and flexible approach to obtaining context-relevant information than the use of a custom application to refine data pulled out from a data source or having to write logic in the consuming application to refine and polish the data obtained.

This section below illustrates what a shared data services platform might typically require in order to perform data access and update.

Data query interfaces can be implemented using standards based syntax like XQuery [1]. These interfaces can by themselves act as reusable components towards creation of other interfaces. XML generated from an XQuery interface can be mapped to a commonly agreed business component model. Data updating interfaces can be created based on service object mapping to business component mapping using tools like Morph [2] and finally updating multiple data sources using a JTA compliant transaction manager and tools like Hibernate [3]. The primary motive is to create an infrastructure that will allow definition of reusable interfaces for data retrieval and updation. Such a platform responsible for executing the XQuery on multiple data sources takes care of distributed query generation, distributed transaction management, result aggregation and updating multiple data sources and is termed as shared data services platform.

The platform abstracts out data access and management functions and provides infrastructure which can be used across multiple projects. Apart from the core data access functions, the platform also helps in metadata management, role-based access, caching, distributed transaction management and above all re-usage of data access interfaces.

SHARED DATA SERVICES PLATFORM

Shared Data Services (SDS) is a concept evolved by applying Service Oriented Architecture (SOA) to the data access layer. An SDS platform built on the principles of SOA should allow data to read/update functionality in a manner that is agnostic to the underlying platform, protocols or technology involved in extracting

data from a data source. In other words, the SDS platform should be able to provide data access to heterogeneous data sources which may be present across the enterprise. The basic concepts of SOA including service reusability, flexibility and open standards-based development must be incorporated in the building of an SDS platform.

Fig 2 shows an architectural view of the different layers present in an SDS framework.

Data Source Layer: This layer consists of all the different data sources that are present within an enterprise. The data source layer presents a single unified view to the layers above but is composed of various different data sources dispersed across the enterprise.

Infrastructure Layer: This layer consists of components that provide certain non-functional core functionalities. Transaction management, Caching, XML Schema mapping, metadata management, role-based access etc., are provided at this level.

Shared Business Services Layer: This is the topmost layer in the SDS platform hierarchy. This layer exposes data access functionalities based on open standards. Client applications invoke the services provided at this layer to access data in the enterprise.

In order to make the SDS framework extensible, flexible and SOA-compliant, Web Services are a natural fit for building services. The data services exposed at the Shared Business Services layer are built as Web Services that can be accessed by client applications. This allows clients from varied platforms to read/update data sources that are managed by the SDS platform. Having Web Services as the primary means of data access also enables the SDS platform to be easily plugged into an existing EAI or ESB framework that is deployed in an organization. With industry further consolidating and agreeing

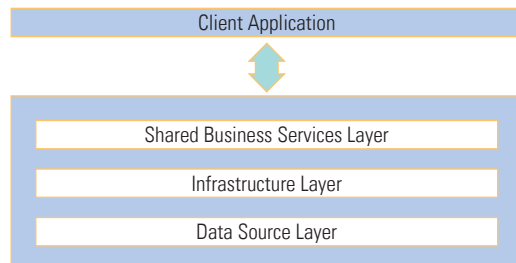


Figure 2: SDS Platform

Source: Infosys Research

on WS-Standards, interoperability issues can be tackled smoothly as different data sources are increasingly brought under the preview of the SDS framework.

DESIGNING SDS LAYER FOR DATA ACCESS

As illustrated in the previous sections, SDS layer abstracts the issues such as distributed transaction management, caching, role based access etc., from the end user who needs to perform database read/write on single or multiple databases. The ensuing section will describe the detailed design view of the platform, data flow, object mapping and accessing data through single access point.

Fig 3 explains how data is accessed through different layers in the SDS platform. As you can see at the database layer there can be multiple data sources, both relational and flat files. At business component level we have object level representation of the database tables or nodes (in case of XML documents). For reading/writing data from/to distributed databases many proprietary and open source tools are available like, XQuery, Hibernate etc. XQuery platform allows to query different databases through standard based XQuery syntax and returns result of the query in the XML format, which can be mapped to the business

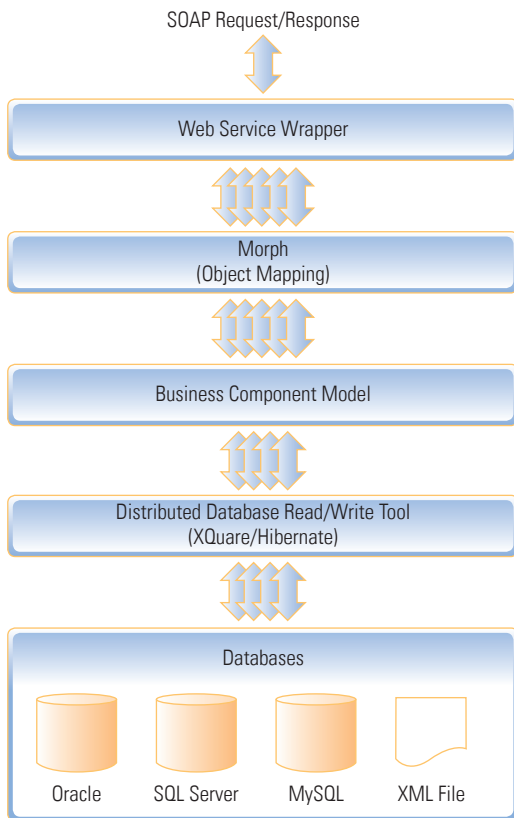


Figure 3: Data Access in SDS Platform

Source: Infosys Research

component model objects, at the layer above [4]. Using Hibernate, one can perform write operation to multiple relational databases just by setting the values in the corresponding objects at business component model level, and the issues like database connection and transaction management are handled by the platform itself. But for updating the flat files such as XML files, a bit more needs to be done as it doesn't have its own transaction manager as compared to other relational databases. For flat files a change summary needs to be maintained so that if there is a need to rollback the transaction due to any failure during the transaction process it can be

reverted back to its original state with help of change summary.

As described earlier, business component model layer contains objects mapped corresponding to the tables/data that one is using for data access. It is this layer where data caching is done for the data accessed from the different data sources. The cached data can be used in another transaction based on the requirement.

Object Mapping layer comes in between the web service wrapper and business component model layers. As the name suggests, it performs mapping of Web services objects (populated from SOAP request) to the business component model objects and vice versa. Morph is one such tool which can be used for such operations.

Web Service wrapper exposes various re-usable interfaces through which data can be accessed or updated based on the application requirements. Re-usage is not limited only to the interface level; the cached data can be reused for various transactions, which in turn boosts the performance of the application as a whole.

The most important part in a scenario where one has to perform data access across multiple data sources is the issue of transaction management. This is especially pertinent when one updates in various data sources. The SDS platform uses the concept of a User Transaction to achieve distributed transaction management. A User Transaction encompasses all individual transactions where each individual transaction refers to the transaction with every data source in which data has to be updated. A User Transaction is deemed successful only when all individual transactions complete successfully. The responsibility of managing the User Transaction and performing rollbacks/commits lies with the SDS platform.


As mentioned earlier the SDS platform will be able to perform transaction management across both relational data sources and flat files, such as XML files.

CONCLUSION

With the increasing consolidation of Web Services standards and SOA initiatives becoming mainstream across enterprises, the concept of Shared Data Services has come of age. The cost of maintaining and managing legacy and varied data stores represents a challenge that enterprises are finding increasingly difficult and expensive to deal with. SDS provides a solution that addresses the concerns of maintaining and integrating legacy and newer data sources into the organization. As firms move towards SOA-ization of their enterprise IT architecture, SDS

provides a platform to leverage and optimize the strengths of SOA in providing faster and unified data access across the enterprise.

REFERENCES

1. W3C specification of XQuery. Available at <http://www.w3.org/TR/xquery/>
2. Morph Framework. Available at <http://morph.sourceforge.net/index.html>
3. Hibernate Home Page. Available at <http://www.hibernate.org/>
4. XQuareFusion: XQuery-based information integration engine. Available at <http://xquare.objectweb.org/fusion/index.html>
5. Shared Data Services: An Architectural Approach, ICWS, Niranjana V, Dr. Sriram Anand, and Krishnendu Kunti, 2005 

Value Dimensions of Service Oriented Architecture

By *Jai Ganesh*

Value realization is driven by business processes, reuse and standards

Information Technology architectures include the technology, strategies, plans and principles that guide an organisation's new technology investments as well as manage the existing technology investments. The objective behind having a good IT architecture is to enable an organisation to operate with a high degree of flexibility while at the same time keeping the cost for the technology investments within justifiable limits. IT architectures are derived from the business architecture and once the architecture is defined, it supports the business. IT flexibility has been defined as the ability of software to adapt to the changing business [1]. Previous research has espoused three types of flexibility in the context of IT infrastructure: system functionality -- where the system or component remains stable while inputs and conditions change; use -- which refers to outcomes and opportunities; and modification -- which deals with the ease of making changes.

Information Technology (IT) resources and capabilities supporting organizations to

rapidly execute their business models and processes are now key differentiating factors in fast-paced, volatile environments. IT systems need to be agile and flexible and the ability to modify or add new business processes without the need to completely overhaul the systems is an important requirement. Agile IT systems are systems that are malleable enough to address business uncertainties. Such systems have the capability to effectively respond to internal and external stimuli within a very short period of time. Flexible IT systems imply that the IT architecture underlying them itself is flexible and lends itself to incorporation of changes in a dynamic fashion. This suggests that IT architectures have an impact on enterprise flexibility and hence on enterprise business value.

The development of component-based techniques for encapsulating legacy IT systems and mapping business process change onto them is leading to IT architectures for direct system support of business processes, and hence to more economical evolutionary change

[2]. Architectural approaches such as Service Oriented Architecture (SOA) are transforming the way IT systems are designed by bringing in a high degree of reuse and loose coupling of applications. This opens up avenues for organizations to deliver their services in new and effective ways to their internal and external partners. Value dimensions and measurable sub-dimensions of IT architectural paradigms such as SOA in enhancing business value is a key area of interest. Value dimensions are the ways in which value is instantiated. In the context of our discussion, value dimensions are the values or benefits which IT architectures present. Understanding the value dimensions of IT architectures are important as they typically involve large scale changes and such decisions need to be supported by the benefits which they generate. In this article, we aim to understand the various value dimensions and sub-dimensions of Agile IT architectures such as SOA in creating business value. Specifically, we aim to answer the following questions:

- What are the value dimensions of SOA based IT architectures?
- What are the most important value dimensions of SOA based IT architectures and how can they be measured?

NEED TO IDENTIFY VALUE DIMENSIONS OF SOA

Service Oriented Architecture envisages delivering of IT functionality as services over a distributed network. The idea behind SOA is to treat business and IT functionality being delivered as a set of services, wherein the services are self-contained, and do not depend on the context or state of other services. SOA-based systems can be seen as a collection of services having well defined interfaces.

In technical terms, SOA is an approach to loosely coupled, protocol independent, standards-based distributed computing wherein IT resources are available on a network as services.

The key idea differentiating SOA from earlier models of distributed computing lies in the notion of a service as the least common denominator. Here, a service refers to a piece of functionality, which is defined by a strict contract via a well-defined interface that is independent of any underlying implementation platform of the service. This kind of interface is loosely coupled in that a service is a standalone entity with no tight coupling to the underlying environment or to the other services. These characteristics are key to delivering the required flexibility for enterprises by adoption of SOA.

There is loose coupling between the services consumers and providers offering maximum decoupling between any two entities. In order to benefit from loose coupling, it is mandatory that the services need to be described in detail and they also need to be self-contained. Factors such as loose coupling, re-use, flexibility etc. lead to the question of identifying the value dimensions of SOA based IT architectures, which have a combination of unique characteristics, prominent among which include re-use, flexibility and open standards.

VALUE DIMENSIONS OF SOA

We identified six value dimensions of IT architectures based on previous literature survey (See Table 1 for the details of the value dimensions as well as the measurable sub-dimensions). In order to distill out the most important value sub-dimensions, we conducted empirical research using a questionnaire based approach. The questionnaire employed five point Likert scales (Very Low, Low, Medium, High, and Very High). The questionnaire was targeted at various

Value Dimensions	Measurable sub-dimensions
Organizational	<ul style="list-style-type: none"> • Time to market • Support for emerging business scenarios
Technology	<ul style="list-style-type: none"> • Reduced complexity and ease of integration • Reduced cost and time of internal integration, partner integration (customers, vendors) • Reduced cost and time of introducing new applications, modifying existing applications • Reduced cost and time of introducing new IT infrastructure, modifying existing IT infrastructure • Scalability of systems • Loose coupling • Inter-organizational collaboration
Business Process	<ul style="list-style-type: none"> • Cost and time of introducing new business processes • Modifying existing business processes
Re-use	<ul style="list-style-type: none"> • Re-use of business models • Re-use of processes • Re-use of applications • Re-use of infrastructure
Standards	<ul style="list-style-type: none"> • Platform and technology independence • Benefits from low Vendor lock-in
People	<ul style="list-style-type: none"> • IT personnel efficiency • Standardized employee skill levels

Table 1: Value Dimensions of SOA

Source: Infosys Research

stakeholders in IT architectural decisions such as architects, business analysts and developers. The stakeholder interview-based approach is appropriate when the adoption of a technology is in its early stages. Moreover, we also got rich perspectives from our interactions with CIOs and IT directors responsible for architecting their organizational IT strategy and making IT architectural decisions. We examined SOA value across dimensions such as organisational, business process, technology, standards, re-use and people.

DISCUSSION OF RESULTS

The following are the results of the research:

1. Organisational value dimension of SOA has time-to-market (which enables enterprises to come out with products/ services in a shorter timeframe) as the most important constituent. This is followed by adaptability to emerging business scenarios (which facilitates the enterprise to realize faster growth by speedier and responsiveness to market conditions).


2. Technology value dimension of SOA has interorganisational collaboration as the most important constituent. This is followed by benefits from loose coupling (as SOA enables systems to be assembled and disassembled easily as they have less dependency on other systems).
3. Business Process value dimension of SOA has modifying existing business processes as the most important constituent.
4. Re-use value dimension of SOA has business process re-use as the most important constituent. This is followed by re-use of applications and infrastructure (through the potential reuse of services and infrastructure).
5. People value dimension of SOA has development of standardized employee skill set (as the SOA interfaces are abstracted from the underlying implementation) as the most important constituent.
6. The standards value dimension of SOA has low vendor lock-in (allowing enterprises to replace customised applications and products) as the most important constituent. This is followed by Platform and technology independence (leading to greater collaboration

and integration, with less spends on proprietary middleware infrastructure).

CONCLUSION

In this research, we have made an attempt to capture the various value dimensions of SOA based IT architectures. By breaking down the overall value dimensions into Organizational, BusinessProcess, Technology, Re-use, Integration and People, we attempted to create a high level list of dimensions which can be converted into measurable variables. Our research implies that the practitioners need to have a comprehensive understanding of the value dimensions of SOA based IT architectures to arrive at the business value measure to justify SOA based IT architectures. Understanding on the above can guide appropriate investments in flexible IT systems and strategic IT planning.

REFERENCES

1. K. Knoll, and S. L. Javenpaa, S. L., Information technology alignment or "fit" in highly turbulent environments: The concept of flexibility, SIGCPR 94-3/94, ACM.
2. P. Henderson, Laws for Dynamic Systems, International Conference on Software Re-Use (ICSR 98), Victoria, Canada, IEEE Computer Society Press, 1998. 

Enterprise Service Bus: More than promises

By Bijoy Majumdar, Terance Dias and Ujval Mysore

Cost effectiveness and a lightweight configurable architecture is its USP

Many of the enterprises facing integration problems today know what SOA promises and aims to achieve; and feel that many of their problems can be addressed by adopting SOA. Enterprise Service Bus or ESB is one platform that helps achieve enterprise wide SOA. In this article we point out some of the advantages that the ESB offers over other integration solutions.

We will discuss the cost effectiveness of ESB compared to the other options and then we list the benefits of its lightweight and configurable architecture. We also show that it's a safe bet for enterprises that have SOA on their roadmap.

COST BENEFIT ARCHITECTURE

A typical enterprise consists of spaghetti of diverse applications such as:

- Legacy applications like Mainframe applications or other proprietary applications which were developed using variety of technologies and do not provide an interface using which it can collaborate with other applications.
- Packaged applications from vendors such as SAP, Siebel, PeopleSoft and Oracle which are vendor specific and have completely proprietary interfaces.
- Custom Applications developed using J2EE and .Net frameworks and deployed on different application servers.
- ETL Applications for data warehouse acquisition processes of Extracting, Transforming (or Transporting) and Loading (ETL) data from source systems into the data warehouse.
- B2B applications through which businesses collaborate. These may have range of interfaces which may be known only to partners or may be open for anybody to communicate.

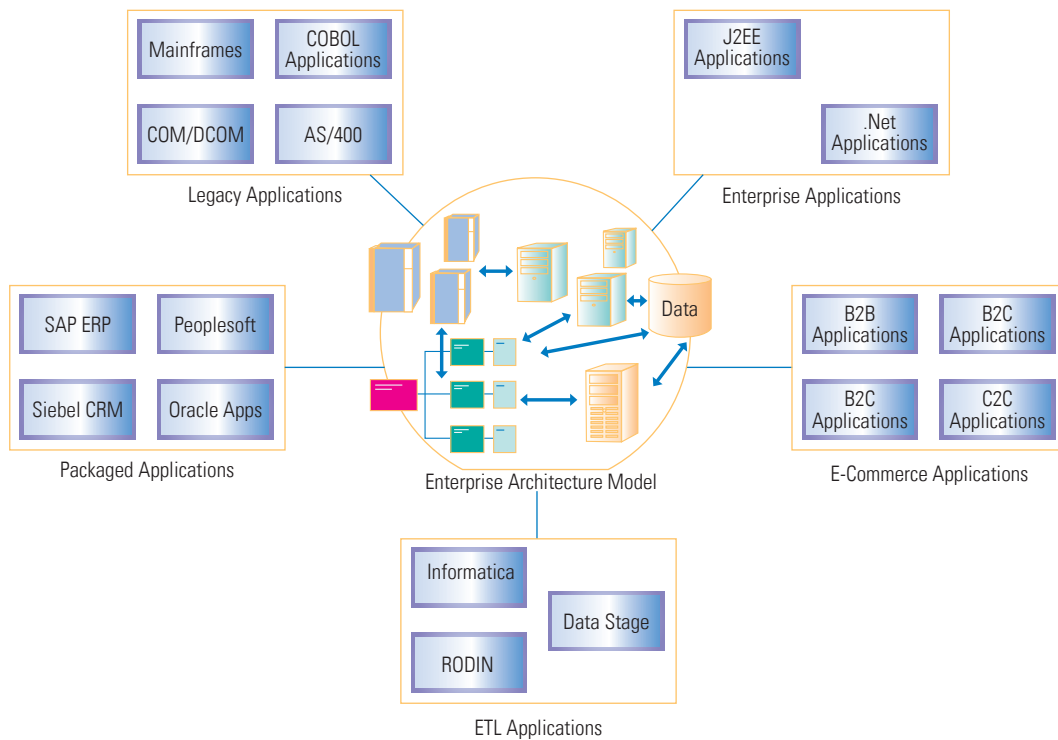


Figure 1: Typical Enterprise Applications

Source: Infosys Research

The varied applications on varied platforms together manage and manipulate the business data based on the business rules. For a business to run successfully, these systems should be able to communicate with each other.

EARLIER SOLUTIONS

Initially to connect these diverse applications, the traditional approach was to develop custom solutions as and when the need arose. But these solutions proved to be costly and time consuming since it had to be developed from scratch taking into consideration both the independent systems involved. Then the concept of EAI tool was introduced, which helped applications integrate in a much easier and faster way. EAI involved every player to this brink for

a revolutionary change in integration platforms and provided solutions that synergized all the systems and platforms in the business scope. All the integration system players had come out with various adapters to connect and talk to various systems in a more robust manner. Now applications could just be plugged to different systems and inter-system communication was easier and faster than ever before.

IS EAI A BEST FIT FOR AN INTEGRATION SOLUTION?

Lately, people have started realizing that just having systems talk is not enough. The solution should also be flexible. Say if there is a change in the enterprise infrastructure due to change in business or due to acquisitions and mergers, the

solution should be agile enough to adapt to the new conditions in minimum time and cost. This is where EAI kind of application integration had its pitfall.

EAI actually creates an agent between application/systems and its underlying middleware. It provides wizards and tools for configuration of the agents. This solution looks good for short term, but for long term this could prove to be a nightmare because the implementation is vendor specific and also requires the use of wrappers/adapters. A version change in the in-scope systems mandates a change in the adapter. And when you consider all the different adapters in an enterprise talking to different systems, the task of managing them was extremely difficult. Hence, this solution proved to be costly. Also the applications that are integrated are tightly coupled, not allowing an SOA kind of environment which is required by businesses in a constantly changing business scenario. This has called for a standards based approach to get the applications to talk, at the same time minimizing the time and cost to adapt to changes.

ESB PROMISES TO SOLVE THIS PROBLEM

An ESB is a platform built on the principles of Service Oriented Architecture (SOA) and other open standards to help applications integrate seamlessly. It is lightweight and extremely configurable. A basic ESB provides a messaging infrastructure along with basic transformations and routing. It mainly uses open standards like web services enabled application to talk. Document style web services provide the sort of loose coupling and reusability required for SOA. Therefore an ESB acts as a SOA backbone for the enterprise. In designing and developing a loosely coupled system, EAI will be anti-pattern. ESB will be

best suited and aligned with the requirements as compared to EAI.

Unlike EAI strategy, it's not applications which are interacting, it is the faction of the application or services coordinating with each other. So ESB fits the bill.

LIGHTWEIGHT, CONFIGURABLE TECHNOLOGY

The ESB architecture is extremely lightweight and configurable container. The integration container permits the creation and hosting of multiple defined services like transformation and content based routing which allows the smooth and intelligent flow of messages and logic in the business backbone.

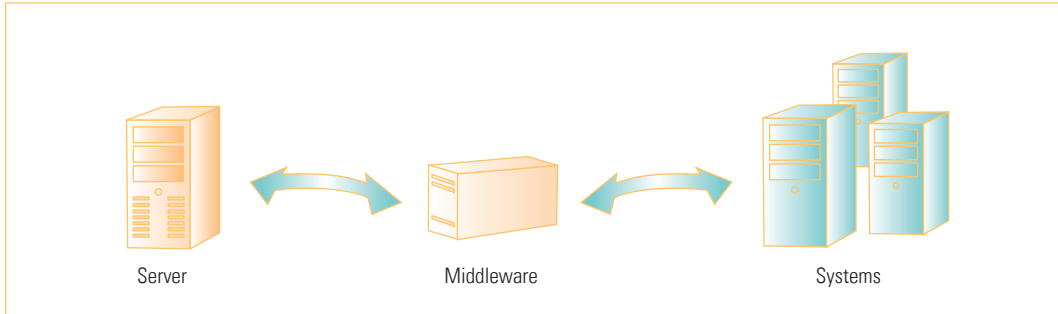
ESB does not come under product category. It is a concept and a framework to be followed or supported for varied services in various platforms to synergize with the business infrastructure. An ESB is a complete backbone upon which to build enterprise Service-Oriented Architectures (SOA). The integrated services are held with the use of standard based messaging that allows the application systems to scale, increase availability, increase throughput, federate security and manage services as well.

ESB is more a virtual layer, an abstraction over the various transport protocols, integration strategies, exchange and endpoint bindings. This phenomenon demarcates the business processes with the other trivial underlying protocols and utilities. ESB allows applications to be brought into a broad-scale SOA at their own pace, in a phased approach that allows the flexible scheduling of the IT resources responsible for doing it.

CONFIGURABLE SERVICE INTEGRATION

ESB model supports the binding component to service engine model for interaction. This results

EAI Infrastructure



ESB Infrastructure

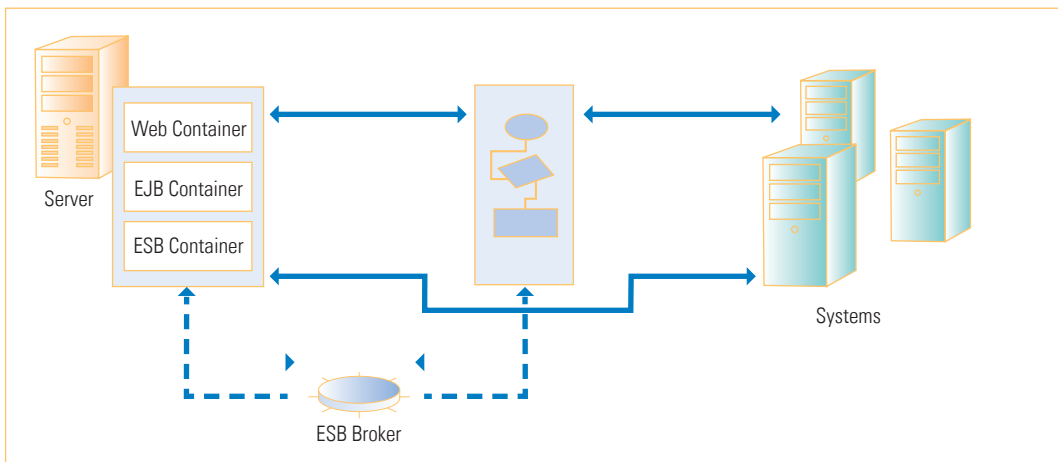


Figure 2: Lightweight ESB infrastructure highlights flexibility in deployment

Source: Infosys Research

in loosely coupled interfaces among various services in the SOA based business system. ESB is mostly configuration of the appropriate binding components to services rather than the conventional coding to integrate with applications on containers like application server, EJB or EAI. The ESB components can be configured and deployed with trivial XML configuration files and once deployed the implementation is resilient to amendments.

Unlike application server or middleware container, ESB container gives the flexibility to configure and deploy only

selected services, those which are only needed and on demand instantiation with its associated components. An ESB is configuration-driven. Process definitions, service configurations, endpoint bindings, routing channel definitions can be configured and re-configured without any coding, and easy and systematic downtime strategy. Unlike the other containers, it does not need the whole underlying component stack for execution of the integration services when only a subset of the container components or utilities is required. This will reiterate the point of low cost in licensing, installation and long term cost benefit strategy.

LIGHTWEIGHT INFRASTRUCTURE

ESB abstract layer can perform independently or can be placed or integrated with any other container like application servers, unlike other middleware or EAI system, which is a different installation and has its own environment setup. There is an exception with few of the proprietary ESB products; some of them are actually a camouflaged version of an EAI. So EAI system has to work alongside with the application servers or other system

UBIQUITOUS SOA WORLD

Currently, an ESB can connect only to those systems which expose their functionalities through web services. Unfortunately, many of the applications do not have that kind of support. Many ESBs today are built with adapters for connecting to systems that do not support web service interfaces. But we believe that since ESB is an SOA enabling platform and SOA cannot be achieved with adapter based integration, adapters should not be used in an ESB. But the

Exposing functionalities as Web Services is catching on - this in turn will have a cascading effect on ESB adoption

to create the integration backbone for the enterprise system. ESB on the other hand can be deployed or migrated across J2EE application servers due to its lightweight and scalable characteristics. This is depicted in Figure 2.

On virtue of its standard based approach and service based integration, it can also utilize some of the features or resources of the application server container. Most users of ESB technology also are users of the application server technology. The business services and utility services are being hosted in the application server or services are being hosted by various EIS systems. ESB collaborates with all these services supporting all with the messaging backbone throughout the enterprise infrastructure. The systems, information systems and data sources, across the enterprise or organization are knitted using the ESB technology.

good news is that awareness and acceptance of web services is increasing. By a large amount, the software package or business product industry is focused on exposing functionalities as web services.

From the list of enterprise applications mentioned above, custom applications and b2b applications are mostly developed in-house or by a software service company with the business inputs from the enterprise. Since these applications are owned by the enterprise, exposing the functionality on them as web services should not be a problem. All that is required is to put a web service wrapper over the existing functionalities. Many of the application servers available today from vendors like Microsoft, IBM, Oracle, BEA, SUN, JBoss, etc. come with inbuilt tools that make it very easy to expose existing functionalities as web services. Only problem then is legacy, packaged and ETL applications. Let us consider them one by one.

LEGACY APPLICATIONS

Most of the legacy code today lies on mainframes. They have existed since the early days of computing and we feel they will continue to be a good option for enterprises having huge amount of transactional data. Let us consider z/OS. The vendors as such do not offer any tool inherent with these systems to create web services but there are tools available in the market (ivory, neon, etc.) which make it possible to expose the applications running on these systems as web services. Using these tools a CICS-COBOL embedded application can be exposed as a web service but a COBOL batch application cannot be web service enabled since it does not use TCP/IP. Going forward these kind of tools only promise to evolve to provide better

close to these two giants as far as market share in EIS software is concerned.

These systems are undergoing changeover over the years to support various upcoming standards for flexibility and ease of integration. Let us take a closer look at their plans for their products and web services.

SAP's new suite of mySAP products is built on NetWeaver. NetWeaver is the runtime environment or technical foundation and acts as an interface between SAP applications. It can also help interface SAP applications to other systems since it is built on http, xml and web services. It enables Enterprise Service Architecture (ESA) or SOA. SAP is re-engineering its products to event-driven SOA.

Leading enterprise platforms such as SAP and Oracle are taking to SOA in a big way - either through reengineering or exposing functionalities as services

functionality and easier operation, making it easier to create web services on mainframe systems.

PACKAGED APPLICATIONS

The most prominent vendors of packaged applications or EIS systems are:

1. SAP
2. Oracle

SAP is the undoubted market leader in the EIS arena. For large organizations it holds 50% of the market share. Oracle, with its acquisition of PeopleSoft and its proposal to acquire Siebel, is not very far behind. No other vendor comes

Oracle recently announced its latest endeavor called Project Fusion. The main objective of this project is to extend and evolve the best features from PeopleSoft, JD Edwards and Oracle product lines. This suite of next generation enterprise applications will leverage Oracle's Fusion Middleware which is SOA enabled. Oracle's E-Business Suite 11i onwards is web service enabled. It is an agreed statement that the solution would bring in the vendor lock in problem. The brighter side is that the functionality is exposed as services that can be consumed by any open source system.

SAP and Oracle featured on the Gartner's WebServicesmagicquadrantLeaders

where leaders are “high-viability vendors with proven track records in Web services, as well as vision and business investment that indicate they are well-positioned for the future. Leaders do not necessarily offer the best products for every customer project; however, they provide solutions that offer relatively lower risk.”

One of the important parameters considered for this report was the company’s business model and the role of Web services in that company’s strategy, as well as its plans and vision for how the overall industry will benefit along with implementation and usage of web services and compliance of standards.

ETL TOOLS

Business Intelligence demand has been increasing for years. It has evolved over years from mere batch process to structured mechanism for each phase of extraction, transformation and loading. Shared Data Services is just one of the major works being done in the area around data-warehousing to help vendors move closer towards their SOA vision. Most of them already have web services support. We’ve just listed a few prominent ones.

One of the best ETL products in this market is Informatica’s PowerCenter. It provides support for invoking existing ETL jobs from within a WS framework. It also allows the client to access the metadata via a web service call.

Oracle Warehouse Builder allows you to publish the mapping and the process flow as a web service allowing the ETL functionality to be called from any other application.

BusinessObjects Data Integrator allows called resources to interact with and modify data. It can publish any job as a web service and call external web services from within its jobs.


CONCLUSION

We have seen the advantages that an ESB offers over other integration solutions. While we say that the perfect SOA environment is far from real today, we maintain that with the way web services and ESB standards are evolving and the rate at which the acceptance of these standards is increasing, ESBs are a safe bet to start building your SOA enterprise.

REFERENCES

1. “Magic Quadrant for Web Services Platforms”, Gartner, http://mediaproducts.gartner.com/gc/webletter/microsoft4_enterprise/2005/article15/article15.html, 2005
2. Sandra Rogers, Stephen D. Hendrick, “Oracle builds comprehensive SOA platform”, http://wp.bitpipe.com/resource/org_1130799543_96/8560_5_edp.pdf?site_cd=ssc&src=KA_RES_20051110, January 2005
3. “SAP NetWeaver: Providing the Foundation to Enable and Manage Change”, SAP, <http://www.sap.com/solutions/netweaver/index.epx>
4. “Enterprise Services Architecture: Blueprint for Services-Based Business Solutions”, SAP, <http://www.sap.com/solutions/esa/index.epx>
5. Paul Krill, “JavaOne: Oracle sets SOA blueprint”, http://www.infoworld.com/article/05/06/29/HNoraclesoa_1.html, July 29, 2005
6. “Enabling Adaptive Business Processes: Oracle E-Business Suite and Service-Oriented Architecture”, An Oracle white paper, http://www.oracle.com/technology/products/applications/integration/Oracle_EBS_and_SOA.pdf, August 2005

8. "BusinessObjects Data Integrator and Web Services", http://www.businessobjects.com/pdf/products/dataintegration/di_web_services_information.pdf

9. "Oracle Warehouse Builder 10g", http://www.oracle.com/technology/products/warehouse/sdk/sdk_home.htm 

Service Oriented Infrastructure

By Shubhashis Sengupta, Hariprasad Nellitheertha and Srikanth Sundarrajan

Embracing Grid and Virtualization technologies is imperative to realize the true potential of SOA

There is a palpable sense of anticipation about SOA – the “new face” of enterprise computing and IT services paradigm. Lot of debate is going on the important SOA issues like application models, service granularity, interfaces, re-use economics etc. The story that is left untold is how some of the so called esoteric technologies like Grid and virtualization make true SOA realizable in practice; well, almost. There are many significant technology shifts happening in those dark non-descript gargantuan warehouses that pass by the name “data centers” hosting thousands of computing resources. Research efforts worth millions of dollars are being spent on infrastructure layer virtualization and associated technologies. If we add to this the decades of extensive research done in the area of distributed heterogeneous computing or Grid, we now have a set of technologies that can usher in service orientation in the infrastructure fabric layer. We term this technology paradigm as Service Oriented Infrastructure (SOI). In this article, we discuss how the fabric layer technologies can play a vital role in realizing SOI

in an enterprise. We also show how “thresholds” of service orientation can be achieved through increasing maturity of fabric technologies like virtualization and Grid.

LAYERS OF SERVICE ORIENTED INFRASTRUCTURE

The enterprise architecture is inexorably migrating towards an open and service oriented structure to facilitate malleability. This progression is happening at different layers of the enterprise IT stack. Business processes are getting flexible and globally integrated. The application layer is moving towards service creation and re-use. The infrastructure hosting models are adopting virtualization. Physical assets are getting commoditized and modularized. The underlying theme for this movement is sharing of resources. We will revisit this journey down IT optimization path in the later part of this article and identify the main benefits and challenges. For the time being, let us focus our attention on depicting what the layers of an ideal SOI would be. Figure 1 represents the typical operational

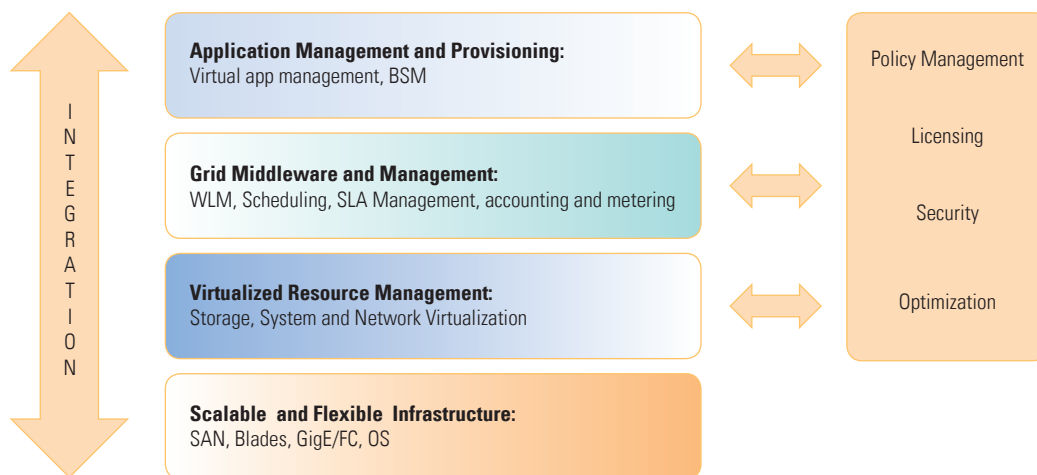


Figure 1: Layers of Infrastructure service orientation stack **Source:** Infosys Research

and manageability layers in a Service Oriented Infrastructure.

SCALABLE PHYSICAL INFRASTRUCTURE

The most visible change in tomorrow’s SOI will be in the layer of physical infrastructure. We are witnessing a shift towards modular platforms, X86 based architecture, commodity hardware and scale-out deployments. Standalone and rack mounted servers will pave way for integrated blade server chassis. Blade Servers help in creating scalable architecture and also help in dynamic provisioning and plug-and-play deployment, along with space and form-factor optimization and reduced power requirements.

Critical data paths will be connected through ultra high speed Fiber Channel or Gigabit Ethernet or Infiniband connectivity. These paths will include the connectivity between the database storage and engines, the blade servers and the Storage Area Networks (SAN) and other paths requiring high data bandwidth. Such a design will eliminate data transfer latencies from becoming a bottleneck. Using SAN, it is practical

and possible to store multiple terabytes of data and still obtain consistent performance.

Edge processing functionalities like web servers, firewall, identity management and other light-weight enterprise services are being farmed out to X86 platforms on Windows and Linux platform. The same is true for back-end shared services like file servers or printing services. The blade servers are now increasingly being preferred to host application containers and business logic. Some of the vertical specific enterprise packaged apps are also being tried out on the blades, especially on those platforms having dual or higher core or processors and specialized management capabilities. This means that commodity processors and OS are now increasingly getting mainstream and the manageability stack on this commodity architecture is also maturing rapidly.

VIRTUAL RESOURCE LAYER

The resource layer comprising the system software like operating systems in the set-up will go “virtual”. New generation system, network

and storage virtualization tools make it possible to aggregate resources such as heterogeneous machines, storage and network bandwidth into a virtual pool and then logically partition servers into multiple independent virtual machines (instances of operating systems) and reserve virtual storage space and network bandwidth. Such an abstraction brings in multiple benefits including:

- Server consolidation, increased utilization; isolation, security and system level (CPU, memory or bandwidth) SLA guarantees for applications.
- “Carve Out”: With virtualization, it becomes very easy to “carve” resources out of operating environment and provision them for a certain customer. This helps in providing per-customer guaranteed resource consumption and performance guarantee. The virtual machines can be allocated a certain quantity of resources such as memory and CPU bandwidth and the actual allocation can be decreased or increased based on load and performance considerations.
- Multi OS Support: Since virtualization tools like VMWare or Xen [1] support multiple operating environments like Windows and Linux, it becomes very easy for hosting and maintaining legacy applications.
- Improved availability: If the load on a particular cluster increases beyond a threshold, the virtual machines (and therefore the entire application set-up) can be migrated to a completely new layout with almost zero downtime.
- “Pools” of virtual machine images can be maintained in repositories. The images can be used for fast and efficient

deployment of virtual machines on-demand and will help in implementing environments only when needed.

- Storage Virtualization aggregates the various storage devices into a common pool and provides a common logical viewpoint. It presents the advantages of the SAN and NAS (Network Attached Storages) to the upper layers of the software stack.

GRID MIDDLEWARE LAYER

The canonical definition of Grid is a network of heterogeneous resources collaborating for a purpose. We would like to stretch this definition in the context of which Grid services will be used in the SOI. In our scenario, the Grid middleware and associated components act as the coupling between the applications and the “raw capabilities” provided by the virtualized systems and physical layers. This layer ensures that the applications make the most efficient use of the resources so that high performance and throughput can be achieved even with high levels of utilization. Theoretically speaking, the enterprise data center can be viewed as a Directed Acyclic Graph (DAG) [2]. The resources, interconnects and application elements form the nodes of the DAG. Grid can group the nodes of this graph in the most optimized manner, assign workloads to these groups and manage the dynamic group behavior in the run-time. The power of Grid in the SOI comes primarily through:

- Scheduling and load balancing: Grid schedulers can schedule jobs seamlessly across heterogeneous machines and clusters.
- Capacity on demand (CoD): Grid allows applications to seamlessly scale across

a Grid infrastructure and also facilitate addition and removal of “nodes” to enable capacity on demand. If the need for excess capacity arises, Grid as the middleware and virtualization at the server level allow automated discovering and re-provisioning of the replacement nodes in the spare server pool.

- Workload management: Enterprise workload management technologies can work in tandem with Grid technologies to provision workload and capacity management.
- Monitoring and management: The nodes can be effectively audited, monitored, and managed through Grid. Grid, in conjunction with virtualization, can effectively manage application SLAs (availability, resource guarantee). Support for and enforcement of flexible usage based licensing also comes under the ambit of manageability.
- Heterogeneous policy management: Grid makes heterogeneous policies, such as cross domain security policies, interoperable.

Instead of talking about classical Grid usage about scheduling and load balancing, let us see how Grid proposes open standards and protocols to make integration possible. In other words, weaving a set of multi-vendor and multi-platform landscape together into a dynamic utility enterprise fabric critically depends on Grid based integration middleware. For Grid to be manageable and interoperable, a resource participating in the Grid should be modeled and administered as Grid managed entity (GME). Organization for the Advancement of Structured Information Standards (OASIS) has proposed Web Services Resource Framework

(WSRF) to represent stateful GMEs and facilitate management through publish-subscribe framework of WS-notification. Global Grid Forum has ratified extensible Open Grid Services Architecture or OGSA [3] which works on WSRF based resource annotation framework. One such interoperable protocol for resource information and metadata repository schema is the Common Information Model (CIM) propounded by the Data management Task Force [4]. Open group has proposed Web based Enterprise Management on top of CIM to facilitate flow of information across managed resources through an XML format. Yet another framework of managing and mapping heterogeneous resources is Web services based System management (WS-Management) framework.

The Data Centre Mark-up Language (DCML) is the first standard to provide a structured format for describing the complete data centre environment for the purpose of ITIL-based service support management.

APPLICATION MANAGEMENT LAYER

The layers described thus far enable a shared resources model in an SOI. To get the story complete, one needs to transition applications too into a shared services model. Managing and provisioning of applications in the data-centre has to be significantly different from traditional “static silo” based approach. In the context of a Grid, the first attempts of decoupling applications from the infrastructure have been in use of code mobility, adaptive configuration and building support for cross-platform application deployment like Crosslets [5]. There is an emergence of a new breed of application virtualization, where, instead of viewing standalone applications running on a server, we need to create a virtual model of application groups, which can be

provisioned and deployed to adaptive groups of virtual servers in a configuration that is self optimizing. Applications with similar run-time characteristics and resource requirements can be clubbed into the application groups. This type of configuration eliminates the requirements for redundancy for each server in the pool.

Using properly configured application management and provisioning tools, any of the spare virtual servers can potentially replace a failed server. Through policy management, the technology offers increased availability of the key business services, allowing more virtual servers to be provisioned dynamically for more critical applications when the corresponding user loads are high and makes centralized manageability easier.

To take this idea further, we can even conjure up a vision of scripting an application which can make use of the virtual resources and dynamic integration. Suppose that at any point in time during execution of a batch application we require the data to be archived in virtual storage, the application would invoke an archival service end point and a set of complex interactions in the DAG would set into motion. For example, operating system utilities must be able to communicate with application APIs, mediated by server, which must be able to describe table requirements to a database, which must be able to request disk space from a storage array, which in turn must be able to initiate an automated tape backup, which in turn will negotiate for bandwidth reservation in the LAN. All these services are virtual services, not tied to an application server instance, or a database or a particular storage device driver and so forth. From the perspective of an end user, the entire scheme of services remain hidden – giving him a notion of true utility on demand and charging him on actual resource consumption.

THRESHOLDS IN SOI REALIZATION

The journey towards realization of SOI has few technology steps to be taken. In Figure 2, we have depicted essentially four thresholds in the technology paradigm. The first two stages mark largely where enterprise infrastructure stands today. There is a big rush towards server consolidation and standardization. This infrastructure is typified by server co-location, standardized operating environments (into Linux, Unix or Windows) and by shared hosting on dedicated servers. Consolidation is mainly achieved through first generation static physical partitioning strategies and workload management. For example, Dynamic System Domains (DSD) in SunFire servers and n-Partitions in HP or CMPs in Fujitsu. The benefits of this model accrue from consolidation (aggregation of IT resources and maintenance efforts to fewer points) and improved server utilization.

In the second threshold, Software partitioning through Virtualization marks the beginning of SOI. It is now possible to host several instances of disparate execution environments such as operating system images in the target server to achieve highest level of isolation. The full system virtualization, one of the oldest tricks in the trade, came from Virtual Machine Monitors of IBM MainFrames and got percolated in other servers like IBM zSeries, iSeries family, HP vPARs and pure software Virtualization systems like VMWare's ESX servers etc. Some of the problems of first generation virtualization techniques were that most hardware faults could not be isolated from the applications, the performance overhead of the monitors, sharing of hardware buffers and I/O. All these ensured that virtualization is seldom tried out in production environment. The second stage confines the notion of SOI and

Stage 1 Consolidated	Stage 2 Virtualized	Stage 3 Optimized	Stage 4 True Utility
<p>Features Shared hosting in discrete servers, consolidation</p> <p>Benefits Reduced costs, Improved utilization</p> <p>Challenges Integration, Performance, Manageability</p>	<p>Features Resource Virtualization, Grids, Consolidated management</p> <p>Benefits Security, Higher utilization High performance, availability</p> <p>Challenges Optimization Efficiency Manageability Rehosting</p>	<p>Features Policy based management, Capacity on Demand, Application virtualization</p> <p>Benefits SLA guarantees, Service provisioning</p> <p>Challenges Scale-out IT as a service</p>	<p>Features Pay as you go, Software as service, Metering and accounting</p> <p>Benefits Value based pricing Increased ROI Lower TCO</p> <p>Challenges Pricing and licensing models, Chargeback</p>

Figure 2: Stages of SOI

Source: Infosys Research

shared resources deep down the system bowels. The application stickiness to the servers is not yet done away with.

There are a few main drivers to improve the SOI technology, in terms of pervasiveness of applicability and performance benefits, which will take us to the third threshold of service orientation. It is worthwhile to note these briefly here:

- At the virtualization technology level there is a tremendous thrust towards enhanced performance. Intel's VT technology (also AMD's Pacifica) will change the image of IA32 architecture being unfriendly to virtualization and improve performance. These new improvements will make it very easy for hyper-visors to support disparate operating systems on a common virtualization platform.
- Hyper-visor's integration with kernel will be standardized in future and will allow

virtual machines running over different hyper-visor to interoperate or migrate over seamlessly.

- The Xen hyper-visor will provide near-physical server performance to virtualized containers and will drive the adoption faster.
- At the Grid middleware level, service provisioning engines will create the intelligence of Capacity on Demand. The CoD middleware will map applications to resources dynamically and will replicate / clone resource groups as per application group workloads. Manageability of the SOI will be very easy. Some vendors, call such groups as "virtual appliances" [6].
- Finally, the business services management or BSM technology will mature to provide the critical linkage of enterprise processes to SOI. The BSM software, in addition to providing extensible policy

description, makes seamless service-oriented configuration management for infrastructure possible.

The final frontier for Service oriented Infrastructure will be “true utility.” The enterprise computing infrastructure will be seamless, pervasive and utility-like, where anyone can plug-in, submit and run any applications using multiple access mechanism. This notion is akin to having switching on a light without bothering where the electricity comes from – the pithead thermal power, hydroelectric power or Nuclear power.

Software as a Service or SaaS will be one of the key enablement of utility Grid. We are already witnessing Amazon’s EC2 and S3 computing and data services [7]. Microsoft and Google are looking aggressively at SaaS in the B2C scenario, where huge virtual infrastructure is optimally tuned and primed for delivering software through self-service.

This article is not about SaaS, but let us look at how SOI will enable true utility. True utility depends on sharing and re-use of infrastructure, data, applications and processes. Data security will be of paramount importance as the providers will embrace a multi tenant model. In the fourth epoch, SOI will provide a computing cloud. The key features of the computing cloud will be:

- Compute resources will no longer be discrete, this means virtual servers will span across multiple, possibly heterogeneous physical machines to provide a different type of single system image that goes beyond cluster computing.
- Capacity planning will mean having right resources available just in time to host services and not pre-configured servers.

- The CoD or just in time capacity should be elastic with demand. In order to overcome the issue of finite resources (will not be much of a problem with storage and computing getting cheaper by the day), the Grid manager should allow virtual appliances to be suspended and resumed based on business priority and resource availability without losing application state.
- Rich semantic Web annotations will make resource description and discovery easier.


Applications crafted for utility Grid will use high level wrapper services for resource discovery, negotiation, lease and run-time execution. SOI will provide abstraction for infrastructure resources which can be asked for and consumed on the fly. Only then can true SOA be realized. It is difficult to ascribe any timelines to this journey. Various enterprises are implementing SOI at different paces. However, we see that by the turn of the decade, there will be an acceleration of enterprise IT towards embracing SOI.

CONCLUSION

Service Oriented Infrastructure should form the bedrock of the overall enterprise SOA strategy. Aligning infrastructure with business in a scalable and flexible manner facilitates IT optimization. Application services that are not tied to particular infrastructure but can dynamically map to a dynamic and elastic infrastructure “cloud”, will provide true SOI. Infrastructure fabric technologies like Grid and virtualization are rapidly progressing to make this vision a reality.

REFERENCES

1. Xen and the Art of Virtualization, Paul Barham et. al, Proceedings of

- the nineteenth ACM symposium on Operating systems principles, 2003
2. Enterprise Grid Alliance Reference model and Grid use case version 1.5, available at <http://www.gridalliance.org/en/workgroups/referencemodel.asp> accessed on 11-sep-2006
 3. The Open Grid Services Architecture Version 1.5, Ian Foster et. al., , available at <http://www.ggf.org/documents/gfd.80.pdf> , accessed on 08-sep-2006
 4. Common Infrastructure Model Reference version 2.12, details available at <http://www.dmtf.org/standards/cim/>
 5. Crosslets: Self-Managing Application Deployment in a Cross- Platform Operating Environment, Stephen Paal et. al., in proc. IEEE Middleware 2005 conference
 6. 3Tera Applogic server purports to support on-line mapping of virtual appliances to underlying infrastructure. Details can be found at <http://www.3tera.com/applogic.html>, accessed on 11-sep-2006.
 7. Amazon's EC2 and S3 are concept computing and storage utility applications sold by Amazon Web services LLC. details can be found at <http://developer.amazonwebservices.com/> 
-

SOA and BPM: Complementary Concepts for Agility

By Parameswaran Seshan

Enterprises can realize better success in aligning IT systems with business by combining the power of SOA and BPM

SOA (Service Oriented Architecture) and BPM (Business Process Management) are two technologies that have been popular in the recent times. Enterprises are looking to make their business systems more flexible by making use of these technologies. This article looks at what SOA and BPM bring to the enterprise architecture and gives a practical perspective on the synergy possible between them. It puts forward how these two are really complementary technologies and not conflicting ones. Enterprises can derive maximum benefit in the alignment of business and IT, by best leveraging their complementary aspects and by coordinating the efforts in both the areas.

WHAT THEY ARE

There is some confusion in the business and technology world on the relationship between SOA and BPM. Respective groups backing these technologies appear to be putting forward arguments and counter arguments that do not

seem to be helping the cause of companies that want to make use of these technologies. Both, BPM Systems (BPMS) and SOA, have been rising in popularity in enterprise architectures with SOA being pushed more by IT and BPM being pushed more by business. Though both promise flexible and agile IT systems, the haziness over their relationship is impacting their adoption in enterprises.

SOA is an architectural pattern that promotes design of software elements as services that can be re-used and then combined to create applications. Each service has a well published interface and provides a well defined functionality. The service interface is abstracted out and separated from the design and implementation of its functionality and the invokers of the service are not exposed to the service implementation.

SOA based architecture makes IT systems agile by allowing applications to loosely couple to each other through the service interfaces.

Composite applications can be built quickly, to service business needs, by chaining together such services and orchestrating their invocation at run time. Such assembly of an application from available services rather than writing code is a value proposition that SOA offers. These services become available to the entire enterprise and some of them even to the external world to customers, partners or suppliers. Services need to be managed and supported to ensure their adherence to SLAs (Service Level Agreements) and evolution including version management without impacting the clients.

BPM is the set of strategy, tools and techniques to design, deploy and simulate

the process by executing the activities in the order specified, provides process performance reporting, and allows the business analyst to monitor the process real-time and analyze the process and improve it. Since the process definition drives the execution of the business logic in BPMS, it makes processes flexible. This is because the formal process definition can be changed to effect change in system behavior with no or minimal changes in underlying applications.

WHY THE DEBATE

Historically, business groups in organizations had put in significant efforts to re-engineer

BPM views business processes as core assets of organizations - this enables their effective design, deployment and simulation

processes to enable rapid process-based change in the organization aligned to meeting organizational objectives. The key aspect of BPM is to see business processes as core assets of companies and managing those processes over their life cycle. BPMS is the technology part of BPM and is an architecture concept that takes a process-centric view of applications to provide flexibility and business alignment to applications. BPMS is a convergence of workflow, EAI, process modeling and business rules.

Each process is an ordered sequence of activities performed by system or humans to achieve a business objective. Processes are defined by a business analyst in a graphical modeler provided by BPMS. BPMS automates

business processes to improve and respond to market demands. However their efforts did not become successful since IT systems at that time were not flexible enough to support those changes. Business, then took to the promise of the then new, ERP (Enterprise Resource Planning) packages that offered standardized business processes and canned business functions. But they learnt while implementing that it involved a huge effort to tailor the packages to suit their specific business needs, as systems were still rigid. The perception that, IT systems are inflexible and business cannot control these systems only started growing and business got more skeptical about IT. Recently BPM technologies

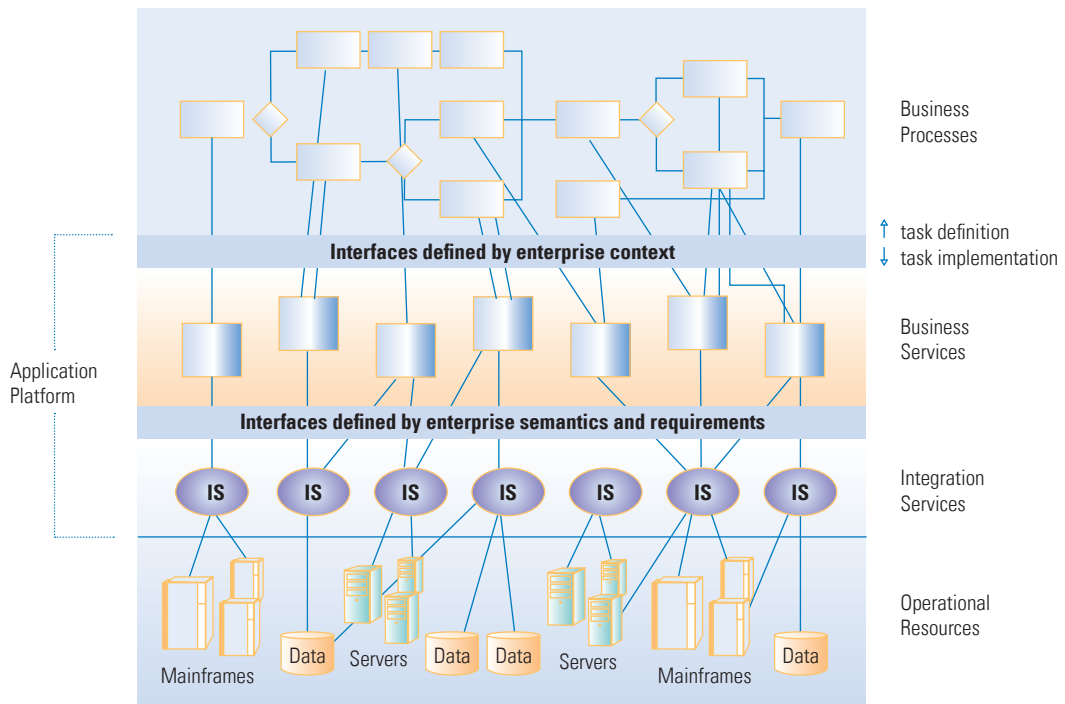


Figure 1: Platform view
 © 2006 AZORA Technologies, Inc.

Source: Adapted from *BPM and SOA-where does one end and the other begin*; Mike Rosen, <http://bptrends.com>

have emerged to provide the opportunity for business groups to take more control of their processes, and business groups have started believing that they finally have a technology addressed to them using which they themselves can effect changes in the behavior of IT systems by changing process models. At the same time IT has been trying to make IT architecture more flexible. IT believes that SOA is a good opportunity to deliver the promise of business adaptable IT systems.

The community is tending to back BPM and IT community tending to push SOA. There is a view point difference, with the BPM community viewing enterprise architecture from top-down (business process perspective) versus the SOA community viewing it from

bottom-up (services). So a gap appears to have formed, with their concerns assuming different meanings for terms like services, processes etc. This is apparent from the way companies like IBM has chosen to address both views in the same WebSphere product by providing Business process modeler and Integration developer components.

SERVICES TO PROCESSES

BPM provides a high-level abstraction for building IT systems namely, the process layer. This is the layer at which the business can effectively take part in development and evolution of IT systems. Each activity in a business process is expected to be a componentized, reasonably high-granular

service which performs a logically complete business operation. If the activity is a system activity, then the BPMS expects SOA to provide this service. Let us take for example a sample process in a bank namely account opening. Here deposit money into account is a system activity and this activity is realized by a service which updates the amount in the system. Thus the activity needs to be linked to this service in the process model and the service is expected to perform the complete business function expected of the activity like in this case performing the required validations like account status, applicability

reused in another process say, manage customer account or funds transfer process. To realize such services that are truly re-usable BPM requires the underpinnings of SOA. SOA can provide BPM this re-usability framework so that activities in a process can get reused easily in different processes instead of them getting implemented as functions bound to the specific context and requirement of one process alone. SOA gives the power to expose the process itself as a service so that it becomes usable to other processes. In a higher level process, it becomes an activity which denotes a sub-process. In B2B or B2C integration scenario, where this process needs to

Reusability in BPM needs to be built on a foundation of SOA - as SOA is best positioned to provide a reusability framework

of differential interest rates etc., and make the account reflect the money deposited. Such business services need to be exposed to the process so that they can be invoked with the process in BPMS becoming a client to the services consuming the services, one that orchestrates their invocation faithfully in the order defined by the business analyst (Figure 1) SOA is the underlying model that is service provider to BPMS. On the other hand, for a manual activity in the process, BPMS expects the service to be performed by a user belonging to the appropriate role and it routes the work to the user.

At a process level, re-use of high-granular services is possible. For example, the deposit money into account service can be

integrate with the process of a partner, supplier, or customer, this process can be provided as an external service which the external process will invoke.

BPM visits the application integration (EAI) problem with a process-based integration approach and makes the process layer drive the integration rather than traditional point-point or hub-spoke hard-coupled integration approaches. SOA enables this loose coupling with each application exposing its core functions as services and then the business process determining the highly flexible order in which the services need to be chained. And this is without the application getting impacted with any change in the invocation sequence.

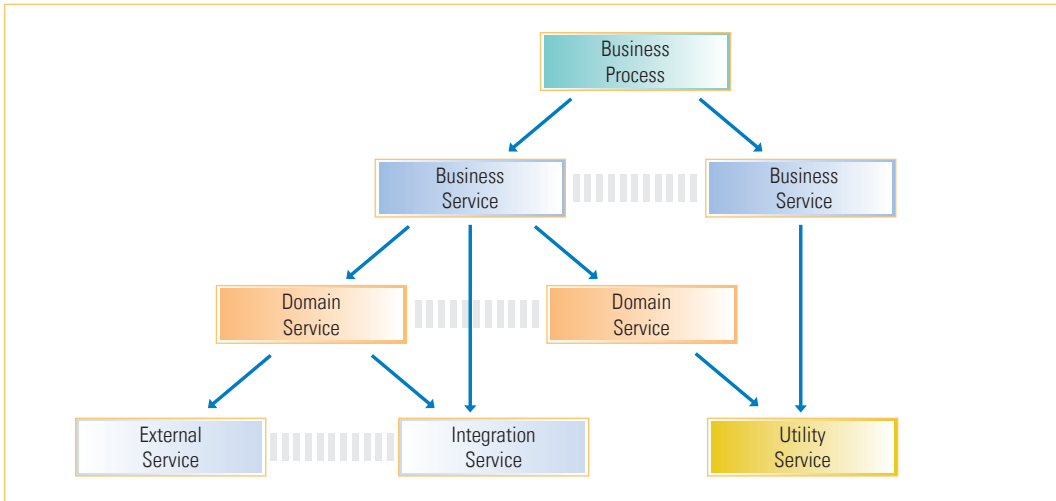


Figure 2: Services Hierarchy
© 2006 AZORA Technologies, Inc.

Source: Adapted from *BPM and SOA-what kind of services does a Business Process need*, Mike Rosen, <http://bptrends.com>

WHAT SOA ALSO NEEDS

In an enterprise, it is neither feasible nor cost-effective to visualize and build all the services that are ever required by applications. One of the reasons is that the business requirements also evolve and the functionality that services need to support is usually determined by what the business thinks the business process should do. It is more feasible to let the business first design or re-design their business processes according to business need and then determine the services required for each activity in the process. The services so identified would be re-usable across various processes as business can better visualize the various process contexts where the same highly granular business functionality is required.

With SOA, we can have a service-hierarchy where we have various layers of services with each layer re-using a lower layer service (Figure 2). A lower level service will typically be a wrapper service to service-enable

a legacy function, a re-usable infrastructure service (i.e., one that provides an architecture level function like logging service) or an external service, or a data access service which gets specific information from the database. One example is “Get Customer Details” given a customer id as input. The business services need to be composed from low granular services and are the ones that are mapped to activities in process. Hence their identification and interface contract design would be influenced by business processes.

Though SOA’s core focus is overall life cycle management of services, it needs a macro context to be present that knows the way these services are to be chained together to deliver value. This context, SOA assumes, would be specified by business analysts or managers who are expected to do this with the formal approach and tools provided to them directly by BPM, and at run time processes executed in BPMS provide this context. The process maintains the state

for the interactions spanning across multiple services. Services primarily exist to serve a business need and their true value is realized only when some client orchestrates their invocations to create value. This client would ideally be the process in BPMS; without the context provided by processes, their existence would not have much meaning.

SOA would need help of BPM to give business more clarity, visibility, flexibility and control of the process flow plus control flow in a composite application. The choreography in a composite application may involve some human actions where a human role performs a function, say approval and system actions. This choreography, being buried in the execution

Firstly, the approach should follow top-down modeling of the process architecture of the enterprise (AS-IS). Process analysts can look at the value chains in the organization and model the core and support processes for that in BPMS starting from level 0, iteratively to lower layers (level 1, level 2 etc.) of the process hierarchy. After analysis, for BPMS and SOA implementation the analyst can pick up one or two lower layer processes from this that are of high impact, say for example, Account opening. This way with cost savings in initial investment, immediate benefits could be gained.

This can be extended further on to other processes incrementally, applying the learning from the pilot experiences. Now before

SOA and BPM are truly complementary - they go hand in hand to impart businesses with more clarity, visibility and flexibility coupled with greater control of the process flow

piece of the application remains invisible and outside-of-control of the business analyst unless it is modeled at the business process level abstraction explicitly by the business analyst in BPMS.

THE IMPERATIVE

Both SOA and BPMS are emerging technologies and standards in both the areas are still firming up. Clearly SOA and BPM enable each other and mandate existence of each other for value realization. This calls for enterprises to approach their implementations in a coordinated fashion rather than with independent efforts.

identifying the services to be enabled, instead of just implementing the existing process AS-IS, the analyst needs to re-design the process for optimization (model the TO-BE) as otherwise, the SOA effort would not be effective. SOA cannot make any positive difference when implemented on a flawed as-is process and this is not the problem of SOA or BPM per se. In the To-Be process, identify service required for business functionality of each activity in the process, keeping in mind the potential for re-use of the service across processes.

Business services thus identified and to be implemented would, only be in a small number when compared to what we would

end up having if we identify the services from a bottom-up fashion. Now IT needs to design and implement these business services on SOA. Hence there is a cost advantage here as opposed to building full set of services from bottom-up and then ending up not using a good number of them.

IT should design the business service in such a way that it is composed from lower level services including wrapper services that service-enable legacy functionality to leverage existing applications wherever possible. To enable maximum re-use across enterprise, a single point of truth for business information and uniform semantics across the enterprise must be maintained for data-access services. For example, customer profile information must be available through a single service that accesses it from one data source and it should mean the same customer as viewed in the enterprise, through out.

SLAs for the service also must be defined and the services must be managed by a core services group to manage service evolution, version management and SLA assurance. A repertoire of re-usable business services can be built by adding such services incrementally based on implementation of new or re-designed business processes. So the upfront investment and risk for SOA implementation is low.

Coming to execution, the process visually modeled in BPMS is made executable by the BPMS by converting it into an executable-process definition based on a standard like WS-BPEL. WS-BPEL, which chains together each service involved in the process flow in the order mandated by the visual model, is executed by BPMS at run time. For runtime, the binding of the concrete service end points to the abstract service interfaces specified in the process definition is taken care of by the BPMS using capabilities in the BPMS to discover, negotiate and select the

right service provider or using the infrastructure of ESB (Enterprise Service Bus).


PROCESS IMPROVEMENT

The process execution performance is monitored by process manager through BAM (Business Activity Monitoring) console provided by BPMS which provides the metrics related to the process. The process analyst can do analysis and make optimizations in the process by changing the process model. Among other things, the analyst looks at the process cycle time, service level SLAs etc. As part of optimization, analyst may change the order of invocation of services, drop some services, introduce new activities etc in the process. The underlying executable WS-BPEL process definition automatically reflects this for run time. In the BPMS process modeler, a repository of existing business services could be made available while modeling process so that they can be picked and chosen. SOA thus helps in reconfiguring processes and with BPM it can improve business-IT alignment.

CONCLUSION

SOA and BPM are two technologies showing the promise of making business systems flexible and agile. There seems to be some disconnect between the views of groups advocating these. Holistically, to provide value, the power of process logic separation provided by BPM should be combined together with the re-usable service foundation that SOA delivers (which BPM can leverage to deliver effective and valuable business processes). The key is to make SOA deliver services that are aligned to the needs of business and flexible enough to support process changes in BPM resulting from evolving business needs. Re-design of process should precede design of business services to ensure SOA effort does not get wasted in supporting wrong processes.

REFERENCES

1. BPM and SOA - What Kind of Service Does a Business Process Need?, Mike Rosen, <http://bptrends.com>, July 2006
2. BPMS Watch: BPM and SOA: One Technology, Two Communities, <http://www.bpminstitute.org>, September 2005
3. BPM and SOA - Where Does One End and the Other Begin?, Mike Rosen, <http://bptrends.com>, January 2006
4. The OMG and Service Oriented Architecture, <http://www.omg.org/attachments/pdf/OMG-and-the-SOA.pdf>, accessed August 2006
5. The assimilation of BPM, <http://www.information-age.com/home>, April 2005 

SOA Technology Competency Center

By Shreyas Kamat

Organizational need to manage SOA transformation effectively has bolstered the growth of SOA TCC concept

As SOA momentum is picking up in the industry, more and more companies are adopting SOA on a larger scale. Also as the SOA hype is slowly undergoing metamorphosis to real world implementations, organizations are realizing there are impediments to a successful SOA adoption and growth.

SOA is just not about integration at a platform or an application level but it is a culture change within the organization about how the traditional business-IT interaction happens and a paradigm shift from IT as a 'technology provider' to IT as a 'business enabler.' Understanding this impact of SOA adoption on the enterprise and creating a structure to effectively manage this change is no longer an option - it's mandatory.

SOA AND THE NEED TO MANAGE ITS ADOPTION

In order to successfully adopt the right technologies, manage change and growing them successfully within an enterprise, there is a need

for a structure to be in place to look at the holistic picture, to create effective strategies and to take the organization to the correct maturity level.

In a typical IT world, the role of IT is viewed as a 'technology provider' and hence there exists a customer-supplier relationship between the business and the IT organization. Business provides requirements and IT provides technology platform as per the business requirement. This pattern has led the IT world to build 'applications' and create technology platforms in silos. In order to reuse the 'applications' or 'platforms' already created, IT started thinking in terms of integrations at the platform level and there the complexities of technology integrations were created.

Business needs agility and a complex infrastructure with multiple integrations fails to provide that. In order to achieve agility, the infrastructure needs to absorb the business process changes quickly. This can be achieved by aligning the technology components closely

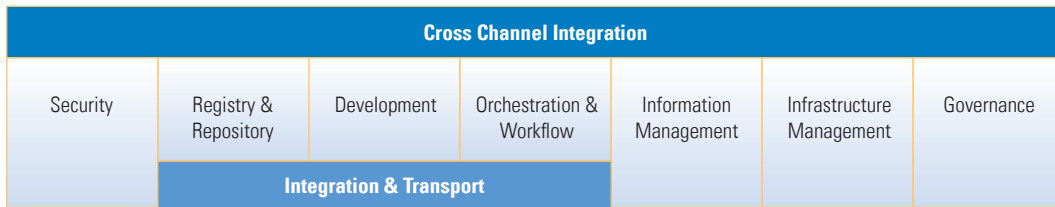


Figure 1: Sample SOA Reference Architecture

Source: Infosys Research

to business process and perhaps giving adequate control to business to create new processes.

Behind creating services which align with the business processes and creating infrastructure to create processes and manage the metadata about the services, is the need to align IT closely to the business and provide business the much needed flexibility. This is a radical change in the IT and business world.

With the dual pressures of meeting aggressive launch timelines and meeting the stringent SLAs of the current operations, bringing in this cultural change is next to impossible. There needs to be a focus around bringing in this change and a structure to manage this change effectively. The questions are

- What exactly is this change? In other words what are the various ‘building blocks’ or ‘capabilities’ which the organizations need to build in order for successful SOA adoption?
- How can organizations achieve managing this change successfully, when they are under such pressures?

ESSENTIAL BUILDING BLOCKS FOR SUCCESSFUL SOA ADOPTION

The concept of SOA is not new now and as more and more organizations have started embracing SOA, some common capabilities which are

required to be built, have started emerging. The three primary building blocks are:

- SOA Reference Architecture
- SOA Roadmap
- SOA Governance

Reference architecture provides a comprehensive capabilities stack to the organization, which it should build as part of the SOA adoption process. These capabilities need to be considered regardless of products and tools the organization needs to consider for successful SOA implementation.

A typical capability stack which is also known as ‘Reference Architecture’ is shown in Figure 1. These capabilities to be built vary based on the organization size, its current capabilities and its SOA adoption goals.

The SOA roadmap provides the organization a sequence of events that need to happen in order to adopt SOA capabilities incrementally and also provides the snapshots of different transition states.

SOA governance provides the control mechanism to control the adoption process and technology growth.

These three building blocks provide the essential foundation for taking the organization to the next level in the SOA maturity. The details of these three building blocks are out of scope for

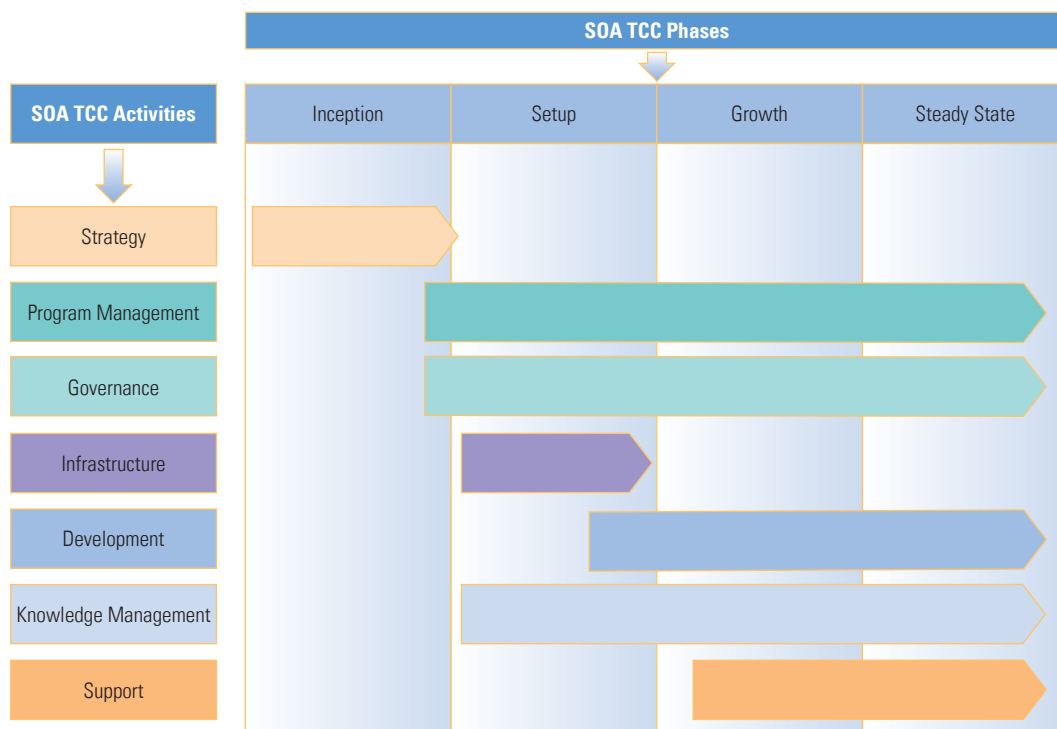


Figure 2: SOA TCC Phases and Activities

Source: Infosys Research

this article; however we will briefly discuss the activities those need to be done for creating these building blocks in the following sections.

Once we have understood the basic building blocks for successful SOA adoption let us look at how SOA Technology Competency Center can provide the structure and processes required to be put in place to successfully adopt SOA and grow it within the organization effectively.

SOA TECHNOLOGY COMPETENCY CENTER

SOA TCC is a central group within an organization which provides direction, guidance and provides the necessary resources in terms of technology, people and processes to the rest of the organization in order to successfully adopt

and grow SOA within the organization. The various other names used currently within the industry for this kind of a group are 'Center of Excellence' or 'Shared Services' or even 'SOA / Web Services PMO (Program Management Office)'. Despite the name, the type of industry vertical and size of the organization, the activities to be performed by this group remain more or less the same. Figure 2 represents the various activities performed by a TCC and the phases during which these activities are performed.

TCC PHASES AND ACTIVITIES

The TCC organization typically undergoes a four-phased evolution.

Inception: This is the phase when the organization is going through a process of understanding

SOA and how should it be adopted within the organization. A business case is established to adopt SOA and SOA strategy/roadmap is defined.

The objective of the SOA strategy is to establish the business case for SOA adoption, define the SOA maturity level where the organization wants to reach and provide a high level roadmap for the transition states including the investment plan.

The roadmap establishes the current capability of the organization after assessing the business/technology drivers and the current IT landscape. Based on industry standards and technology trends, a capability mapping exercise is carried out to define a standards or technology adoption roadmap. The final part of the roadmap is to provide a migration plan from the current state to the 'to be' state.

Another important activity which should happen in the inception phase is establishing the SOA governance framework. There are two aspects of SOA governance: Design time governance and Run time governance. Run time governance activities are usually carried out in the Set up phase. During the inception phase, the TCC policies/principles, investment model, organization structure and engagement processes are planned.

Set up: Set up phase is a more tactical phase when the SOA strategy/roadmap and governance recommendation start getting implemented. The team structure, governance model, program management plan, reference architecture, infrastructure architecture and communication plan get defined/elaborated as part of set up activities.

SOA TCC usually either resides within the EA organization or becomes an extension of the EA organization. The team usually comprises of SOA strategist/architects/developers mixed

with management roles. The governance activities in this phase focus on creating the standards, review processes, engagement processes and feedback processes. The PMO starts tracking the issues/risks and creates communication plan for the TCC.

The technology team starts defining/elaborating on the reference architecture and creates the required frameworks and components. This also includes the infrastructure required to support SOA/web services enterprise wide. Some organizations do not include infrastructure as part of the TCC since infrastructure managed by individual units within the organization may suit the organization. As part of the architecture definition buy/build/reuse analysis, vendor/product evaluation/recommendation and building proof of concepts activities are carried out.

Growth: In the growth phase the actual roll out of the TCC gets started to the specific groups within the organization. The early part of the growth phase is usually the pilot projects. Based on the pilot learning and feedback, the TCC program is rolled out onto the rest of the enterprise over a period of time.

In this phase pilot team/project is engaged as defined in the migration plan of the TCC roadmap. The architects/developers from the TCC team are heavily involved with the pilot team to architect, design, build and implement the pilot project. The criteria for selecting the pilot cover most of the ideas conceived during the inception and set up phase. The learnings from the pilot project are used to revise the documents, standards, models, frameworks, infrastructure and processes. This pilot serves as a major de-risking strategy at an early stage of adoption and prevents any cost or timeline escalations down the line.

Based on the results of the pilot the TCC is rolled out to the rest of the organization

referring to the roadmap/migration plan. The SOA maturity goal as defined in the SOA strategy and the roadmap stages influence the activities of the growth phase to a large extent. Most of the SOA adoptions are a hybrid approach which is a 'meet in the middle' approach rather than a pure top-down or bottom-up approach. There are pockets within the organizations where the teams have implemented web services to meet the business need. TCC needs to consider a bottom up approach to align these groups with the overall SOA standards.

The growth of TCC also depends upon how clearly the TCC mission and goals are articulated and communicated to the rest of the organization. This communication falls under the 'knowledge management' activity of the TCC. Usually a portal is created which contains TCC documents, white papers, workshop material and engagement and support requests. Increasing the awareness and SOA knowledge level within the organization is an important aspect and the success of SOA adoption depends largely on this aspect. TCC usually creates the training material and conducts training sessions and workshops to achieve this goal.

Steady State: Steady state happens when all of the TCC aspects are fully functional enterprise-wide. This is where the TCC is functioning as a governing/consulting group to the rest of the organization. The activities during this phase are primarily geared towards supporting the infrastructure, frameworks, continuing to conduct workshops/reviews and providing consultancy to rest of the organization.

The support activities are restricted to supporting the frameworks, component and infrastructure created by the TCC. These do not include supporting the projects using these assets. However upon request from the project

team TCC may provide consultancy to resolve a specific issue.

The steady state of the SOA TCC may get disrupted due to organizational changes such as internal structure changes, merger and acquisitions, or even major changes in any of the underlying technology directions/infrastructure. When such a change happens TCC may need to run through the four phases as a mini-cycle to manage the change.

DE-MYSTIFYING SOA TCC

Some of the common myths seen in the industry about the SOA TCC are:

- SOA TCC is a resource staffing organization.
On the other hand, SOA TCC provides the strategy, direction and the required foundation for successfully adopting and growing SOA. TCC will enable the rest of the organization by providing adequate structure and processes. TCC may also provide consulting resources but will not staff the project teams.
- SOA TCC is a governing body only.
While TCC is definitely involved in establishing the governance principles and standards, governance is one of the many functions which TCC performs.
- SOA TCC is a profit center.
This assumption is not necessarily true. SOA TCC definitely needs some Enterprise funding to begin with. Later during the Growth and Steady state phases, the size of the organization and the volume of projects may fuel the profits due to charge back model, consulting hours etc. However without adequate funding in the initial phases the TCC may well end up becoming a resource staffing organization.


CONCLUSION

SOA spans across multiple technologies and hence acquiring expertise from different technology areas is critical to the success of the TCC. Executive sponsorship, adequate funding and establishing clear mission and goals are some of the other critical success factors. An organization would have a slim chance of success in SOA adoption without an effective TCC, analogous to a basketball or football team trying to win the championship without a competent coach.

There is no gainsaying the fact that TCC type of organization is required to help the companies successfully adopt and grow SOA. However every organization is unique and the drivers behind SOA adoption also vary. TCC helps if there is an enterprise-wide adoption of SOA. For implementing, say, one or two projects

using SOA does not mandate the need for a TCC. Hence there is a need to carefully evaluate the business drivers, management readiness for funding a TCC organization and the availability of the required skills - either internal or external consultants - to form a TCC before deciding to establish a SOA TCC.

REFERENCES

1. Effective SOA Governance, Governing Service Oriented Architecture, Kerrie Holey, Jim Palistrant and Steve Graham, March 2006
2. Making Sense of SOA Governance, Service Lifecycle Management, Registries and Repositories, Jason Bloomberg, Zapthink paper, March 2006.
3. <http://www-304.ibm.com/jct09002c/isv/soa/> 

Evolutionary Approach to realizing SOA: A Microsoft Platform example

By Ananthalakshmi Vallapuzha and Manish Srivastava

An SOA approach based on business drivers and expected benefits helps in provisioning the right capabilities and strategies for successful realization

Organizations today are increasingly looking at SOA as a key business strategy to building an agile enterprise. However, the drivers for adopting an SOA approach and the benefits that organizations seek from the exercise are quite varied. For instance, a large utilities company wanted to be able to extend business services available on their agent desktop to a customer portal using SOA. A government transport agency used SOA to improve its interaction with partners and customers. A large financial firm wanted a central access point for services distributed across diverse legacy applications [1]. Other reasons often cited for wanting to adopt SOA are, to improve business flexibility and ability to respond to market changes, and better align IT to business. While SOA promises to help on all these fronts, the approach, capabilities and technical infrastructure required to achieve each of these can be quite different.

The drivers for organizations to adopt SOA may be attributed to one or more of three key primary needs:

Information Availability

The IT systems of most organizations are composed of mostly large, independent systems that were built in isolation. These legacy systems are generally not designed for interoperation; rather, they are fine tuned to meet specific functional and operational requirements. Tapping into the data and information encapsulated by these systems is no easy task. However, organizations are increasingly finding the need to integrate their systems in order to streamline processes and better utilize information available in these systems. Previous attempts to integrate systems through ad-hoc point-to-point interactions have resulted in a “spaghetti network” of links that are both difficult to manage and change [2].

Replacing these systems with modern systems architected for integration or reuse is simply not an option considering the business-critical nature of these applications and the enormous investments involved in terms of cost and effort. For instance, a leading utilities company wanted to consolidate and standardize its customer service processes, while still retaining its reliable and stable mainframe backend. The need then is to be able to unlock the information in legacy systems and make them available to other systems across the enterprise in a consistent fashion. Organizations then look to SOA as an option to do this. The utilities company mentioned earlier decided to SOA-enable the legacy backend and consolidate the front-end applications. The driver for SOA adoption in such a case is to enhance interoperability of the systems, to enable integration with other systems, or to address consolidation or reuse requirements.

Flexibility

In a highly competitive environment, the key to differentiation is innovation. Innovation requires agile systems that can help quickly get new ideas implemented and out into the market, ahead of competition. Increasing regulatory compliance requirements also step up the pressure on organizations to put in place mechanisms to ensure that enterprise systems are able to support changes to processes quickly and in a predictable manner. Such flexibility of systems and processes is then another key driver for organizations to adopt SOA. Here the requirement is to improve agility and adaptability of the systems that support the business, and be able to effect changes in business processes or put together new services for consumers quickly. Equally important is the ability to better predict the time to market for a new service or a change.

Manageability

Businesses are often tied down by the limitations of their IT systems. Business processes are run on systems that are defined, supported and managed by IT. Visibility into process performance is inadequate due to various reasons like the right data not being captured, inability to use the application level metrics to derive meaningful process information, data latency issues, inflexible reports amongst other things. While there are SLAs (Service Level Agreements) defined at IT application level, there is no clear means to translate or map these to business SLAs. The combination of low visibility and low control over business processes acts as a roadblock to increased competitiveness. Businesses then find the idea of management units on the lines of business processes composed of services rather than IT applications very attractive. This is the third key driver for organizations to adopt SOA. Here the main requirements are increased visibility and control. Businesses want to be able to define and manage service level agreements at business process and service boundaries, rather than at IT-defined application boundaries, so that they have much better control over the processes supported by IT.

APPROACHES TO SERVICE ORIENTATION

Realizing expected benefits from SOA requires the right kind of capabilities and investments. For instance, just implementing web services will not directly result in improving flexibility of the enterprise systems. Similarly, simply service enabling legacy systems will not result in bridging the business IT divide. It is necessary that the strategy and approach adopted be oriented towards delivering specific benefits.

Enabling Information Availability

The approach here is to tap into information

already available in the existing enterprise applications and build a façade of services over them making them available through a consistent interface that facilitates simple integration.

Enabling availability of information locked in legacy systems requires mechanisms and the necessary technical infrastructure to interface with the legacy systems and help expose the information as services. Tools that can help assess the service-orient-ability of legacy applications and determine the services to be exposed at the appropriate level of granularity and reusability are invaluable. Based on the different kinds of legacy systems supported in the enterprise, technology and application adapters that can integrate with them and help expose their functionality in a standardized format will be required. The architecture and design of certain legacy applications as well as the technology they are built on may constrain the ability to expose the information within. In such cases, it will be necessary to employ engineering techniques such as migration or re-factoring to enable the legacy applications to be service oriented. Tools that help in migration and re-factoring to whatever extent will be very useful. Also required are service creation and description tools that can help simplify creation of the service façade and publish the services in accordance with SOA standards.

It is to be noted that while this approach can help to quickly leverage and integrate legacy systems, the services thus exposed may not necessarily be at the right level of granularity or reuse to be suitable for an enterprise wide use, and may require additional wrappers or changes to match the common information model.

Enabling Flexibility

Enabling enterprise agility or flexibility requires different SOA capabilities. Typically, legacy

The four tenets of SOA

- Boundaries are explicit
- Services are autonomous
- Services share schema and contract, not class
- Compatibility is based upon policy

applications are monolith applications that have business logic embedded in application code. This makes it very difficult to make changes to the processes or introduce new processes. In order to bring in flexibility through SOA, the approach is to transform the application functionality into a set of modular services that can be orchestrated.

To enable true agility, services need to be designed in accordance with good service design principles such that they can be independently developed, maintained and managed.

Additionally, to leverage the full benefit of adopting SOA at an enterprise level, services need to be designed to be reusable across the enterprise. This requires the organization to put together capabilities and mechanisms to identify and define services at the optimal level of granularity and reuse. Common services such as data translation and transformation, message routing, caching, transactions and security are essential, preferably supported at the platform level. There may also be a need for establishing a common information model across the enterprise to enable exchange of common data entities necessary for reuse of services across business functions.

Once well-defined services are available, these can be orchestrated into meaningful business processes. The workflow components need to be modeled as separate entities that can be determined and changed independently from the services that they orchestrate. Process management tools that can help create and

manage processes will be required. Business rules must be modeled separately and managed independently from the business processes through dedicated rules engines and management tools. At this stage, the proliferation of services within the enterprise will necessitate mechanisms for cataloging and categorization of the available services. As more functionality is made available as services and the process of service orienting applications matures in the organization, building a new service or process increasingly becomes a matter of identifying the right set of services from the catalog and orchestrating them using orchestration tools. Not only does this greatly reduce the time and effort required for adding new services or making changes to processes, but also makes the whole service creation process that much more predictable, thus greatly improving the time to market for a new service or change.

Enabling Manageability

The approach for enabling manageability is to facilitate increased visibility and control for business users into the business process performance, and provide mechanisms for enhanced collaboration between business and IT in mapping business needs to IT capabilities. Increased visibility implies making the right information about the business available at the right time through the right channel to business users. This requires not only business service monitoring, business intelligence and reporting capabilities, but also technology that can help deliver the information in near real-time. The information generated thus should be available to business users at anytime on their channel of choice, necessitating multi-channel delivery capabilities. Mechanisms that track the state of the process, help identify deficiencies in the process and take corrective actions are also required for better control.

Moving to enterprise SOA also requires a cultural shift in the way that business and IT work together. There is a need for organizational governance mechanisms and the establishment, enforcement and management of security and business policies. This implies the need for supporting technical infrastructure for advanced security and policy management.

The most important benefit that SOA brings at this stage is the ability to define SLAs at service and business process boundaries, enabling much better levels of corporate performance management. This requires business and IT monitoring capabilities so that both business SLAs and technical SLAs such as availability, reliability and performance can be tracked and managed. There is a need also for mechanisms to enable business users to define SLAs and KPIs (Key Process Indicators), track them through performance dashboards, and initiate and track processes for corrective actions.

Since services effectively model business processes, the flow of data and transactions through service oriented applications can produce valuable business data. What can really take this to the next level of better business-IT alignment is the capability to churn this rich service usage information to identify potential optimization possibilities or untapped business opportunities that can help in business innovation and differentiation.

EVOLUTIONARY LEVELS OF SERVICE ORIENTATION

A strategic move to SOA is a challenging proposition that requires transformation of the organization's IT infrastructure, re-alignment of applications and processes to the SOA paradigm, and establishment of necessary controls and governance mechanisms to manage the organizational shift to SOA. Adopting a big-bang approach for service orientation of

the enterprise can hence be a risky proposition. A recommended approach is a gradual and structured evolutionary adoption that helps realize immediate benefits, while at the same time, laying the foundation for the next level of adoption. This kind of SOA roadmap helps provide incremental benefits along the way, building a stronger business case for greater adoption.

Approaches based on the key business drivers discussed above also correlate quite well to increasing levels of SOA adoption [Fig. 1] and these approaches may be used as stages of realization of enterprise SOA.

Realizing these approaches requires putting together necessary platform capabilities and employing relevant engineering techniques for each of the levels. The platform chosen for such a realization should be flexible and modular enough to support an evolutionary realization by allowing capabilities to be added in stages, rather than a package kind of implementation that requires everything to be built in one go. The Microsoft platform allows a modular build up of capabilities and hence is a good choice for realizing an evolutionary approach. Here, the evolutionary realization is described through an illustrative Microsoft platform example. Note that in this example, some of the products and technologies mentioned are yet to be released. However, these are slated for release in the near future and have been considered in this illustrative example because of their significance in an SOA realization on the MS platform.

Consider a hypothetical telecom organization XYZTel that is planning a long term SOA realization strategy, but has a primary need to improve its call center operations. It performs an assessment of its current enterprise architecture and decides to adopt an evolutionary roadmap to realizing SOA that

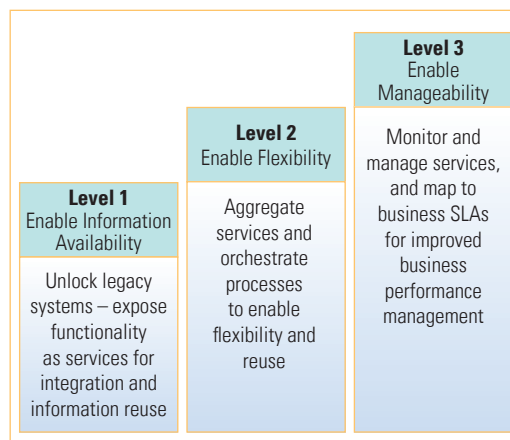


Figure 1: Increasing levels of SOA adoption

Source: Infosys Research

helps address immediate needs while building up to the end state realization. XYZTel has also made a strategic decision to use the MS platform as its platform for SOA realization.

Realizing Information Availability

One of the main issues is that CSRs (Customer Service Representatives) have to individually access multiple applications to handle a single call, impacting productivity and call handling times, and XYZTel wants to change this and make consolidated information available. Since all necessary information is already available in the existing applications, it decides to go the SOA way and build service wrappers around each of these applications, making the information available externally as services. Custom modules will access these services and aggregate the data and deliver consolidated information to the CSR's desktop [Fig. 2].

For its business applications running on IBM mainframes, HIS is chosen to interface with them at application or data level. The Biztalk SAP adapter is used to integrate with its SAP system.

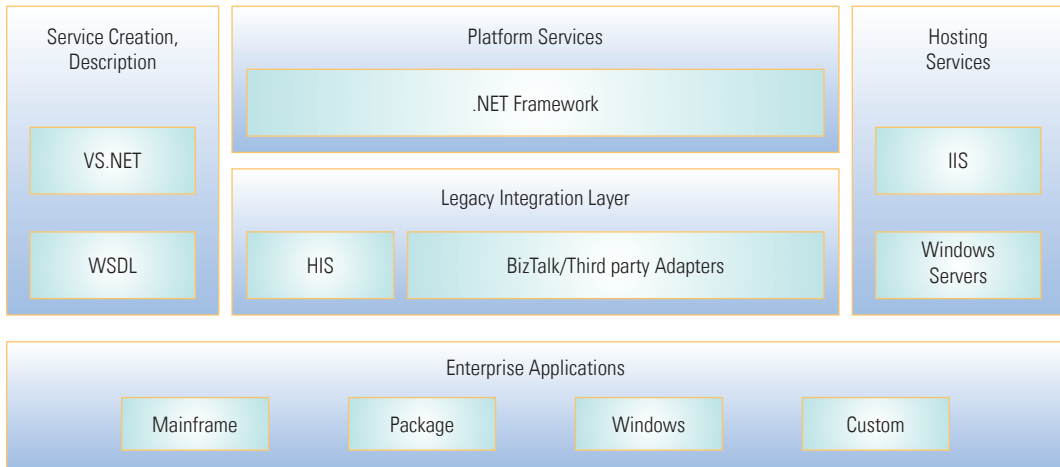


Figure 2: Illustrative MS technology stack for realizing Information Availability **Source:** Infosys Research

Biztalk also provides application adapters for SAP, Siebel, PeopleSoft, Oracle, TIBCO, etc., as well as transport and messaging adapters such as the IBM MQ Series adapter, IBM DB2 adapter and SOAP adapter.

The services are exposed to external applications by building .NET service wrappers. Custom .NET business components access these services and provide consolidated information to the agent desktop.

Realizing Flexibility

Another issue faced by XYZTel is the IT constraints involved in bringing out new services and campaigns. Since these involve changes to application code, process latencies and the time required for implementation and testing result in delay in these services hitting the market, effectively negating the potential benefits of these schemes.

In order to improve the flexibility of its applications, the organization takes up the exercise of modularizing business functionality and modeling them as services, building on the

services exposed from the legacy applications [Fig. 3]. It goes through a detailed process of service identification and definition. To help modularize services on its mainframe based business applications, XYZTel uses third-party tools such as the Enterprise Application Modernization Framework, a mainframe modernization product from Relativity Technologies, to assess service orient-ability of the applications, re-factor code where necessary, and separate out business rules.

WCF is used to provide a service connectivity layer that helps address service security, reliability and transactions, and also supports flexible data formats, transport protocols, and synchronous and asynchronous invocation mechanisms.

Business processes are modeled as workflow that orchestrates the services. XYZTel decides to use Biztalk to create and run these processes with any necessary data transformation and mapping between the coordinating services. Embedded rules are identified and captured separately, and made accessible through the

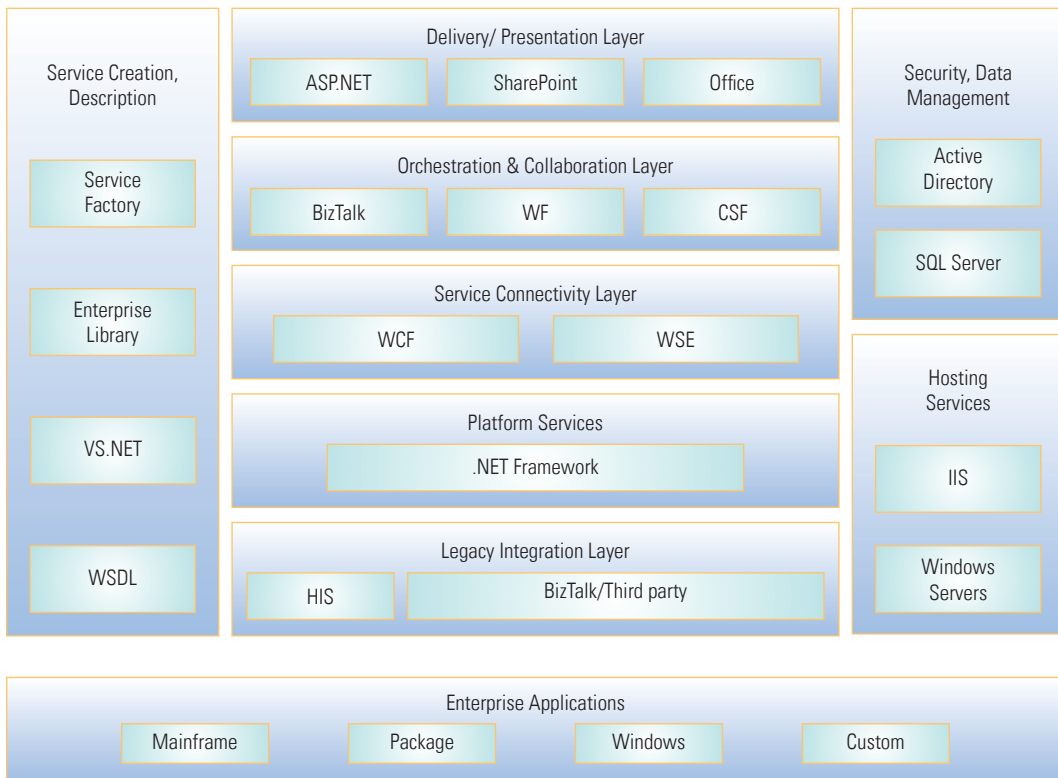


Figure 3: Illustrative MS technology stack for realizing flexibility **Source:** Infosys Research

Biztalk rules engine. With these changes in place, XYZTel finds that rules can be changed easily and independently, business processes changes often require only modifications to the workflow without impacting individual services, and new services can be created and plugged in comparatively easily and quickly.

XYZTel also wants to reduce the load on its call center operations by making some of its services directly available to the customer through a self-service portal. In order to do this, the relevant business processes are themselves exposed as services using Biztalk, and the self-service portal applications call these services directly, making the same functionality available through the portal. XYZTel discovers that it

can now extend the same model to improve information availability to its field agents on their mobile devices, helping them provide better customer service.

Realizing Manageability

XYZTel goes the next step and implements service monitoring and management mechanisms [Fig. 4]. It uses Microsoft Operations Manager (MOM) 2005 in conjunction with other third party partner products that provide both operational (performance, availability) as well as business services monitoring support for service management. It finds that analyzing the process tracking data obtained from Biztalk helps provide much better visibility into operations,

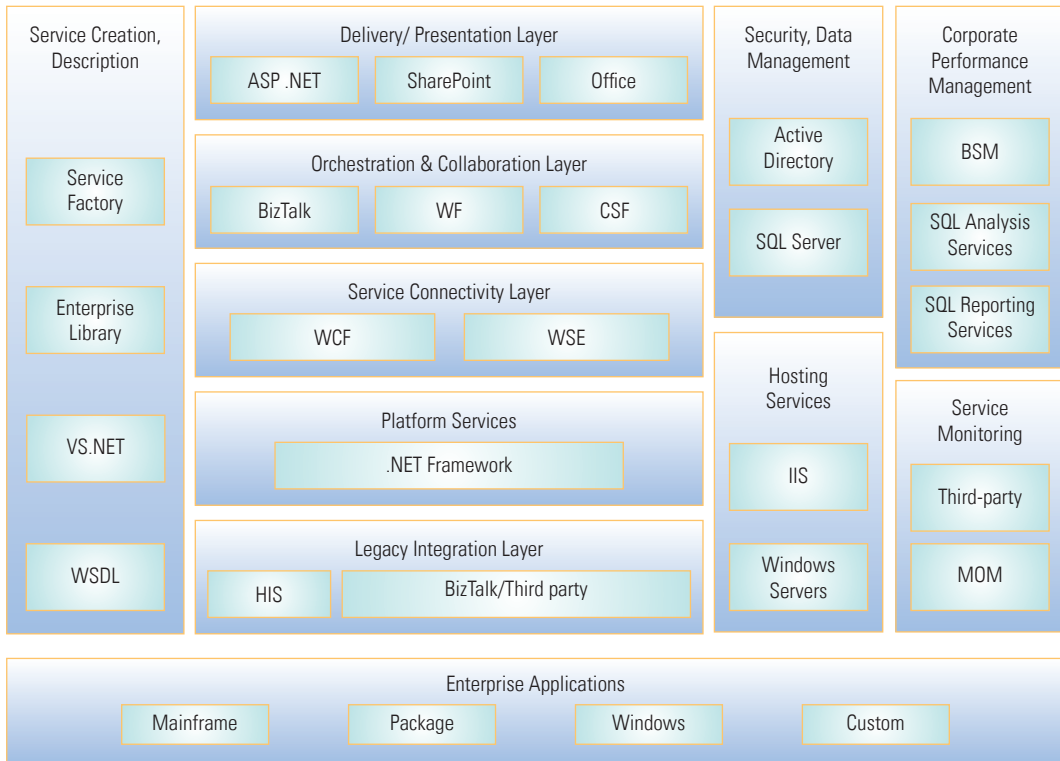


Figure 4: Illustrative MS technology stack for realizing manageability **Source:** Infosys Research

helping better monitor agent productivity and identify process optimization possibilities. With the Biztalk Business Activity Monitoring (BAM) it is also able to easily track the statuses of all processes, quickly identifying delays or bottlenecks. Using SQL Reporting and Analysis services to build in business intelligence features to analyze the tracking data, XYZTel is also able to provide a richer and near real time view of operations to its senior management, enabling them to make better informed and quicker decisions.

As the circle of adoption increases and the number of services increase, XYZTel uses CSF identity management capability to manage identities where processes cross business

function boundaries. It additionally taps into CSF collaboration features such as sharing of customer profiles across services to provide value-added benefits to customers and exploit cross-sell opportunities.

Key processes are increasingly composed of services, some of which are themselves processes, and XYZTel is able to manage business SLAs much better by assigning and managing SLAs at the composite services level. XYZTel implements an enterprise business score-carding process using the Business Scorecard Manager tool from Microsoft. This helps quickly identify and drill down to where SLAs have been breached and take corrective action. It creates a Sharepoint

Drivers	Approach	Platform Capabilities required	Supporting Technologies (for a MS platform realization)
Information Availability	Service enable legacy systems and expose functionality as services for external access.	<ul style="list-style-type: none"> Technology and application adapters SOA assessment and migration tools Messaging capability 	<ul style="list-style-type: none"> HIS Biztalk Adapters MSMQ, MQ Series Relativity, SEEC
Flexibility	Process-orient applications by orchestrating modular services. Separate business rules.	<ul style="list-style-type: none"> Ability to identify and design modular and reusable services Establishment of Common Information Model Support for data translation and transformation, message routing, caching, transactions, security and other platform services Workflow modeling capabilities and process engine Rules engine Multi-channel delivery 	<ul style="list-style-type: none"> Biztalk orchestration and rules engine Windows Communication foundation Workflow Foundation ASP.NET Windows Sharepoint technologies Connected Services Framework MS Office
Manageability	Use service monitoring to enhance process performance visibility. Enable process level SLAs and performance management	<ul style="list-style-type: none"> Service monitoring and management Service collaboration Business intelligence and performance management tools Policy management and governance SLA definition and management tools 	<ul style="list-style-type: none"> SQL Server Reporting and Analysis services Active Directory Connected Services Framework Microsoft Operations Management Partner Service Management products

Table 1: Snapshot of adoption strategies and capabilities **Source:** Infosys Research

collaboration environment for business users that provides a one-stop, personalized view of reports, operational dashboard and business scorecard. It additionally uses this to build in a closed loop system of analyzing, actionizing, tracking and measurement of business SLAs, thus greatly improving corporate performance manageability.

XYZTel has thus realized enterprise SOA by addressing it in evolutionary steps, instead of a big-bang approach, while at the same time realizing targeted benefits at each step.


CONCLUSION

SOA is a buzzword today and many organizations are in the race to adopt SOA. However, it is important to ensure that the right approach is selected and the right capabilities are provisioned to ensure successful realization. Here, we recommend selecting an approach based on the primary business drivers for adoption and expected benefits.

A snapshot of the approach based on the driver, the platform capabilities required and the technologies that can provide those capabilities is provided [Table 1].

We also recommend adopting an evolutionary approach to SOA rather than a big-bang approach. The approaches described can be leveraged for such an evolutionary adoption.

REFERENCES

1. Real-World SOA: SOA Lessons Learned, Randy Heffer, Forrester, September 2005
 2. Applied SOA: Conquering IT Complexity Through Software Architecture, Yefim V. Natis, Gartner, May 2005
 3. Your Paths To Service-Oriented Architecture, Randy Heffner, Forrester, December 2004
 4. What WCF brings to SOA, John DeVadoss, http://searchwebservices.techtarget.com/qna/0,289202,sid26_gci1159627,00.html, January 2006
 5. Your Strategic SOA Platform Vision, Randy Heffner, Forrester, March 2005 
-

SOA : Saving Grace for Legacy Applications

By Anubhuti Bharill, Biji Nair and Binooj Purayath

Piloting a service oriented legacy modernization effort first, can shorten the path to success

As one of the core benefits, Service Orientation envisions enhancing the integration of existing, heterogeneous systems in a flexible, collaborative manner. A significant portion of the systems that should participate in Service Oriented (SO) Integration were developed in a computing era dominated by mainframes. These systems, termed as legacy applications in today's parlance, encode critical business rules and according to current estimate, comprise of more than 250 billion code lines and it is increasing [1]. Naturally, enterprises continue to maintain their significant investment in legacy systems and strive to reuse their critical functionalities. Hence, any credible service orientation strategy must address the essential reality of enterprise IT landscape dotted with numerous legacy systems.

It is worthwhile to consider a detailed view of an enterprise initiative aimed at demonstrating the business value of service enablement. Even before embarking on the big program, the organization we look at makes a small investment to assess the risks and acquire

sufficient learning that can be applied when the program is finally underway. Concurrently, the initiative targets at bootstrapping a technical environment, say a Sandbox, and experiments on the risky aspects of modernization.

THE CHALLENGES OF LEGACY INTEGRATION

The idea of integrating legacy systems' functionalities often presents many cross cutting concerns and challenges. Starting from the task of defining an explicit business case for service orientation to the finer details of implementing system level transaction demarcations for service calls - many such complex situations present themselves that need careful planning and execution.

One of the challenges of legacy integration arises from a mismatch between the essential characteristics of legacy systems and the expected behavior of participating systems in a collaborative framework [2]. For example, a monolithic legacy implementation with limited modularization may not meet the more

fine-grained, service oriented behavior typical of participating systems in a service oriented environment.

As another example, flexibility of a Service Oriented Architecture comes from related but slightly varying functionalities offered by the service providers. While it is common to find such polymorphic behavior in modern object-oriented systems, legacy systems are often not designed for such features, so realizing similar functionalities can be costly both in terms of time and effort.

process automation enabled through SO Integration.

- Availability of accurate and real-time data - eliminate the inconsistencies and redundancies between back office systems and customer channels.

One of the modernization options was to harvest business services from the core enterprise applications and adopt SO Integration while continuously aligning with the Future State Architecture (FSA). The

Typically organizations, as in this case, embark on legacy modernization initiatives when faced with rapid business growth and to balance strategic needs such as business agility and reduced time to market

THE ORGANISATION UNDER STUDY

The organization under study is a leading global bank that had envisioned to embark on a legacy modernization initiative to align with the expected rapid business growth and the strategic needs such as business agility and reduced time to market. As is common to most firms, some of the key business drivers of the modernization were:

- Agility and Speed to Market - introduce new products in a reduced timeframe.
- Higher customer satisfaction by improved operational efficiency - result of straight through processing and

objective of the initial effort, discussed in detail here, was to validate this modernization option, more precisely

- Evaluate the feasibility of SO Integration with a small investment, before deciding on making a bigger investment.
- Establish the business value of service oriented legacy modernization.
- Establish the scale down version of technical environment to bootstrap the major activities.
- Establish the technical feasibility of the modernization.

SPECIFIC CHALLENGES IN CONTEXT

There were three specific challenges in this organization. One had to do with the mindset of the people involved, another had to do with the knowledge about the applications under consideration and the third had to do with managing the transition into the new environment

The legacy of the legacy: Majority of the IT staff were of the opinion that a replacement, which was costly, was the only option available for the legacy modernization. The business stakeholders shied away from costs and preferred to continue on the existing platform, even while acknowledging the

Introducing change into well established

ecosystem: Introducing unfamiliar concepts of modernization and SOA to an enterprise offered considerable challenges. The task of establishing the integration environment (sandbox) exposed many of these issues; many driven from the size of the organization and existing departmental structures. As an example, collaborating with an array of internal technology organizations, external vendors and philosophically differing developers presented unforeseen challenges in communication, planning and execution of the development activities.

Collaboration in legacy modernization initiatives remain a major issue - development activities run into unforeseen hurdles originating from the large number of internal technology organizations and external vendors

system limitations and its impact on day-to-day business. These ideas of system replacement and forbidden cost had been urban legends for more than 7 years. Due to this, the stakeholders were of mixed opinion regarding the expected success of the modernization.

Knowledge about the legacy: The core legacy applications were more than a decade old, adopted from a product vendor, and had grown organically. The firm wanted to harvest the business knowledge from existing code rather than depending on the product vendor (with whom the firm wanted to relinquish the relationship). Additionally the documentation available was not so up-to-date.

ASCERTAINING THE COMPLEXITY AND SUCCESS OF MODERNIZATION

This initial effort was envisioned to provide an understanding around a set of influential factors based on which the success of the modernization depended. Some of these factors were

- Integration with a state of the art Web Interface/Dashboard: Is it viable to integrate with an intuitive user interface that can enhance user productivity and reduce the time for transaction processing?
- Code/Asset reusability & introduction: How much of the existing software assets

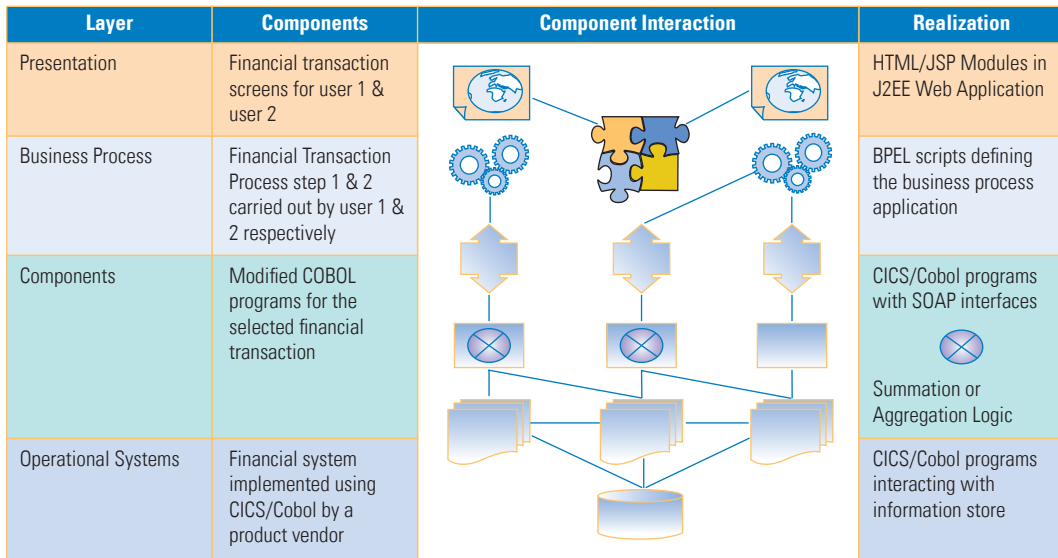


Figure 1: The Component Layering in the To-Be Architecture

Source: Infosys Research

and code elements can be reused without modification and how much needs to be introduced to achieve the service oriented integration?

- Stateless nature of services: Considering the spaghetti implementation of user interfaces, workflow and business processing logic, how much of stateless principles can be adhered to?
- Separation of human workflow and processing flow from the core processing steps: How to achieve the intended separation of the above concerns so that those can be re-implemented on a flexible, standard based Business Process Management platform?
- Usage of standardized web service technologies: Validate the existing support of Web Services standards such as WSDL, SOAP, XML and so on, on the legacy environment.

- Loose coupling: The ability to layer the existing code to different layers of concerns so that they are independent, but interoperable. Such concerns are presentation/user interfaces, navigation of user interfaces, human workflow, business processing flow, core business processing steps and data store access.
- Service granularity and aggregation: How to decide the right level of service granularity to be exposed in the legacy system; how and where to deploy the aggregation (finer to coarse granularly) logic? Options were i) using an orchestration/choreography layer outside legacy system ii) use legacy constructs to create the coarse grained wiring.
- Impact on the presentation layer or user experience due to separation of concerns: The intended separation

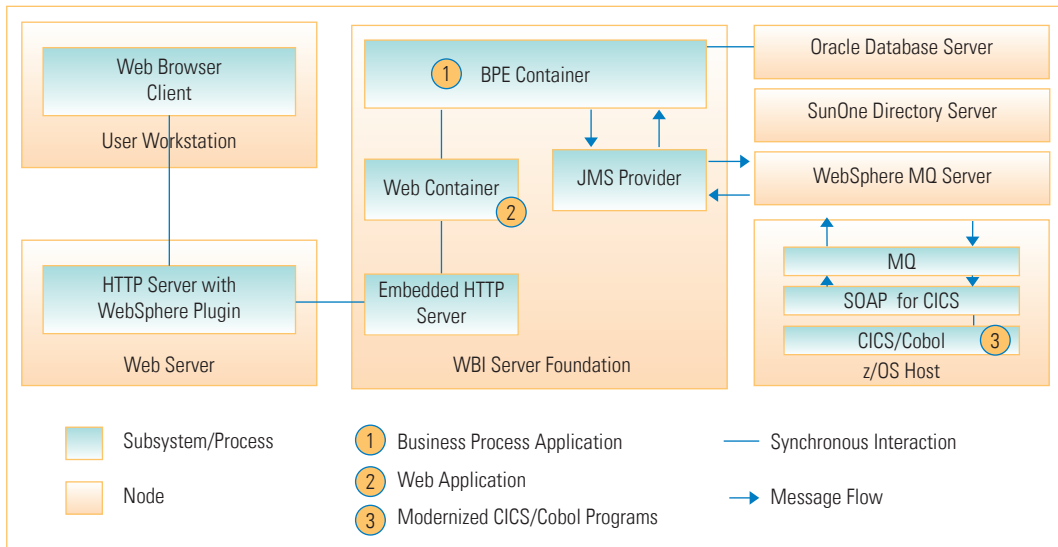


Figure 2: The Physical Infrastructure in the To-Be Architecture

Source: Infosys Research

of user interfaces and other concerns were expected to introduce some limitations/differences [than existing] to the user interaction, such as deciding on persistence points in the human workflow. What are some of the limitations/differences that a separation of concern may bring along?

A TWO-PHASED PILOT APPROACH

This exercise was conducted in two phases.

Phase 1- Planning included the following steps:

- Defined a set of architecture aspects that need to be proved based on the factors in deciding the value of modernization.
- Defined & designed i) a set of Web based UI as a prototype of a Web Dashboard, ii) a set of service interfaces/contracts iii) Logical and Physical Architecture (Figs. 1 and 2)

- Identified a candidate set of code assets and performed an analysis on the above for understanding the impact of future adherence to the SOA standards [3].

Phase 2 - Execution included the following steps:

- Established a sandbox - a scaled down version of physical architectures, established the system interfaces, and deployed the service oriented components in the runtime environment.
- Analyzed and validated the influential factors in assessing the complexity of FSA and SO Integration.
- Documented the learning - both technical and operational, so that it can become version of physical architectures, established the system interfaces, and deployed the service oriented components in the runtime environment.

- Analyzed and validated the influential factors in assessing the complexity of FSA and SO Integration
- Documented the learning – both technical and operational, so that it can become helpful in planning for the future initiatives.
- Defined a legacy modernization costing framework based on the effort numbers, influential factors and benchmarks, so that future financial decisions around modernization can be driven in a systematic way.

TECHNOLOGY SOLUTION

The solution essentially included three different elements – modernized CICS/COBOL programs

using J2EE, Struts and Web Services.

The Development Tools: Developed the components using NetManage RUMBA, WebSphere Studio (Integration and Enterprise Editions) and WebSphere BI Modeler.

THE OUTCOME

Large modernization initiatives often aspire, but struggle to define an effective cost benefit analysis framework to provide visibility into the payoff on the intended IT investment. Historically, such upfront analysis activities often results in avoiding IT disappointments [4]. Such payoff analysis can help in making prioritized and value driven decisions. This analysis need not be a pure spreadsheet based one. Additionally, by demonstrating the prototypes, explaining

An elaborate cost-benefit analysis is typically a chimera for most modernization initiatives, notwithstanding the fact that such analyses help convince stakeholders

as the service provider, business process application as the service broker & composer and Web based UI application as the service consumer. The technologies used at each of these elements are provided below.

The Service Provider: IBM Mainframe, Z/OS, CICS, DB2, VSAM, COBOL, SOAP for CICS and Web Services API from Google.

The Service Broker & Composer: Web Sphere Business Integration Server Foundation and MQ, J2EE, BPEL4WS and Web Services.

The Service Consumer: The next generation web dashboard for the back office operator

the concepts and the benefits, the business community often gets a better ‘touch and feel,’ which helps them be more comfortable and convinced on the potential IT investment.

In this situation, the newly constructed, intuitive and powerful user interfaces acted as the primary vehicle that could provide the real feel of the benefit of the modernization. When the real business information appeared on the UI after a data fetch from the backend legacy system, it provided a discussion framework to generate interesting ideas and opportunities that could make the business operations more efficient and cost effective.

As an example, to demonstrate the ability to seamlessly integrate with third party systems, this implementation included integration with Google search through their Web Services APIs [5]. Through this integration, the application was able to display Google search results based on a specific customer name, next to the customer business data retrieved from the data store. In reality, for the actual business operations, the above search results might not have had any significance. Still, this demonstration opened doors for contextually significant discussions on potential integration scenarios, with many internal and external systems. For many stakeholders, especially non-technologists, these discussions provided

target technology platform. As an example, Business Process Management capabilities of IBM WBI Server Foundation (Version 5) had many limitations such as in creating sub-processes, integrating and invoking web services in the legacy platform and so on. From that context, this exercise also acted as a technical proof of concept, identifying the platform limitations, validating the product roadmaps from the vendor and more importantly, identifying risk elements of the future state architecture.

The experiences from the execution this initiative provided significant learning in the program management front too. As an example, to make the overall program

Service oriented integration offers a viable alternative for organizations looking for a workaround when faced with multiple uncertainties while modernizing

a quick validation of the potential benefits from the architecture in consideration.

On the technical front, many of the influential factors identified in the planning stage were analyzed such as code/asset reusability and introduction, stateless nature of services, workflow separation, level of loose coupling, service oriented integration and aggregation, and usage of standardized web service technologies on the legacy system. Out of this analysis, a learning documentation was created with the intention of fine tuning the modernization strategy.

This exercise also exposed technical limitations on many of the elements on the


a successful one, many more internal and external organizations participated than initially anticipated. According to this learning, the team fine tuned the program management practices that should be applied for the overall modernization program.

Overall, this initiative demonstrated the technical viability and the business benefits, such as agility and ability for integration. After this, various stakeholders, including the business team, became confident on the modernization strategy. It provided them the confidence to move into the next set of activities to modernize the rest of the applications, which constitutes 90% of the overall legacy system.

CONCLUSION

Historically, many organizations avoided or delayed modernization decisions because of uncertainties around these initiatives. SO Integration provides a low risk option in modernizing systems relative to other options such as rewriting to a newer platform. It helps to leverage existing system investments while concurrently deploying newer technologies. SO Integration thus helps organizations reconsider their abandoned or delayed modernization plans.

REFERENCES

1. COBOL for the 21st Century by Nancy Stern, Robert A. Stern, James P. Ley, John Wiley & Sons, Aug 2005
 2. Legacy Systems: Transformation Strategies by William M. Ulrich, Prentice Hall PTR, Jun 2002
 3. Options Analysis for Reengineering (OAR): A Method for Mining Legacy Assets by John Bergey, Liam O'Brien, Dennis Smith, Software Engineering Institute, Carnegie Mellon University, <http://www.sei.cmu.edu/publications/documents/01.reports/01tn013.html>
 4. Making Technology Investments Profitable: ROI Roadmap to Better Business Cases by Jack M. Keen, Bonnie Digrius, John Wiley & Sons Inc, Nov 2002
 5. Google SOAP Search API (beta) <http://www.google.com/apis/> 
-

“SOA is an architecture meant to handle change”

Ronald Schmelzer, founder of ZapThink, in conversation with Srinivas Padmanabhuni, emphasized that the key value of SOA lies in its ability to address change without re-coding and reconfiguration efforts

Q. *As a keen observer of enterprise computing trends, what do you perceive as the core difference of Services versus previous models of distributed computing like objects or component?*

Ron: SOA represents an evolution of architectures for handling complex, distributed computing, but significantly different from other approaches. It puts an abstraction layer over an implementation. Due to this, SOA doesn't focus on how you build or run Services. The implementation in turn can be an object, a component, a database item or a legacy code module. Further, SOA relies on metadata for configuration, assembly and composition.

In SOA, change is the primary issue addressed. SOA allows changes in business processes, policies, rules, and composition of different IT assets without significant redevelopment or

recoding effort. Sitting on the top of IT layer, this heterogeneous abstracted layer enables loosely-coupled interactions, and thus enables companies to achieve significant agility in the face of complexity and heterogeneity.

Q. *You mentioned about composition of different services. Does that fall under SOA or BPM (Business Process Management)?*

Ron: Before SOA arrived on the scene, for composition of applications, two technologies were available - Integration using EAI and middleware, and BPM. BPM allows business processes to be defined and composed. SOA has effectively removed the need for conventional EAI/middleware approach to standardize integration. BPM and SOA are two sides of the same coin. BPM tooling can be used to compose

services. While SOA is a methodology, BPM is a technology.

Q. From a trend perspective, what do you think is the level of maturity of SOA in enterprises today?

Ron: Most enterprises are not quite implementing SOA yet, but are implementing ABOS. It's very simple to expose a service, make it implemented in java, C#, or on a mainframe or a database. However, they are realizing that it is harder to build an SOA. SOA is architecture, not implementation, and this requires building an architecture oriented towards services. Building an enterprise SOA is a difficult task, as there is a need to deal with security, management, composition, reliability, and, most importantly, change management. Many companies are just moving from OO programming to SOA, but this doesn't translate well. Early, successful adopters of SOA-based enterprise architecture have already benefited from using registry, governance framework, and loosely coupled composition logic. However, majority of enterprises still have an integration-centric mentality for SOA.

Q. You are a big supporter of service oriented architecture and its implications for business flexibility. What are the main principles in SOA systems architecting and designing which in your view will help derive maximum business flexibility?

Ron: Key architectural practices for successful SOA include:

A) **Contract first development:** In SOA, what is to be built is the service abstraction and not its implementation. The idea is to reuse the interface without recreation of business logic. It is important to define the contract of a service provider in detail including metadata related to operational

and non-functional requirements like security, reliability, and define even specific service consumer requirements sometimes.

B) **Top-down approach:** In this, business processes are analyzed and services of right granularity and level of functionality are identified, instead of bottom up approach which analyzes code modules and exposes them as services.

C) **Combining MDA with Agile Methodologies:** We suggest a combination involving an agile approach first, for rapidly translating business requirements to the service contract, followed by an MDA based approach to convert the contract to service implementations. The missing part today is effective tooling for this, however some right tools like those capable of executable UML, are emerging.

Q. Service identification is a major pain point in SOA and still remains an art. What are your thoughts on bringing science and methodology to service identification?

Ron: Service identification depends a lot on the system. It's the responsibility of the architect to identify which services to build and what granularity and functionality they should contain. This depends on a) frequency with which business process changes, b) how much the business processes should be allowed to change, c) the level of reuse in enterprise etc. Typical requirement for a service is to capture those activities/processes that allow change without reconfiguration. Those that require most configuration changes need to be constantly re-factored. Tools might help an architect get reports on these dimensions which

can streamline the decision on the right set of services.

Q. Any prescriptions or guidelines for service identification?

Ron: We can view the approaches to this at different levels:

A) Departmental level

At a departmental level the aim is to reduce cost of integration. Ideal candidates for services are those components with fixed cost of maintenance cost etc. A bottom up approach with a systems analysis needs to be done with proper identification of opportunities for reducing change costs.

B) Strategic level

At a strategic level, SOA is an architecture meant to handle change. As a result, in SOA you are never done. The aim in SOA is to build reusable and composable services with the target of achieving reconfigurability and reducing redundancy and time taken to build new systems. Some concepts which can help achieve these are:

i) Service domain exercise: The identification methodology should identify service domains, which refer to a group of like services, and common across multiple processes. The idea in a service domain is to rationalize the services based on commonality. Typically many like services may have upto 80% common and reusable component with 20% variation. Hence the service domain exercise is a good approach to yield services of right granularity.

ii) Top down approach: Use a top-down approach to define services. The goal should be to achieve business agility. It should start from analysis of business requirements with a rough definition of the business process. At this stage all details need not be known about the process. Later, with inputs from process specialist, the process should be specialized and rationalized to identify a pile of services. We should do a

service rationalization as well so that it would give a nice collection of services at the end. Then these services should be composed in order to build the process.

Q. Yet another pain point in SOA is the holy grail of designing a service of ideal granularity. You may design a highly useful coarse grained business service, but if it is too coarse grained it may lose on the reuse potential. What in your opinion are key considerations when it comes to designing services of right granularity?

Ron: Simply using a single process as an atomic service does not provide agility. Agility is determined by a service's applicability in multiple, different business process scenarios. This might mean that an architect would have to refactor and decompose the atomic services into finer grained services to achieve agility, or vice-versa, compose fine-grained services into more coarse-grained to achieve the right level of loose-coupling. Likewise, if a service wrapper is put over a big piece of code, it would not provide the right granularity. As already mentioned, there might be many processes composed from like components with just 20-30% difference. Hence, as part of the service domain exercise, the services of the common part should be finely developed in order to enable sharing between processes. This coupled with constant refactoring of the processes ensures the handling of both the generic and specific components of the process while achieving agility.

Q. The twin process of deriving services from a top down approach and harvesting services by bottom up analysis of application portfolio leads to a mismatch of granularity of services. How shall we overcome this gap?

Ron: It is the role of skilled architects to know how to match services of different levels of granularity.

Another concept is that semantics of data creates some of this gap. Differences in the meaning of data become a real issue. That would necessitate a value-added intermediary to transform data, to massage the information, and to abstract the complexity from different consumers. In addition it is important to constantly analyze the service portfolio and refactor services so that granularity is maintained.

Q. What are your thoughts on software architects and developers effectively carrying on contract first development in SOA?

Ron: The real issue is that designing services and developing them are two different roles which need different expertise. It's like designing a house and building a house where each role needs different expertise. In general, this expertise would be found in different people. Our recommendation is that architect should take control of all the design part of the contract in the services and must control the implementation, and the developer should try to faithfully build to the contract. Architect should make sure that the contract has captured all the changes in business requirements that are going to affect the business processes in near future.

Q. From an architect's perspective, what are the different tools which can streamline SOA systems architecting?

Ron: For an SOA architect, schema based tooling and metadata tooling are very important. Due to differences in schemas, and changes which can happen in due course, change management becomes an important element in SOA. Thus it is important to have tools for manipulation of and visibility into schemas (e.g. Contivo etc.). Simply doing development using IDEs like eclipse does not help in SOA.

Likewise, it is important to have tools to enable management of, and visibility into, metadata. The metadata registry also assumes importance as it enabled run-time binding of consumers with providers depending upon consumer requirements and provider constraints.

Overall, however, today enterprises invest in loosely-coupled services management (like WSM tools) tools in addition to investing in change management. It is important that the enterprises invest in tools for managing metadata and changes in metadata, contracts and policies, schemas and other artifacts.

Other tools that are for essential for SOA architects are those that allow composition of services, enterprise service bus or message-oriented service infrastructure for doing service integration and build /runtime environment to run services.

Q. In your book "Service Orient or be Doomed", and numerous other articles, you have stressed a lot on SOA governance. What are your thoughts on SOA governance?

Ron: SOA governance is a critical necessity for SOA development. For example, if a version change occurs in a service, how would you determine what systems are affected? How do you make sure that all versions are kept in a way that enable backtracking? Governance, which gives people control to do possible changes in policies, should be restricted to expert hands. Else, anyone can make unwanted changes, adversely affecting services.

Governance in SOA is basically about: enabling policies, enforcing policies, visibility into compliance, and mitigation of exceptions. This requires a coherent approach from the people, process and technology dimensions.

Web Services Management tools which have been used till date in SOA deployments, provide visibility after an exception occurs and hence incomplete from a governance point of view. In fact these WSM tools can be viewed as stop gap arrangement for SOA governance before generic IT management policies and processes can be applied to SOA governance.

Erstwhile formal approaches to governance based on Enterprise Architecture (EA) methods have been weak. But SOA definitely appears to be a practical and potentially realistic version of EA.

Q. In that connection, you have written about the interplay between ITIL and SOA. Can you explain?

Ron: ITIL is the standard for IT Service Management (ITSM), for managing the most important services that IT provides to the rest of an organization. ITIL has well-documented processes, procedures, and approaches for dealing with the most common business needs of IT.

In fact, there is a convergence of the notions of “service” as ITIL would cater to in ITSM with the notion as in SOA. In that context, SOA and ITIL can learn a lot from each other. On one hand services developed under ITIL should be able to be represented in a Service-oriented manner, leveraging the abstraction, loose-coupling, and composability capabilities of SOA for meeting ITIL needs. Likewise, SOA architects can borrow some of the artifacts and documents created for ITIL and repurpose them to meet SOA-specific needs. Thus, practices for SOA and IT processes overlap with each other, instead of containing each other.

Q. Can you comment on reuse and ROI in SOA?

Ron: Returns from SOA require ongoing commitment, since returns might be over

multiple projects. Hence, it's important to show industry the ROI of SOA specially in terms of visibility and agility. It's not easy though to predict returns on reuse. SOA gives long term return, because high investment is required to create SOA at an enterprise scale. It's important to maintain and monitor the matrix associated with services, create service models and manage economics related to them.

Q. You have predicted the rise of Service Consumer for SOA in 2006. In that context, how do you think SOA and the current trends in Web 2.0 play out. What are the interrelation between each?

Ron: Today, collaborative, web-based mashups are popular, with exchange of information in a loosely-coupled manner. If you look at Flickr, Google Maps etc, the ways to exchange ideas and data are changing. They are loosely coupled and developed in large scale. AJAX and similar technologies are being used for building Rich Internet Applications to make service consumption a reality among common people. Now, logic is residing on client side than server side. Certainly, this trend would finally push portal software away, which have been used conventionally to create mashups of services from different sources.

Conventional view is that SOA can provide back-end integration point for Web 2.0 based rich front-ends. However, we believe in power of Web 2.0 to extend the power of SOA beyond this. Web 2.0 at service consumer end will enable composition of services at front-end. This may indirectly create a reduction in need for portal servers, which were conventionally used to do this. Further, it will enable unforeseen and additional ways of leveraging the same service. Hence Web 2.0 has the potential of extending SOA to front-end applications.


Q. ESB is a major buzzword today. What is your take on ESB? And what do you feel is the role of ESB in SOA infrastructure today?

Ron: Most of the companies already have some form of ESB infrastructure. However, most companies feel they need to buy new ESBs to enable SOA in their enterprise. That's not exactly correct. Often, there is no need to buy another one. What they more often need to invest in is metadata management tools, which capture changes in policies, contracts etc. required for maintenance of services.

Q. What is your advice to system integrators like Infosys regarding SOA?

Ron: Responsibilities are even bigger for you. You have been implementing e-business for a long time. The SOA opportunity is bigger

than to e-business movement. You should be having a lot of expertise in designing as well as implementing e-business systems. Now, you should offer expertise in SOA with repeatable methodologies for defining service contract, policies, composition logic etc.

Ronald Schmelzer, senior analyst and founder of the analyst firm ZapThink, is a well-known expert in the field of Service-Oriented Architecture (SOA), Web Services, and XML-based standards. Ron has been featured in and has written for periodicals, and has spoken at numerous industry conferences and in front of some of the largest businesses in the world. Ron Schmelzer was the lead author of XML And Web Services Unleashed (SAMS 2002) as well as co-author of Service-Orient or Be Doomed (Wiley 2006) with Jason Bloomberg, released in 2006 

Improve Integration Efficiency with Semantic SOA

By Bhavin Raichura and Shaurabh Bharti

Empower the SOA implementation with Dynamic Integration and Runtime Process Orchestration

Business integration problems are still a challenge for most enterprises today and they are struggling to find a simple, reusable and cost effective solution. The solution to the integration problem has to deal with multiple technologies, data formats and application connections. The Service Oriented Architecture (SOA) goals of higher visibility, accessibility, and interoperability enable enterprises to develop standards-based integration solutions for information sharing, information aggregation, collaboration and mediation across systems and technologies.

In the recent Forrester's Business Technographics® survey, it is reported that a large number of enterprises have started adopting SOA tactically to address the internal, external and multi-channel integration problems. Many small and medium enterprises are also adopting SOA in a strategic way to achieve business transformation goals. As SOA implementation becomes mature in enterprises, the demand for information interoperability, process automation and enterprise reuse increases to sustain

innovative business models. Over a period of time, the SOA and XML based integration solution will be challenged for Automatic Service Discovery, Dynamic Information Exchange and Runtime Process Orchestration capabilities, which are not fully addressed in the current state of SOA.

The abovementioned shortcomings come into fore, especially in context of integration solutions requiring high level of dynamism and automation. In what follows, these specific shortcomings shall be detailed.

SOA SHORTCOMINGS IN CONTEXT OF DYNAMISM AND AUTOMATION

The key shortcomings of SOA implementations within the context of automation and dynamism are listed below.

Point-to-Point Data Transformation Mappings

Traditionally SOA implementations use XML and XSLT to achieve data interoperability. The XML and XSLT approach forces point-to-point

mappings between single-source and single-target systems. If an enterprise has N data formats, it results into N² - N XSLT mappings. Any change in the data formats will force changes in multiple mappings. Increase in the number of data formats also increases the number of XSLT mappings, which adds to maintenance issues in the enterprise. In many-to-many information exchange scenarios, it becomes very complex to process these mappings dynamically. It restricts dynamic information exchange capabilities while implementing SOA.

The static and syntactic XSLT scripts and mappings violate loose-coupling: a basic principle of SOA. The point-to-point integration schemes are brittle, less scalable and inflexible to resolve integration issues for complex enterprise scenarios.

Standard XML Schema Based Integration

The traditional SOA solution relies on standardization of XML schemas (Rosetta Net etc) to address the issue of multiple data formats used by different enterprises. The standard schemas are difficult to implement globally across all the enterprises. Significant effort is required to agree on “how-to use standards” between the enterprises. Besides agreeing on which standards to use, enterprises also have to agree on specifics of message contents, message structures, message sequencing and service execution constraints. Thus, the interoperability issue remains a critical challenge for external integration.

Syntactic Data Interoperability

XML schema enforces only the syntax not the meaning, constraints, capabilities and usage. As systems mature, the demand for information interoperability, automation and intelligence increases. The sophisticated decision making

process requires “knowledge” across multiple systems and multiple information domains. This crafts a solution requirement to know the meaning of the data at runtime for dynamic information exchange. Since the XML schema enforce only the syntax, it is unable to meet the dynamic and automated information interoperability with “automation” and “knowledge”.

Static Process Orchestration

With wider implementation of SOA, there is a strong call for automatic service selection, dynamic service mediation and automatic process orchestration to realize additional business-value of SOA. The current scope of traditional SOA description in WSDL, and UDDI is limited to representing the capability and properties of a service. The traditional SOA approach facilitates static process orchestration model which depends on fixed rules and fixed services. It works well with the traditional business models which assume fixed business processes and fixed partners. But, the futuristic business models will demand Automatic Partner Selection and Runtime Process Orchestration with Automated Service Negotiation capabilities, which in-turn demand a dynamic SOA implementation.

The aforementioned shortcomings demand dynamic capabilities in SOA to support the futuristic business models. These capabilities in-turn translate into Automatic Service Discovery, Dynamic Information Exchange and Runtime Process Orchestration solution features.

THE SAVIOUR: SEMANTIC SOA

Semantic SOA is a standards-based, ontology-enabled, constraint-driven approach that enables automatic service discovery and selection, dynamic information exchange and runtime process orchestration correction and optimization.

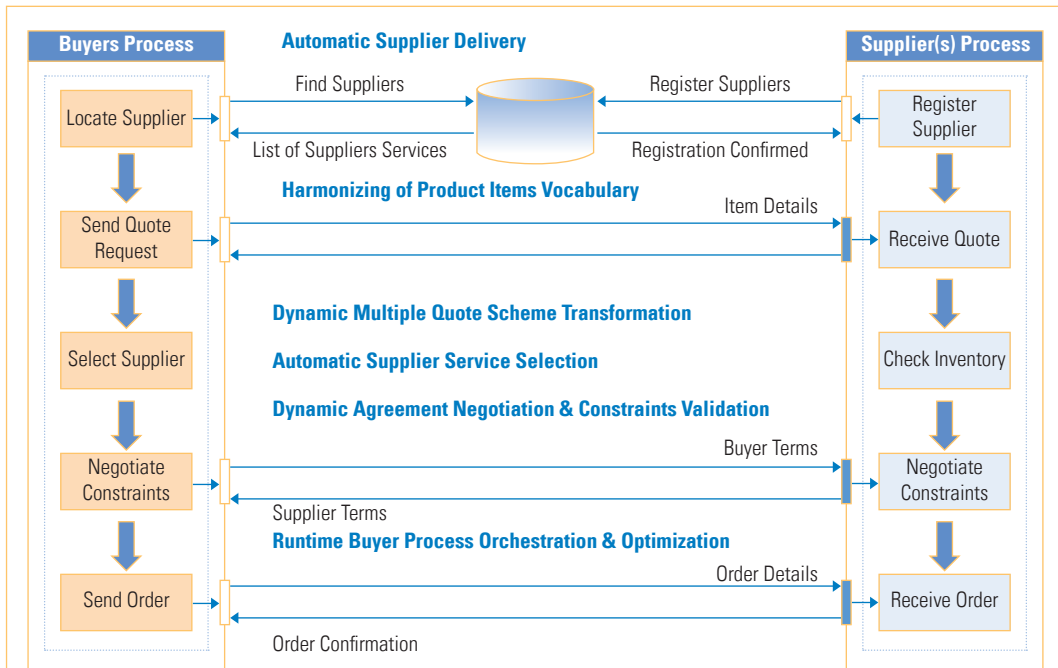


Figure 1: Semantic SOA Business Value

Source: Infosys Research

Semantic SOA is about applying ontology to the SOA implementation to enable the required automation and interoperability capabilities in traditional SOA to improve the enterprise efficiency in integration, information aggregation, collaboration and mediation.

In the enterprise scenario, the complexity of the integration and mediation solutions increases while traversing from single domain to multiple domains. The current state of such application integration and mediation is tightly coupled, point-to-point and thereby inefficient. Semantic SOA brings following capabilities:

- Automatic Service Discovery
- Dynamic Information Exchange
- Dynamic Service Constraints Negotiation
- Automatic Service Selection

- Runtime Process Orchestration Optimization and Correction

The power of semantics simplifies and automates complex mediation scenarios by bringing ontology-based dynamic data interoperability and automated process orchestration to the traditional SOA.

SEMANTIC SOA BUSINESS VALUE

To understand the business value of Semantic SOA, let us take a B2B Partner Collaboration Case Study. Consider a hypothetical scenario for Supplier Selection and Item Procurement Process [Figure 1]. The following are the important use cases in the given scenario:

- Buyers want to locate suppliers who can fulfill the request

- Buyers want to receive quotations from different suppliers
- Supplier wants to submit quotations for the required Items
- Buyers want to select the best deal supplier who also conform to the constraints and agreement for the order processing
- Buyer finally wants to place an order with the selected supplier.

The salient features of new Semantic SOA solution to the given scenario are listed below:

- Automatic Supplier Discovery using the service definition and related metadata to avoid any manual intervention
- Widening the reach to increased number of suppliers by supporting different vocabularies for the item descriptions at runtime
- Dynamically Processing the Quote Response Schemas of each supplier to extract the required data for automatic supplier selection
- Dynamic Negotiation and Validation of Buyer and Supplier Constraints such as delivery location, date and payment terms
- Automatic Supplier Selection based on the constraints negotiations and quote data
- Runtime Buyer Process Orchestration Optimization and Corrections to automate the entire buying process.

These capabilities offer improved business efficiency by reducing the lead-time for the buying process. Next we will discuss characteristics of Semantic SOA which enables to achieve these capabilities.

CHARACTERISTICS OF SEMANTIC SOA

Semantic SOA brings concepts of data semantics, functional semantics, execution semantics and

Quality of Service (QoS) semantics to resolve the shortcomings of the traditional SOA approach and achieve the earlier described capabilities.

Data Semantics

Data Semantics describe ontology for the definition of request and response message structures for a given service operation. The ontology annotations for the message structure help perform dynamic data transformation. It provides the Dynamic Information Exchange capability to the SOA solution.

The data semantics annotate data fields with domain ontology which facilitate flexible and automatic data transformations.

Functional Semantics

Functional semantics describe ontology for the definition of service capabilities and operations offered. The ontology annotations for the service definition and capabilities help to perform Automatic Service Discovery.

The functional semantics provide a template based semantic service discovery with matchmaking algorithm, which provides ranked available service results.

Execution Semantics

Execution semantics describe ontology for the execution flow of services in a business process or operations in a service. The ontology annotations help to determine the flow of dynamic service execution and invocation. It facilitates Runtime Process Orchestration capability to the SOA solution.

QoS Semantics

Quality-of-Service (QoS) Semantics describe ontology for the qualitative and quantitative constraints for service execution and invocation. The ontology annotations help to dynamically negotiate the constraints and agreements before

actually executing the services. It facilitates Runtime Process Orchestration capability to the SOA solution.

SEMANTIC SOA: STANDARDS AND TOOLS

Many standards have come up in industry to address “complete-interoperability” issue between services. They define different data, relationships between them and various ontologies to models. We briefly touch OWL-S, WSDL-S, WSMO and SWSO. They use Resource Description Framework (RDF) to describe semantic data.

OWL-S

OWL-S, derived from DAML-S, is a standard which helps software robots to discover, invoke, compose and monitor web services offered from multiple resources using their service descriptions. It uses Web Ontology Language (OWL) to describe its semantics and exposes Service Profile for advertising and discovering services, process model that gives detailed information about its operations and grounding to facilitate communications between services using messages.

WSDL-S

Web Service Semantics (WSDL-S) takes a step further from OWL-S and defines procedures to include semantic annotations to WSDL documents. It uses external ontologies described in different semantic models and maps various WSDL components to it. Datatypes, operations and interfaces of WSDL can be directly associated with appropriate concepts in semantic model. Moreover, it also gives freedom of choosing particular modeling language (be it OWL or UML or others) to the user, so that pre-defined models can also be annotated in the WSDL documents. This incremental approach to semantically annotate WSDL documents also offers easy updates and maintenance in future.

WSMO

Web Services Modeling Ontology (WSMO) provides a conceptual framework and a formal language to describe semantic aspects of web services to improve their discovery, composition and invocation. WSMO is richer than WSDL-S and has the capability to describe preconditions, post conditions, assumptions and effects on operations as well as orchestration and choreography of interfaces. WSMO contains four basic elements namely, ontology, web service description (describes functional and behavioral aspects of service), goals (describes user’s interests in services) and mediators (handles interoperability between other three elements). The formal logic language used in WSMO is described using Web Service Modeling Language or WSML.

SWSO

Semantic Web Services Ontology (SWSO) describes conceptual model of ontology as well as a first-order-logic axiomatization (called FLOWS) to define semantics of ontology. FLOWS enable reasoning between web services using semantics for interaction between each other and other applications. A similar attempt from Meteor-S also aims at adding semantics BPEL4WS, a standard for orchestration of web services.

HOW SEMANTIC SOA ADDRESSES THE TRADITIONAL SOA SHORTCOMINGS

Ontology Based Data Transformation

Ontology is a graph of abstract concepts, relations and logical assertions that comprise an information model for a given set of requests and responses. Semantic SOA approach uses ontology to describe message structures. The data semantics based ontology definitions message structures and auto-linked ontology mappings resolve the N2 mappings problem with the traditional SOA approach. The conversion from

one information domain to another information domain is achieved through a single ontology mappings file. The ontology mappings can be used to bridge information models asserting the heterogeneous data representations of the same concept. The mapping of two information domain ontologies automatically links any other ontologies to which they are connected.

Collection of data schemas for a given enterprise is equivalent to just a single ontology information domain entity. Hence, the single ontology mapping entity between two information models replaces a number of XSL mappings between them. Also, ontologies are transitive in nature and are capable of integrating themselves across enterprises.

Ontology Wrapped XML Schemas

The Semantic SOA approach does not force the participants in an SOA to tightly bind to common shared vocabulary as is required in a traditional SOA practice. The semantic SOA based approach to define the information models resolves all the issues related to standardized XML schema based integration in the traditional SOA approach. The ontology definition of the information model for a given enterprise wraps the XML Schema used by the enterprise. The Ontology Information Model harmonizes the vocabularies for request and response structures and thereby enables dynamic data transformation. It gives complete freedom to enterprises to consume the standards of their interest. This helps enterprises to leverage the legacy systems in place and save a lot of time to come to an agreement on how to use the standards. The enterprises will define their own information domain using ontology. The consumer enterprise will create an ontology mapping to transform the data from source information domain to the target information domain.

Application of ontology and semantics on-the-top-of SOA brings automation and

interoperability that is required to enable business cope-up with the future business models.

Dynamic Information Exchange

Semantic SOA uses registered ontology, ontology mappings and message metadata to dynamically determine how to convert one data format from one information domain to another data format in another information domain. It performs dynamic transformation using ontology and metadata introspection. It enables enterprises to address the dynamic information exchange challenge. The Semantic Enabled Data Mediation Engine will consume the data semantics of the message structure, metadata for the request content, ontology definition and ontology mappings to dynamically convert one data format to another. Enabling SOA approach with semantics (ontology) provides “explicit meaning” to the data exchange and business operations to infer “knowledge” and achieve required interoperability with automation.

Runtime Process Orchestration, Optimization and Correction

The Functional Semantics enable Automatic Service Discovery and the Execution and QoS Semantics enable Runtime Process Orchestration, Optimization and Corrections. It resolves the issues related to static process orchestration with traditional SOA. Semantic SOA approach enforces to implement standards specifications to achieve to functional semantics, execution semantics and QoS semantics. The Semantic Enabled Service Discovery Engine facilitates Automatic Service Discovery and Selection by consuming functional semantics ontology definitions. The Semantic Enabled Process Mediation Engine facilitates Runtime Process Orchestration, Optimization and Correction by consuming execution semantics and QoS semantics ontology definitions.

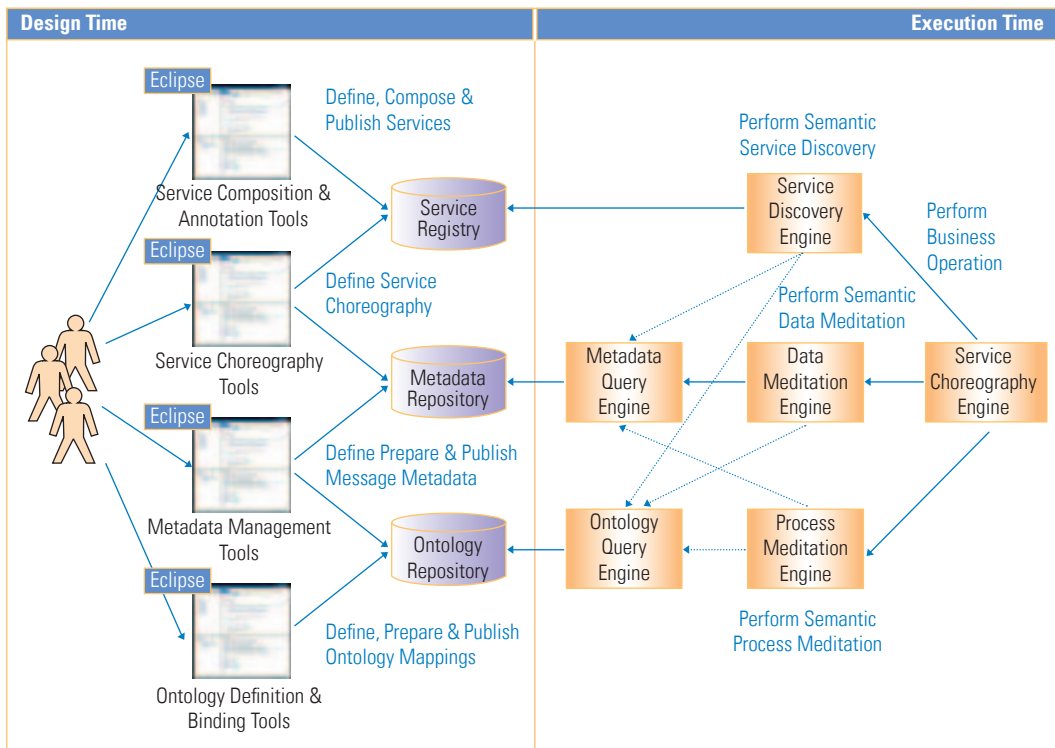


Figure 2: Semantic SOA Logical Architecture

Source: Infosys Research

Thus, Semantic SOA empowers business to perform automatic service discovery and dynamic information exchange to realize improved integration efficiency and process automation.

REALIZING SEMANTIC SOA

The Semantic SOA architecture goal is to support automatic discovery, selection, mediation, invocation and interoperation of services. The following conceptual components of Semantic SOA architecture help realizing these capabilities:

Service Registry enables automatic service discovery. The service definition should be stored in the service registry. It can be implemented as UDDI registry for web services.

Metadata Registry provides the metadata required for automatic discovery. Also,

the metadata registry will store the metadata for data formats used in service requests and responses, which will help achieving dynamic integration.

Ontology registry stores the ontology for the request and response messages and enables dynamic data transformation using ontology mappings.

Semantic Enabled Service Discovery Engine uses an ontology based reasoning interface to provide the intelligence required to automatically discover the services using the introspection of the service metadata.

Semantic Enabled Data Mediation Engine also uses a similar Reasoning Interface for dynamic integration. The data mediation process dynamically determines how to convert request data format into the required data format for further processing.

Semantic Enabled Process Mediation Engine facilitates dynamic process orchestration. A typical business operation requires execution of multiple services dynamically. The process mediation engine will provide that intelligence using the Reasoning Interface on the top-of-Service Definition and Business Operation Service Choreography Engine wraps the complete business operation and performs the various steps that are required to complete the business operation using Semantic Service Discovery, Data Mediation and Process Mediation. The Choreography Engine also wraps the communication mechanisms that should be used to communicate with services.


CONCLUSION

Semantic SOA brings Automatic Service Discovery, Dynamic Information Exchange and Runtime Process Orchestration Optimization and Correction capabilities to the traditional SOA. It offers efficient, scalable, agile and future-proof integration solution to enterprises. It can act as a business transformation catalyst and open new prospects for business, services and revenue models for the enterprises.

At the same time, Semantic SOA is comparatively new, innovative and an ambitious approach that offers process automation and information intelligence. Many industry standards are still evolving; also the visual tools that provide reasonable development productivity are under development. Apart from that, the ontology definitions and mappings require additional skills in the bucket of enterprise.

To conclude, enterprises cannot ignore the power and agility that Semantic SOA brings even-though it looks ambitious at this stage. It is recommended that enterprises adopting SOA should approach semantics step-by-step, keeping balance between innovation and realization. SOA implementation should be planned and designed for semantics to make it future-proof.

REFERENCES

1. Survey Data Says: The Time For SOA Is Now - April 14, 2006, Forrester <http://www.forrester.com/Research/Document/0,7211,39289,00.html>, accessed 11th Jul, 2006.
2. Web Service Semantics - WSDL-S <http://www.w3.org/Submission/WSDL-S/>, accessed 17th Jul, 2006.
3. Aligning WSMO and WSDL-S <http://www.wsmo.org/TR/d30/v0.1/>, accessed 19th Jul, 2006.
4. OWL-S: Semantic Markup for Web Services <http://www.w3.org/Submission/OWL-S/>, accessed 16th Jul 2006.
5. OWL Web Ontology Language <http://www.w3.org/TR/owl-guide/>, accessed 15th Jul, 2006.
6. RDF - <http://www.w3.org/RDF/>, accessed 12th Jul, 2006.
7. METEOR-S: Semantic Web Services and Processes <http://lstdis.cs.uga.edu/projects/meteor-s/>, accessed 19th Jul 2006.
8. WebService Modeling Ontology (WSMO) <http://www.w3.org/Submission/WSMO/>, accessed 18th Jul, 2006.
9. The Web Service Modeling Language WSML <http://www.wsmo.org/wsml/wsml-syntax>, accessed 18th Jul, 2006. 

Rejuvenate, Collaborate and Innovate: CPG industry context

By Vaidyanatha Siva and Vishal Puri

A consumer-driven replenishment system represents a huge business opportunity - here's how SOA and EDA can make this a reality for CPG companies

CPG (Consumer Packaged Goods) is an industry segment comprising businesses that manufacture or sell packaged goods for individual (as opposed to commercial) consumption [1]. PepsiCo, P&G, Kraft, Clorox are examples of CPG companies. There are three key imperatives that drive today's CPG companies -

- Pressures to drive operational efficiency by doing more with less
- Enhanced need to collaborate with retailers and suppliers to drive growth and profit
- Customer focus to drive innovation and stay ahead of the competition

This article deals with how SOA, in conjunction with the tremendous convergence in key technologies, can enable CPG enterprises to:

- Rejuvenate - unlock the value of existing IT investments

- Collaborate - within and outside organizational and geographical boundaries, as real-time as needed
- Innovate - get intimate with the customer and innovate to deliver unprecedented value

REJUVENATE

The adoption of technology and its acknowledgement as a strategic differentiator in CPG companies can be considered passive, rather than active. On top of this, the industry has been rife with mergers and acquisitions, mudding the waters with quick fix integrations to enable business as usual. Only recently have companies started taking a holistic view of the entire supply chain, spurred by supply chain visibility issues as well as programs such as VMI seeing adoption. Traditional SCM solutions, because of their localized optimization approach, never attempted to have an end-to-end view of data.

The first attempts at an integrated view of the enterprise were visual tools that integrated various data sources (via proprietary integrations) across SCM, ERP and legacy mainframe applications to provide enterprise wide collaborators a common view into the supply chain. While this is a move in the right direction, and in itself can give critical insights to all partners in a chain, making for better collaborative decisions, it is not enough. There will have to be a next step of enabling the various collaborators to dynamically source the data and access the services they need, in order to enable accurate forecasting, applying performance metrics and having a closed loop supply chain.

Much of today's enterprise IT assets lay buried in yesterday's legacy applications. This is a result of putting years and years of fixes on top of inflexible monolithic applications, to reflect business changes in the IT applications. These monolithic application architectures implicitly encourage silos of IT applications to spring up, with manual workarounds to stitch together a process, whenever the process requires integration across these silos. The move towards a connected enterprise is imperative to drive down supply chain inefficiencies. It may sound daunting to overthrow the existing regime and bring the incumbent forces of modernization and integration, but fortunately there exist many viable alternatives for going about this incrementally.

- 1) Web service enablement - Today there exist highly productive environments for developers to selectively expose mainframe based applications as modern web services -
 - That can be composed using RAD tools, which work with undocumented interfaces.
 - That can be easily tied into a standards compliant process orchestration engine.

Options exist such as green screen scraping, JCA connectivity to IMS, CICS transactions and also visual modeling and flow support for CICS service flow.

- 2) Legacy migration - If a business case can be made for migrating to a modern application infrastructure based on the prohibitive cost of maintaining and enhancing obsolete proprietary legacy software and hardware systems, there exist technology alternatives that can make this transition rapid. Some of the components of this technology stack are-

- Business rule extraction - This is a new breed of technology solutions that enables out-of-the- box business rule extraction capabilities on a myriad of platforms and languages. Relativity offers the Modernization Workbench™ that is an extensible platform to parse, profile and analyze existing enterprise applications [2]. IQ Server™ from Metallett goes one step further and creates a searchable knowledgebase of the extracted business rules, using advanced semantic inferencing and meta-modeling [3].
- Business service mapping - Once business rules and information models have been extracted, a service mapping exercise enabled by tools helps in actualizing the service.
- Code migration - Using tools provided by the business rules extraction engines, or by using best of breed tools for converting code, such as COBOL to Java or COBOL to COBOL.net, one can rapidly execute the migration exercise.

Using the business service mapping, the newly migrated code can then be exposed as a service, at the appropriate granularity.

COLLABORATE AND ADAPT

The Business Context

In a market where margins are being driven down due to lack of brand premiums, reduced shelf space and competition from home brands, it is becoming increasingly pertinent for CPG companies to transform themselves into operationally efficient and responsive organizations.

Out-of-stock in retail, especially during promotional periods, has been a severe problem. Out-of-stock is one manifestation of the several problems of demand invisibility that affect the supply chain today. Other concerns include higher markdown, supply returns by manufacturing/distributing nodes, time lag in product design etc.

There are a few success stories, from which CPG companies can take a cue. One such success story is that of fashion retailer Zara, which epitomizes a responsive supply chain [4]. Zara does a great job of sensing product movement, consumer demand and responding to it quickly through automation. Signals are sent to sourcing sites all across Europe and South East Asia and the product is brought to the market in record time. Cisco systems e-hub enterprise application, is another great example, where the demand signals percolate deep into the supply chain to enhance responsiveness [5].

The solution context

CPG companies are looking at many strategies in order to make this transformation, all of which involve collaboration internally and externally. The existing source to customer supply chain is based on hard assets, which leads to inflexibility and unresponsiveness as we see today. However, a demand driven supply network (DDSN) promotes a self renewing interaction among three strategic business domains - demand,

supply and product. Some of these strategies include -

- Demand forecasting and upward flow of trends
- Supply network design and reaction processes to meet changes in demand and supply, constraints and casuals
- Product lifecycle management and downward flow of product and promotional information

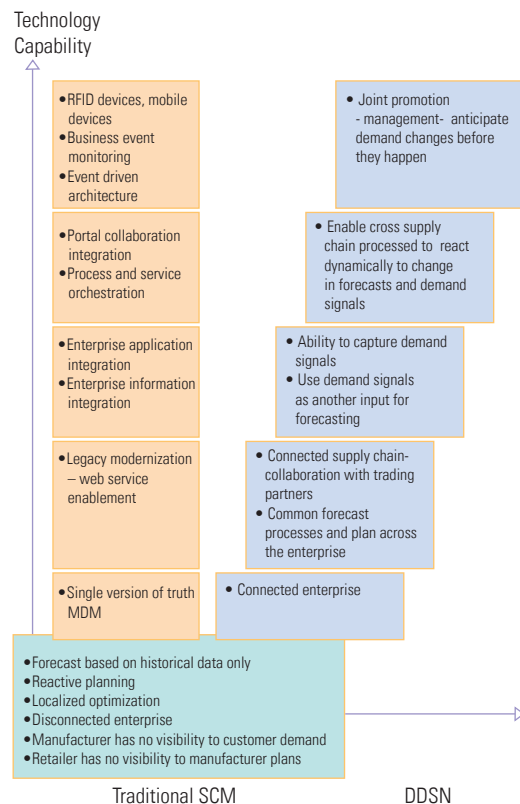


Figure 1: SCM versus DDSN

Source: Infosys Research

Collaboration

The primary step in enabling DDSN is to establish collaboration and synchronization between the value-chain partners.

While CPG companies may implement time phased planning and achieve benefits internally, they would need greater collaboration from their trading partners.

Some of the areas of collaboration are -

- Demand planning - Review demand and distribution estimates by retailers, share demand influencing factors
- Replenishment planning - Jointly determine multi-echelon safety stock strategies
- Promotional planning - Share promotional lifts, forecasts, calendar and information
- New product planning - provide feedback, plan changes and collaborate in logistics

These collaborations are enabled through effective synchronization of data between the trading partners. Setting collective standards for information exchange, data normalization, security and data management processes ensures collaboration with high degree of trust and accuracy. The maturity of service enablement of an enterprise, speaks to the level of collaboration that it can participate in. An organization with higher maturity levels should at least be able to quickly build together applications supported by processes that expose clean, meaningful services externally in a meaningful way to the trading partner.

Adaptiveness

A demand forecast based purely on historical data, to drive inventory replenishment, is not enough. A supply chain geared to fulfill demand on such a forecast will be unlikely to respond to causal events such as weather related disasters

or sudden disruptive shifts in consumer behavior. While a lot of examples of the former can be found, to give an example for shift in consumer behavior, consider the release of "The Incredibles" on DVD [6]. The usual trend has been that DVD sales for animation films that are successful on the big screen, translate to a successful DVD launch. In anticipation of a huge demand, large numbers of the films DVD inventory was stockpiled. This resulted in huge losses.

A truly adaptive supply chain will be able to respond to these demand signals and respond by dynamically switching to alternate strategies. There are two key aspects of such an adaptive supply chain -

1) Enablement of appropriate sensors in the enterprise and its value chain partners. Examples of these sensors are -

- RFID tags that provide inventory out of stock information when product is being shifted from the back of the store to the shelf.
- Event triggers based on causal events such as disasters, weather pattern changes etc
- Event triggers based on correlating clickstream data from the ecommerce channels, generating demand signals, and customer feedback signals.
- People responsible for making personal connect with the local customer to sense demand. In the case of retailer Zara, these people are called Commercial.

2) Enablement of the supply chain to receive these signals, react to them appropriately and propagate them deep into the supply chain.

There are a number of technologies that need to come together in order for an enterprise to implement an adaptive supply chain network. These are -

1. Agents-based event driven architecture -

Event driven architecture consists of streams, agents and an event processing engine. There are two kinds of streams:

- The control stream - what to watch out for? These are the subscriptions the event subscription agent asks the event publisher to look out for. The power of EDA lies in its ability to build sophisticated model driven subscriptions [7]. In a commodity-trading application, a model may specify relationships between prices of a commodity at different places, prices of substitutable commodities, costs of shipment between different places, exchange rates, and so forth. A variation from the model may signify a threat or opportunity that is worthy of exploration.
- The message stream - the filtered messages that match the subscriptions which are registered with an event publisher.

Event publishers and event subscribers are loosely coupled lightweight agents. These agents can be used to monitor databases, for post facto event generation, as well as on business applications to handle real time events that may or may not be persisted, but are critical sources of relevant business events. Advancements in agent technology has enabled the possibility of agents being deployed on a multitude of middleware platforms, where the business logic resides, and attach them to very fine grained services exposed through platform specific discovery technology.

2. Enterprise Service Bus - An ESB implementation as specified by JSR 208, more popularly known as Java Business Integration (JBI) is designed with open standards and enterprise agility in mind. It complements the asynchronous event generation agents by providing a highly scalable messaging backbone, supporting content based routing, correlation capabilities and service orchestration through integration with BPEL compliant engines. This is especially key in dealing with the large volume of events being generated from the edge servers, such as RFID servers.

3. RFID technology - Both RFID sensor and server technology are seeing slow and steady adoption with mandates from large retailers (Wal-Mart, Albertsons etc), government agencies and departments (US DoD). Considerable challenges, such as low read rates, directionality problems and lack of process details, remain. Some companies have chosen a careful plan for adoption of the technology in full cognition of the current technology limitations as well as the potential for competitive edge for early adopters. Early research has shown considerable improvement in OOS events in stores that use RFID.

INNOVATE

Product innovation is key to success and, sometimes, survival for a CPG company. From a closeted paradigm where most of the research happened in-house, today there are many open innovation networks that enable a CPG company to enable faster go-to-market. The marketing organization can outsource market research to some third party and then leverage an open innovation network such as innocentive to provide quick solutions to product opportunities.

One obvious use of SOA, EDA and webservices is to enable this enhanced collaboration between various partners. This collaboration at the edge of the enterprise is what John Hagel calls the only sustainable competitive advantage [8].

The other use of this group of technologies is in Enterprise Information Management (EIM). encompasses the harnessing of structured, semi-structured and unstructured data to provide insight into the business.

Traditional approaches for customer insight such as business intelligence would be augmented by leveraging the constant stream of information that is now available from diverse sources - market analysis, financial data, customer blogs, call center reports (voice to text conversion and analysis), customer product feedback, etc. Taxonomy, Ontologies to get context sensitive, semantically aware information and expert systems to understand this information and act on it would be the next stage in the evolution of EIM.

Agent-based, event-driven architecture, SOA and web services, Natural Language Processing and Expert Systems will be key technologies that will enable customer insight and drive product innovation.

CONCLUSION

In the final analysis, SOA and the convergence we have seen in related technologies provide one underlying advantage to CPG companies - a faster go-to-market solution. Legacy modernization (rejuvenation), collaboration and product innovation are enhanced due to the ability to harness these technologies to provide faster results. Of course, businesses and people

need to be able to leverage the technology to reap benefits. While people and businesses should want to collaborate, technology will just enable the collaboration.

REFERENCES

1. http://www.gartner.com/6_help/glossary/GlossaryC.jsp
2. <http://www.relativity.com/pages/modernizationworkbench.asp>
3. <http://www.metalect.com/solutions.php>
4. <http://www.optimize-mag.com/issue/026/financial.htm?articleId=17701020&queryText=&pgno=5>
5. A consumer-driven replenishment system represents a \$325 billion opportunity: <http://newsroom.cisco.com/dlls/tln/newsletter/2002/april/part2.html>
6. <http://money.cnn.com/2005/07/01/technology/dvds/>
7. Clarifying the terms Event Driven and Service Oriented Architecture, Roy Schulte, Gartner Research, February 2005.
8. The Only Sustainable Edge: Why Business Strategy Depends on Productive Friction and Dynamic Specialization, John Hagel III and John Seely Brown, Harvard Business Press, 2005.
9. Key Metrics can Help Justify Investments in SOA, Michael Barnes, Gartner Research, May 2006.
10. Defining Service is Key to Implementing a Service-Oriented Architecture, Darryl Plummer, Gartner Research, May 2006. Also available on <http://www.psdn.com/library/kbcategory.jspa?resultOffset=30&categoryID=58> 

SOA – Impact on Enterprise Architecture Service View

By Peter Jarman

‘Service’ as a concept existed long before SOA came to prominence. Today, how then does SOA impact the service concept?

SOA is not a new concept, rather it is an evolutionary view of a capability which has existed as long as distributed computing has existed, namely the ability to access services and data in a remote fashion. Many of the SOA concepts already existed in the EAI world, although in a more proprietary fashion. This article identifies some generic service layer categories and where they are used in SOA as well as assess the impact of SOA on pre-existing service based architectures.

WHAT IS A SERVICE?

One of the fundamental foundations of SOA is the concept of a service. However the concept of a service existed long before SOA came to prominence. SOA has developed and leveraged a foundation of open standards for the usage and delivery of services, while using a definition of service which has remained focussed on “plumbing.”

W3C defines a Service as an abstract resource that represents a capability of performing tasks that form a coherent functionality from the point of view of providers entities and requesters entities [1]. Wilkes defines a Service as providing a simplified mechanism to connect applications regardless of the technology or devices they use, or their location [2]. Both definitions are focussed on the mechanism of providing a service based on the W3C standards.

There is continued debate around whether a service in SOA implies web services only. Although the standards based approach is a key part of isolating the functionality from the underlying delivery technology, it should not preclude the delivery of an SOA based implementation. Services can be effectively delivered using technologies such as MQ, CORBA, or vanilla J2EE or .NET APIs. The key concept is the delivery of service based functionality, able to be re-used, delivered on a

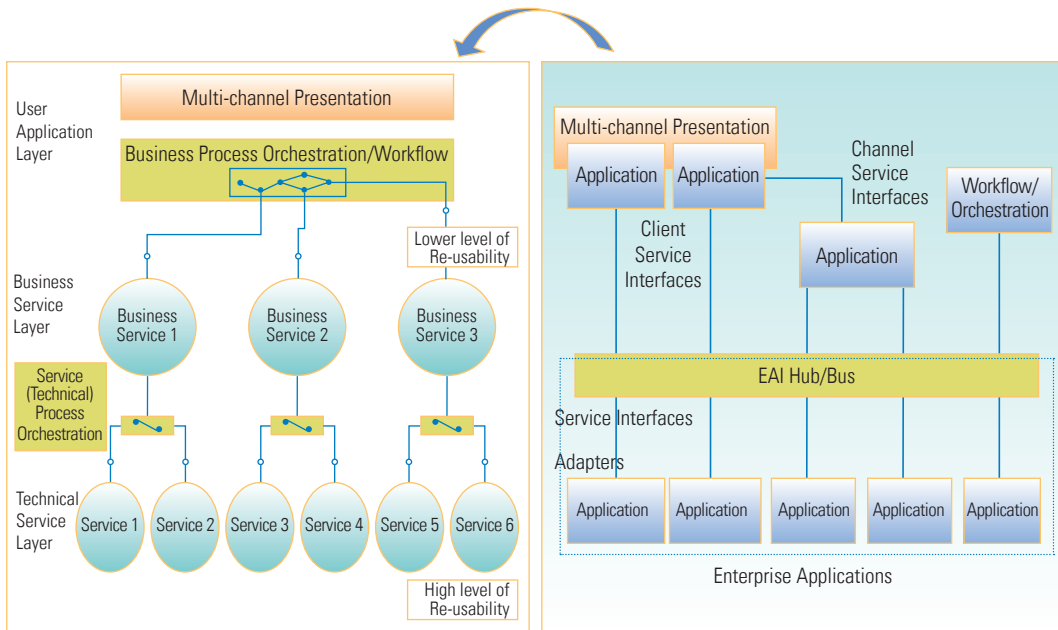


Figure 1: SOA Services Layers and Business Processes vs Traditional EAI Service View

Source: Infosys Research

technology acceptable to an organization and its partners, while supporting re-use, orchestration and scalability without additional development effort.

SERVICE LEVELS

Most medium to large organisations, especially those already using EAI technology, already have the services concept in place, often accompanied by a common information model. Lawrence Wilkes finds that using an integration layer not only supports connecting together services, but also enables business process and service delivery to have independent change cycles [3].

Concepts such as enterprise services exist in the implemented enterprise architectures for many organisations, as part of a common shared infrastructure. However a variation of this concept which is coming to the fore as part of SOA, is the concept of a “business service.”

In a SOA layered architectural view there are two key service layers, the business service layer and the technical services layer [Fig. 1]. In both cases the relevant orchestration capability also intrudes into the layer as coarser grained services can be delivered as composite services at both the technical and business levels. This tends to be more prevalent at the business layer but is equally relevant in the technical layer.

Historically there have been no obvious differentiators between business and technical services, as the service concept was focussed on technology alignment, e. g., EAI rather than business concepts.

The concept of a business service has been more driven by BPM and the expectation of the business to develop, manage and monitor its processes. Thus even though the same SOA type technologies support both technical and

business services layers, differentiation is more driven by organisational responsibilities. In fact the need to split these layers is inducing some organisations to use different technology sets, to deliver the same technical functionality, just to ensure responsibility for these layers remains discrete in the organisation.

TECHNICAL SERVICES

Some technical services are already in place in many organisations although they may not be enabled as Web Services. A key aspect of technical services is that they provide support to business functions, rather than being a specific business service themselves. These can be both low level services directly supporting the business functionality or support services which are more closely aligned with the IT infrastructure (software and hardware). For example operational management services are not business services but support services.

Technical services can be linked to form business services composites, using process orchestration. Some examples of Banking system technical services are shown in Table 1.

BUSINESS SERVICES

Some key differentiators between business and technical services are the business value and the relevance to the actual processes being enacted across the business. It is at the business service layer where IT and business alignment becomes most obvious. This is also the layer where business change can be more traumatic if the relevant business rules and application integration are not well architected.

Business services can be linked to form business processes using process orchestration or workflow. Alternatively they can be enabled directly through a user interface to provide user functionality. Composite business services can be provided via service orchestration of other

Technical Services
Identity Management - validateCustomer
Identity Management - getAccountList
Identity Management - getCustomerDetails
getAccountBalance (Banking, Cash management & Credit Card)
getAccountTxnList [Banking, Cash management systems] debitAccount [Banking, Cash management & Credit Card systems]
creditAccount [Banking, Cash management & Credit Card systems]
changeAccountDetails [Banking, Cash management & Credit Card systems]
CreditCard – getAccountTxnsForPeriod
Operations – GetDailyPerformanceStatistics

Table 1: Sample of Technical Services

Source: Infosys Research

business services or technical services.

In order to support flexible business processes, the business services need to be abstracted enough to be flexible. It is better to have finer grained business services which are orchestrated to deliver a coarser grained service aligned to the relevant business process. Over time as business processes change, there will be changes in the service requirements. If a business service is optimised for one process it will often need changes to support the changed process and the use of orchestration/workflow over finer grained business services will better support changes without requiring significant IT development effort.

According to Sprott and Wilkes, the successful enterprise will plan its service application architecture as a series of maturity stages, each establishing foundations for the next [4].

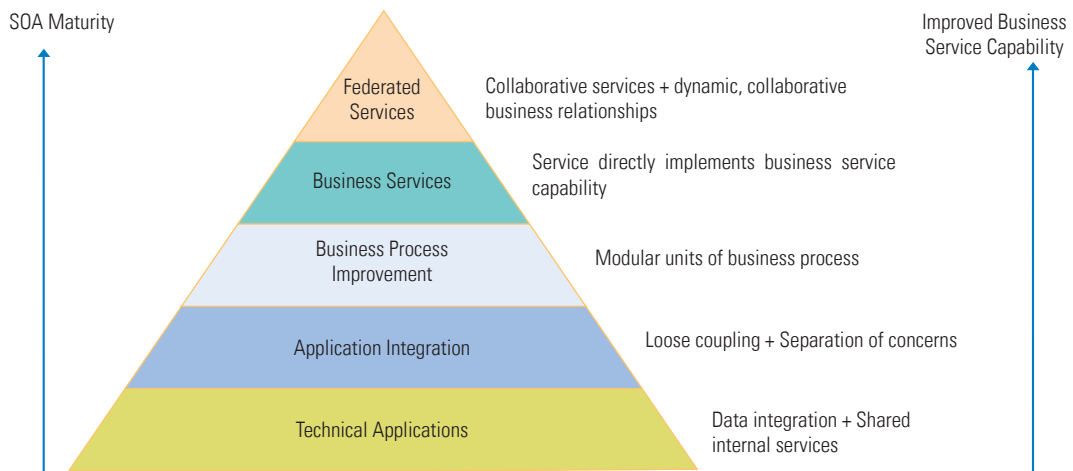


Figure 2: SOA Maturity Model

Source: Adapted from Sprott and Wilkes

As is shown in Figure 2, early stages of SOA are more focused on enabling existing applications and implementing SOA type technologies. These services developed at this level tend to be more technical services as they reflect existing applications. Lawrence Wilkes identifies that until now many organisations have systems and packages which provide functionality that do not align with the services required by new business processes. However, in organizations where the existing applications are optimised and well-aligned with their current business processes, there is a higher probability of being able to deliver usable business services. Some examples of banking business services and their composition services can be seen in Table 2.

SERVICE SEGMENTATION

Many organisational enterprise architectures segment services based on organisational responsibilities, into enterprise and channel/business unit. This type of structure has arisen out of EAI architectures. Overtime, EAI services/

interfaces have tend to migrate between channel and enterprise as the organization changes.

With SOA, services tend to be segmented into technical and business layers, but a similar EAI aligned organisational responsibility can also apply. In addition, the SOA and web service paradigm has increased the prevalence of external services, mainly for business services, and also for some technical services. This results in a matrix, rather than simple layers (Fig. 3). A possible breakdown of the sample services across these segments is show in Table 3. In this case the channel would be focussed on self service channel for the financial organisation.

The migration of services between enterprise and channel is still relevant. In addition, however, some business services may migrate to technical services as more complex business services are developed to support changing business needs. Also as the business changes over time some business processes and services will no longer be seen as core elements, resulting in potential delegation of some services to external suppliers.

Business Servicev	Service Composition
viewIBAccountsList	IM - getAccountList B - getAccountBalance CM - getAccBalance CC - getAccountBal
viewAccountDetails	B - getAccountBalance B - getAccountTxnList CM - getAccBalance CM - getAccTxns CC - getAccountBal CC - getAccountTxnsForPeriod
accountTransfer	B - getAccountBalance B - debitAccount B - creditAccount CM - getAccBalance CM - debitAcc CM - creditAcc CC - getAccountBal CC - debitAccount CC - creditAccount
payBill	B - getAccountBalance B - debitAccount CM - getAccBalance CM - debitAcc CC - getAccountBal CC - debitAccount B2B - getBillersList (business) B2B - makeBillPayment (business)
viewIBCcustomerDetails	IM - getCustomerDetails
updateIBCcustomerDetails	IM - getAccountList B - changeAccountDetails CM - changeAccDetails CC - changeAccountDetails
B2B – makeBillPayment	
B2B – getBillersList	

Table 2: Sample of Service Orchestration

Source: Infosys Research

SERVICE GRANULARITY

Separation of business and technical services will naturally result in differing levels of granularity for each service layer. Business services are coarser grained and their re use is restricted by how often the same business function is used

	Technical	Business
Channel	Operations – GetDailyPerformanceStatistics	viewIBAccountsList payBill viewIBCcustomerDetails updateIBCcustomerDetails
Enterprise External	B - getAccountBalance B – debitAccount B – creditAccount CM– getAccBalance CM – debitAcc CM – creditAcc CC–getAccountBal CC – debitAccount CC – creditAccount IM – getAccountList IM- getCustomerDetails	viewIBAccountsList payBill viewIBCcustomerDetails updateIBCcustomerDetails
External		B2B – makeBillPayment B2B – getBillersList

Table 3: Sample of Service Segmentation

Source: Infosys Research

across multiple business processes. Technical services tend to support a higher level of re-use as their capabilities may be re used across a number of business services.

There is no cookbook approach for determining optimum service granularity. In fact, over time, even an optimal service definition will become sub-optimal due to business changes.

For business services, determining appropriate levels of granularity can be based on identifying aspects like:

- Business alignment
- Ease of composition
- Abstraction/isolation from underlying application technology
- Business value
- Industry standards
- Security requirements

For technical services determining appropriate

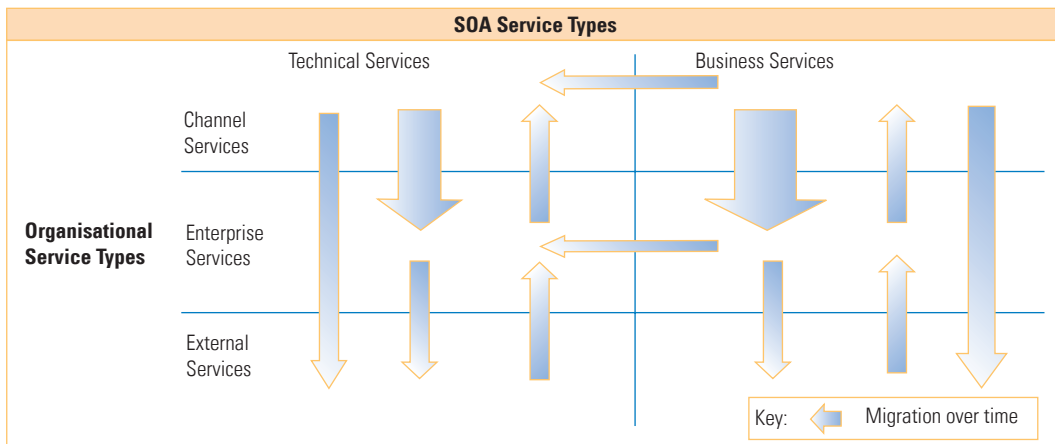


Figure 3: SOA Service types

Source: Infosys Research

levels of granularity can be based on identifying aspects like:

- Ease of composition
- Opportunities for re use
- Alignment with existing service
- Security requirements
- Delivery/maintenance cost

CONCLUSION

SOA is all about delivery and usage of services. We have looked at how SOA enhances the service concept and impacts existing enterprise views on service architecture. It also raises the need to identify different service layers and types to better support the conflicting business and technical needs.

One impact identified is the need to separate technical and business services over any existing service distribution architecture to better support SOA and business alignment. Another impact is the need to support migration of these services across organisational boundaries as this migration is likely to be more pronounced in an SOA enabled organisation.

REFERENCES

1. W3C Web Services Glossary - W3C Working Group Note 11 February 2004 <http://www.w3.org/TR/ws-gloss/>
2. CBDi - ROI - The Costs and Benefits of Web Services and Service Oriented Architecture, Lawrence Wilkes, 2003 <http://roadmap.cbdiforum.com/reports/roi/>
3. CBDI Report - Business Integration Architecture and Implementation - Critical Success Factors, Lawrence Wilkes, May 2002. http://www.cbdiforum.com/bronze/downloads/bus_int_architecture.pdf
4. CBDI Journal SOA Fundamentals - Understanding SOA, David Sprott & Lawrence Wilkes, 2006 <http://roadmap.cbdiforum.com/reports/fundamentals/>
5. InSOAP: An Architectural Framework for Service Definition and Realization by Abdelkarim Erradi, Sriram Anand & Naveen Kulkarni (Infosys research paper) http://setlabs/setlabs/Thought_Leadership_Docs/SOAFICSEpaper.pdf

Enterprise Application Framework for SOA Realization

By Shyam Kumar Doddavula, Sandeep Karamongikar

Enterprise Application Frameworks are critical for SOA Realization

SOA concepts are primarily designed to achieve the vision of an agile enterprise with a flexible IT infrastructure that enables the business to respond fast and cost effectively to changes through reuse of existing investments. SOA advocates developing reusable business services and building applications by composing those services instead of building monolithic applications in silos. These SOA concepts are pretty much accepted now-a-days. The challenge before enterprises today is in taking SOA from concepts to execution.

STRATEGY FOR SOA REALIZATION

Figure 1 provides an overview of the strategy for SOA realization. The strategy involves performing the following 4 steps iteratively

1. Define SOA Reference Architectures
2. Define an enterprise SOA application framework that provides the structure and the basic building blocks based on the reference architectures

3. Standardize on enterprise tools that will enable use of the frameworks better
4. Define a the SOA methodology that lays down the process to be followed with the reference architectures, application frameworks and tools incorporated at appropriate stages.
5. Define the SOA Governance Model

This strategy enables providing prescriptive instead of providing reactive solutions, with strong downstream support during service development.

One of the key aspects of the strategy is developing an enterprise application framework which provides the infrastructure needed while designing and developing applications based on the SOA concepts.

In this article we explain how to go about defining reference architectures with the needed design elements for SOA through a systematic 'requirements driven' approach and how to develop an enterprise application

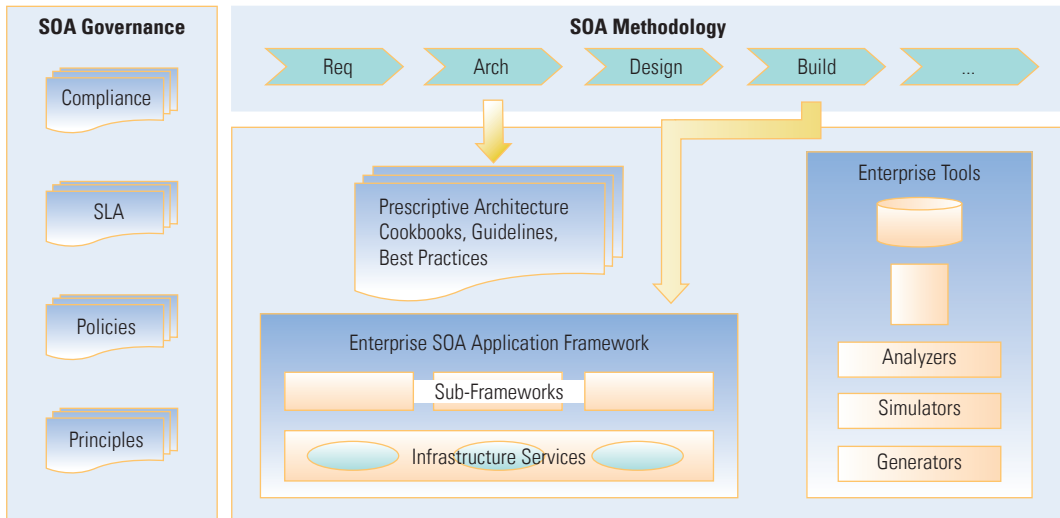


Figure 1: SOA Realization Strategy

Source: Infosys Research

framework that will enable realize those reference architectures.

OUR APPROACH TO FRAMEWORK DEVELOPMENT

Our approach is to first identify the significant requirements for developing services and then identify the key design elements needed to address those requirements based on applicable design patterns and then develop an application framework that provides the basic design elements identified in the reference architecture.

REQUIREMENTS: SOA CONSIDERATIONS

From a technical perspective, the core principle of SOA is: the business functionality is provided as a service and in order to use the functionality the service consumer should be able to lookup the corresponding service and use it. Service design should be interface driven.

The design implications are:

A) There should be a well defined service lookup

mechanism that the service consumers can use to get a handle of the implementation of the service interface.

- B) The user code shouldn't be tied to the implementation specifics of the service.
 - i) Ideally user code shouldn't change if the technology used for the service implementation changes say from Cobol, to a simple java object or to an EJB or to a .NET.
- C) The user code shouldn't have to deal with the life cycle aspects (ideally all aspects) of a service like creating, initializing, configuring, deploying, locating and managing a service.
 - i) There should be well-defined mechanisms that take care of creating, initializing, configuring, deploying and managing a service that finally provides a mechanism for the end user to look up the service and use it.
 - ii) There should be mechanisms that will allow defining other service aspects like access control to the services, audit of

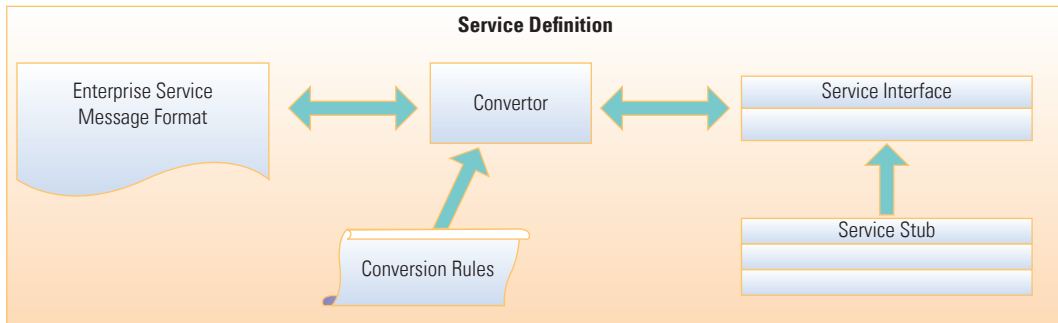


Figure 2: Service Definition

Source: Infosys Research

service access where the user can plug in their logic.

REQUIREMENTS: NEED FOR VARIATIONS

A key consideration that often gets ignored is the requirement for variations in service behavior based on context like the line-of-business, geography, marketing channel etc., under which the service is invoked. This is especially important because SOA strategy results in services that are shared in an enterprise. If we take the insurance domain for example, insurance calculations usually vary based on various factors like the state, country etc., or the marketing channel that the product is sold through like self service, assisted, retail, wholesale and partner. Similarly, the service orchestrations steps for process services like the insurance application processing, claims processing etc., are typically different for different lines of business like for say property insurance, auto insurance and so and so forth. What is needed is an application framework that provides Service Variation mechanisms.

DESIGN: FRAMEWORK COMPONENTS NEEDED FOR SOA

Based on the analysis above, the architecturally significant features identified are:

- A) A clearly defined mechanism to define a service interface with the available operations and input and output parameters.
- B) A registry of services that the service providers can use to register their service implementations and which the service consumers can use to lookup a service implementation.
- C) An enterprise service bus into which the service implementations can plug in and out and which supports multiple calling semantics (like synchronous, asynchronous etc.), and features like transformation, routing etc.
- D) A well-defined service orchestration mechanism to take care of flow based and long-running interactions.
- E) A well defined mechanism that takes care of service aspects like configuration, management, access control, audit etc.
- F) Some well-defined service invocation mechanisms with adaptors that will allow the service to be invoked and implemented through multiple technologies (like WebServices, EJBs, POJOs, Cobol etc)
- G) Well-defined Coarse-Grained Variation

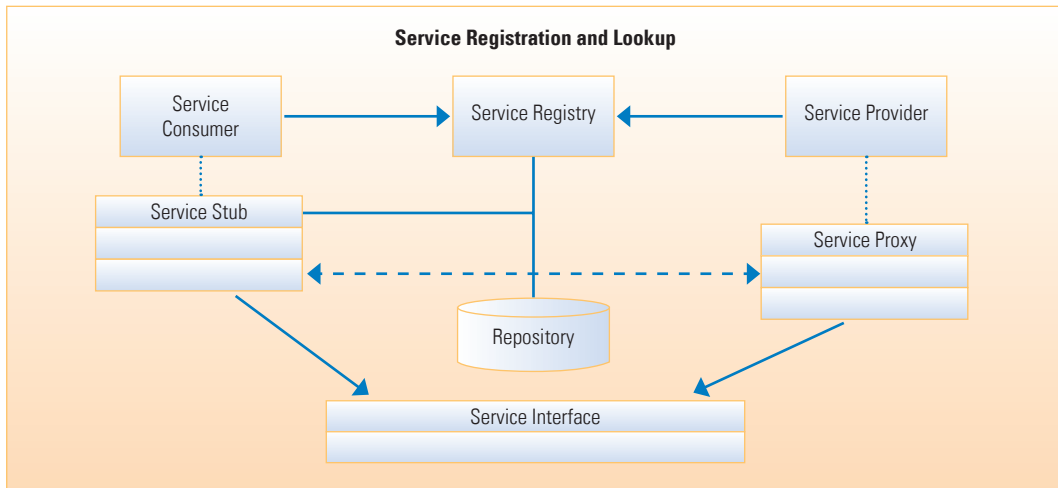


Figure 3: Service Registry

Source: Infosys Research

Points that will allow the behavior of shared services to be varied based on the context of invocation.

Service Definition

The framework should primarily provide a standard mechanism for defining the service interface. Since most enterprises use applications and systems that are implemented using multiple technologies and platforms, an XML based mechanism like WSDL is recommended for service definition.

There are tools that will generate the service implementation components like the service stubs, proxies etc. from the interface representations and vice-versa.

Service Registry

One of the important requirements to be addressed by the framework is to provide a service registry with details of the service interfaces and the service providers.

The framework needs to provide a service registry design element which provides

the API to register and lookup the service stubs that implement the service interface. The service Stub encapsulates the invocation details for the consumers and it interacts with the service proxy which encapsulates the invocation details for the service providers.

Service Invocation

The infrastructural logical components that are needed for service invocation include, Service Stub, Service Proxy, Adaptors, Message Broker, Message Bus and Gateways.

While the Service Stub implements the delegate pattern and provides service interface to the service consumers hiding the invocation details, ServiceProxy implements the proxy pattern and provides the abstraction of the invocation details for the service providers.

Adaptors provide the technology specific integration mechanisms for the service stubs and proxies. For a J2EE technology based implementation the adaptor can provide the listener mechanisms that the stubs and proxies can use to receive the messages and the API to send a message.

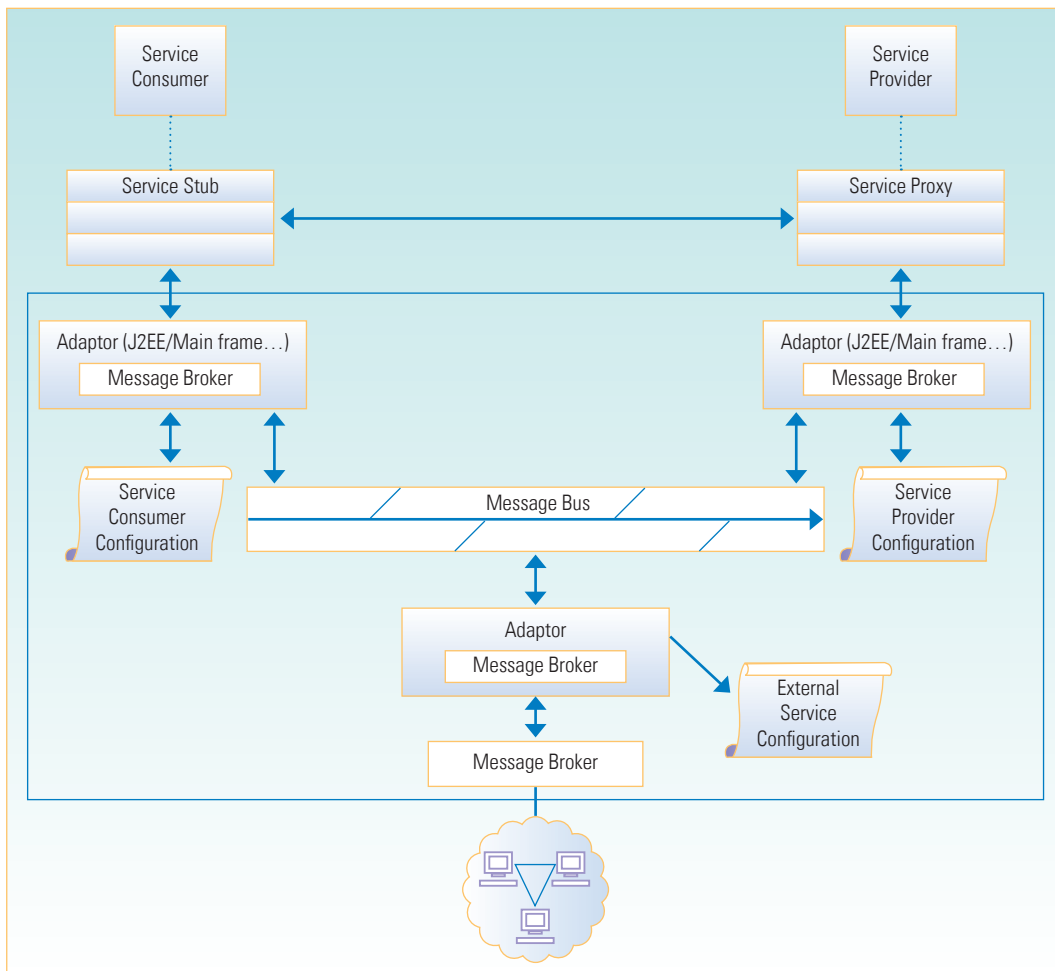


Figure 4: Service Invocation

Source: Infosys Research

Message Broker and the Message Bus provide the transformations, routing and transport. The broker and the bus take care of transforming the message representations from the service consumer and service provider internal formats to the Enterprise Message Format and vice-versa and also provide the routing of the messages and the abstraction of the transport with store and forward, message retries, prioritizing of messages etc.

Gateways provide the mechanisms for external integration. The gateways provide the single point-of-contacts for the external partners and transform the invocation protocols and message formats from the external partners to the internal enterprise message formats using a Message Broker and an adaptor, and also enforce the security checks, audit requirements etc.

Service Orchestration

The framework should define a Service

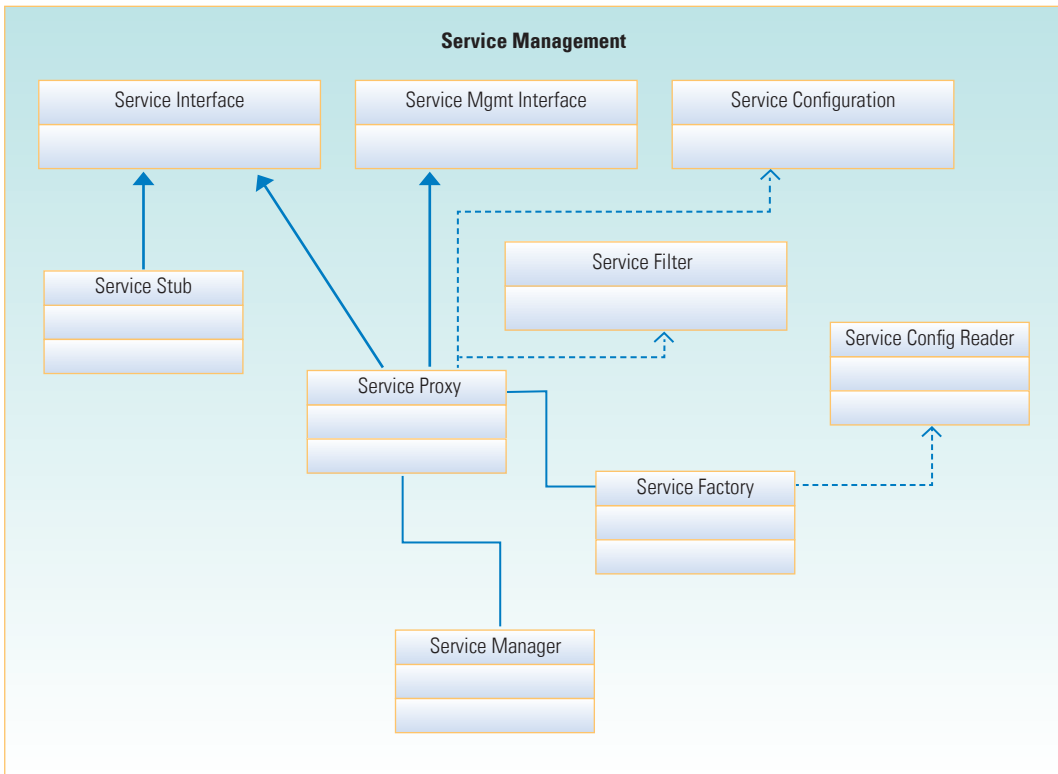


Figure 5: Service Management

Source: Infosys Research

Orchestration Adaptor that helps abstract the interactions with the orchestration implementations (BPM/BPEL product). The adaptor should provide API to initiate processes, get the list of process instances, get the list of activities and their state, be able to manipulate the state of activities, list of exceptions etc with abstractions over the implementation specifics.

Service Management

The next important requirement to be addressed include providing a standard mechanism for the management of services, for configuration of services, for taking care of the cross cutting concerns like the access control, audit etc., that apply to all or most service requests driven by centralized processes.

One of the common requirements in service design is to ensure that the service is configurable so that a service instance can be localized to a particular context and deployed. The framework should therefore provide a standard mechanism for service configuration.

The framework should define a standard Service Configuration Format, a Service Configuration Reader component, a Service Configuration component to represent and hold the service configuration information and a Service Factory component that takes care of the creation of the service, loading the service configuration and initializing the service with the desired configuration.

The framework should provide for a mechanism to allow the separation of the service

core functional logic from the logic for enforcing the cross cutting concerns like access controls, audits etc. The framework should define a Service Filter component which can be plugged into the service invocation mechanism at service proxy to intercept the service requests and apply the QoS aspect logic.

Service Variation Points

The approach we recommend to address the need for variations in the behavior of shared services based on invocation context is to first identify the common variation requirements, then identify the architecture strategies to enable such variation and then, based on those strategies, define an architecture with well-defined variation points, and finally define mechanisms for bundling such variations. Table 1 has the list of common scenarios and the architecture strategies for the same.

The recommended logical architecture with the recommended Variation Points is shown in Figure 6.

CONCLUSION

In this article we described the strategy for SOA realization and then defined the essential characteristics of an enterprise application framework which is a critical part of the strategy. J2EE technologies with all the associated open source solutions offer a good platform for developing such a framework. A well-defined process for services identification, definition, implementation supported by a standard R&D infrastructure for executing Service Oriented Architectures including reference architectures, application framework and tools, with an overall services governance model will help enterprises realize the promised benefits of SOA.

Scenario	Strategy
Vary service implementation based on context	<ul style="list-style-type: none"> Use Service Locator to lookup different Service Implementations based on context
Vary service request handling logic	<ul style="list-style-type: none"> Design Service implementation to use pluggable Service Request Handlers
Vary sequence of steps in a business process involving service orchestrations based on context	<ul style="list-style-type: none"> Use Configurable BPEL engines to model service orchestrations Define multiple processes for multiple business contexts Provide an indirection between logical process ID and actual process ID
Vary sequence in processes involving human interactions based on context	<ul style="list-style-type: none"> Use Workflow engines to model processes involving human interactions Define multiple workflows for multiple contexts Provide an indirection between the logical workflow ID to the actual workflow ID
Vary business logic based on context	<ul style="list-style-type: none"> Design business logic implementation to use a rules engine and provide mechanism to plug different rule-sets based on configuration Design business logic implementation logic to be parameter driven and provide a service configuration mechanism Design the core business logic to be Object-Oriented and provide a mechanism to plugin different implementations based on configuration
Vary service invocation mechanisms based on context	<ul style="list-style-type: none"> Interface driven services with technology wrappers like EJB/JAX-RPC wrappers Configuration driven service implementation technology selection

Table 1: Common scenarios and architectures strategie

Source: Infosys Research

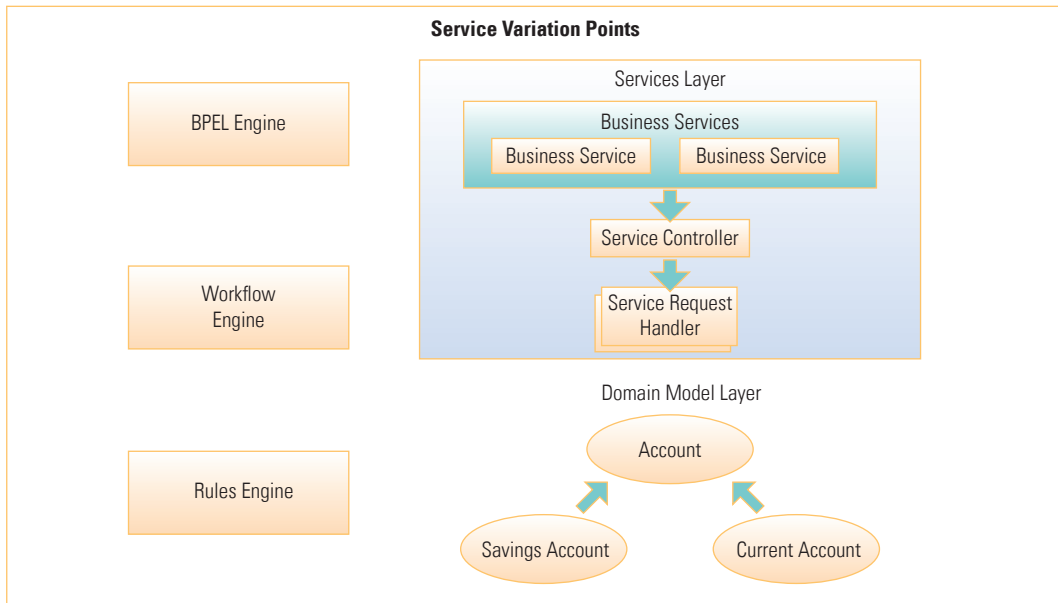


Figure 6: Service Variation Points

Source: Infosys Research

REFERENCES

1. SOA Center SOA Blueprints, available at <http://soacenter.com/>
2. SOA Design BluePrints Catalog by Sun, available at <https://bpcatalog.dev.java.net/nonav/soa/index.html>
3. Service-Oriented Architecture Portal by Bea at <http://dev2dev.bea.com/soa/>
4. Building SOA Solutions with the Service Component Architecture available at http://www-128.ibm.com/developerworks/websphere/techjournal/0510_brent/0510_brent.html
5. Understanding Service- Oriented Architecture, available at <http://msdn.microsoft.com/architecture/soa/default.aspx?pull=/library/en-us/dnmaj/html/aj1soa.asp>
6. Service-Oriented Architecture (SOA) and Web Services: The Road to Enterprise Application Integration (EAI), available at <http://java.sun.com/developer/technicalArticles/WebServices/soa/>
7. Designing Enterprise Framework for SOA, available at <http://today.java.net/pub/a/today/2005/04/28/soadesign.html>
8. Building SOA based Solution for a Workforce Management available at - <http://doi.ieeecomputersociety.org/10.1109/SCC.2006.27>

SOA is not just Integration+ !



“...Nor can SOA be equated with implementing just an ESB.”
Consulting Editor, Dr. Srinivas Padmanabhuni who heads the SOA Center of Excellence (CoE) at Infosys attempts to delineate a pragmatic roadmap to SOA adoption

Much has been made of hyped technology waves over the past few decades as the effectiveness of the technologies at the frontier of each wave. Is the current wave of SOA any different? Will SOA be able to bridge, if not obliterate the gap between business and IT? The real question is whether there is actual gap between the desired expectations and the delivered value of SOA.

It might be too early to make judgment on this front, because as a mainstream trend in distributed computing it is a fairly new phenomenon. However in this column I would like to leave behind some thoughts on overcoming the key challenges and pitfalls in SOA adoption. In the myriad SOA interactions with clients, some of the thoughts which come to my mind as key to address when adopting SOA are outlined below:


1. SOA requires STRATEGIC approach from a planning perspective, for maximum Reuse and flexibility, with models for sharing costs and benefits across the organization, to get the return on investment.
2. The treatment of SOA as Integration+ should be avoided to get a broader applicability of SOA. Just implementing an ESB is not SOA.
3. A pragmatic approach to gaining business process flexibility should be a key driver for SOA adoption. Tie SOA adoption to BPM initiative in the organization.
4. There is a strong need to develop both processes and people dimension in addition to technology dimension. This means tweaking the organizational models and putting in appropriate best practices in software processes to take care of SOA.

5. Architects need to get expertise in contract formation, in terms of richness of contract/interface descriptions, while developers need to develop skills in contract first development.
6. While it is important to get SOA projects to have a long term view for strategic reuse, in terms of getting buy in from key stakeholders for funding, it is advisable to identify touchpoints in the enterprise application portfolio, which can be addressed in short term with quick ROI, to provide proof of value to stakeholders. While a strategic view of SOA is important, waiting for all stakeholders to come to realize the value of SOA, and come to a common understanding of the benefits is a fruitless exercise.
7. Scope of deployment of SOA needs to be clearly understood for specific contexts, because SOA can be applied in varied contexts ranging from plain data and information integration, service oriented application integration, to more strategic initiatives like enterprise architecture transformation, with even mega transformation exercises like infrastructural virtualization via SOA.
8. To get appropriate governance infrastructure for SOA, it is important to understand the importance of both process and people dimension of governance and the run-time dimension. While people dimension of SOA governance is about institutionalizing appropriate IT organization models (let us call it SOA IT organization) amenable to SOA lifecycle (key is to understand the difference between the demand and the supply organizations for SOA, and the role of structures like demand side architecture review boards), the process dimension is about the overarching steps involving the different stakeholders in the SOA IT organization (finer realization of policies, responsibilities, roles etc.), and the run-time governance in the form of appropriate tools and techniques for working with rules, policies, and monitoring SLAs for services in the organization (a repository may be a key facet of this).
9. SOA should not be treated as a one-way journey with prescribed unidirectional steps, but should be treated as a continuous improvement journey with retrospective refinement of any step based on feedback from execution of a step. In this way, it is all about dynamic reconfigurability of systems with a view of constant improvement.
10. Treatment of SOA as yet another IT initiative can be minimized with the adoption of a "business - oriented" design of IT systems, in terms of business needs driving the service interfaces; this is something which is independent of how the need is fulfilled by an IT system. Within this business-oriented requirements step, if additional thought is put on unplanned reuse of the potential business function, that can go a long way in designing a future proof way of doing SOA.
11. Last but not the least, service identification and design needs to adopt middle-out or meet-in-the middle approaches for pragmatic SOA realization. A pure top-down step from business processes will leave a big disconnect from existing application infrastructure and a pure bottom up step will fall short of the business-oriented requirement of the interface.

12. Service design needs to accommodate a flexible loosely coupled formulation capable of handling change, in terms of minimality of effort required to adjust to changes in the system. Such design mechanisms need to put additional efforts in making sure that reuse is captured via schema and not via class as in conventional object oriented systems.

With these caveats we hope enterprises can adopt a pragmatic path to SOA with positive ROI and benefits. I hope that this column appeals to all SOA practitioners in terms of getting value from SOA deployments.

About the Author

Dr. Srinivas Padmanabhuni is a Principal Researcher and heads the SOA Centre of Excellence in SETLabs, Infosys. He specializes in Web services, Service Oriented Architecture, and Grid technologies alongside pursuing interests in semantic web, intelligent agents, and enterprise architecture. He has published extensively in international conferences and journals alongside speaking at thought leadership forums in the areas of SOA and Web services. Prior to joining Infosys, Dr. Srinivas has worked in multiple capacities in startups in North America. Dr. Srinivas holds a PhD in computing science from University of Alberta and undergraduate and graduate degrees in Computer Science from Indian Institutes of Technology (IIT) at Kanpur and Mumbai respectively 

Index

- .Net 13, 48, 81,88
- AJAX 65,
- Applications
 - B2B 13, 18
 - composite 30
 - custom 13, 18
 - ETL 13
 - legacy 13, 18, 23, 43, 45, 48, 53, 55, 76
 - packaged 13, 18
 - rich internet 65
- AMD 26
 - Pacifia 26
- Amazon 27-28
 - EC2 27, 28
 - S3 27, 28
- Automated
 - service negotiation 68
 - process orchestration 69
- Automatic
 - partner selection 68
 - process orchestration 68
 - service discovery 67-70, 72-74
 - service selection 68-70
 - supplier discovery 70
- BEA 17
- BPM Systems, also BPMS 29-30, 32, 34-36
- Business Process Management, also BPM 29-32, 34-36 56, 59, 61-62, 82, 92, 95
- Carve Out 23
- CICS 18, 58, 76
- Clorox 75
- COBOL 16, 58, 60, 76, 88-89
- Common Information Model, also CIM 24, 45, 51, 82
- Consumer Packaged Goods, also CPG 75, 77, 79-80
- CORBA 81
- Coupling
 - loose 10-12, 15, 32, 56, 59, 63, 65, 68
 - tight 3, 10
- Crosslets 24, 28
- Directed Acyclic Graph, also DAG, 23, 25
- Data Centers 21
- Data Centre Mark-up Language, also DCML 24
- DVD 78
- Dynamic Information Exchange 67-70, 72-74
- EIS 17-18
- EJB 16, 88-89, 93
- Enterprise Application Integration, also EAI 6, 14-17, 30, 32, 61, 81-82, 84, 94
- Enterprise Resource Planning, also ERP 30
- Enterprise Service Bus, also ESB 6, 13, 15-17, 19, 35, 64, 66, 79, 89, 95
- Future State Architecture, also FSA 54, 57-59
- Gartner 19, 52, 80
- Global Grid Forum 24
- Google 27, 58-60
 - maps 65
- Grid 21, 24, 26-28
 - middleware layer 23
- Grid Managed Entity, also GME 24
- Hibernate 5, 7, 8
- Hyper-visor 26,
- IBM 17, 31, 42, 94
 - DB2 Adapter 48
 - iSeries 25
 - Mainframes 25, 47, 58
 - MQ Series Adapter 48
 - WBI Server Foundation (Version 5) 59
 - zSeries 25
- Integration Approaches
 - hub-spoke 32
 - point-point 32
 - process-based 32

- Intel
 - VT Technology 26
- IT
 - architecture/s 8-10, 31
 - investment/s 58, 75
 - service management, also ITSM 65
- ITIL 24, 65
- IT Systems 9, 10, 12, 29-31, 43-44, 96
 - agile 9
 - flexible 9
- J2EE 13, 17, 58, 81, 89, 93,
- JBoss 17
- Knowledge Management 41
- Kraft 75
- Layer
 - abstraction 61
 - application 4, 21
 - application management 24
 - business component model 7
 - business service 82-85
 - components 4
 - data access 5, 6
 - data source 6,
 - grid middleware 23
 - infrastructure 4, 6, 21
 - integration 4, 82,
 - object mapping 7
 - orchestration/choreography 56
 - presentation 56
 - process 4, 31, 32,
 - SDS 6,
 - service connectivity 48
 - shared business services 6
 - technical service 82, 84-85
 - virtual 15
 - virtual resource 22
 - webservices wrapper 5
- Metamorphosis 37
- Microsoft 17, 19, 27, 43, 47, 94
 - Business Scorecard Manager 49
 - Operations Manager 49
- Migration 45, 84, 86
 - code 76
 - legacy 76
 - plan 40-41
- Morph 5, 7, 8
- NetManage 58
- Ontology 69-74
 - based dynamic data 69
 - based data transformation 71
 - enabled 68
 - wrapped XML Schemas 72
 - data transformation
 - domain 70
 - information model 72
 - semantic web services, also SWSO 71
 - web services modeling, also WSMO 81
- Open Grid Services Architecture, also OGSA 24
- Optimization 35, 46, 69-70, 72-73
 - approach 75
 - form-factor 22
 - IT 21, 27
 - process 50
 - to-be 34
- Oracle 13, 17-20, 48
 - E-Business Suite 11i 18,20
 - Fusion Middleware 18
 - Project Fusion 18
 - Warehouse Builder 19
- OWL-S 71
- P&G 75
- PeopleSoft 13, 18, 48
- PepsiCo 75
- Planning 47, 53, 55, 57, 58
 - capacity 27
 - demand 78
 - new product 78
 - perspective 95
 - promotional 78
 - replenishment 78
 - stage 59
 - strategic IT 12

time phased 77
 Polymorphic 54
 Process 19, 30-38, 44, 46, 48-50, 63, 64, 71, 76, 79, 83, 87, 92-93, 96
 -based 30
 -centric 30
 analysts 34
 architecture
 as-is 34
 automated/ic orchestration 68-69
 automation 54, 67, 73-74
 batch 19
 BPEL 35
 BPMS Modeler 35
 buying 70
 cycle time 35
 data mediation 74
 dimension 96
 dynamic orchestration 74
 item procurement 69
 latencies 48
 quality of service, also QOS 70, 93
 runtime orchestration 67-72, 74
 Semantics 70, 72-73
 specialist 63
 static orchestration 68, 72
 to-be 34
 transaction 7
 RFID 78-79
 RUMBA 58
 Storage Area Networks, also SAN 22-23
 SAP 13, 18-19, 48
 NetWeaver 18-19
 Biztalk adapter 47
 Service Level Agreements, also SLAs 23-24, 30, 35, 38, 44, 46, 50-51, 96
 Shared Data Services, also SDS 3, 5, 8, 19
 layer 6
 framework 6
 platform 6-8
 Service Oriented Infrastructure, also SOI 21-27
 Shared Data Services 3, 5, 8, 19
 Siebel 13, 18, 48
 Software as a service, also SaaS 27
 Storage Virtualization 23
 Struts 58
 SUN 17, 94
 Supply Chain Management, also SCM 75-76
 Technology Competency Center, also TCC 37, 39-42
 User Transaction 7-8
 VMWare 23, 25
 Websphere 31, 94
 Studio 58
 BI Modeler 58
 Web 2.0 65
 Web Service Enablement 76
 Web Services 6-8, 15, 17-20, 24, 28, 39-41, 44, 56, 58, 59, 65-66, 71, 73-74, 76 79-81, 83, 86, 89, 94, 97
 API 58-59
 BPEL 35
 management framework 24
 modeling ontology, also WSMO 71
 program management office, also PMO 39
 resource framework , also WSRF 24
 semantic ontology , also SWSO 71
 WSDL 3, 56, 68, 71, 74, 90
 XQuery 5, 7, 8
 XML 5-8, 16, 18, 24, 56, 66-68, 72, 90
 schema mapping, 6
 XSLT 67-68

SETLabs Briefings

BUSINESS INNOVATION through TECHNOLOGY

Editor
George Eby Mathew

Consulting Editor
Srinivas Padmanabhuni

Deputy Editors
Praveen Bhasa Malla
Sumanta Deb

Copy Editor
Anupama Gummaraju

Graphics/Web Editors
Ramesh Ramachandran
Sanjay Eknath Nayak

ITLS Program
Ajay Kolhatkar

Marketing Managers
Jacob Varghese
Kanupriya Sindhu
Vijayaraghavan T S

Production Managers
Sudarshan Kumar V S
V H Suresh Kumar

How to Reach Us:

Email:
SETLabsBriefings@infosys.com

Phone:
+91-80-41173896

Fax:
+91-80-28520740

Post:
SETLabs Briefings,
B-19, Infosys Technologies Ltd.
Electronics City, Hosur Road,
Bangalore 560100, India

Subscription:
vijaytsr@infosys.com

Rights and Permission:
george_mathew@infosys.com

Reprints:
jacob_varghese@infosys.com

Editorial Office: SETLabs Briefings, B-19, Infosys Technologies Ltd.
Electronics City, Hosur Road, Bangalore 560100, India
Email: SetlabsBriefings@infosys.com <http://www.infosys.com/SETLabs/briefings/>

SETLabs Briefings is a quarterly published Infosys' Software Engineering & Technology Labs (SETLabs) with the objective of offering fresh perspectives on boardroom business technology. The publication aims at becoming the most sought after source for thought leading, strategic and experiential insights on business technology management.

SETLabs is an important part of Infosys' commitment to leadership in innovation using technology. SETLabs anticipates and assesses the evolution of technology and its impact on businesses and enables Infosys to constantly synthesize what it learns and catalyze technology enabled business transformation and thus assume leadership in providing best of breed solutions to clients across the globe. This is achieved through research supported by state-of-the-art labs and collaboration with industry leaders.

Infosys Technologies Ltd (NASDAQ: INFY) defines, designs and delivers IT-enabled business solutions that help Global 2000 companies win in a flat world. These solutions focus on providing strategic differentiation and operational superiority to clients. Infosys creates these solutions for its clients by leveraging its domain and business expertise along with a complete range of services. With Infosys, clients are assured of a transparent business partner, world-class processes, speed of execution and the power to stretch their IT budget by leveraging the Global Delivery Model that Infosys pioneered. To find out how Infosys can help businesses achieve competitive advantage, visit www.infosys.com or send an email to infosys@infosys.com

© 2007, Infosys Technologies Limited

Infosys acknowledges the proprietary rights of the trademarks and product names of the other companies mentioned in this issue. The information provided in this document is intended for the sole use of the recipient and for educational purposes only. Infosys makes no express or implied warranties relating to the information contained herein or to any derived results obtained by the recipient from the use of the information in this document. Infosys further does not guarantee the sequence, timeliness, accuracy or completeness of the information and will not be liable in any way to the recipient for any delays, inaccuracies, errors in, or omissions of, any of the information or in the transmission thereof, or for any damages arising there from. Opinions and forecasts constitute our judgment at the time of release and are subject to change without notice. This document does not contain information provided to us in confidence by our clients.

NOTES

NOTES

NOTES

Authors featured in this issue

ANANTHALAKSMI VALLAPUZHA

Ananthalakshmi Vallapuzha is a Senior Technical Architect with the Microsoft Technology Center, Infosys. She can be reached at ananthalakshmi@infosys.com.

ANUBHUTI BHARILL

Anubhuti Bharill is a Senior Analyst with the Europe Middle East and Africa business unit, Infosys. She can be reached at anubhuti_bharill@infosys.com

BHAVIN RAICHURA

Bhavin Raichura is a Technical Architect in the Hi-Tech and Discrete Manufacturing business unit, Infosys. He can be reached at bhavin_raichura@infosys.com

BIJI NAIR

Biji Nair is a Project Manager with the Europe Middle East and Africa business unit, Infosys. She can be reached at biji_nair@infosys.com

BIJOY MAJUMDAR

Bijoy Majumdar is a Technical Architect at the Web Services Center of Excellence in SETLabs, Infosys. He can be reached at bijoy_majumdar@infosys.com

BINOOJ PURAYATH

Binooj Purayath is a Principal Architect with the Technology Consulting Group, Infosys where he focuses on SOA and Legacy Modernization. He can be reached at binooj_purayath@infosys.com

HARIPRASAD NELLITHEERTHA

Hari Prasad Nellitheertha is a Technical Architect with the Grid and High Performance Computing research team at SETLabs, Infosys. He can be reached at hariprasad_v01@infosys.com

JAI GANESH

Jai Ganesh, PhD, is a Senior Research Associate with the SOA Center of Excellence in SETLabs, Infosys. He can be reached at jai_ganesh01@infosys.com

KRISHNENDU KUNTI

Krishnendu Kunti is a Technical Specialist with the Web Services CoE, SETLabs, Infosys. He can be reached at krishnendu_kunti@infosys.com

MANISH SRIVASTAVA

Manish Srivastava is a Principal Architect with Infosys Technologies. He is leading the solution development efforts for the Infosys Microsoft joint technology led transformation program called Catalytic IT. He can be reached at manishsv@infosys.com.

MOHIT CHAWLA

Mohit Chawla is a Software Engineer with the Web Services CoE, SETLabs, Infosys. He can be reached at mohit_chawla@infosys.com

PARAMESWARAN SESHAN

Parameswaran Seshan is a Senior Technical Architect with SETLabs, Infosys. He can be reached at parameswaran_seshan@infosys.com

PETER JARMAN

Peter Jarman is a Principal Architect with Infosys Australia. He has 25 years experience in IT across a number of domains. He can be reached at peter_jarman@infosys.com

SANDEEP KARAMONGIKAR

Sandeep Karamongikar is a Principal Architect and heads the Infosys J2EE Center of Excellence at SETLabs. He can be reached at sandipmk@infosys.com.

SHAURABH BHARTI

Shaurabh Bharti is a Junior Research Associate with the Web Services SOA CoE in SETLabs, Infosys. He can be reached at shaurabh_bharti@infosys.com

SHREYAS KAMAT

Shreyas Kamat is a Principal Architect with the Technology Consulting Group, Infosys. He has several years of experience as a Technology Strategist and has helped many organizations across industries achieve IT transformation. He can be reached at shreyas_kamat@infosys.com

SHUBHASHIS SENGUPTA

Shubhashis Sengupta, PhD, is a Principal Researcher with Grid and HPC research team at SETLabs, Infosys. Shubhashis has a decade of research, publication, consulting and solutions experience. He can be reached at shubhashis_sengupta@infosys.com

SHYAM KUMAR DODDAVULA

Shyam Kumar Doddavula is a Senior Technical Architect at Infosys J2EE Center of Excellence. He can be reached at shyamkumar_d@infosys.com

SRIKANTH SUNDARRAJAN

Srikanth Sundarajan is a Technical Architect with Grid and HPC research team at SETLabs, Infosys. He can be reached at srikanth_sundarajan@infosys.com

TERANCE DIAS

Terance Dias is a Software Engineer at the Web Services CoE in SETLabs, Infosys. He can be reached at terance_dias@infosys.com

UJVAL MYSORE

Ujval Mysore is a Software Engineer at the Web Services CoE in SETLabs, Infosys. He can be reached at ujval_mysore@infosys.com

VAIDYANATHA SIVA

Vaidyanatha Siva is a Principal Architect and heads Technology Architecture and Innovation at the Retail, CPG and Distribution business unit, Infosys. He can be reached at siva_vaidyanatha@infosys.com

VIKRAM SITARAM

Vikram Sitaram was a Software Engineer at the Web services CoE, in SETLabs, Infosys.

VISHAL PURI

Vishal Puri is a Senior Technical Architect in the Retail, CPG and Distribution business unit, Infosys. He can be reached at vishal_puri@infosys.com

Subu Goparaju
Vice President
and Head of SETLabs

"At SETLabs, we constantly look for opportunities to leverage technology while creating and implementing innovative business solutions for our clients. As part of this quest, we develop engineering methodologies that help Infosys implement these solutions right first time and every time".

For information on obtaining additional copies, reprinting or translating articles, and all other correspondence, please contact:

Telephone : 91-80-41173878

Email: SetlabsBriefings@infosys.com

© SETLabs 2007, Infosys Technologies Limited.

Infosys acknowledges the proprietary rights of the trademarks and product names of the other companies mentioned in this issue of SETLabs Briefings. The information provided in this document is intended for the sole use of the recipient and for educational purposes only. Infosys makes no express or implied warranties relating to the information contained in this document or to any derived results obtained by the recipient from the use of the information in the document. Infosys further does not guarantee the sequence, timeliness, accuracy or completeness of the information and will not be liable in any way to the recipient for any delays, inaccuracies, errors in, or omissions of, any of the information or in the transmission thereof, or for any damages arising there from. Opinions and forecasts constitute our judgment at the time of release and are subject to change without notice. This document does not contain information provided to us in confidence by our clients.

Infosys[®]

POWERED BY INTELLECT
DRIVEN BY VALUES