INTELLIGENT-BASED TRANSLATION MODEL FOR CROSS-PLATFORM COMMUNICATIONS IN DISTRIBUTED SYSTEMS

Najhan M.Ibrahim, Mohd Fadzil Hassan, Ahmad Faizul Shamsudin

Department of Management and Information Technology
Univerity Sultan Azlan Shan, Bukit Chandan, Kuala Kangsar, 33000, Malaysia
Department of Computer and Information Sciences
Universiti Teknologi PETRONAS, Bandar Seri Iskandar, 31750 Tronoh, Perak, Malaysia
hanfast@gmail.com, afaizuls@usas.edu.my

ABSTRACT : *AIl technology has an important potential to facilitate connectivity in a distributed and cross-platform application. The main reason for the great success of agent technology in this area is an intelligent of the connection and integration among different applications over a distributed environment. The agent software is suitable for three environments: the first is the distributed system, the second is for a dynamic environment and the last is for the system that needs a flexible interaction. An environment of distributed and cross-platform communications systems is well suited for an agent-based system because of it is distributed in nature and dynamically flexible. Agent-based Translation Model is proposed to ensure the translation of communicated messages among different Web services hosted in different platforms. The aim is to provide a flexible translation process to support the cross-platform communication.*

***KEYWORDS*:** *Service Oriented Architecture (SOA), Message Oriented Middleware (MOM), Web Services, Cross-platform, Translation Model.*

I. INTRODUCTION

Unlike the traditional software framework, which is mostly unable to reuse the service, an SOA-based application is flexible. It may compose some services at runtime using existing services. At this point of time, SOA has provided a new direction for a software framework where the architecture is determined at runtime and the framework can be dynamically changed at runtime to meet the new software requirements [1]. In addition, web service is one of the significant attributes for transactions and communications in SOA application, which will be described in detail in the following section.

Furthermore, the SOA implementation is the enabling technology for an efficient development of applications and business processes using SOA principles. It provides users, developers, and administrators an operational framework and tools to configure, use, and manage their services that form inter-operative capabilities between applications and business processes. This framework uses a service-centric concept and non-service-centric capability for a trusted partner [2-7]. The following are generic characteristics of an SOA application[8-10].

1. Dynamically connects clients with other services in an independent protocol
2. Constantly handles Synchronous and Asynchronous communications modes during execution
3. Defines and handles events
4. Automatically converts data formats and services between clients
5. Controls distributed SOA resources in a centralized manner

6. Manages exceptions in the service implementation process
7. Monitors and observes various events and metrics during transactions
8. Grants a reusable service library code for the usage of client applications.

Nowadays, internet is growing from general information sharing to a business operation and transaction for the most of the modern business, educational institution and government agency. A matured technology allows organizations to communicate with each other more efficiently. Out of these Internet roots, Web Service technology was developed with a general goal to construct elements of business logic and services, which can be used and reused by other applications. The services themselves hide the complexity of their business logic from the consumers through simple interfaces allowing the services to be reused in many different applications [11]. The service and the consumer are described as being loosely coupled by an approach that allows complex composite solutions to be developed through leveraging multiple Web Services.

II. METHODOLOGY

The validation of the proposed research is based on benchmarking the proposed cross-platform framework with existing message oriented middleware. The major benchmark parameters are communications round-trip time compared to the performance of the proposed research. In this research, the benchmarking method and the parameters are apparent and hence a qualitative method is adopted in this study with details as outlined below. This section discusses the research setting, research design, data collection approach, and analysis.

Research Setting:

As discussed in the earlier sections, this research addresses the issues of cross-platform communications among SOA applications. To reflect the real world scenario in the research process, a case study involving of three different SOA-based applications were implemented. In the proposed approach, each SOA application is attached with proposed solution as a standard communications medium for cross-platform communications to study a flexibility of the proposed framework. The three SOA applications were placed to achieve a communications between multiple types of SOA environment.

Research Approach:

To make the communications process successful, the request application is needed to receive a respond from the partner application. The performance of both message sending and message receiver was hypothetically analyzed. The proposed framework with communication model implemented in JADE 4.3.2 prototype that has develop and adopted from the standard of TLAB (Telecom Italia Lab). The proposed framework has been evaluated in various communications process and in this paper presented only communications round-trip time

The measurement obtained above was compared with hypothetical analysis to formally validate the claims made by the proposed solution.

Data Collection Approach:

In the prototype implementation, three different SOA applications are connected with each other for testing and evaluating the communications process between them. In the case study, each of SOA application can be published a service to be used by other application and can also be a consumer, which is used a service from others. The communications process is measure by round-trip time which is start from sending a request until receive a respond. The measurement code embedded at the beginning of message sending to record the staring time and at the receiving of respond to calculate the round-trip time.

Analysis:

The following analysis would be required for this study with respect to the comparison of the existing solutions and the proposed framework for cross-platform communications. The effect of the communications process of the existing cross-platform solutions compared to the proposed cross-platform communications process where scalability of the framework, reliability of the framework, throughput of the communications process were analysed. Also, the system overhead of both existing solution and the proposed framework during normal load (Minimum request) and full load (Maximum request) was analysed.

III. RESEARCH WORKFLOW

In order to achieve research objective, different methods throughout our research work have been used. The work is divided into three different phases. In this section, different method and activities used for every phase of this research are presented as the following.

Phase 1: Identify the requirements for cross-platform communications

- A comprehensive literature survey has been conducted in order to get a clear idea about cross-platform. The different classifications of available solutions have been consolidated to understand the requirement of cross-platform communications.
- A comparative study of existing solutions for cross-platform has been conducted to find the advantaged and limitation of each requirement to be considered in the proposed framework.
- After selecting the requirement that supports the cross-platform communications, the evaluation process for each requirement has been conducted in detail.

Phase 2: Building the framework for cross-platform communications:

- At the beginning of this phase, the existing cross-platforms framework in SOA have been examined to figure out current limitations and problems.
- The main component of the cross-platform framework has been identified to build an abstract model.
- Building an efficiency translation model that supports multiple types of web services, which is the main engine in the propose framework.

Phase 3: Developing multi agent system for an intelligent cross-platform communications using agent technology:

- Using easyABMS to specify the roles, functions, and responsibility of each agents. This methodology used to verify the logical and computation model as well.
- Building Multi Agent System (MAS) that manages all communications activities in the cross-platform framework. Multi Agent Systems is also facilitating the communications process, which ensures the cross-platform communications
- Prototyping the proposed system using JADE environment with web services developed using Java to simulate the case study.
- Evaluate the framework reliability and performance and compare it with some of the existing solutions.

IV. THE PROPOSE TRANSLATION MODEL

Agent technology has an important potential to facilitate connectivity in a distributed and cross-platform application. The main reason for the great success of agent technology in this area is an intelligent of the connection and integration among different applications over a distributed environment [12]. The agent software is suitable for three environments: the first is the distributed system, the second is for a dynamic environment and the last is for the system that needs a flexible interaction. An environment of distributed and cross-platform communications systems is well suited for an agent-based system because of it is distributed in nature and dynamically flexible. Literature studies have shown that the enhancement and practicality has been proved from the field of agent technology as they have been deployed to solve many issues of distributed systems and cross-platform environments [13]. For example, modeling and simulation of the Agent Platform for Reliable Asynchronous Distributed Programming [85], Multi-agent Systems: Overview of a New Paradigm for Distributed Systems [14-16].

The suggested model has been developed as a sub-model to support agent-based MOM in managing translation and mapping process in cross-platform communication. In addition, the level of automation in the current transmission and communication has become significant. Therefore, in the conceptual message mapping the concept of basic ontology mapping has been applied. Furthermore, our research work as shown in Fig 1 put the initiatives to improve the level of automation and message mapping with the similarity method of the service description for efficiency in the cross-platform communication model.

Typically, SOA architecture needs to deal with a middle server such as a communication authority (CA) and message brokers. According to [17], the SOA system is not required to have any third party in the system among the trusted domains. Therefore, the agent-based MOM is connected directly with each other. By decreasing such complexity of the workflow, it will increase the autonomous level that does not need to wait for any response from a third party application. In this proposed model, the agent translator is connected directly with the DF (directory facility) in the JADE platform where the message description located to support all the message types that facilitate the mapping process [18]. The following sub section will describe the translation process and the translation mapping in detail.

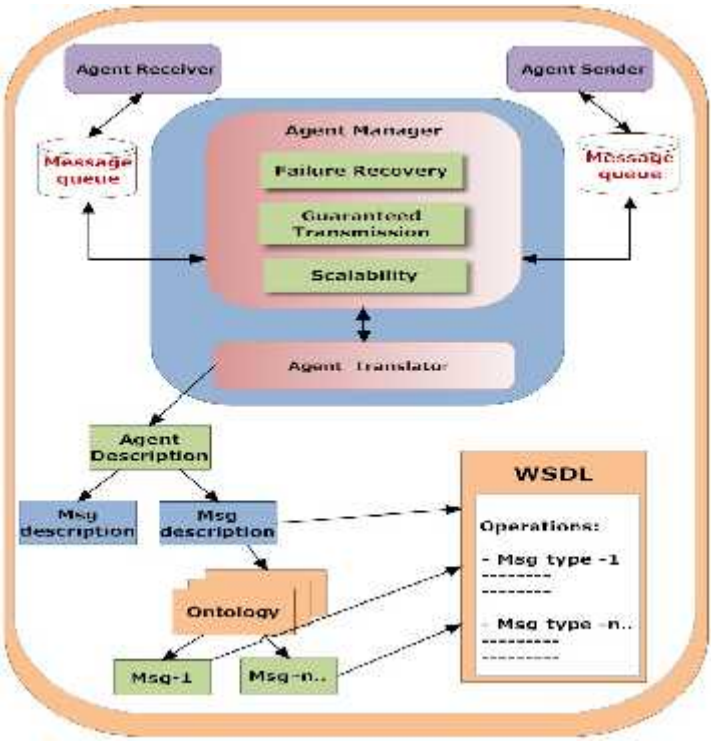

Figure 1 Suggested Model

A. Translation process

In order to overcome the issue of the cross-platform communication, a flexible translation model has been developed and included in the proposed framework to ensure the integration of the Web service and agent technology. It also supports multi-type applications and the translation processes which is managed by agent software, automatically, without any manual process from the user. It can be considered as a plug and play system to support different applications of SOA. This translation model concentrates on message translation between WSDL and ACL which are the main standard languages of this generic framework. The translation processes respond to send and receive messages from each application. In addition, the proposed translation model is a multi-agent system (MAS) where the agents act based on the requested message of every transmission. It will translate the sending and receiving message into an agent communication language which is the standard agent communication proposed by FIPA.

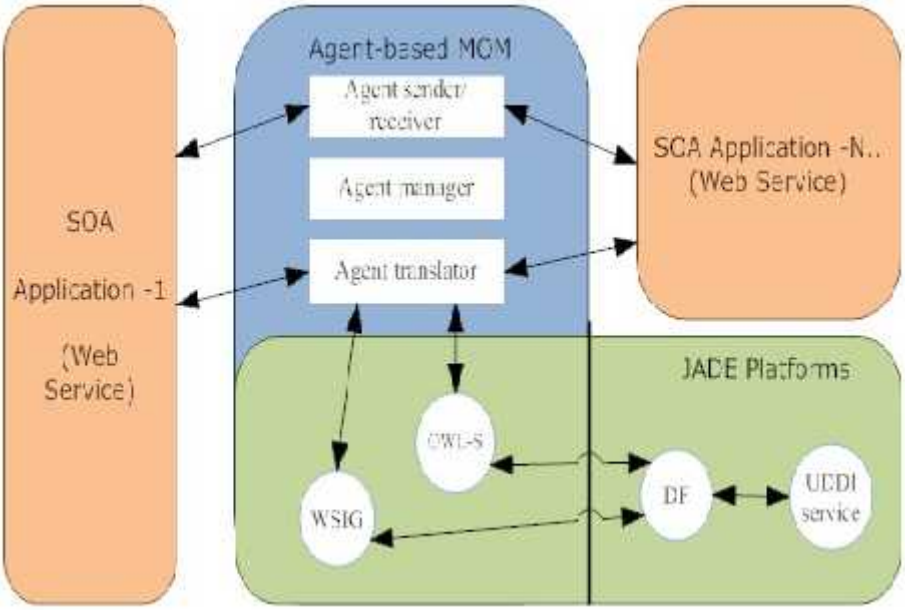

Figure 2 Agent Translator Model (Overview)

In Figure 2, the translation model presents the different components within the Agent-based MOM system and how they are connected to the JADE Platforms. There are four main systems included in the overview model which are the Agent-based MOM, JADE Platforms, SOA applications 1 and SOA applications N. SOA applications 1 and N are represented as different applications that are connected to the Agent-based MOM where the Agent-based MOM is seen as that main system that manages all the translation processes by way of the translator model. The JADE Platform is the agent platform that provides the facilities for the Agent Management system (AMS). An agent translator is connected with WSIG and OWL-S in the JADE platform where the WSIG plug-in is used for the translation of WSDL to ACL and the OWL-S plug-in is for the translation of ACL to WSDL. Both WSIG and OWL-S are connected to DF and UDDI (Universal Description, Discovery and Integration) for the mapping process that will be explained in detail in the following section [19-20].

B. Translation mapping

In the previous section, the overall view of the translation model has been presented which consists of the related systems and components. In this section, the step by step process of the translation mapping will be described in detail as shown in Figure 3.

The translation model consists of two main components which are the WSIG (Web service Integration Gateway), the OWL-S (Web Ontology Language – Semantic) that are connected to the JADE platform.

Web service represents a different application of SOA that will send the Web service message to the agent translator. There are two cases of the translation process which are the message from WSDL to ACL and the message from ACL to WSDL. In the first case, the translation is required to enable the communication between agents for different applications. In the second case, the ACL message will be translated into WSDL for message mapping and matching with the supported message type of the partner SOA application. The library file of DF is written in the WSDL format which makes it possible to encode and store all the supported SOA applications of the Agent-based MOM [21].

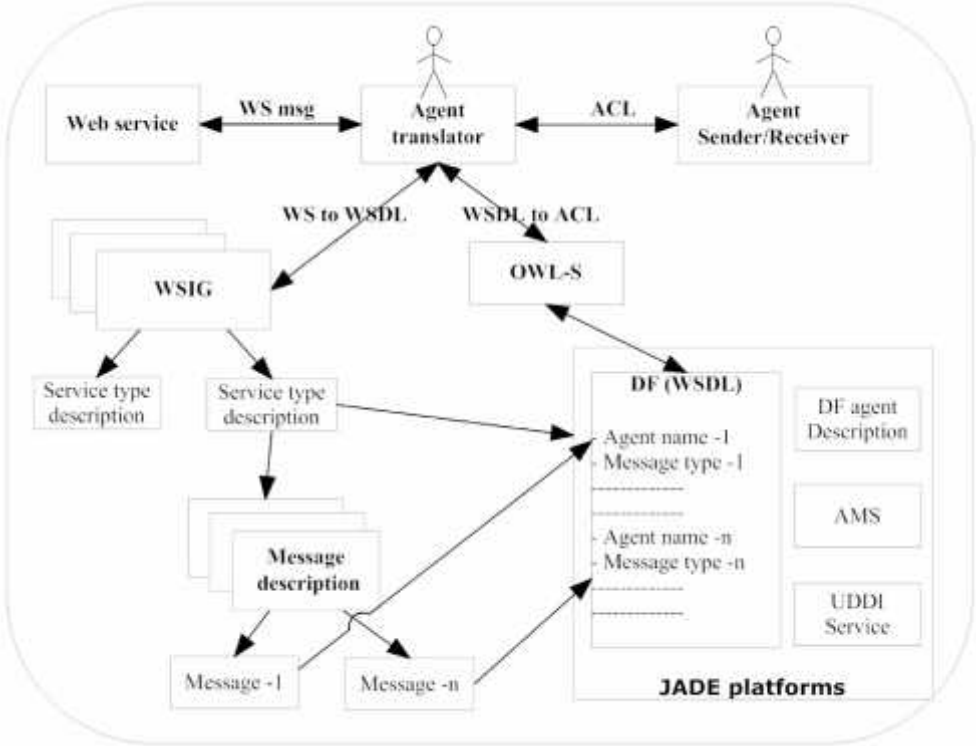

Figure 3 Translation Mapping

In addition, the mapping process starts when the translation model has received a message from an attached SOA application and a manager agent. As has been described in the earlier section, the message from the attached SOA application is a web service message and the message from the manager agent is an ACL message. When the agent translator has received the message, it first needs to analyze the message format as to whether it is WSDL or ACL. If the message is Web service, it will be forwarded to WSIG to describe the message into the WSDL format. WSIG will collect the information of all the supporting messages in the DF library to create an accurate WSDL. Afterwards, the message will also be translated into ACL by OWL-S. Therefore, it allows the agent to communicate among different SOA applications. If the message is in the ACL format, it will be translated to WSDL by OWL-S which allows message mapping with the DF library. The mapping will be matched and will collect information in DF based on the message request of the partner application to facilitate WSIG in converting the message to its own Web service format. The real case study-based implementation will be presented [21-25].

C. Experimental setup

The case study used in the experiments will be adopted in two different manners, which are existing Message Oriented Middleware (MOM) and existing cross-platform solutions. The experimental setup of each manner is different in terms of experimental setup and environment. Firstly, existing MOM and Agent-based MOM are implemented using the similar experimental setup, which all the control parameters for simulation are exactly same as shown in table 1. These simulation metrics are the standard implementation parameter in Java agent development framework (JADE). JADE has been proposed as a standard implementation tools by TILAB (Telecom Italia). Secondly, existing solutions for cross-platform is implemented in JADE environment where it is also using to evaluate the performance of the proposed solution.

Table 1: Simulation parameters

Parameters	Values
Round-trip time (request/respond)	Seconds
Bandwidth	10 Mbps
Packet size	30 bytes
Minimum load	< 10 tasks at a time
Maximum load	> 10 < 100 tasks at a time

As shown in table 1. The experiments evaluate the effectiveness of the proposed cross-platform framework in term of the measured attribute, the communications round-trip (request/respond) time in seconds, against workload indicated by the number of request invoking at the same time. The link bandwidth was set to 10 Mbps and the packet size was set to 30 bytes. If the task is more than 30 bytes then the system is reported as a complex message, which will be not executed. Others experiments have been controlled in a way that some violation to the communications will occur in pre-specific time to check the system ability to detect these violations correctly. We show the variation of the data measured by communications agents from that monitored at the source code level of an agent. Several directions were classified that were used for evaluation and validation purposes in this work. The java method was used (long System.currentTimeMillis()) to measure the time intervals that returned the number of milliseconds as a standard agent-based measurement. Additionally, the case study is composed of three different SOA-based applications to evaluate multiple types of SOA application where most of existing solutions are supported only two different type of SOA. Each of application is attached with agent-based MOM (ABMOM) to facilitate the communications process among them. There are SOAP-based applications, CORBA-based applications and REST-based applications. These SOA-based applications represent the different partners of the SOA-based applications that are required to communicate with each other. First, the ABMOM is installed into each of the SOA application as a plug and play application. Then, some task is allocated to each SOA-based application to communicate with each other. The original form of request is in the web service format that will be described by WSDL as explained in the proposed translation model. Afterwards, the multi agent system of ABMOM executed a task based on each role of the agent.

As shown in Figure 4, ABMOM resides in the SOAP-based application, CORBA-based application, and REST-based application. All applications run in JADE environment and Apache Server. Three important elements are present in the FIPA compliant platform. The Agent Management System (AMS) controls the access of the platform and the Directory Facility (DF). DF provides a library service and Agent Communications Channel (ACC) that facilitates the message transport service for FIPA ACL message delivery among agents living in different agent platforms [26-27]. In the case study, there are three different SOA-based applications to evaluate the communications performance of ABMOM as described earlier. In the next section, the case study of Multi-Agent Book Broker application will be presented

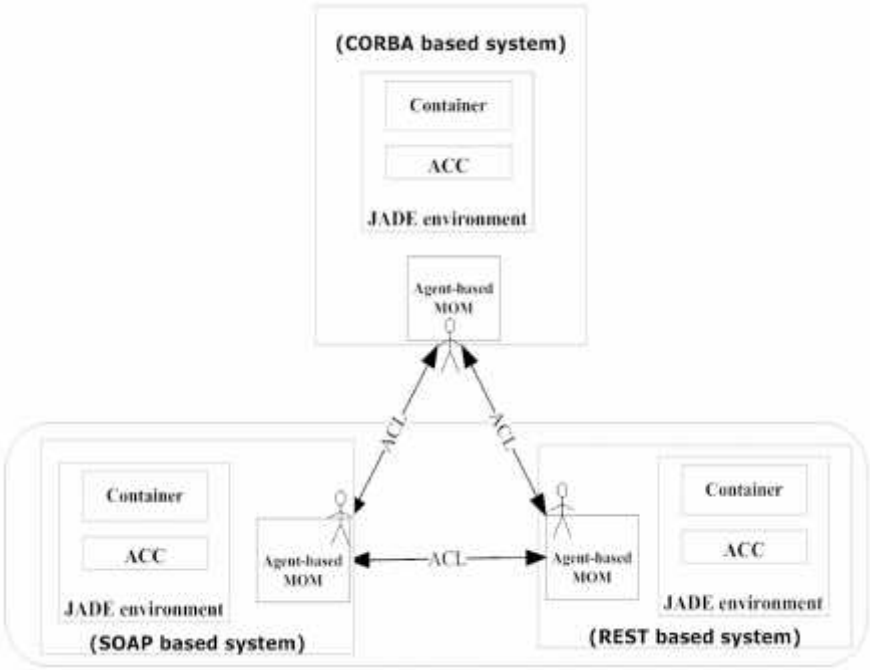

Figure 4 Simulation Metric

V. RESULT ANALYSIS AND COMPARATIVE STUDY

In order to evaluate and validate the proposed framework, the communications between multi-agents systems in the experiments are monitored in two different levels. First level is monitoring the communications using code level monitoring instrumentation in the source code. In the second level, we used the proposed cross-platform framework to monitor the communications among them. Additionally, all the experiments scenarios have been implemented in two scenarios. First implementation is the proposed cross-platform applied in minimum requests and second, implementation at the maximum request to check the actual performance between a proposed framework and existing solutions. Minimum task execution means that less than ten tasks were allocated at the same time and the maximum task execution mean that more than ten but less than hundred tasks were assigned to the system at a time. Finally, all these implementations have been repeated 20 times to increase the accuracy.

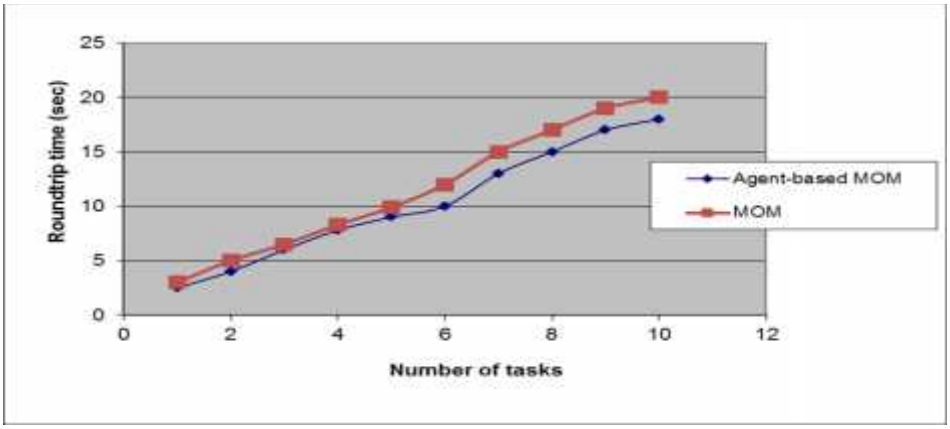

Figure 5: Round-trip time (Minimum Load)

In the first run of the experiment, two different scenario of task execution were included, an execution of existing MOM and the proposed agent-based MOM. This experiment was to evaluate the round-trip time performance of the communications as shown in figure 5 the changing of the round-trip time value depends on an increasing number of tasks. From the results, it was found that there was not much difference between the proposed agent-based MOM and existing MOM at minimum task execution. However, the agent-based MOM did show a slightly less round-trip time performance.

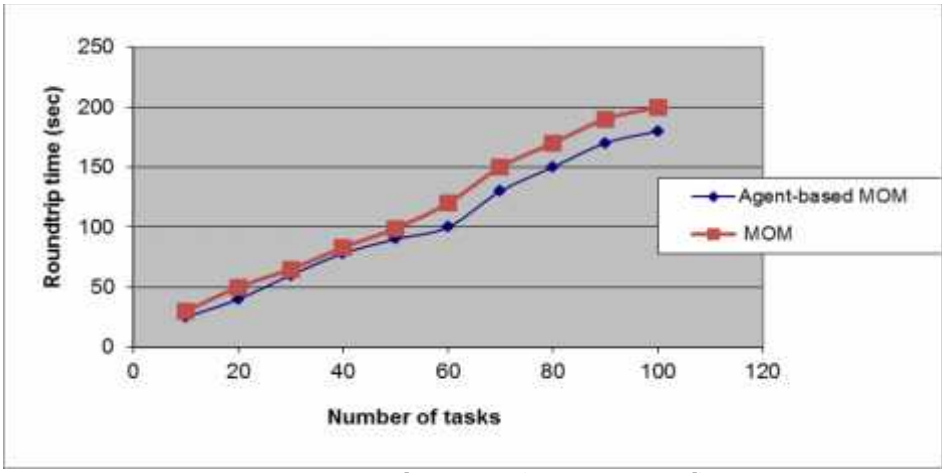

Figure 6: Round-trip time (Maximum Load)

In figure 6, the changing of the round-trip time at this specific value of the graph is clearly shown. In the second run at the maximum number of tasks allocated, the graph of the round-trip time due to the change in the number of tasks is slightly different between the proposed agent-based MOM and existing MOM; with an agent technology, the application was able to manage the communications more efficiently during the full load of tasks.

The linear pattern of the graph also had a substantial effect on the value of the round-trip time and the number of tasks executed. As shown in figure 5 and figure 6, the average number of tasks allocated at the minimum number of workload in the system was slightly similar. However, agent-based MOM was able to achieve the communications more effectively than the traditional MOM.

In addition, the difference in the round-trip times between the minimum and maximum number of tasks allocated was caused by the network, the associated software components and the number of tasks which were necessary to invoke the communications. Figure 6 shows the round-trip times for the first 10 requests to execute the allocated tasks. The average round-trip time was 20 seconds. The difference in the round-trip time between the requests was caused because each request had different message requirements in order to get a response.

Furthermore, an experimental comparative study was conducted between the proposed solution and Java message service (JMS). JMS has been selected based on the similar functionality provided for the cross-platform communications where most of existing solutions are focus on some particular problem. It is also can be considered as a standard benchmarking middleware provided by Java application for distributed and loosely couple application and cross-platform communication

According to literature review, the existing solutions can be divided into two different types. Firstly, the existing solutions inform of middleware that enable integration and communications between distributed SOA applications. It consists of a standard functionally to be configured and all the communications processes are required a respond from human to complete the communications task. Secondly, the existing solutions for cross-platform those are developed for a specific and particular problem or issue. However, most of them did not consider the general requirement for the cross-platform communications process.

Therefore, the first type of existing solution is the most suitable for a comparative study and once of well-known middleware has been selected to conduct a comparative study. In this section, the evaluation of the proposed framework and empirical comparative study between the proposed solution and Java message service (JMS) will be presented. JMS is a messaging standard that allows application components based on the java enterprise edition to create, send, received, and read message. It allowed the communications between different components of a distributed application to be loosely coupled, reliable, and asynchronous such SOA.

Table 2: Experimental Test Results for Round-trip Time

Comparison Metrics	**JMS (Java message Oriented Middleware)**	**Agent-based Message Oriented Middleware**
Minimum Load	96.50%	99.41%
Maximum Load	92.30%	95.59%

Java message oriented middleware has been chosen for comparative study due to java application program being widely cited in literature study. It can be consider as establish application for distributed and different application and they are well known in message exchanging enterprise system. The experimental comparative studies evaluated two functionalities of the system in the overall experiment test. First, was the round-trip performance of the communications both with the minimum and maximum workload as shown in table 2. For the minimum load ten requests (R) sent for twenty times and ten responses were received with several different received times in milliseconds, which will be converted into percentages as shown below.

Where,
Percentage Round trip time (PRT) = Average respond time (ART) / Request (R) * 100.

JMS(Java message Oriented Middleware)	Agent-based MOM
PRT = (ART / R) * 100 = (9.65 / 10) *100 = 96.5 %	PRT = (ART / R) *100 = (9.94 / 10) *100 = 99.4 %

For maximum load of one hundred requests (R) sent for twenty times and one hundred responses received with several different received times in milliseconds, it is also converted into percentages as shown in the calculations below.

JMS(Java message Oriented Middleware)	Agent-based MOM
PRT = ART / R *100 = 92.30 / 100 *100 = 92.30%	PRT = ART / R *100 = 95.59 / 100 *100 = 95.59%

VI. CONCLUSION

In this research, a novel agent-based Translation Model has been suggested to ensure the cross-platform 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] W. Chou, L. Li, and F. Liu, Web Services for Service-Oriented Communication. 2006, IEEE: USA.
[2] N. Looker, and J. Xu, Assessing the Dependability of SOAP RPC-Based Web Services by Fault Injection. International Workshop on Object-Oriented Real-Time Dependable Systems, 2003.
[3] Lexington and Massachusetts, Services Oriented Architecture (SOA) Market Shares, Strategies, and Forecasts, Worldwide, 2011 to 2017. 2011, WinterGreen Research.
[4] M.H. Selamat, and A.A. Kharusi, Service Oriented Architecture in Education Sector. IJCSNS International Journal of Computer Science and Network Security, 2009. 9 No.5, May 2009(Computer Science and Network Security).
[5] J.P. Lawler, and H. Howell-Barber, Service-Oriented Architecture (SOA strategy, Methodology, and Technology). 2008: Springer.
[6] J.M. Pullena, R. Bruntonb, D. Brutzmanc, D. Drakeb, M. Hiebd, K.L. Morseb and A. Tolke, Using Web services to integrate heterogeneous simulations in a grid environment. Future Generation Computer Systems 21, 2009: p. 9.
[7] Mintchev, A. and S.L. Bulgaria, Interoperability among Service Registry Implementations: Is UDDI Standard Enough, in International Conference on Web Services. 2008, IEEE
[8] A.N.K. Chen, S. Sen, and B.B.M. Shao, Strategies for effective Web services adoption for dynamic e-businesses. Decision Support Systems, 2005. 42: p. 789– 809.
[9] P.A. Buhler, J.M. Vidal, and H. Verhagen, Adaptive Workflow = Web Services + Agents. 2003.
[10] H.P. Huy, T. Kawamura, and T. Hasegawa, Web Service Gateway – a step forward to e-business, in International Conference on Web Services. 2004, IEEE.
[11] Y. Charif, and N. Sabouret, An Overview of Semantic Web Services Composition Approaches. Electronic Notes in Theoretical Computer Science, 2006: p. 33–41.
[12] B. Li, Research and Application of SOA Standards in the Integration on Web Services, in Second International Workshop on Education Technology and Computer Science. 2010, IEEE: Chaina.
[13] T. Takase, and K. Tajima, Efficient Web Services Message Exchange by SOAP Bundling Framework, in International Enterprise Distributed Object Computing Conference. 2007, IEEE.
[14] N. M.Ibrahim, M.F. Hassan, Enhancement of Message Oriented Middleware for multiple type of SOA system, special edition of the Int. Journal of Science, Lahore (ISSN 1013-5316), 2014 ,Indexed ISI.
[15] Z. Peng, Y. Shi, Z. Zhang and W. Fan, Analysis and Implementation of Web Collaborative Environment Based on SOAP, in Pacific-Asia Workshop on Computational Intelligence and Industrial Application. 2008, IEEE.
[16] N. M.Ibrahim, M.F. Hassan, and M.Hussin Abdullah, Agent-based Service Oriented Architecture (SOA) for Cross-platform, International Journal of Soft Computing and Software Engineering (JSCSE), 2013, Indexed ISI and Scopus.
[17] C. Werner, C. Buschmann, Y. Brandt and S. Fischer, Compressing SOAP Messages by using Pushdown Automata, in International Conference on Web Services. 2006, IEEE.
[18] L. Li, C. Niu, N. Chen and J. W, High Performance Web Services Based on Service-Specific SOAP Processor, in International Conference on Web Services. 2006, IEEE.
[19] T. Jepsen, SOAP Cleans up Interoperability Problems on the Web. 2001, IEEE.
[20] H. Liu, X. Lin, and M. Li, Modeling Response Time of SOAP over Http, in International Conference on Web Services. 2005, IEEE.
[21] N. M.Ibrahim, M.F. Hassan, M.Hussin Abdullah, ABMOM for Cross-platform Communication in SOA Systems, 3rd International Conference on Research and Innovation in Information Systems – 2013 (ICRIIS'13), University Tenaga Malaysia, IEEE, Kuala Lumpur.
[22] X. Li, An Agent/XML based Information Integration Platform for Process Industry, in 2010 2nd International Conference on Computer Engineering and Technology. 2010, IEEE.
[23] T.M. Chester, Cross-Platform Integration with XML and SOAP, in IT Pro September , October 2001. 2001 IEEE.
[24] X. Li, An Agent/XML based Information Integration Platform for Process Industry, in 2010 2nd International Conference on Computer Engineering and Technology. 2010, IEEE. p. 527.
[25] N. M.Ibrahim,. and M.F. Hassan, Cross-platform Communications Model for different SOA Applications, in ICCOINS2016. 2016, IEEE: Kuala Lumpur, KLCC.

.