Proceedings of the 5th WSEAS Int. Conf. on Software Engineering, Parallel and Distributed Systems, Madrid, Spain, February 15-17, 2006 (pp48-54)

Service Oriented Architecture for Enterprise Applications

SHANKAR KAMBHAMPATY and SATISH CHANDRA
Technology Architecture Group
Satyam Computer Services Limited
C5, TSR Towers, Raj Bhavan Road
Somajiguda, Hyderabad – 500 082
INDIA
{Shankar_Kambhampaty, Satish_Chandra} @satyam.com http://www.satyam.com

Abstract: - While large organizations develop applications with new technologies, it is often necessary to leverage the investments made in legacy applications and reuse business functionality provided by them. Hence, such organizations need scalable distributed applications that are integrated based on an enterprise-wide strategy. Service Oriented Architecture (SOA) is an answer to address these requirements. With the advent of web services, SOA based enterprise applications have become vendor independent to a large extent. The paper provides insights gathered through providing SOA based solutions in consulting engagements. It discusses the Strawman Architecture for Enterprise-Wide Service Oriented Architecture. Reference Architectures for SOA based Enterprise Applications on J2EE and .NET platforms, using Design Patterns, are detailed. It presents the SOA methodology recommended to customers. The solutions discussed in the paper are reusable across domains and the implementing organizations would benefit in terms of considerable time and cost savings.

Key-Words: - Service Oriented Architecture, Enterprise Application, Legacy Systems Integration, Enterprise Service Bus, Strawman Architecture, SOA Based Enterprise Applications, Design Patterns, Reference Architecture, SOA Best Practices.

1 Introduction

An enterprise usually has a large number of applications. Each of the applications caters to the specific needs of different units within the enterprise. It is imperative that the applications collaborate to provide a uniform view of the enterprise and also leverage on the investments made in legacy applications, through reuse of business functionality provided by them. Service Oriented Architecture (SOA) enables exposing the reusable functionality of an application as services that can be consumed by other applications.

A central aspect of Service Oriented Architectures is the loose coupling between applications (services) that is achieved when services publish their functional and non-functional behavioral characteristics in a standardized, machine-readable format as indicated in [2].

This paper is based on customer engagements that involved recommending and providing SOA based solutions. In this paper, large-scale business applications built on distributed systems are referred to as “Enterprise Applications”. It discusses a Strawman Architecture that serves as a starting point for developing Service Oriented Architecture for Enterprise Applications. The paper also discusses Reference Architectures for SOA based J2EE and .NET applications, using Design Patterns. A unique feature of the Reference Architectures is the adoption of best practices relevant to the platform, independent of specific implementation mechanisms. The solutions and reference architectures discussed in the paper are reusable across domains and the implementing organizations would benefit in terms of considerable time and cost savings.

The rest of this paper is organized as follows. Section 2 discusses the concept of Service Oriented Architecture and the motivation for considering SOA as a viable option in developing enterprise applications. Section 3 discusses the Strawman Architecture that serves as a starting point for the development of Service Oriented Architecture for large Enterprises. Section 4 describes the Reference Architectures for SOA based applications developed on J2EE and .NET platforms, using Design Patterns. Section 5 presents SOA Best Practices and Section 6 concludes the paper.

2 Service Oriented Architecture

2.1 Service Oriented Architecture

Service Oriented Architecture (SOA) is an approach in which an application exposes services that are consumed by clients. SOA differs from the more general client/server model in its emphasis on loose coupling between software components. Heterogeneous systems, both in terms of hardware and software, can communicate to leverage on the business functionality (rather services) provided by each other. The higher level of abstraction provides a strategic advantage of facilitating more focus on the business requirement [12].

A business service can be mapped to one or more technology services for implementation. From a technology perspective, a service is a defined interface with input and output parameters.

Services can be implemented in various ways. Tools like RMI in Java and Remoting on .Net platform can provide services. The problem with these technologies is that they are platform specific and are not interoperable. There are other technologies like Web Services, which can be used across platforms and networks. Web Services work with technologies like XML, SOAP, WSDL and UDDI. They are platform-independent, vendor independent and language independent and therefore are better suited for exposing and consuming services.

With the advent of web services, SOA enabled enterprise applications have become vendor independent, to a great extent. As predicted in [5], IBM, SUN and BEA have recognized Web Services as a major technology and have products that support SOA.

2.2 Motivation for Service Orientation

From a business perspective, it becomes necessary for enterprises to protect the investments already made in existing solutions despite the need for new applications that incorporate changing technologies and business requirements. From a technology perspective too, it becomes necessary to have a scalable architecture that allows for reuse of business processes implemented in the existing solutions. Also, several customers require an approach to plan an enterprise wide strategy that leads to a scenario of well integrated IT systems within their organization.

One of the options to address these requirements is the recommendation of SOA. SOA, when properly implemented, addresses the business and technical integration considerations elegantly.

With industry analysts such as Gartner [3] predicting that over 80% of the business applications sold between 2005 and 2008 will be based on the principles of Service Oriented Architecture (SOA), it becomes imperative to consider SOA for Enterprise Applications to benefit from the product/technology offerings that support SOA and remain competitive.

3 Strawman Architecture for Enterprise-Wide SOA

Typically, an Enterprise is a large organization having forward and backward linkages with other organizations. It deals with a variety of external agencies. The Enterprise would, in all likelihood, be involved in B2B and B2C transactions. This section presents Strawman Architecture for developing Enterprise-Wide SOA. Strawman Architecture is the initial architecture that serves as a starting point for developing the target architecture. It is refined over number of iterations and results in the development of the target architecture.

The Strawman Architecture presented here could serve as a starting point for developing a SOA based solution for an Enterprise.

In this section, discussion is not restricted to a particular technology. A generic view of a service is considered.

The following figure represents Strawman for Enterprise-Wide SOA.

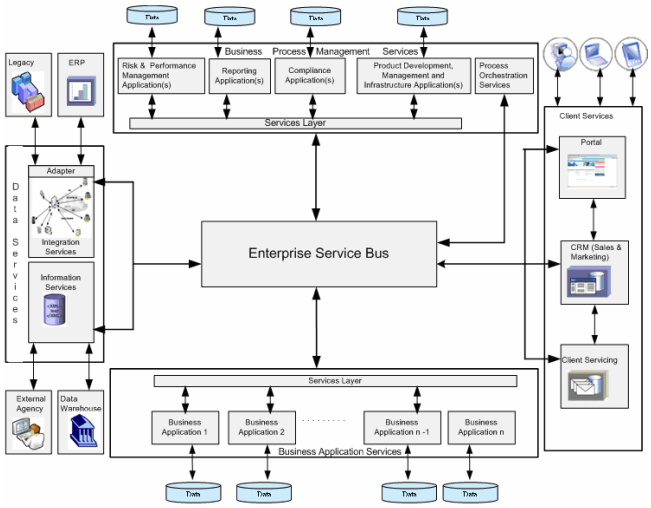

Fig.1 Strawman for Enterprise-Wide SOA

It can be seen from the figure that the enterprise has several applications that need to talk to each other. A key feature of the architecture is the use of Enterprise Service Bus (ESB) that enables a smooth communication between the applications. ESB is often described as a product, especially in the marketing literature of various vendors. But, in a strict sense, ESB is an architectural style. The Strawman architecture for Enterprise-Wide SOA has the ESB as the heart of communication between applications [1,6]. The business processes of the enterprise may be exposed as services that can be classified as follows:

i. Client Services
ii. Business Process Management Services
iii. Business Application Services
iv. Data Services

The services exposed by each of these categories can be specified as a set of messages of a XML framework. The Services Layers implement the necessary functionality to transform the request from a service consumer application to a suitable format, communicating with the application/product implementing the functionality and returning the result.

A brief description of the services is as follows:

i) Client Services

Client Services enable and deliver content to users. These include the Enterprise Portal, CRM functionality of Sales and Marketing and the distribution functionality including Client Reporting.

ii) Business Process Management Services

Business Process Management Services handle orchestration of business processes implemented in business applications, provide functionality of Compliance and Reporting and also infrastructure services such as security. These services control the flow and interaction of business services.

iii) Business Application Services

The Business Application Services encapsulate access to the functionality of the business processes of the enterprise implemented in the various business applications.

iv) Data Services

Data Services encapsulate access to data in various sources. There are typically two types of Data services that need to be provided in an enterprise:

Information Services that transform and replicate data stored across the enterprise in various data sources.

Integration services that provide the ability to interact with structured and unstructured data services (like the databases and flat files). These services establish connectivity of ESB with Legacy and other packaged implementations such as ERP and encapsulate access to functionality and data in those applications.

The key aspects of the architecture are in line with the following four tenets of service orientation [7]:

a) Boundaries are Explicit.
b) Services are Autonomous.
c) Services Share Schema and Contract.
d) Service Compatibility is based on Policy.

4. Reference Architectures for Enterprise Applications

The Reference Architecture based approach has been proven to be useful in several customer engagements.

According to Rational Unified Process (RUP), “a Reference Architecture, is in essence, a predefined

architectural pattern, or set of patterns, possibly partially or completely instantiated, designed, and proven for use in particular business and technical contexts, together with supporting artifacts to enable their use. Often, these artifacts are harvested from previous projects"[4].

The Business applications shown in Fig.1 need to support the Service Oriented Architecture for the enterprise. Many of the Enterprise applications in organizations today are developed for the J2EE and .NET platforms. For these platforms, use of Reference architectures for the Business applications that conform to the Strawman (in Fig. 1) can help in adopting best practices and in reducing the effort for architecting and implementation. This section discusses Reference Architectures for SOA based Enterprise applications for J2EE and .NET platform which are based on Design Patterns.

4.1 SOA based Enterprise Application – J2EE

The logical architecture of the enterprise application can be represented as layers and services. Layers group a set of services, and each layer has public interfaces and provides a cohesive set of services. The Reference Architecture of a SOA based J2EE Application is shown in Fig. 2.

The Presentation Layer of a thin client application is implemented using the MVC pattern. The request from the browser of a thin-client application is passed to the Controller, which then passes the request to the classes implementing the Business Delegate pattern.

Struts Open Source Framework is recommended for implementing the MVC pattern shown in the Reference Architecture.

The Business Layer implements business processes for the different modules, including static data maintenance, transactions, workflow, batch processing and report generation. To encapsulate the business logic and to decouple it from the Presentation Layer, the Business Delegate pattern is used.

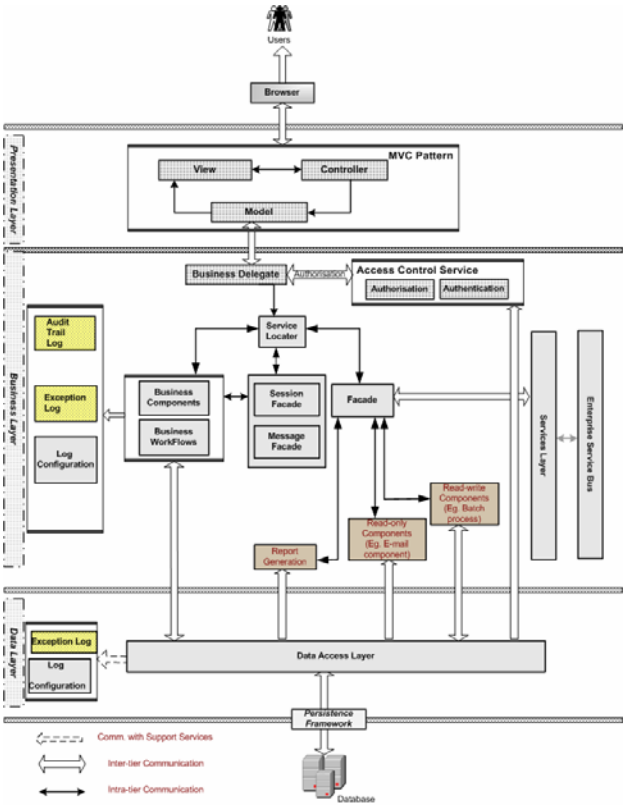

Fig. 2 Reference Architecture of a SOA based J2EE Application

The Access Control Service includes authentication of users and authorization of access to functionality of the application based on the privileges defined for the users. Use of JAAS framework is recommended for authentication and authorization. The classes that implement Business Delegate call Access Control Service to authorize the user for every request made by the user.

Session Façade exposes business components implemented as Enterprise Java Beans (EJB).

Service Locator provides mechanisms to transparently locate business components and services in a uniform manner. This is used to implement and encapsulate service and component lookup. The Service Locator hides the implementation details of the look up mechanism and encapsulates related dependencies.

Message Façade is used for asynchronous communication between different Java components.

The classes implementing the Façade pattern provide an interface for modules such as the Services Layer, Reporting module, Batch-processing module, e-mail, etc.

The Services Layer encapsulates the service-oriented communication between the system under context and all the external systems, for which it is either a consumer or producer of services.

The Services Layer basically enables the location and communication between a service consumer and its corresponding service provider. The Services Layer calls Enterprise Service Bus (ESB) for communication with other applications.

The Data Access Layer connects to database using the persistence framework. The data access mechanism could be implemented using any of the following technology options:

- JDBC
- Container managed entity EJB
- Bean managed entity EJB
- Object-relational mapping tools, like Hibernate

4.2 SOA based Enterprise Application – .NET

As in the case of J2EE based application, the logical architecture of the .NET application can be represented as layers and services. Layers group a set of services, and each layer has public interfaces and provides a cohesive set of services. The Reference Architecture of a SOA based .NET Application is shown in Fig. 3.

The request for business functionality, which comes from the client over HTTP, is passed to the Service Interface of the Business Layer. The Business Layer implements business processes for the different modules, including batch processing and report generation. To encapsulate the business logic and to decouple it from the Presentation Layer, the Service Interface pattern is used.

The Access Control Service includes authentication and authorization. The Service Interface calls Access Control Service to authorize the user for every request made by the user.

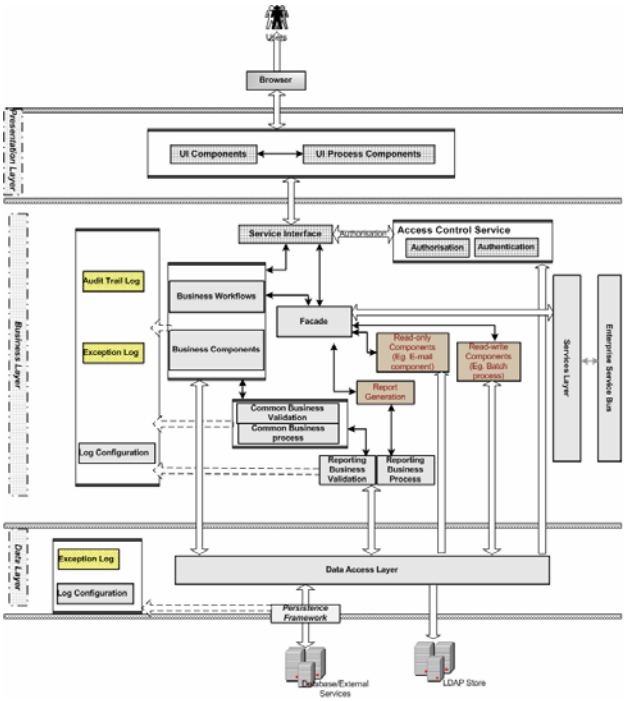

Fig. 3 Reference Architecture of a SOA based .NET Application

The classes implementing the Façade pattern provide an interface for modules such as the Services Layer, Reporting module, Batch-processing module, e-mail, etc.

The Services Layer encapsulates the service-oriented communication between the system under context and all the external systems, for which it is either a consumer or producer of services.

The Services Layer enables the location and communication between a service consumer and its corresponding service provider. The Services Layer calls the Enterprise Service Bus (ESB) for communication with other applications.

The Data Access Layer connects to database using the persistence framework such as ADO.NET. The data access components encapsulate the data sources from the business layer.

5 SOA Best Practices for the Enterprise

The Best practices that have been identified for SOA implementation projects include the following:

i. Assessing the suitability of SOA
ii. Developing an SOA Strategy
iii. Preparation of SOA Guidelines
iv. Defining an SOA Roadmap

i) Assessing the suitability of SOA

One of the Best Practices that have been identified, is to go beyond the hype of SOA, and establish clearly that SOA is the right architecture to address the given business requirements.

In order to establish the suitability of SOA, it is important to look at SOA in the context of the business of Enterprise in addition to the context of specific applications for which SOA is being considered. When Reusability and EAI are key concerns of the enterprise, SOA is a good option.

It is also important that the enterprise considering SOA does a COST/Benefit analysis and establishes the Return on Investment (ROI). SOA implementations require investments in service enabling existing applications and in ESB infrastructure and commitment at enterprise level for such investments is only possible when the ROI is justified.

ii) Developing a SOA Strategy

Another Best Practice that has been identified in SOA implementations is for an Enterprise to develop a SOA Strategy.

An example of a SOA strategy suitable to many organizations is as follows:

a) As part of the strategy, a framework is defined so that different applications exposing services know precisely how to define a service. Typically, this would be an XML framework. Such a framework could form the basis of the design of a Services Layer for applications participating in SOA. The next step in the strategy is to have consumer applications invoke the services exposed by the Services Layer as Web Service calls.

By this approach, each service constitutes a "message" that the consumer application invokes and each type of message has an associated Schema. A service invocation will be intercepted first by the Services Layer that recognizes the "type of message". The Services Layer then converts it into the appropriate function call of the service exposed by the application and fires the function. The response message is converted to the appropriate format by the Services Layer and passed back to the consumer.

b) With a framework defined for exposing and consuming services, the final step of the strategy is one involving a mechanism for integration and to search and locate services exposed by applications.

c) Enterprise Service Bus (ESB) is an important mechanism to be considered as part of the strategy for integration as it provides capabilities like content based routing, message transformation that work with enterprise applications and service orchestration.

iii) Preparation of SOA Guidelines

Before embarking on a SOA implementation with in an enterprise it is important to come up with a set of guidelines. It helps develop a common view among key stakeholders across the enterprise and also helps in building consensus.

Essentially, this would involve preparation of Architecture "Blue Book" that has architecture principles and reference architecture for the enterprise. The guidelines would include strategic Technology stack for the enterprise and also would refer to model SOA implementations done in and outside the enterprise.

The Architecture Blue Book can be the instrument to get buy-in from key stakeholders and to ensure their views are addressed suitably. It is also important to establish a Governance Model and identify a "Champion" who ensures that different units in the enterprise implement SOA as per the guidelines.

iv) Defining a SOA Roadmap

A roadmap that targets SOA implementation in an incremental manner needs to be defined.

As a first step, less complex activities can be targeted first. One of the units in the enterprise can be chosen for a pilot and a few applications can be service enabled. A Proof-of-Concept (PoC) involving an ESB can also be developed.

For developers, service-orientation means learning new programming models and techniques.

Changing developers' mindsets to effectively use these new models might be the biggest hurdle for successful adoption of SOA as discussed in [9].

The next step in the Roadmap could be to target medium complexity. Applications in multiple departments can be made to participate in SOA with services being exposed and consumed through an ESB.

The final step in the Roadmap could be to implement SOA at Enterprise level.

6 Conclusion

This paper has captured experiences in providing SOA based solutions for enterprise applications. Strawman Architecture which would serve as a beginning point is discussed. Reference Architectures for SOA based Enterprise applications have been presented in the paper. Best practices have been discussed. The use of SOA would reduce the time-to-integrate any new application. Further, the use of techniques and tools discussed in the paper would reduce the time to arrive at the architecture and design of the SOA based solution.

7 References

[1] Rick Robinson, *Understanding Enterprise Service Bus scenarios and solutions in Service-Oriented Architecture, Parts 1 to 8,* www-128.ibm.com/developerworks/xml/library/

[2] Nirmal K Mukhi Ravi Konuru and Francisco Curbera,, *Cooperative Middleware Specialization for Service Oriented Architectures,* International World Wide Web Conference archive Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters table of contents New York, NY, USA SESSION: Semantics and discovery table of contents Pages: 206 - 215 Year of Publication: 2004.

[3] Charles Abrams and David Mitchell Smith, *Service-Oriented Business Applications Show Their Potential,* Gartner Research.

[4] Paul R. Reed, Jr., *Reference architecture: The best of best practices,* Published on IBM Website, Jan 30 2004.

[5] Michael Stal, *Web Services: Beyond Component-Based Computing,* Communications of the ACM, October 2002.

[6] Mark Endrei, Jenny Ang, Ali Arsanjani, Sook Chua, Philippe Comte, Pål Krogdahl, Min Luo, Tony Newling, *Patterns: Service- Oriented Architecture and Web Service,* IBM Red Book, www.ibm.com/redbooks.

[7] John Evdemon, *The Four Tenets of Service Orientation,* article from bpminstitute.org at http://www.bpminstitute.org/articles/article/article/the-four-tenets-of-service-orientation.html

[8] Shalil Majithia, David W. Walker, W.A.Gray, *A Framework for Automated Service Composition in Service-Oriented Architectures,* www.wesc.ac.uk/resources/publications/

[9] Gregor Hohpe, *Developing Software in a Service-Oriented World,* White Paper from ThougthWorks, www.enterpriseintegrationpatterns.com/docs/SOA_World.pdf

[10] Harold Carr, *The PEPt Service Oriented Architecture,* 2nd International Conference on Service Oriented Computing New York City, NY, USA November 15-18, 2004 (ICSOC 2004).

[11] Ali Arsanjani, *Service-oriented modeling and architecture,* www128.ibm.com/developerworks/webservices/library/ws-soa-design1/

[12] Soumen chatterjee, *Messaging Patterns in Service Oriented Architecture, Part 1,* www.msdn.microsoft.com/library/