SOA BASED C4I COMMON-VIEW INTEROPERABILITY MODEL

Zeeshan Siddiqui, Abdullah S. Alghamdi
C4I Centre of Advance Systems (C4ICAS), King Saud UniversityRiyadh, Kingdom of Saudi Arabia
[zeeshan.siddiqui, ghamd]@ksu.edu.sa

ABSTRACT: *Message oriented communication using platform dependent techniques and middle-wares are a traditional approach. Difficulty begins when dissimilar platforms are communicating together having different interfaces. Military organizations are also pursuing towards interoperability solutions. The goal to achieve is to interoperate interrelated mission critical systems running under the hood of one system called System of Systems (SoS) or often termed as Command Control Computer Communication and Intelligence (C4I) System. To achieve maximum interoperability we have introduced a common platform middleware that communicate on the bases of messaging, routing, invocation and mediation services. This paper covers and discussed a Service Oriented Architecture (SOA) based common view interoperability approach by integrating different dispersed applications together under a common-view. We will discuss the functionality, integration and implementation by adopter configuration and coupling.*

Keywords— **Service Oriented Architecture (SOA), Interoperability, Software Engineering, C4I, middleware**

.INTRODUCTION

Interoperability is most discussed topic as organizations are growing and communication is taking place. Different integration and communication techniques based on different approaches were presented and implemented and many approaches are yet to be discussed.

The future enhancements are pushing the technology towards more discreet level. The interoperability between different applications and softwares working under identical policies and proposals is still a challenge, depending on the size of their organizations. When there is a case of applications based on different platforms needs to communicate to each other, the interoperability issue turn towards a more complex and difficult situation.

In this study, our focus is not a particular organization, rather, *military* organization having there own architectures dependable on a specific platform. These architectures, based on the platform customized according to their need and requirements. More over, for military point-of-view it is mandatory to accurately understand or receive what is sent from other architecture or specified applications.

A multinational organization comprises of hundreds of applications running, that may be customized according to their need or bought from a third-party vendor or can be a part of an inherited system. Number of application is not the case; the infrastructure should be capable of exchanging applications and absorbing new scenarios. The term basically used in this aspect is Enterprise Application Integration (EAI) [1].

The Message-Oriented Middleware [2] is the traditional solution to EAI, the commodities build using MOM comprises of a central message line up system frequently termed as *message broker* (figure 1). Applications are connected to the message broker to put message in and out. A unique capability of the message broker is to store messages, in a way that dispatcher and recipient do not have to be logged in at the same time. Message broker can route the message to transmit it to more then one receiver and it can help the receiver to fit the message according to its own requirement. This facility permits the connected applications to use their own required message formats. [2]

One of the big interoperability issue of MOM is the platform dependency, protocols and platforms are specific not independent. Requirement of a reusable service that is independent of any specific platform and that can help implement the complex business processes is mandatory. Service Oriented Architecture (SOA) [3] is the modern and efficient use of this phenomena; SOA can be build and put into practice by several web service technologies i.e. SOAP [4] or REST [5]. With the usage of SOA in identical platforms, it also gives benefit to those enterprise application providers which are based on dissimilar OS platforms and using different programming languages.

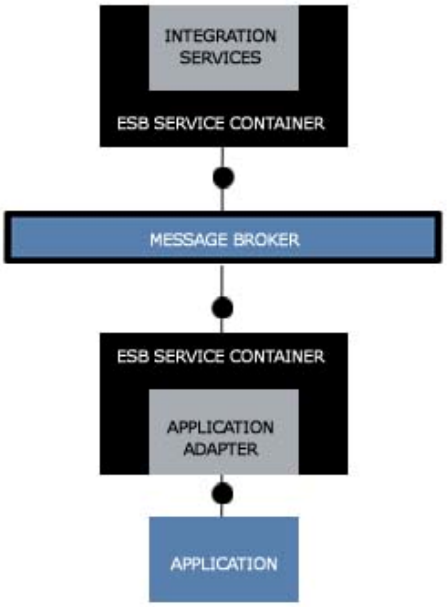

Fig.1. Message broker

Preliminaries & Background

Military domains are considered the most critical domains as compared to other domains. Information transfer between different military nodes is simultaneously crucial and highly desirable for terrorist and hackers to break the security and destroy the country stability and peace. The integration or interoperability issue of different military domains i.e. Air, Sea and Ground are increasing. Taking into consideration the current global situation, the need of a *highly* interoperable, secure and efficient *System of Systems (C4I)* i.e. Command Control Computer Communication and Intelligent System [28] is rising.

A. Purpose & Value of this Research

Up-to this extent, the planners think that achieving a true and absolute *interoperability* is not possible, however, practically

speaking, there are many successful projects running and being implemented successfully these days which were a fantasy few years back. In military domains, since the *joint nation collaboration* is observed, the interoperability issues has increased, and therefore, if we compare the ratio of interoperability solutions and percentage of the current successful scenarios with the previous studies then solutions are increasing with the percentage level.

These days, every country around the globe is facing intimidation at some intensity level. Therefore, the need or value of an efficient, secure and non-breakable interoperable system in military domain is increasing. This paper is also an intention to cover-up the interoperable issue up-to maximum extent. In any SOA based architecture, the functionality and interoperability depends on the efficiency of its adapter. We have demonstrated the maximum interoperability by practically coupling and testing the adapter. The goals and aims of this implementation and testing are as follows:

- Interoperability of Joint collaboration between battalions of different military nodes
- Successfully configure the middleware adapter to integrate the domains
- Successfully define and route the SOAP services
- Successfully couple the adapter with both battalion cores and test the full service activity test result.

To accomplish this task, the rest of the paper organization is divided into following sections: Section III consists of related study and literature review, Section IV will propose the approach of interoperability. Section V covers implementation, Section VI will have final testing and finally Section VII will cover the conclusion & result

I. RELATED STUDY & LITERATURE REVIEW

To evaluate the interoperability issues, and to review current solutions, following literature reviews are among many that we did:

Michael R. Hieb et al, compares an army C4I Data model of the JCDB to a simulation representation to identify the areas which are not aligned. The analysis discuss that the current army scenarios are not completely aware of the standards data models. [Michael R. Heib and Major James Blalock, Data Alignment between army C4I databases and army simulations, Spring simulation workshop, orlando Florida, 1999]

Andreas Tolk, summarized ongoing related efforts in common framework and discussed that a common framework is not only technically feasible but necessary to increase the efficiency of the War-fighter. [Andreas Tolk, A common framework for military M&S and C4I Systems, 2003 Spring Simulation Interoperability Workshop, Orlando Florida, 2003]

David Perme et al, summarized the current integration of air operations BML with ground operations BML and recommends how to approach future challenges. [David Perme, Andreas Tolk, William P. Sudnikuvich, J. Mark Pullen, Michael R. Hieb, Integrating Air and Ground Operations within Common Battle Management Languages, Spring Simulation Interoperability Workshop, Sandiago, CA, 2005]

Alghamdi A. et al, proposed a common information exchange model for multiple C4I architectures based on W3C standards that follow the recent DoDAF product line to give maximum interoperability while integrating different military architectures on a common battle field. [Abdullah S. Alghamdi, Zeeshan Siddiqui, Syed Shah Aman Quadri,]

Thea Clark et al, used two models of interoperability i.e. US DOD LISI model and Organizational Interoperability Model o determine technical interoperability of air combat system and organizational interoperability for coalition force. [Thea Clark and Terry Moon, Interoperability for Joint and Coalition Operations, Australian Defense Force Journal, Vol. 151, p.23-36, 2001]

R.M. Colombo and M.E. Orlowska, preserved that the strong design autonomy is not always possible to resolve semantic. [Information System Journal, Blackwell Publishing Ltd, Vol. 5, p. 37-50, 2010]

Bernhard H. and Wolfgang K., provided categorization of existing interoperability techniques, characteristics definition, and quality comparison. [ACM computer Survey, vol. 42, number 1, 2009]

A. Oracle ESB, A Recommendation

On the basis of the SOA Foundations, our proposed approach use and recommends *Oracle Enterprise Service Bus* [11] and *Oracle Integration Platform* [12]. In our further discussion we have highlighted its working architecture.

A.1.1 Working

For inter-application messaging Oracle ESB is a fundamental component that provides a loosely coupled framework. *Oracle JDeveloper* and *Oracle ESB control user* interfaces are configured and designed with *ESB service.* [13]

Message delivery is supported by ESB server by multiple protocol bindings, such as WSIF, HTTP/SOAP, JCA, JMS and java, by utilizing asynchronous/synchronous, reply/request, subscribe/publish models. [13]

A.1.1.1 3-Tiered architecture

A common but most essential architecture approach these days, *Oracle ESB* fully understand and support all three layers i.e. *User Interface Tier, Middle Tier and Data Tier.* The *UI Tier* consists of design level JDeveloper and monitoring ESB Control, *import/export and monitor* the routing or updated routing with *middle tier and data tier*. The middle tier consists of J2EE based *Metadata Server* communicating with J2EE *Runtime servers* through JMS. The data tier comprises of *Artefacts (XSD, XSLT, WSDL, Maps)* and *Relational (Service MID, Routing Rules, Instances, and Errors).* The *Middle Tier* is communicating with *Data Tier* using JDBC. [13]

A.1.1.2 OracleAS Interconnect an Oracle Integration platform

The integration infrastructure of *OracleAS Interconnect* [16] is packed with *OracleAS* [17], specifically, *iStudio designer* [18] and *Oracle Database* [19] with (AQ) *Advance Queuing.* As a part of the *Oracle Integration Platform* [14] including *Oracle Business Process Execution Language Process Manager (BPEL PM)* [15] for standards based workflow orchestration; *Oracle B2B* [20] for connecting to partners using industry standards B2B protocols, and *Oracle*

Adapters [21] which provides JCA based connectivity to virtually any external data source.

B.2.1 ***OracleAS Interconnect Functionality***

iStudio is the combination of Business rules and transformation logic. These rules and logics help to integrate heterogenous applications. By using *iStudio specification design tool* one can model the integration logic, result is metadata that is stored in the *OracleAS Interconnect* repository. In integration design, during runtime, the metadata is treated as runtime instructions to exchange the conversation between applications. [22]

OracleAS Interconnect uses common view; application that are communicating and exchanging information uses the common view, no direct communication. If any change or upgrade in the linked application, then the changes reflects in that particular application view and maps to the common view. In other words, only remapping is required, rest of the spokes and relationships with the hub remains unaffected. [13]

II. PROPOSED APPROACH

Our study is orbiting around military architectures; interoperability issues related to different applications integration combining different applications in related or non-related platforms are still a challenge. In broader view, when communication is spaning and exchanging information across different systems having different platforms and standards, the interoperability issues become a painstaking exercise.

After our review we can summarize Interoperability issues as follows;

- Separation of concerns,
- Lack of multi-protocol bus dependency,
- Virtualization,
- Transformation & Routing,
- Availability/scalability and Open Standards Support.

The solution is to propose a common information exchange framework in which services are loosely but securely coupled to overcome these problems.

In figure 2 on next page, we propose our state-of-the-art model that demonstrates the connectivity between multiple apps instances using a common-view.

The proposed model is based on the idea of connecting different heterogenous applications with going through a series of different integration steps. The model connects the application using a common view that comprises of an *adapter* which is responsible for providing metadata availability to newly or already connected applications. When ever there is a small or huge changes in any connected application, they will thing which is required is a re-map with the common view. This all is accomplished by the adapter configuration and routing between the metadata repository and the common view. The asynchronous flow course-plot or route the newly added military battalion <*newBattalion*> with the help of the adapter.

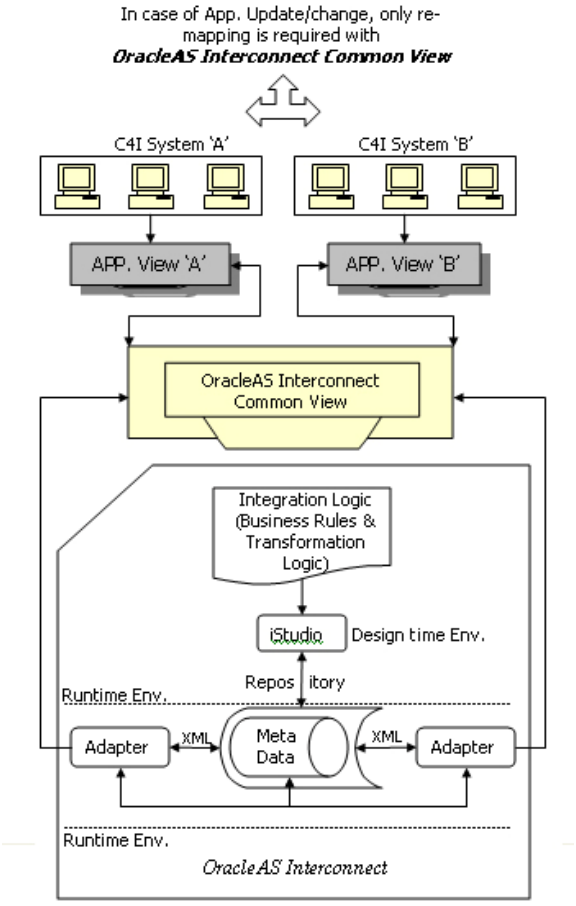

Figure 2, proposed integrated common-view model Situation Synopsis

It add-on with the <*newBattalion*> data with Email address incorporation using SOAP call facility. By doing this will finally bring out the new battalion data as an <*AddBattalion*> result in the common-view centre. This is done using Inter connect Inbound service course-plotting.

III. IMPLEMENTATION STEPS & IMPLEMENTATION DEMONSTRATION

A. *Overview*

The common view uses adaptors to communicate with meta-data repository. Therefore, we will use *Siebel* [23] to impeccably integrate different software and applications. By doing this the concept of writing long codes will be eliminated completely. The runtime platform uses the Metadata as instructions at runtime and enables the cross platform interoperability between different applications.

Implementation Steps

Our main step is to configure the adapter in an efficient and smooth way to have multiple information flow. This section demonstrates the adapter's coupling and final testing. We have followed the following two stated steps:

- Step 01: *IC Adapter Configuration and Routing*
- Step 02: *Adapter coupling*

Step 01: *IC Adapter Configuration and Routing*

The Siebel adapter is a finest adapter that practically enables all sort of platform dependant applications to integrate and work together. It supports synchronous asynchronous

interaction, robotically detain metadata for event or service interaction. [23]

In our exercise, we have pursued following deployment steps to deploy the IC adapter:

Set up of IC Adapter

java –jar admin_client.jar deployer:oc4j:opmn://<<c4icsr>>:<<8088>>/oc4j_soa <<c4idb>> <<*******>> -deploy –file ORACLE_HOME\integration\esb\lib\icAdapter.rar -deploymentName IcAdapter

Set up icwsilplugin

java –jar admin_client.jar deployer:oc4j:opmn://<<c4isr>>:<<8088>>/oc4j_soa <<c4idb>> <<*******>> -deploy –file ORACLE_HOME\integration\esb\lib\icwsilplugin.ear –deploymentName icwsilplugin –parent default

Icwsilplugin binding

java –jar admin_client.jar deployer:oc4j:opmn://<<c4isr>>:<<8088>>/oc4j_soa
<<c4idb>> <<*******>> -bindWebApp –appName icwsilplugin –webModuleName icwsilplugin
–webSiteName default-web-site –contextRoot /ic

Configuration

IC adapter can be configured in the Oracle ESB after altering *oc4j-ra.xml* file below:

```
<?xml version="1.0"?>
<connector-factory location="eis/ICAdapter" connector-name="ICAdapter">
        <config-property name="c4isr" value="c4iesb"/>
        <config-property name="driverClassName"
        value="oracle.jdbc.driver.OracleDriver"/>
<config-property name="Database Connection String"
        value="jdbc:oracle:thin:@esb-db-as:1521:ORCL"/>
<config-property name="c4idb" value="ichub"/>
<config-property name="*******" value="Manager1"/>
<config-property name="repoName"
value="InterConnectRepository"/>
<connection-pooling use="none"></connection-pooling>
        <security-config use="none"></security-config>
        </connector-factory>
        </oc4j-connector-factories>
```

Second, we added the following line in *Server.xml* to add the Oracle Application Integration libraries:

```
<code-source
path="E:\Ora\mid\integration\interconnect\lib\oai.jar"/>
```

Step 01: Routing service creation

In order to create Email routing service, we have used designer window by dragging *Routing Service* to the window. We have generated WSDL from project schema files. We have named our service *C4iEmailRoute.*

For *<<C4iEMailRoute>>* we have to set the *<<Definitions>>* and *<<Routing Rules>>*. For setting up *<<Definitions>>*, we have selected the *<<BattalionService>>* from the System/Group drop down. To invoke the route from external service we have checked the *<<Invocable from an external service>>* check. In the *<<Route Rule>>* section the operation which we have defined as *<<routeNewBattalion>>* is already being displayed. We have added a new operation for this rule named by *"setEmailforBattalion"*. To add transformation for messages to route, we have created a *mapper* file named by *"Battalion_To_setEmailforBattalionElement.xsl"*. We have used the designer *Automap* facility to make the transformation between *source* and *target.* Following is the end-result of this auto mapping illustrated in Table 1.0 below:

Source: BattalionServices_C4iEmailRoute		
Target: EmailWSSoapHttpPort?WSDL		
<<source>>		<<target>>
Inp1:Battalion		tns0:setEmailforBattalionElement
→Bid		tns0:Btitle←
→BTitle		tns0:BCore←
→Bname		tns0:BState←
→BCode		tns0:BEmail←
→BEmail		tns0:BPo←
→BAdd(+)		tns0:Bname←
		tns0:BCity←
		tns0:BCode←
		tns0:Bid←

Table 1.0: Automap Target Operation

Now, the next step was to define the reply operation for this rule. To accomplish this we choose the adapter event operation for publishing. To create the transformation for reply we have to follow the same above routine i.e. to create a new *mapper* file and use the Automap facility. Now as a result the target and the resource will be vice versa as stated above in table 1.0. In the designer window we need to concate the *"source" and "target"* together and define/edit the concate functions in the xls editor window. Following 03 function parameters are defined:

1. /tns:setEmailforBattalionElement/tns:BCode
2. ","
3. /tns:setEmailforBattalionElement/tns:Bname

Email Routing registration in ESB

In order to register our Email routing service to ESB we have to register it to *LocalESBServer.* Following is the successful registration proof of the Email routing service to the ESB:

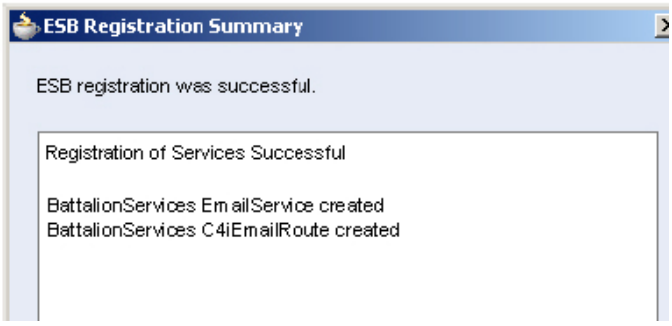

Figure 8.4: Successful *C4iEmailRoute* registration Proof in ESB

A. Step 04: Adapter Final Coupling

The Siebel adapter covers these inbound services; receive an XML service request document, transformation of the document, workflow processing, DML operations on query data, format the data in XML service response document and returns the data to the appropriate application. In Oracle ESB control panel under services pane, we have to define routing rules for *Battalion_newBattalion_RS.* Our target operation is

"*EmailRoutingService::routeNewBattalion"* which we have to select.

After defining the rule our flow is complete, the result is illustrated in the following figure 9.2 below:

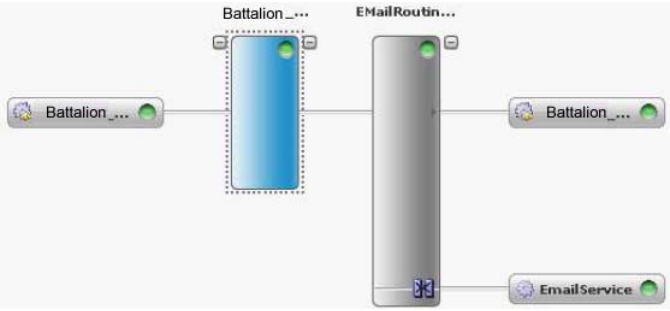

Figure 9.2: Final Routing Flow

IV. FINAL TESTING

The only thing which is waited is the final test; we did this by copying the *"Battalion.xml"* file in Interconnect FTP adapter outbound directory. In the ESB console under *"Messages"* the status of all the services should be "*Green"*, as showed in the following figure:

Figure 10.1: Message Service Final Testing

When clicking on any service for example; by clicking on *Battalion_newBattalion_RS* service following will show up in Figure 10.2 and 10.3:

Figure 10.2: Status & Tracking Data result

V. ORACLE ESB USAGE BENEFITS ON OTHER ESB(S)

After comparing different ESB(s) review in our literature study the following lacks were observed in open source ESB(s):

- No complex Business Data Transformations
- No functionality to customize ESB to integrate with other domain specific ESB(s)
- No Content based routing
- Limited Open Standard support

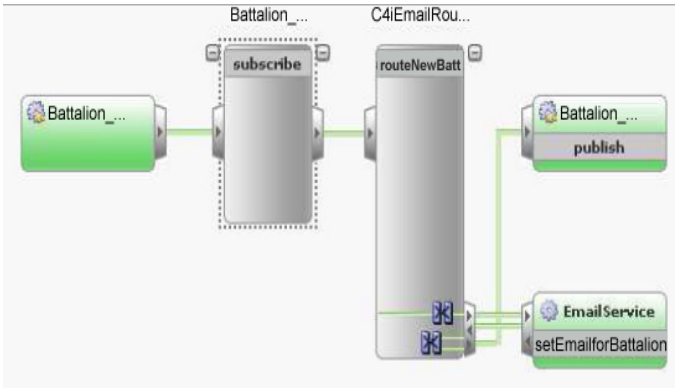

Figure 10.2: Full service activity test result

Oracle ESB has covered almost every aspect of interoperability issues, following features are concluded:

- Business Data Transformations
- Pervasive Enterprise System Connectivity
- Flexible Content Routing
- Open Standards support more then any ESB: WSDL, SOAP, HTTP(s), Reliable SOAP, WSIF, WS-*, JMS, XSLT, BPEL, JCA, XPATH, XQuery, UDDI, JNDI, J2EE, JDBC, SMTP, FTP

The Mule ESB is one of the top competitors involved in this business. Recently they have launched Mule IDE 2.0, but the latest Oracle 11g product has *way-a-head* comparable features:

- Cross-referencing, domain value map, java connectors architecture adapters and business activity monitoring sensors controls integration
- Oracle ESB (OSB) is now authoritative to operate on third-party java platform, IBM Web Sphere and Red Hat JBoss along with Oracle WebLogic Server.

The Oracle ESB road map is less risky, more convincing and present clearer options.

VI. CONCLUSION & RESULT

Oracle ESB is providing complete and comprehensive integration to different domains related to different organizations following different architectures. It gives a far more ahead of approach that overcomes the traditional hard-coded integration methods. *Oracle ESB* focuses on functional aspects of integration, giving absolute platform independency. It is not just gives ***platform-to-platform*** integration, rather, by using pre-packed adapters; it focuses more on ***cross-platform*** integration techniques and methodologies. This research has established following results and overcome following ***interoperability*** issues:

- ***Loose Coupling,*** by giving a common view not a direct connection to decrease integration interfaces count.
- ***Ease of Customization,*** any sudden or certain change in any application will not affect the common view; rather, require a re-map to the common view as discussed in Section VI.
- ***Easy Expansion,*** the integration of any new application in the system is as easy as *plug-n-play.* The particular application view needs to be connected to the common-

view and any of this will not affect the existing applications connectivity.

ACKNOWLEDGMENT

This project is supported by the Research Centre in College of Computer & Information Science, King Saud University. The authors are grateful for the whole research centre continuos support which has brought this project to successful completion. A word of thanks to all faculty members of the CCIS college and Software Engineering Department and on top of all, a grateful thanks to Almighty Allah, who never closes down in affectionate us.

REFERENCES

[1] Siddiqui. Z., Ghamdi. A., *"Node level Information Security in Common Information Exchange Model (CIEM)",* Science International, ISSN 1013-5316, December 2010

[2] Abdullah S. Alghamdi and Zeeshan Siddiqui, *Common Information Framework b/w Defense Architecture, A Web Semantics approach*, 16th International Conference on Distributed Multimedia Systems, Hyatt Lodge at McDonald's Campus, Oak Brook, Illinois, October 14-16, pp. 149-152, 2010 USA

[3] Linthicum. D., *Enterprise Application Integration,* Addison-Wesley Longman Ltd. Essex UK, UK, Published in 2000.

[4] Li. H. and Jiang. G., *Semantic Message Oriented Middleware for Published/Subscribed Networks,* proc of SPIE, 2004 – actcomm.thayer.dartmouth.edu

[5] Agrawal. R., Bayardo. R., Gruhl. D., Papadimitriou. S., *Vinci: A SOA for rapid development of web applications,* Elsevier Science, march 16 2001

[6] W3C, *SOAP Version 1.2 Part 1: Messaging Framework (Second Edition),* W3C Recommendations, April 27, 2007

[7] Fielding. R. and Taylor. R., *Principled Design of the Modern Web Architecture,* ACM Transaction on Internet Technology, Vol. 2, No. 2, Page 116 of 115-150

[8] Fulton. L, *The Forrester Wave: Enterprise Service Buses, Q1 2009,For Enterprise Service architecture professionals,* January 26, 2009

[9] D. Karastoyanova et al, *Semantic Service Bus: Architecture and Implementation of a Next Generation Middleware*. ICDE Workshops (2007), 347-354.

[10] Bedner. P., Furdik. K., Lukac. G. and Sabol. T., *Design of a Semantic Service Bus for Networked Enterprises*, Technical University of Košice, Slovakia InterSoft, a.s., Slovakia, *Spike Project,* 2009
Hinton. H., Hondo. M. and Hutchison. B., *Security patterns within SOA*, IBM Websphere, November 23, 2005

[11] Sun Developer Network, *Java Messaging Services (JSM), News & Updates,* December 17, 2008

[12] Oracle ™, *Oracle ESB Developer's Guide 10g(10.1.3.1.0)* ,September 2006

[13] Oracle ™, *Oracle Data Integration Platform Concepts & component* ,release 2, part number A96574-01

[14] Oracle ™ , *Oracle Enterprise Service Bus: The Foundation for S-O-A,* Technical documentation, September 2006

[15] Oracle ™, *Oracle Directory Integration Platform,* http://download.oracle.com

[16] Oracle ™, *Oracle BREL Process Manager 10.1.2.0x*, A Quick Start tutorial, http://download.oracle.com

[17] Oracle ™, *Oracle AS Integration Interconnect,* an Oracle Technical White paper, January 2005

[18] Oracle ™, *Oracle Application Server 10g, Part No. B13992-02,* September 2004

[19] Oracle ™, *Oracle Application Server InterConnect User's Guide 10g(9.0.4),* Part Number 10404-01

[20] Oracle ™, *Oracle Database 2 Day + Security Guide 11g Release 2(11.2),* Part Number E10575-04

[21] Oracle ™, *Oracle Fusion Middleware Oracle B2B technical note,* Version 1.2

[22] Oracle ™, *Oracle Fusion Middleware Data Sheet,* Oracle Adapter Library

[23] Oracle ™, *Enterprise Service Buses, Where are they going?,* An Oracle white paper, October 2006

[24] Oracle ™, *Oracle Application Server Adapter for Siebel User's guide10g release 3(10.1.3.4.0).*

[25] Woolley. R, *Enterprise Service Bus (ESB) Product Evaluation Comparisons,* Utah Department of Technology Services, Oct 18, 2006.

[26] D. Stein., V. Bruno, A. Steven., W. Dietrich., D. Bart. And T. Filip, *Throughput Evaluation of Different ESB Approaches,* Conference on Software Engineering Research and Practice, Jun 25-28, 2007, ISBN: 1-60132-033-7, 1-60132-034-5, CSREA Press, pp. 378-384.

[27] V. Ken and G. Mike, *The Forrester Wave: Enterprise Service Bus, Q2 2006,* BEA Systems, Nov 17, 2006.

[28] F. Larry, *The Forrester Wave: Enterprise Service Bus, Q1 2009,* BEA System, 2009

[29] A. Abdullah, S. Zeeshan and Q. Syed Shah, *A Common Information Exchange Model for Multiple C4I Architectures,* 12th International Conference on Computer Modeling and Simulation, Cambridge University UK, pp. 538-542, 24-26 March, 2010