International Journal of Advance Research in Science and Engineering
Vol. No.5, Issue No. 04, April 2016
www.ijarse.com
IJARSE
ISSN 2319 - 8354

SERVICE ORIENTED ARCHITECTURE (SOA) AND SPECIALIZED MESSAGING PATTERNS ORIENTED MIDDLEWARE WITH MULTIPLE TYPES OF SOA APPLICATIONS

Er. Govind Dev Lodha[1], Er. Vijay Malav[2] , Dr. Amit Sharma[3]

[1,2, 3] *Associate Professor Department of Computer Science & Engineering,*
Vedant College of Engineering & Technology ,Bundi ,Rajasthan, (India)

ABSTRACT

In this research Paper I am focusing on the To A Service Oriented Architecture (SOA) is set of principles that define an architecture that is loosely coupled and comprised of service providers and service consumers that interact according to a negotiated contract or interface. These services provide the Interfaces to Applications in the IT landscape. The primary goal of SOA is to expose application functions in a standardized way so that they can be leveraged across multiple projects. This approach greatly reduces the time, effort and cost it takes to maintain and expand solutions to meet business needs.

It is important to note that SOA is not an off-the-shelf technology; rather it is a way of architecting and organizing IT infrastructure and business functionality. SOA is a paradigm for designing, developing, deploying and managing discrete units of logic i.e. services, within a computing environment, and has the following attributes:

• Functionality is organized as a set of modular, reusable shared services

• Services have well-defined interfaces and encapsulate key business processes

• Customer facing solutions serve as customized views of these services for different segments, and can access these shared services as needed.

• The reusable shared services are built without making any assumptions of who (portal or another service) will consume these services

SOA is vital to an enterprise as an architecture paradigm because it allows the business to become agile and respond rapidly to changes and challenges it faces internally and externally from its customers, stakeholders and various federal mandates and regulations to accomplish mission critical objectives.

Keywords: component: Agent technology, Cross-platform, Message Oriented Middleware (MOM), Service Oriented Architecture (SOA), Web Services.

I. INTRODUCTION

Service-Oriented Architecture (SOA) is a software framework for integrating loosely-coupled and distributed services into an interconnected workflow process. SOA-based systems may integrate both heritage and new

services created by an organization, either internally or via external trusted partners. It allows an application to be connected and executed across multiple platforms at geographically distributed locations. Within an SOA system, services are composed and integrated using different standard and specifications of deployment to enable flexible connections and collaborations among SOA applications. The loose coupling of services and reusable properties across multi-platforms make SOA as very flexible software architecture. Moreover, the communication between legacy systems becomes significant due to the rapid growth of software and technology development. This is because the purpose of the new software innovation is to support the limitation of legacy system with a new specifications and requirements. Therefore, SOA can help an organization to overcome integration issues between legacy and current system with specific technological deployment such as by using extensible Markup Language (XML) file for sending messages across different software applications.

The problem of SOA applications is, its implementation using different web service standards, specifications or platforms which need to acquire the services of others in achieving the specified goals. As such, cross-platform Communication is critical. It means the ability of diverse application systems to work (inter-operate) or communicate together efficiently. Without an ability of cross-platform communication, the difference SOA application will not able to connect or communicate for any purpose of transmission. In addition, the loosely coupled environment of SOA, separate resources do not need to know the details of each applications, but the problem is that they need to have enough common ground to communicate and exchange messages [5]. In this respect, there is a growing research interest in the cross-platform communication on different approaches and standards to implement SOA applications (i.e XML schema, SOAP, WSDL, and WS-* (WS Splat)). [6]. In spite of this, the cross-platform communication still has some of unique challenges for instance, there are no standard requirements and specifications to implement SOA applications that support multiple types of applications during runtime.

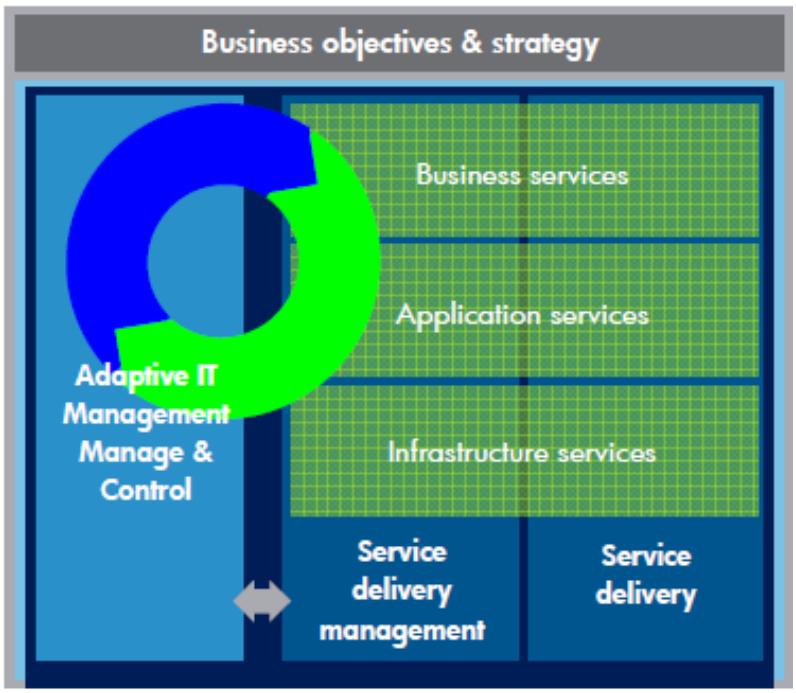

Figure 1: Adaptive Enterprise Darwin Architecture

The main challenge that limits the growth of SOA is because of the communication is restricted to a specific type of SOA applications. Difference SOA applications may deploy or implement using different specifications and requirements as mention above. Therefore, the communication between them becomes complicated and cross-platform communication is necessary to enable the communication between different types of SOA applications. Attempting to address an increasing problem of cross-platform communication among SOA applications, the existing cross-platform framework within SOA environment is investigated with a goal to evaluate the communication process between different SOA applications. It involves different phases of the cross-platform communication processes. These have been segregated to identify specifications and requirements of the communication attributes to be included into cross-platform communication framework which capable to support multiple type of SOA applications during runtime. Furthermore, this work has conducted a comparative study among existing cross-platform communication frameworks within SOA context. As a result, most of existing frameworks are supported only one-to-one type of communication, which is not support more than two different types of SOA applications during runtime.

II. BENEFITS OF SOA

SOA promotes loose coupling of software resulting in services that are easier to integrate as they utilize lower-cost tools and have standards based exchange formats and interface. Some of the key benefits that an enterprise can achieve from SOA are:

2.1 Agility to collaborate – SOA provides ability to securely and easily share information, with partners and stakeholders by presenting a standard coarse grained service which any authorized business partner can use, there by extending the enterprise.

2.2 Agility to adapt – SOA promotes the ability to rapidly reconfigure the business process by selecting from the available set of services, thereby providing the agility to adapt to the business requirements introduced by stakeholders and business partners.

2.3 Reduction of cost - SOA primitives are standards based For example, WSDL, SAML, SOAP & UDDI, providing a modular architecture, thereby enabling sharing and reuse of services.

2.4 Improvement in efficiency - SOA promotes a modular enterprise, promising a high degree of reusability of business services, ensuring consistency.

2.5 Better business operations: SOA based services enable a common architecture and approach to be pervasive in an enterprise containing heterogeneous legacy systems.

2.6 Ease of introducing new technologies – SOA's coarse grain modularity allows enhancements to service by introducing new more efficient technologies as they are introduced without actually changing the service interface.

III. TRENDS IN HIGH PERFORMANCE COMPUTING

Typical implementation strategy for SOA consists of leveraging existing IT applications and assets to create "grass roots" web services implementation. It should be noted that *web services* do not equate to *service-oriented architecture*. Web services primitives are merely a set of technology tools that enable the implementation of service oriented architecture.

This "grass roots" web services implementation transforms heterogeneous systems with proprietary interfaces and wraps legacy systems into standards based coarse grain web based services interfaces that can be leveraged by authorized business process or entity. This is followed by creation of Mission Critical Business Processes using underlying web services and secure service interface. The Mission Critical Process and services must then be managed to provide efficient operation. Creation of this environment is typically done in a SOA pilot within the enterprise, which is then extended to the enterprise as shown in the roadmap in Figure 2.

Figure 2: SOA Implementation Roadmap

Current best practices and state-of-art technology has evolved from operational and platform centric paradigm, where IT was in a reactive mode, with distributed architectures and tightly coupled custom applications, to a service and process centric paradigm, where business processes govern infrastructure and IT systems correlated with business processes using shared services.

This new service and process paradigm can be visualized using HP's Darwin Architecture Framework and consists of four layers (as shown in Figure 3), namely the business process layer, the business services layer, the application services layer and the infrastructure services layer. These layers help demonstrate how a low lying

asset is leveraged to create application services and further abstracted to create business services, which are then used as building blocks for creation of mission critical business processes.

IV. THE PROPOSE AGENT-BASED MOM FRAMEWORK

According to the literature study, most of the current works in cross-platform and distributed systems are on middleware technology as an interoperate application between different applications. Even though web services over SOAP are the most used communication mechanism for SOA, other communication techniques also used, such as REST based technology. Nevertheless, an organization cannot truly take advantage of SOA application's benefits without a well-integrated among distributed and cross-platform applications. Middleware enables this integration by sending approaching applications out to cross-platform environments and releasing the domain-specific value of each application.

According to S. Kuppuraju, A. Kumar and G. P. Kumari "Web Service standards and specifications are used to connect different products in an SOA paradigm. However, cross-platform communication cannot be guaranteed due to various reasons like differences in the versions of Web Service standards, specifications supported and differences in SOA deployment style". As a result, to overcome the weaknesses of multi-domain communication, some researchers have conducted studies to extend the MOM architecture. Most of these efforts have addressed one or two issues of the MOM limitations and have not considered the general requirement for cross-platform communication. Therefore, it is strongly argued that a study in this area is needed to consolidate these distinct issues of cross-platform integration and communication to find a common ground towards the establishment of a generic framework for cross-platform communication to be applied within the SOA context. Development of a generic and flexible framework to support the communication among cross-platforms and geographically distributed systems would be a good steps towards achieving cross-platform communication in SOA systems. The suggested framework is adopting agents to address the generic and flexible concern in Message Oriented Middleware where agents will manage all the communication processes between applications. At the same time, there will be a translation agent involved in this process. The general view for the framework is shown in the figure below In Figure 1, the cross-platform framework includes four agents within Multi-Agent System (MAS) in each side of application to be used in the communication process between different SOA applications. Included is the Agent Sender, Agent Receiver, Agent Manager and Agent Translator; these multi agent systems represent communication enabled for different SOA applications which use different types of messages in their system communication. At the left side of the diagram is a representation of the SOA-based application number 1 that using different message type from application number N and the right side has a representation as the SOA-based application number N which can be more than one type of different SOA applications.

V. MODELING AND SIMULATION

Agent-Based Modeling and Simulation (ABMS) can be considered as a significant software methodology for developing and modeling complex agent-based systems and which has become difficult to deny for its

usefulness in several application domains. Currently, ABMS models and their implementations for system simulation can be obtained

easyABMS consist of seven consequent processes from the System Analysis to the Simulation Result Analysis . The following sub section will describe each process in detail.

The key achievement of the cross-platform communication in SOA applications is the ultimate transaction over different application systems that use different types of messages in communication, for example, SOA application A may use SOAP messages for their system communication but SOA application B uses REST messages. This section will focus on analyzing the basic requirements of the application system. The System Representation is based on the foundation of the system and the simulation objectives are presented in Figure 2. All the entities represented in Figure 2 are described along with their relationships and rules which represent the basic interaction among the different SOA applications.

Figure 3. Suggested Frameworks

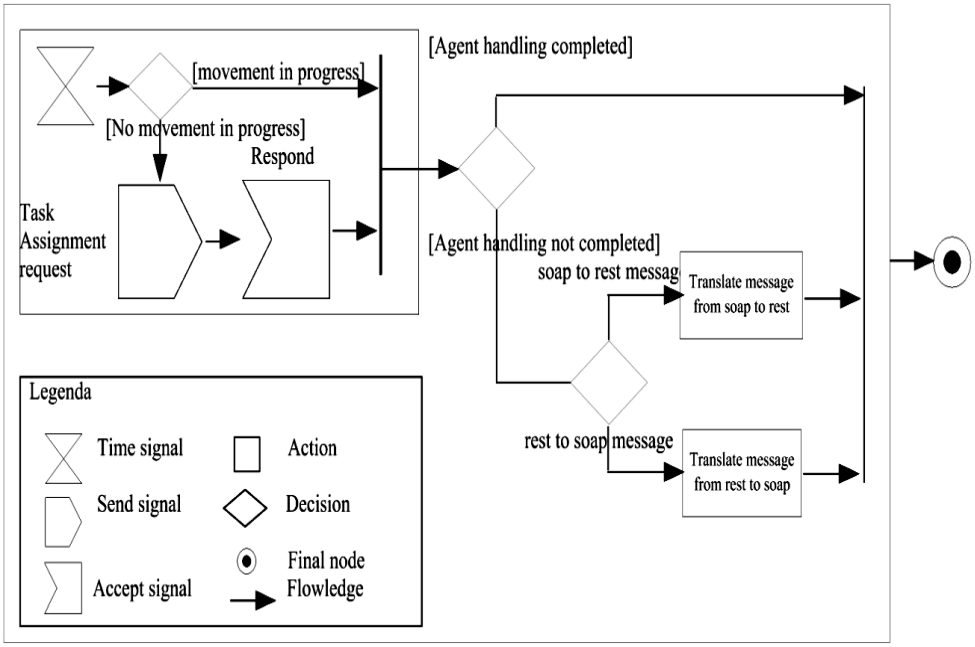

Figure 4. Part of the Agent-based MOM Model

VI. CONCLUSION

In this research, an Agent-based Message Oriented Middleware has been suggested to enable the communication among different SOA-based applications. In order to validate and evaluate the system performance and its effectiveness, a system experiential test had to be executed. The experiential test was challenging based on a lot of extended solutions being invented in the cross-platform with different particular perspectives. Most of the research works in literature were proposed to solve some particular issue which would fix their specific problem. Therefore, three dimension case study implementation was conducted to prove how generic and flexible the proposed cross-platform communication framework. In the proposed solution, the concentration was on generic cross-platform communication which would be able to support multiple types of SOA-based applications.

REFERENCES

[1] R. L. Baskerville, M. Cavallari, K. Hjort-Madsen, J. Pries-Heje, M. Sorrentino, and F. Virili, "The strategic value of SOA: a comparative case study in the banking sector," International Journal of Information Technology & Management, vol. 9, pp. 30-53, 2010.

[2] S. BEA. (2005, 30. Jan). BEA SOA domain model. Available: http://www.ebizq.net/white_ papers/6196. html

[3] C. Legner and R. Heutschi, "SOA Adoption in Practice - Findings from Early SOA," in Proceedings of the Fifteenth European Conference on Information Systems (ECIS 2007), St. Gallen, Switzerland, 2007, pp. 1643-1654.

[4] W. Lam, "Investigating success factors in enterprise application integration: a case-driven analysis," Eur J Inf Syst, vol. 14, pp. 175-187, 2005.

[5] R. Hirschheim, R. Welke, and A. Schwarz, "Service - Oriented Architecture : Myths , Realities , and a Maturity Model," MIS Quarterly Executive, vol. 9, pp. 37-48, 2010.

[6] G. Viering, C. Legner, and F. Ahlemann, "The (Lacking) Business Perspective on SOA–Critical Themes in SOA Research," in Wirtschaftinformatik Proceedings, 2009, p. Paper 5.

[7] T. G. J. Schepers, M. E. Iacob, and P. A. T. V. Eck, "A lifecycle approach to SOA governance," in Proceedings of the 2008 ACM symposium on Applied computing, Fortaleza, Ceara, Brazil, 2008, pp. 1055-1061.

[8] M. Niemann, J. Eckert, N. Repp, and R. Steinmetz, "Towards a generic governance model for service-oriented architectures," in Proceedings of the Fourteenth Americas Conference on Information Systems (AMCIS 2008), Toronto, ON, Canada, 2008.

[9] A. Ordanini and P. Pasini, "Service co-production and value co-creation: The case for a service-oriented architecture (SOA)," European Management Journal, vol. 26, pp. 289-297, 2008.

[10] J. Antikainen and S. Pekkola, "Factors influencing the alignment of SOA development with business objectives," in Proceedings of the 17th European Conference on Information Systems (ECIS 2009), Verona, Italy, 2009, pp. 2579-2590.

[11] J. W. Ross, "Creating a strategic architecture competency: Learning in stages," MIS Quarterly Executive, vol. 2, pp. 31-43, 2003.

[12] R. Welke, R. Hirschheim, and A. Schwarz, "Service-Oriented Architecture Maturity," Computer, vol. 44, pp. 61-67, 2011.

[13] P. Weill, M. Subramani, and M. Broadbent, "Building IT Infrastructure for Strategic Agility," MIT Sloan Management Review, vol. 44, pp. 57-65, 2002.

[14] R. Varadan, K. Channabasavaiah, S. Simpson, K. Holley, and A. Allam, "Increasing business flexibility and SOA adoption through effective SOA governance," IBM Systems Journal, vol. 47, pp. 473-488, 2008.

[15] E. A. Marks, Service-oriented architecture (SOA) governance for the services driven enterprise: Wiley, 2008.

[16] T. Erl, S. G. Bennett, B. Carlyle, C. Gee, R. Laird, A. T. Manes, et al., SOA Governance: Prentice Hall, 2011.

[17] M. N. Haines and M. A. Rothenberger, "How a service-oriented architecture may change the software development process," Communications of the ACM, vol. 53, pp. 135-140, 2010.