Evaluating ESB for C4I Architecture Framework Using Analytic Hierarchy Process

Abdullah S Alghamdi, Iftikhar Ahmad, Muhammad Nasir
Department of Software Engineering, College of Computer & Information Sciences,
King Saud University, P.O. Box 51178, Riyadh 11543,
Kingdom of Saudi Arabia.

Abstract -*An ESB is a platform that provides services such as message routing and transformation. Along with this, it has the capabilities to ease the pains of connecting heterogeneous C4I systems among various defense forces. This paper describes an assessing mechanism of optimum selection of Enterprise Services Bus (ESB) for C4I architecture framework. We use Analytic Hierarchy Process (AHP) for analyzing different Enterprise Services Buses (ESBs). In this paper, we present experimental results that may be utilized for architecting C4I system and can further help organizations in selecting an ESB in an optimized way.*

Keywords: Architecture Frameworks (AFs), Command, Control, Communications, Computers and Intelligence (C4I) system, Analytic Hierarchy Process (AHP), Enterprise Services Buses (ESBs)

1 Introduction

The interest of researchers, analyst, designer and developers in C4I system made it more imperative and attractive because of adaptation from civil and defense area. Now a day there are many ESBs available in the market and it is very difficult to choose which one is suitable. There are many issues in the integration of heterogeneous C4I systems that may be minimized using ESBs. Selecting an ESB is very difficult for an organization because many factors involve for selection. This paper describes an assessing mechanism of four ESBs namely Mule, Glassfish, WSO2and Fuse keeping in view the C4I System as a base, so as to ascertain which ESB fulfills the requirements of the system of systems. The assessing mechanism consists of two criteria such as main criteria and sub-criteria. We evaluated and rated the ESBs by assigning priorities, and calculating weights on the basis of main criteria and sub-criteria. This paper is divided into the following sections background, related work, methodology and implementation, results, and conclusion.

2 Background

The C4I systems are used in various departments where command and control scenario exits such as; defense, police, investigation, road, rail, airports, oil and gas. The C4I systems mainly focus in defense applications. C4I systems play a major role in development of information management, data fusion, and dissemination and it consist of people, procedures, technology, doctrine and authority [1]. The C4I system helps commander to acquire his objective in any crucial situation. C4I consists of Command, Control, Communications, Computers and Intelligence. The Command is authority that a commander exercises over subordinates by virtue of rank or assignment. The Control is also authority which may be less than full command exercised by a commander over part of the activities of subordinate or other organizations. While Computers and Communications process and transport information. Intelligence refers to information and knowledge obtained through observation, investigation, analysis, or understanding [2].

Enterprise Service Bus (ESB) is a major component of Service Oriented Architecture (SOA). An ESB provides interoperability using web services and related technologies and secure message transfer service between applications. ESB provides loosely coupled services. ESB can be used for communication and sharing certain information between different army wing's system. Using ESB applications communicate with each other by services invoking in a location independent fashion [3].

For SOA applications and services and ease enterprise integration ESB assists as an infrastructure backbone. Through reutilization of existing applications and data ESB prominently reduces cost and time to create new processes. In critical circumstances like network or software failure, shot messages are buffered by ESB and securely delivered when system is up and running again. ESB is considered much reliable for delivering messaging across services even over hardware layer [4].

Marvelous rise in threats on defense system it's a great deal to make a secured defense system. Many ESBs are available in the market to connect different system and synchronize them so that they can easily communicate with each other [5]. Different companies are providing their ESBs. To choose an ESB in accordance with the set parameters and requirements is a very difficult task. Hence, we are going to evaluate Mule, GlassFish, WSO2 and Fuse ESBs.

3 Related Work

It is a challenging task to evaluate ESBs with respect to user needs; so much work has been done in the area. To evaluate and compare ESBs based on certain criteria researchers used different mechanism. But to fulfill the requirements of SOA application is important criteria that lead closely to a particular ESB. Comparisons that have been done by researchers usually on general ESBs, open source ESBs or commercial ESBs. Price is important factor but it useless when open source ESBs are compared. Every researcher represents its own list of criteria to conduct their evaluation and the most commonly base is price.

Woolley had done a patent work who applied Vollmer and Gilpin's evaluation criteria to two open sources ESBs, such as Apache Service Mix and Mule Source Mule. His list of criteria included current offering, strategy, market pressure and integration. Woolley suggested that Mule ESB is the best and after this Fiorono ESB. Other ESBs were BEA System Equalogic Service Bus, IBM WebSphere Enterprise Service Bus and Apache ServiceMix [6].

Comparisons of two open sources ESBs such as Apache ServiceMix and Mule Source Mule and also two commercial ESBs like IBM WebSphare Enterprise Service Bus and BEA Systems Aqualogic Service Bus have been done by Desmet et al. Flexibility of ESBs may turn into bottleneck if complicated messages use it with many processes, so this research was on performance. In this worst case any business or defense process might be paralyzed. Hence, the performance is an important criterion for evaluation. Rates of ESBs were based on the performance test results. Desmet et al. rated Open ESBs first and commercial ESBs after them [7].

Evaluation on eight commercial ESBs with hundred criteria, which were in three grouped as market pressure, current offering and strategy, were conducted by Vollmer and Gilpin. They rated first Cape Clear and second to BEA Aqalogic Service Bus. ESB rates were based on surveys and briefings. Other ESBs were IBM WebSphere ESB, Fiorano, IONA Artix, PolarLake, Software AG and Sonic [8].

Vittie used integration, price and core bus feature as evaluation criteria on commercial ESBs. He rated first BEA Aqualogic Service Bus and second to Oracle SOA Suite. The others were Fiorano, Cape Clear, Tibco Software, IBM Websphare Enterprise Service Bus, Sonic and Software AG. His evaluation based on information provides by the consumers or was taken from the previous studies [9].

On the basis of criteria like stateless, stateful, extensibility and failover, Tobias et al. evaluated open sources enterprise services buses such as Fuse, Mule and OpenESB. They rated first to Fuse and second to Mule and third to OpenESB. Their study discovered the need to identify significant information resources and expose them through loosely coupled, reusable, and composable services for successful composition into workflows. The information consumed by the business process, without the basic raw material of workflow, the value of ESB orchestration would be severely limited. The re-combination of information by the orchestration engine in a new process would be difficult to achieve, without access to information freed of business process presumptions [10].

Alghamdi presented an approach to evaluate different architecture framework for C4I system using MCDM such as AHP. In designing and development process of C4I systems interoperability is an important issue. [11].

4 Enterprise Service Buses (ESBs)

4.1 Mule ESB

Integrating, interoperability and creating services are easy and fast is Mule ESB because it offers simple development model and lightweight architecture. For Mule ESB low CPU, memory, simplify deployment and maintenance required. No need to change or replace any existing system to implement Mule ESB it can easily work and deploy in any topology with or without an application container. For large SOA implementation Mule ESB also provide same performance and reliability challenges. Mule ESB provides common transports such as JMS, HTTP, SMTP, FTP, POP3, and XMPP are supported natively, as are web services and pluggable connectivity. Messages transferred through MULE along one of these protocols can behave like synchronously or request-response [12].

Typically Mule ESB used JMS for messaging system but any other messaging server can also be implemented such as Microsoft Messaging Queuing (MSMQ), IBM WebSphere MQ or TIBCO Rendezvous. We can connect mainframe applications, web services, messaging, sockets etc and interact with them consistently when using MULE ESB because there are no specific rules for integration when using it. Building block facility is also provided by Plug-in architecture of Mule. Mule use Service Oriented Architecture (SOA) that integrate existing system easily. It can be use easily with any application server or as standalone [13].

To run Mule ESB's components does not require any specific programmatic code Application Programming Interface (API). Mule also provides support of integration with Spring Framework and Business Process Management (BPM). It supports XML, CVS, Binary, Streams, Record and Java object etc. It provides the facility of zero code intrusion. Objects are fully portable without any Mule specific API on service object. Messaging framework of Mule provides reads, transforms and sends data as message between applications that are heterogeneous and not able to read or process data coming from another application [14].

Mule has an advantage that it can convert data as needed but other ESBs have to create an adapter for every application and convert the data into single common messaging format. In Mule no need for any kind of adapters to connect applications and not required a common

messaging format. Information sent on any communication channel, such as HTTP or JMS, and is translated as needed along the way. When source applications connect to Mule and want to share data with other target applications, it reads data from one former, change it completely as needed so that can be read by other application, and then sends it to the later. This functionality of Mule enables to integrate all types of applications even that are not built for integration. The main advantage of Mule ESB is it allows different applications to communicate with each other within intranet or over the internet [15].

4.2 GlassFish ESB

GlassFish ESB provides interoperability and lightweight integration platform with fast development tools. GlassFish deploy SOA components with free dependencies and flexibility. GlassFish consist of NetBeans tooling, GlassFish application server, JBI runtime for deploying solutions, integration engines, adapters for external systems, and simple installer. Normalized Message Router (NMR) locates appropriate service providers for transferring messages. NMR is a part of JBI container [16].

Interoperability option provides facility to communicate heterogeneous systems. To operate heterogeneous environment, interoperability make easy to develop secure, cross platform web services that are reliable and faster. GlassFish ESB is reliable and high performance infrastructure that increases interoperability and scalability for different systems with different architecture and provides secure interoperability for exchange of information related to any defense wing. Glass Fish contains Net Beans tooling, Glass Fish application server, Java Business Integration (JBI) runtime for deploying solutions, integration engines, adapter for external system, and simple installer. GlassFish is highly integrated, scalable application integration solution for SOA adapters [17].

Plug and play facility of GlassFish ESB allows users and vendors to plug and play through these components and services that can be interoperable. GlassFish ESB is based on Open ESB that delivers a platform for integration, Enterprise Architecture Integration (EAI) and Service Oriented Architecture (SOA). To build flexible and robust solutions for integration of an organization for their system, using large number of components including binding components (adaptors) and service engines (processors) based on large number of standards, such as JBI, Java EE and SOAP and so on. GlassFish ESB allows vertical and horizontal scalability using JBI component architecture's asynchronous and decupled designed model. Staged Event-Driven Architecture (SEDA) provides synchronous, message based nature, and this provides minimizing blocking threads, scalability applications without explicit code and associated memory requirements [18].

4.3 Fuse ESB

FUSE ESB has pluggable architecture and work with other integration components just like OSGi, JMS, JCA and JMX etc and can easily be embedded at endpoints that allow distributed systems to intelligently interact without mandating a centralized server. FUSE ESB based in Apache ServiceMix and supports JBI and OSGi architectures that allow using their preferred service solutions in their SOA. FUSE ESB is a fully standard base and open source interoperability platform for enterprise information technology organizations. Organizations can use FUSE ESB to use their service solution in their SOA with pluggable architecture. Without mandating centralized server allow distributed systems to interact intelligently in FUSE ESB. Architecture of FUSE ESB and GlassFish ESB is same. FUSE ESB includes FUSE Message Broker, FUSE services framework and FUSE mediation router and is part of a family of application integration and messaging components based on apache projects. FUSE ESB is one of a family of components that includes FUSE HQ, FUSE Message Broker, FUSE Services Framework, and FUSE Mediation Router. The FUSE components are tested for interoperability, certified, and supported to combine the speed and innovation of open source software with the reliability and expertise of commercially provided enterprise services [19].

FUSE ESB provides the support for Spring Framework, which is lightweight container for application components and facility to use their preferred service solution in their SOA with pluggable Java Business Integration (JBI) and Open Service Gateway initiative (OSGi) architecture. The advantage of Spring Framework is provides advantage to write light weight JBI components using POJO. Through Web services FUSE ESB support interoperability to complex and distributed services or standalone service. Inside JBI environment, JBI Components Service Engine provides logic needed to provide services for massage transformation, orchestration or advance message routing, so they can only communicate with other components inside of the JBI [20].

For accessing outside the JBI environment to service, JBI components Binding Components provide access via a particular protocol. They implement the logic needed to connect to a transport and receive a message through that transport [21].

4.4 WSO2

WSO2 is easy-to-use open source ESB available under the Apache Software License and allow administrators to simply and easily configure message routing, virtualization, intermediation, transformation, logging, task scheduling, load balancing, failover routing, event brokering, etc. The runtime has been designed to be completely asynchronous, non-blocking and streaming based on the Apache

Synapse core. WSO2 ESB supports all the widely used transports including HTTP/s, JMS, VFS and domain specific transports like FIX, HL7 and so on. A new transport can be added easily using the Axis2 transport framework and plug-in to the ESB [22].

WSO2 ESB provides browser based Graphical User Interface (GUI) for configuring the ESB, an integration registry for browsing, loading, and configuring services; and graphical management and monitoring tools. WSO2 also provides Web management console that is most useful. Although the XML based configuration file are not difficult to understand, so the console makes mistakes less likely. When operators and administrators are not ESB developer then it comes true in that environment like large organizations. WSO2 management console also provides useful monitoring functions, allowing administrators to see graphical view of all kind of traffic including message traffic to proxies, end points and sequences, as well as details such as min/max/average response times, faults and total message counts [23].

WSO2 ESB automatically optimizes the parsing of messages so it can perform virtualization and routing. WSO2 ESB includes non-blocking http/s transport permits ultra-fast execution and support for large number of connections. Asynchronous JavaScript and XML (AJAX) Web-based administration console facilitate monitoring, management, and definition of policies, routing and transformation [24].

5 Methodology and Implememtation

The methodology incorporated in this evaluation consists of goal selection, decision of criteria; determine the alternatives, building hierarchy, assignment of priorities, calculation of weights, and consistency test as shown in Figure 1. Further, this work is implemented using multi-criteria decision making software.

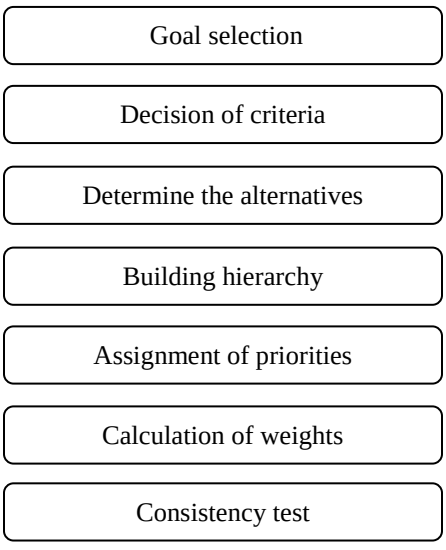

Figure 1: Methodology Process

5.1 Goal Selection

First of all, we selected a goal for this work. The goal is Evaluating ESB for C4I architecture framework. Four ESBs such as Mule, GlassFish, WSO2 and Fuse are selected for analysis purpose.

5.2 Decision of Criteria

Secondly, we decided criteria and sub-criteria. The main criteria consist of Interoperability', 'Extensibility', 'Messaging', 'Easiness', and 'Connectivity support'. The main criteria are further divided into sub-criteria. The criterion 'Interoperability' is divided into sub-criteria namely 'Syntactic', 'Semantic' and 'Network'. In the same way, the criterion 'Messaging' is divided into 'Reliability', 'Security' and 'Speed'. The selection of criteria and sub-criteria is based on the works as done by many other researchers [10,11,6].

5.3 Determine the alternatives

Thirdly, we determined the alternatives such as Mule, GlassFish,WSO2 and Fuse. These alternatives are the focus of this work.

5.4 Building Hierarchy

The hierarchy is built on the bases of criteria, sub-criteria and alternatives as shown in Figure 2. The goal "Evaluating ESB for C4I architecture framework" is at top of the hierarchy. The criteria and sub-criteria are shown in the middle. The alternatives are at bottom of the hierarchy but these are not shown due complexity in the diagram.

5.5 Assignment of Priorities

The assignment of priorities is based on the information obtained from previous works [6, 10,11]. The scale used for pairwise comparison is nine points scale .

5.6 Calculation of Weights

The weights of each node (criteria, and sub-criteria) are calculated on the bases of assigned priorities as shown in Table 1 to Table 3.

5.7 Consistency Test

The consistency ratio is calculated based on the weights. If the consistency ratio is less than 10 percent, the inconsistency is acceptable. Otherwise, we need to revise the subjective judgment. In this work the consistency ratio is less than 10 percent so there is no any inconsistency.

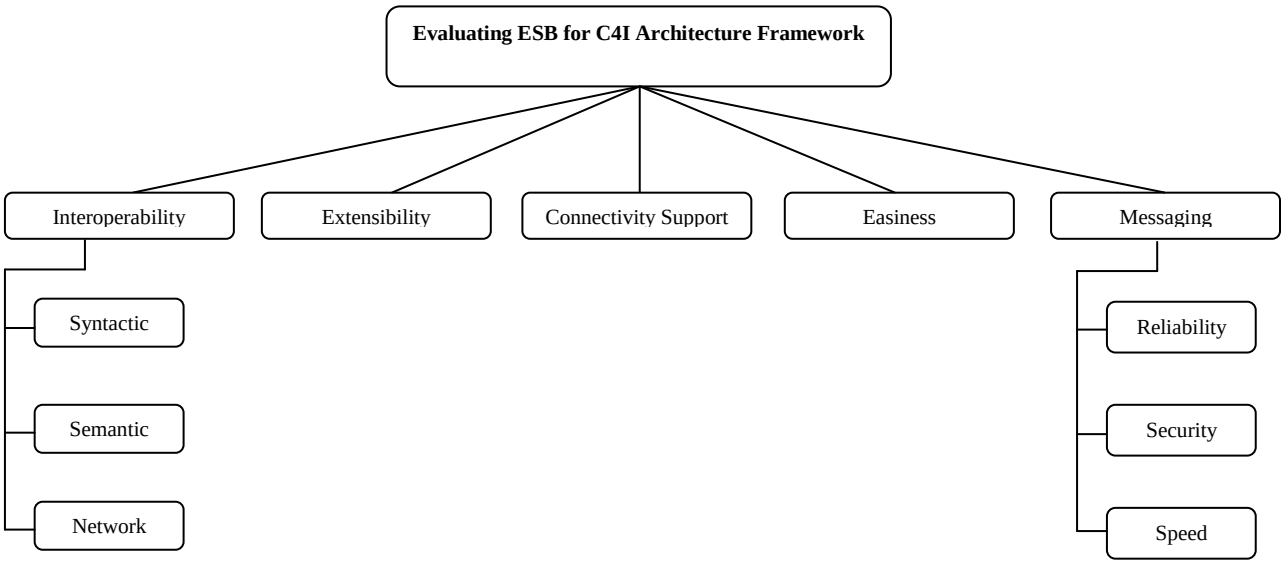

Figure 2: Hierarchy consist of goal, criteria and sub-criteria

TABLE I. CRITERIA WEIGHTS

Weights	Interoperability	Extensibility	Connectivity Support	Easiness	Messaging	Total
Local	0.38	0.09	0.18	0.12	0.23	1.00
Global	0.38	0.09	0.18	0.12	0.23	1.00

TABLE II. INTEROPERABILITY SUB CRITERIA WEIGHTS

Weights	Syntactic	Semantic	Network	Total
Local	0.33	0.33	0.34	1.00
Global	0.12	0.12	0.14	0.38

TABLE III. MESSAGING SUB CRITERIA WEIGHTS

Weights	Reliability	Security	Speed	Total
Local	0.32	0.21	0.47	1.00
Global	0.07	0.05	0.11	0.23

Table 1 shows weights of main criteria. The sum of local weights (0.38+0.09+0.18+0.12+0.23) is equal to 1 and same for global weights (0.38+0.09+0.18+0.12+0.23). Interoperability sub-criterion weights are shown in Table 2. The sum of interoperability local criteria's weights (0.33+0.33+0.34) is equal to 1 and sum of global weights (0.12+0.12+0.14) is equal to 0.38 that is its global weight. Same is the case with sub-criterion Messaging such as the sum of local weights (0.32+0.21+0.47) is equal to 1 and global weights (0.07+0.05+0.11) is equal to 0.23 that is Messaging global weight.

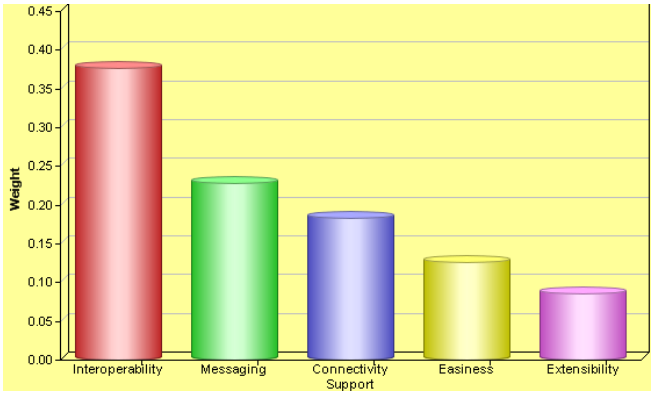

Figure 3: Criteia's Analysis

Figure 3 shows criteria ranking such as interoperability, messaging, connectivity support, easiness and extensibility.

6 Results

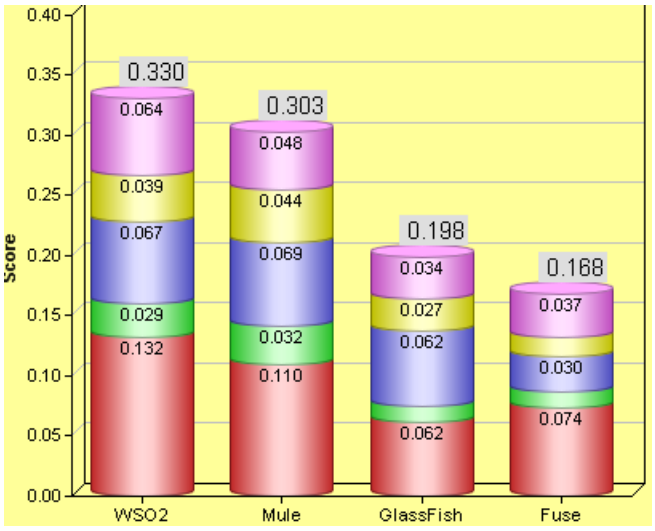

Figure 4: Comparative Analysis of WSO2, Mule, GlassFish and Fuse Enterprise Services Buses

Results in Figure 4 shows that WSO2 is best ESB in the application of C4I architecture framework. The Mule is rated as second Glassfish as third and Fuse as fourth in this work of assessment. Further, Mule is best in case of easiness, messaging and connectivity support while WSO2is preferable in terms of interoperability and extensibility. However, WSO2 and Mule is optimum as compared to other ESBs such as GlassFish and Fuse.

7 Conclusion

The MCDM technique is used to evaluate three ESBs such as Mule, GlassFish, WSO2 and Fuse. The evaluation process consists of main criteria and sub-criteria. According to our study, we have concluded that among all the ESBs, the WSo2 and Mule is most suitable to tackle the current issues of C4I architecture framework such as interoperability, messaging, connectivity suppor, and easiness.

REFERENCES

[1] Lean Weng YEOH, Ming Chun NG, Architecting C4I Systems, Second International Symposium on engineering Systems MIT, Cambridge, Massachusetts, June 15-17, 2009.

[2] United States Department of Defense (USDoD) , Joint Doctrine for Command, Control, Communications &Computer (C4) Systems Support for Joint Operations, Joint Chiefs of Staff, 30 May 1995, Available on website accessed on Sept 01, 2009. http://www.dtic.mil/doctrine/jel/new_pubs/jp6_0.pdf

[3] Luis Garces-Erice, "Building an Enterprise Service Bus for Real-Time SOA: A Messaging Middleware Stack", 33rd Annual IEEE International Computer Software and Applications Conference, pp. 79-84, 2009, doi: 10.1109/COMPSAC.2009.119.

[4] Martin Keen, Amit Acharya, Susan Bishop, Alan Hopkins, Sven Milinski, Chris Nott, Rick Robinson, Jonathan Adams and Paul Verschueren, "Patterns: Implementing an SOA using Enterprise Service Bus", IBM Redbooks, SG24-6346-00 , July 5, 2004. http://www.redbooks.ibm.com/redbooks/pdfs/sg246346.pdf

[5] Saurabh Mittal, Bernard Zeigler, Jose L. Risco Martin, Ferat Sahin and Mo Jamshidi, "Modeling and Simulation for systems of systems Engineering", Chap 5, Wiley [Imprint], Inc. 2008. http://acims.arizona.edu/PUBLICATIONS/PDF/Mo_Chapt_5_MittalZeiglerJoseFeratV4.pdf

[6] Robert Woolley, "Enterprise Service Bus (ESB) Product Evaluation Comparisons", Utah Department of Technology Services, Oct 18, 2006; http://dts.utah.gov/techresearch/researchservices/researchanalysis/resources/esbCompare061018.pdf: Webpage accessed on Dec 15, 2009.

[7] Stein. Desmet, Bruno Volckaert, Steven Van Assche, Dietrich Van Der Weken, Bart Dhoedt, Filip De Turck, "Throughput Evaluation of Different Enterprise Service Bus Approaches", Conference on Software Engineering Research and Practice; Jun 25-28, 2007, ISBN: 1-60132-033-7, 1-60132-034-5 (1-60132-035-3), CSREA Press, pp. 378-384. http://www.ibcn.intec.ugent.be/papers/3032.pdf

[8] Ken Vollmer and Mike Gilpin, "The Forrester Wave: Enterprise Service Bus, Q2 2006", BEA Systems, Nov 17, 2006, http://whitepapers.zdnet.co.uk/0,1000000651,260256988p,00.htm

[9] Lori MacVittie, "Review: ESB Suites", Networking Computing, CMP Media LLC, March 10, 2006. http://www.networkcomputing.com/wireless/review-esb-suites.php.

[10] Tobias Kruessmann, Arne koschel, Martin Murphy, Adrian Trenaman, Irina Astrova, "High Availability: Evaluating Open Source Enterprise Services Buses" Proceedings of the ITI 2009 31th Int. Conf. on information Technology Interface, June 22-25, 2009, Cavtat, Croatia.

[11] Abdullah S. Alghamdi, "Evaluating Defense Architecture Framwork for C4I System using Analytic Hirarchy Process", Journal of Computer Science 5(12): 1075-1081, 2009.

[12] Peter Delia, Antoine Borg, Ricston Lts "MULE 2: A Developer Guide to ESB and Integration Plaeform", Professional and Applied Computing, Apress, ISBN: 978-1-4302-0981-2 (Print) 978-1-4302-0982-9 (Online), DOI 10.1007/978-1-4302-0982-9, chapter 1, pp 1-28, Feb 7, 2009. http://www.springerlink.com/content/978-1-4302-0981-2

[13] http://www.mulesoft.org/display/MULE/Home: Webpage accessed on Sep 10, 2009.

[14] Vina Ermagan, Ingolf H. Krüger and Massimiliano Menarini, "Aspect-oriented modeling approach to define routing in enterprise service bus architectures", ACM New York, ISBN: 978-1-60558—025-8, Pages 15-20, 2008 , Bookmark DOI: http://doi.acm.org/10.1145/1370731.1370735

[15] Ricky E. Sward and Kelly J. Whiteacre, "A Multi-Language Service Oriented Architecture Using an Enterprise Service Bus", ACM New York, Volum 28, pages: 85-90, Issue 3, December 2008, DOI: http://doi.acm.org/10.1145/1454497.1454489

[16] Vasiliev and Yuli, "Beginning Database-Driven Application Development in Java EE Using GlassFish", Apress, ISBN: 978-1-4302-0963-8 (Print) 978-1-4302-0964-5 (Online), Springer Link Date: Apr 21, 2009, DOI: 10.1007/978-1-4302-0964-5. http://www.springerlink.com/content/978-1-4302-0963-8

[17] Sun MicroSystems, "Sun GlassFish Enterprise Service Bus", 2009 http://www.sun.com/software/javaenterprisesystem/javacaps/glassfish-esb-ds.pdf: Webpage accessed on Dec 30, 2009

[18] Mike Somekh, Mark Foster, Rastislav Kannocz "GlassFisg ESB High Availability and Clustring, Sun Micro Systems", White paper, Chapter 1 Introduction, Dec 2009. https://www.sun.com/offers/docs/glassfish_esb_ha_wp.pdf

[19] http://www.fusesource.com: Webpage accessed on Sep 1, 2009

[20] Adam Badura,Bartosz Sakowicz and Dariusz Makowski, "Integration of Management protocols based on Apche ServiceMix JBI platform", CADSM 2009. 10th International Conference, pp: 381-384, Feb 2009, http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=04839857.

[21] Tijs Rademakers and Jos Dirksen, "Open Source ESBs in Action", Manning Publications, ISBN 1933988215, Sample Chapt 1, Sep 2008 http://www.manning.com/rademakers/sample_ch01_ESB.pdf

[22] WSO2 ESB, Last published 25th Jan 2010, http://wso2.org/project/esb/java/2.1.3/docs/index.html Webpage accessed on 20th Feb 2010.

[23] Steven Nunez, "WSO2: A lightweight, fast, and free ESB", InfoWorld, 6th Nov 2007, http://www.infoworld.com/d/architecture/wso2-lightweight-fast-and-free-esb-781.

[24] Emsworth, "WSO2 Enterprise Service Bus", Cover Pages Hosted by OASIS, England, 11th Jun 2007, http://xml.coverpages.org/WSO2-ESBv10.html.