MATHEMATICAL MODELING USING COORDINATION MECHANISMS FOR MULTI-AGENT SYSTEMS IN SERVICE ORIENTED ARCHITECTURE

NAJHAN MUHAMAD IBRAHIM[1*], MOHD FADZIL HASSAN[2]

[1]Kulliyyah of ICT, International Islamic University Malaysia, Gombak, Malaysia
[2]Faculty of Science and Information Technology, Universiti Teknologi PETRONAS, Perak, Malaysia

**najhan_ibrahim@iium.edu.my*

(Received: Day Month Year; Accepted: Day Month Year; Published on-line: Day Month Year)

https://doi.org/10.31436/iiumej.vxxix.x

ABSTRACT: A heterogeneous communications mechanism is essential between different types of SOA applications to ensure the communications will be well maintained. The need for heterogeneous communications within SOA has attracted different research efforts as reported in various literature. This research interest is specifically focused on interoperability of heterogeneous communications in multiple types of SOA applications. The study of the coordination problem is an important issue in the multi-agent system. In the current software development industry, most software products are produced with the expectation that the application will provide intelligent coordination with the minimum level of interruption. This case is not always acceptable especially for the critical system especially those that deal with complex communications that involved multiple types of participants where the partner application needs to know about the basic guide and specification of each other. The proposed solution is dealing with one of these issues as it evaluates the coordination between the partner applications where the accuracy of responding is involved. For this reason, we modeled the proposed system mathematically using Coordination mechanisms to validate and verify the system for heterogeneous communications in SOA system.

KEY WORDS: ***Formal Methods, Mathematical Modeling, Multi-agent Systems, Service Oriented Architecture (SOA), Coordination Mechanisms.***

1. INTRODUCTION

In software engineering, formal methods are mathematical based techniques used to provide the specification, development, and verification of software systems prior to the implementation. The formal methods have been used widely in software development because of its promising benefits in coherence degree and coordination between activities of software application. Coordination is a common issue in distributed heterogeneous communications, which implemented in agent-based software. The proposed research for heterogeneous communications uses and applies theories and techniques from several areas, such as Service Oriented Architecture, Web services, Agent technology.

In addition, the suggested heterogeneous communications framework is elaborating by representing it mathematically using Coordination mechanisms. This process enables to evaluate and validate the proposed solution mathematically. This step is important as it can proof explicitly the validity and completeness of the suggested framework (Hassan et. al., 2014). This section starts by highlighting the importance of mathematical modeling and why it is widely used in software engineering. Follow by introducing the components of Communications Net Protocol that will be used in this section. The definition of the operations in the proposed solution that we will model and then identify their rules that have to be validated. The formal specification of the system will be presented in this section as well which includes basic types definition, global constants definition, the change, and fixed schemas, the initial state, and the operations schemas. This formal definition allows us to validate the system mathematically. The importance of this section relies on the value of mathematical validation of any software system before the prototype implementation. The validation process shows the stability and validity of the system in all the situations.

2. DESIGN AND DEVELOPMENT OF THE PROPOSED FRAMEWORK

In the effort to implement the proposed agent-based communications framework, the agent's framework analyzed and designed using standard agent-based modeling and simulation methods. There are several agent-based modeling methods available to the model multi-agent system such as ABM (agent-based modeling), ABS (agent-based systems or simulation), and IBM (individual-based modeling). However, Agent-Based Modeling and Simulation (ABMS) is the well know and widely used modeling method to analyze and develop a complex application that is comprised of intelligent and flexible interaction (WaiShiang et. al., 2017). ABMS is also considered as a modeling approach that can be exploited to represent some system attributes that are not clearly described as attributes or functionalities in a complex application. It is becoming accepted for its effectiveness in several application-based domains (e.g., Grid computing, Cloud computing, and Service-Oriented Architecture). Therefore, ABMS is as a modeling technique that most suitable to the model multi-agent system for heterogeneous communications (Parhi et. al.,2014; Hamzah et. at.,2017; Menasce et. al.,2000; M. Ibrahim et. al. 2011).

Figure 1: Basic Architecture of Agent in ABMS (Ni et. al, 2017)

In general, an agent-based model and simulation will act and react to the environment as shown in Figure 1 based on attributes rules of the agent. There are six elements of agents that consume and react to the environment which are attributes, behavioral rules, memory, resources, decision making sophistication and rules to modify behavioral rules. An attribute and rules can be identified and describe by the developer to handle some particular task at a precise time. In the next section will be described in detail the agent-oriented modeling techniques to construct the framework.

2.1 AGENT ORIENTED MODELING TECHNIQUES

To implement the proposed agent-based message-oriented middleware, the paper analyzed and designed a multi agent's framework using a standard modeling and designing methods. In software development, the modeling stage is one of the most imperative stages to develop the system as it helps to understand how each component in the system would work and how it could be implemented. The modeling technique that will be used in this research work is easyABMS and AUML (Agent Unified Modeling Language) where both are the most efficient to develop an agent-based framework (Klein, 2017; M.Ibrahim, et. al., 2014). The techniques used will be discussed systematically in this section.

2.1.1 The easyABMS

easyABMS has been recognized as standard agent-based modeling and simulation due to its adaptive and flexible to develop agent-based applications [10]. It consists of seven stages from the Analysis of the system-based requirements for the transaction process, which are Conceptual System Modeling, Simulation Design, Simulation Code Generation, Simulation Set-up, Simulation Execution and Results Analysis. easyABMS identifies four main processes which are integration, model-driven and visualization where Unified Modeling Language (UML) is a virtual diagram and notation. easyABMS also utilizes the advanced features of visual modeling and automatic code generation offered by the Repast Symphony Toolkit (RST), the most well-known open-source ABMS platform. Moreover, referring to the model-driven paradigm, the simulation code is able to generate from the consequent system Simulation Model (Simon, et. al., 2014; Garro, et. al., 2009; M. Ibrahim, et. al., 2016).

2.1.2 AGENT UML (AUML)

Unified Modeling Language (UML) is one of the significant modeling methods in object-oriented environments where it promoted researchers to enhance and support agent-based programming. In order to model an agent, UML does not meet basic requirements where the object environments are usually static nevertheless agent environments are dynamic in nature and communicate with each other independently (Belghiat, et. al. 2016). Therefore, The Agent Unified Modeling Language (AUML) is necessary to enable agent-oriented programming (AOP) and simplify implementation. AUML is an agent software modeling which is an extension of the Unified Modeling Language (UML), proposed as a standard by the

Foundation for Intelligent Physical Agents (FIPA) (Campean, et. al., 2017). In this research work, the proposed multi-agent framework the roles of the agent systems were identified by AUML. Also, the interaction between every different platform was modeled using easyABMS. The AUML sequence and collaboration diagrams were used to describe the communications between all the agents. Finally, the activity and state diagrams were used to describe the internal processing of each agent. In the next section, modeling and simulating agent-based message-oriented middleware will be presented.

3. MODELING AND SIMULATING MULTI-AGENT FRAMEWORK VIA easyABMS

Agent-based Modeling and Simulation (ABMS) can be considered as a significant software methodology for developing and modeling complex agent-based applications. It is become difficult to deny its effectiveness in the development of distributed and heterogeneous applications due to its flexibility and adaptability in the development process. Currently, varieties ABMS modeling and method available to develop an agent-based application which can be obtained from in different domains such as SOA, Grid computing and Cloud computing. In this research work, easyABMS is using to model and construct the proposed agent-based message-oriented middleware. Consist of five main consequent processes, which are the System Analysis, Conceptual system modeling, Simulation design, Simulation execution and Result Analysis (Simon, 2014). The following sub-section will describe each process in detail.

3.1 SYSTEM ANALYSIS

The key achievement of the heterogeneous communications in SOA applications is the success of transaction over different SOA standards and specification, which use different types of web service messages in the communications, for instance, SOA-based application A may use SOAP messages. On the other hand, SOA-based application B may be implemented in REST. Furthermore, this section will focus on analyzing the basic requirements and relationship of the proposed solution. The System Representation is based on the foundation of the system and the simulation objectives as presented in Figure 2. There are two types of relationships, which are inter and intra entry relationships. Inter entry relationship means the communications between different SOA-based application, which interfaces represented by agent sender and agent manager. In contrast, intra entry relationship is the extract of communications between agents within the SOA-based application. All the entities represented in Figure 3 are described along with their relationships and rules which represent the basic interaction among the different SOA applications (Hamzah, et. al., 2017).

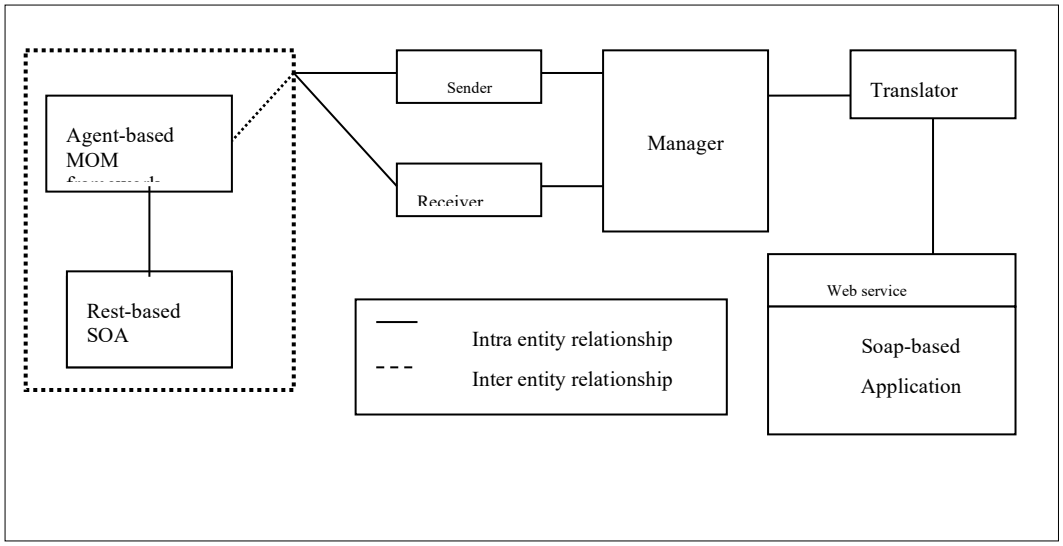

Figure 2: System Representation

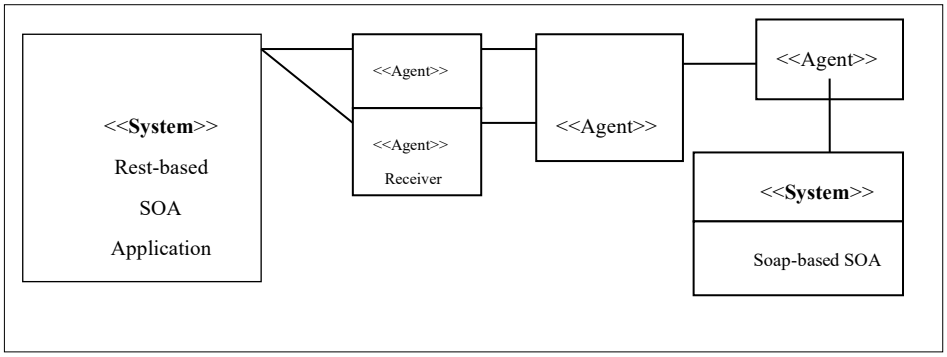

Figure 3: Structural System Model

3.2 CONCEPTUAL SYSTEM MODELING

The Structural System Model resulting from the System Representation is presented in Figure 2. According to the simulation objective, this concerns communications management for the level of representation. The rest-based SOA application is represented on more abstract from respect to the level of resulting in the analysis phase. In Figure 4, the corresponding Agent in each entity of the Structural System Model and System Model are defined. It also described the meaning of each symbol in the model such as time signal, send the signal, accept the signal, action, decision, final node, and flow-ledge. The following subsection's diagram presents the Agent Model for the receiver, manager, translator and system model for the proposed multi-agent framework.

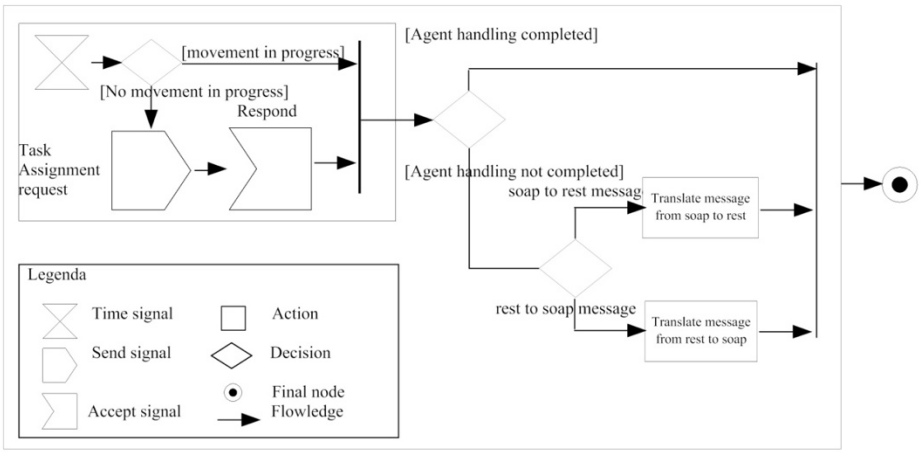

Figure 4: Part of the Multi-agent Framework Model

Figure 5 demonstrates an Agent Behavioral Model of the proposed work. The handling activity identifies the activities of the movement task, which agents execute to accomplish the goals in conjunction with the pre/post conditions and the implementation schedule. Furthermore, as the description of an Agent Behavioral Model requires which each activity in the Agent Activity must be further described by a UML Activity Diagram in the following section (WaiShiang, et. al., 2017).

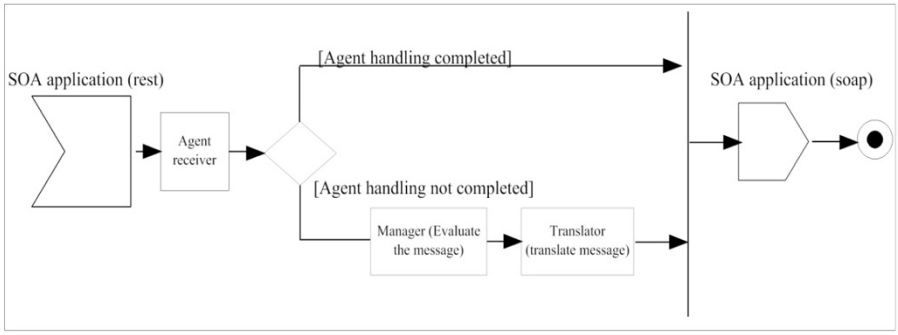

Figure 5: Part of the Movement Task Behavioral Model

3.3 SIMULATION DESIGN

The Simulation Design is the abstraction level of the framework to implement the simulation. Based on easyABMS methodology and several web service integration techniques (theoretical and practical) the proposed agent-based message-oriented middleware (abstraction level) has been developed. It has been adopted from a service-oriented perspective (SOP), with a high degree of intelligent that supports flexible collaborations and computations on a large complex application. This comprehensive framework has been constructed based on AUML modeling. The process of each stage also has been clearly defined by easyABMS (Simon, 2014).

3.4 SIMULATION EXCUTION AND RESULT ANALYSIS

In the proposed research work, the simulation prototype was designed by using the NetBeans programming framework where java agent development framework (JADE) as agent plug-in library that supports all agent's rules and specifications. NetBeans can be referred to as a platform framework for an integrated development environment (IDE) that supports multiple programming applications such as Java, PHP, C++, and JavaScript. It also allows applications to be developed from a set of software components known as modules. Third-party developers can extend the application based on the NetBeans platform. Moreover, JADE is also a software framework implemented in the Java language. The implementation of a multi-agent system is carried out through an included library that consists of FIPA specifications and through a set of graphical tools that supports the debugging and deployment phases (Laftah Al-Yaseen, et. al., 2016).

4. FORMAL METHOD AND EVALUATION

The formal method can be defined as a foundation to describe the complex system and reasoning about the system that supports the development of the program. It uses the mathematical expression for specification, development, and verification of software. Formal methods are usually used to verify the system stability and reliability theoretically before the empirical testing (Bennajeh, et. al., 2015). The behavior can be described by a formal specification where a formal development facilitates the product to meet the specification. The formal method also supports formal reasoning to enable a clear understanding of the system without ambiguous activities. “The formal methods can be used in three essential activities, which are design, modeling, and verification” (Terán, et. al., 2013).

The common methods of design such AUML can only represent the structure of a program that includes the part of the program and how they incorporate with one another. It allowed only a few attributes to check the design correctness, or to decide which design strategy is better to be followed. Mathematical modeling and verification are considered as a more powerful design strategy because it can also model the behavior of the system. Therefore, it provides a scientific way to show that an abstract model derived from the modeling stage and a detailed design created in the design stage are describing the same thing. It also provides the development steps, and capable to check each step's correctness by calculation and proof. Therefore, mathematical modeling helps us to come up with error-free and correct design without coding and testing a program.

Modeling a software system mathematically can be defined as the software behavior comprehensively inaccurate way, and it is often shorter, easier, and more precise than the code. Additionally, we can use the mathematical model to expect accurately the behavior of the program before we start coding it. Therefore, mathematical modeling makes the behavior of the software predictable, which required for complex and critical systems such as safety systems. Moreover, the formal model allows us to calculate the results the program should get. Therefore, the formal model as an oracle will help us determine the necessary test cases and it informs us whether the program would pass the test cases or not. The main facility for the verification methods in software engineering is that it can prove the

stability and reliability of the system. Proof normally provides more confidence than testing because proof considers all the possible causes, while testing just selects some of them. Moreover, proof can be more convincing than appeals to expectations because it is more explicit, easier to check and therefore less fallible than unproved expectations (Bennajeh, et. al., 2015).

4.1 COORDINATION MECHANISMS

In the previous section, the formal method to validate the proposed research work has been presented. In this section, the tool that was used for this purpose will be introduced. Coordination is a central issue in agent-based software, and specifically in a distributed intelligent system. The coordination mechanisms can be divided into two categories, cooperation, and negotiation. Cooperation is a type of coordination that consists of many agents whose actions do not disturb each other. Negotiation is a kind of cooperation that can be achieved via agents of a system, in which each one is delivered the responsibility to make the communications successfully. Commonly, every agent has been designed for a different purpose, in this case, the coordination mechanisms must achieve their objectives. Therefore, in this research contract net protocol has been selected to design and analysis the coordination mechanisms where it is a standard used coordination protocol (Terán, et. al., 2013).

4.2 CONTRACT NET PROTOCOL MODELS

The contract net protocol was developed by Randall Davis and Reid Smith with the purpose to solve collaborative problems. The goal is to coordinate the communications between several agents, where it will be complexity by handle a task in a single agent. Each agent has a specific role and responsibility in the communications process. The protocol uses several basic messages like the following [18].

- Sender
- Receiver
- Manager
- Translator

These messages provide the functionality needed to make cooperation between agents. The FIPA interaction diagram of this protocol has presented as shown in Figure 6.

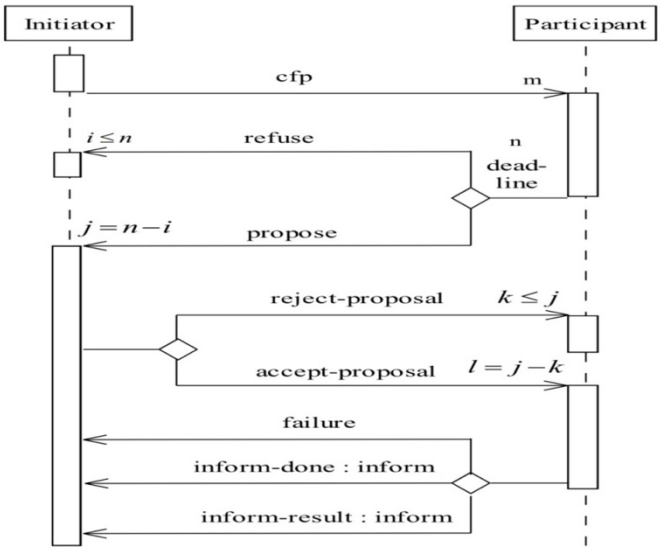

Figure 6: Interaction Protocol of Contract Net (Terán, et. al.,2013)

The following defines the variables, roles and cooperates describing the coordination mechanisms that are studying in this research work.

Let AT be an agent task to be performed at any time.

AT_j designates a set of sub-tasks in which AT can be divided.

A group of agents $AG = (a_a (AT_j), A_{sr})$ is defined, where $a_a (AT_j)$ is the manager agent for a sub-task AT_j. AG_s is the set of agents that can be potential receiver, $AG_s = \{AG_{s1}, AG_{s2}, \ldots, AG_{sn}\}$ and AG_r is the set of agents that can be sender, $AG_r = \{AG_{r1}, AG_{r2}, \ldots, AG_{rn}\}$.

Furthermore, it is assumed that $|AG| > a_a (AT_j)$, this ensures that at least two agents are engaged in the communications. Finally there will be a $AGs (ATj)$ that will be the sender agent for the task, i. e., $\forall AT_j \ \exists \ AG_i \in AG : AG_i = AG_i (AT_j)$. "This protocol can be viewed as a 8- tuple L" [18].

$$L = (M, f (T_j), \overrightarrow{0_c} , g(O_c), M_p, RP, h_c (RP), RF)$$

Where

- IM is the initial message, $IM = (AT_j , I_a , F)$, where T_j is the type of task expecting to be completed, j={1,…,n}, such that j is the sub-tasks identifier. I_a is the information of the manager agent (it can include its virtual address (DIR_a), etc.), and ET is the expiration time to complete the task (it is given by a numeric pair ET = [HH , MM]).

- ET (AT_j) is a function that allows potential sender or receiver to assess their capacities to respond to the notice of request for the performance of the task AT_j.

- $\overrightarrow{O_c}$ is a vector, $O_c \in IM_{m\,x\,1}$ where $O_i \in O_c$, $i = (1,,,,,,m\}$ containing the offer of each contractor i to complete the task AT_j
- $g(O_c)$ is a function that represents the offer evaluation criteria of the manager agent.
- IM_p is the message to the winner contractor. It is a tuple $IM_g = (A_c$, *[accept – offer])* , where WA_c is the name of the receiver agent, with the communicative act "accept-offer". For the rest of agents, the manager agent distributes a message that they did not receive the message.
- RP is a vector, $RP \varepsilon IM_1 \times r$ where each cell RP_k Î RP represents the progress report k of the agent c receiver of the sub-task AT_j (for k={1,…,r}).
- $h_c(RP,k)$ is an evaluation function of the manager agent to know the degree of task execution AT_j by the contractor.

RF represents the final report of the receiver of the task agent AT_j, and it is the last element of vector RP, i. e., RF Î RP .

4.3 CONTRACT NET PROTOCOL FOR THE PROPOSED FRAMEWORK

A multi-agent system has become a valuable technology to assist heterogeneous communications due to the distributed nature of the process and task to be solved in real-time. The problem of fault management has been widely studied and several approaches have been presented [18]. The coordination model has been addressed by defining the main conversations to accomplish the service that multi-agent offers. In the proposed framework, the conversation or communications between agents will be used in the proposed model.

The fault management systems (FMS) are involved in two elements, the first perform the failure analysis where it includes fault detection and diagnosis. The second performs the task of the maintenance management system, which includes the prediction of functional failure, preventive maintenance, and execution of maintenance. The FMS is a sub-system of an intelligent system where some activities of FMS are followed a distributed computing model. A specific methodology is used to define a specification of a multi-agent system. (Bennajeh, et. al., 2015) introduced the methodology in three models, which included the agent model, task model, and coordination model.

Agent model: The determination of agents based on the agent model in the previous section provides the following functions; detect, analyze, predict the occurrence of a fault, and correct in the multi-agent system. These functions have been proposed for evaluation and validate the multi-agent system. In this research work, there are several agents represent the role and defines in the agent model, although some of the agents were divided into several tasks. There are four main agents in each sub-system: Agent sender, Agent receiver, Agent manager and Agent Translator.

Task model: The following Table 1 shows the task model of FMS

Coordination model: Based on the simulation, On-condition maintenance is proposed for the process of evaluation and validation.

Table 1: Task, Sub-tasks and Sub-tasks Type

Tasks	**Sub-tasks**	**Sub-tasks Type**
Observation task (OT)	Interrupt Functional Failure Identification (IFFA) Performance Indexes Calculate (CIF) Maintenance State (EM)	Identification task (IT) Processing task (PTo) Monitored task (MTo)
Detection task (DT)	Fault Occurrence Statistics (EOF) Detection Technique Selection (STD) Incorporating of Detection Methods (IMD)	Statistic Calculation (SC) Selection task (ST) Processing Search (PS)
Location task (LT)	Fault location (UF)	Search
Diagnostic task (DTi)	Failure Mode Statistic (EMF) Failure Cause Statistics (ECF) Failure consequence Analysis (ACF) Identification of Failure Modes and Causes (IMFC)	Statistical Calculation Search Processing Identification
Prediction task (PT)	Reliability Curve Calculation (CCC)	processing
Coordination task (CToo)	Evaluation of Resource (ER)	Evaluation task (ET)

Conversation:

- Objective: Performance of tasks on the system
- Interacting agents: Coordinator (Acoo), Controller (Acon), Database (ABD), Detector (AD), Predictor (AP), Diagnoses (ADi)
- Beginner: Agent sender
- Speech Acts: Send of message, Receive of message, Get process Information, Historical View, and Translate.
- Precondition: Have to receive the initial message to begin
- Ending condition: The communications will be completed after receive the respond
- Description: Every communications process will be stored in a log file for record.

Before begin modeling the proposed framework, the basic entities should define that are going to use within the framework. The main basic entity to use is the agents defined earlier with the sub-elements within the multi-agent system such as agent sender, agent receiver, agent manager and agent translator. Additionally, few schemas need to be declared that represent different types of the records that exchanged between the agents and saved in the log file for tracking and validation purposes. These records are Rules, Status, Metrics, and Responsibilities. Rules record the set of conditions where the communications parameters are violated. Statuses record the set of quality attributes and their values at different times. Metrics store the set of the measurement directives and metrics values read by the heterogeneous communications framework. Finally, responsibilities record all the obligation cases that occurred during the communications process.

The simulation is a book broker trading system, which included three subsystems as the following: CORBA-base system, SOAP-based system, and REST-based system. At one point, some agent act as a seller to send a message (broadcast message) for a book for sale. This message will be answered by several diagnose agents (agent manager) in buyer sub-system.

- Type of tasks: Three diagnostic tasks, DTi_1 (Offer1), DTi_2 (Offer2), DTi_3 (Offer3)
- Three manager agents in three multi-agent framework
- Potential buyer for the offer: for DT_{i1} $A_c = \{A_{11}, A_{12}, A_{13},\}$, for DTi_2 $A_c = \{A_{21}, A_{22}, A_{23}\}$, and for DTi_3 $A_c = \{A_{31}, A_{32}, A_{33}\}$ all diagnoses agents. For each agent the first index represents the task and the second number of agent.
- Initial Message: IM = (TD_i, DIR_a, [DD,MM])
- f(DTi), function to evaluate the ability to do a task by the seller (determines if it will buy). As criteria for offer evaluation, they consider their skill and availability.
- Vector $\overrightarrow{O_c}$

Table 2: Offers DTi1

Offers DTi1	
O_{11}	$[SC, ST, TPo, PS]_{11}$
O_{12}	$[SC, ST, TPo, PS]_{12}$
O_{13}	$[SC, ST, TPo, PS]_{13}$

Table 3: Offer TDi2

Offers TDi2	
O_{21}	$[SC, ST, TPo, PS]_{21}$
O_{22}	$[SC, ST, TPo, PS]_{22}$
O_{23}	$[SC, ST, TPo, PS]_{23}$

Table 4: Offers DTi3

Offers TDi3	
O_{31}	$[SC, ST, TPo, PS]_{31}$
O_{32}	$[SC, ST, TPo, PS]_{32}$
O_{33}	$[SC, ST, TPo, PS]_{33}$

Table 2, 3 and 4 show the agents skills (the nomenclature SC, ST, TPo and PS are taken from Sub-tasks Type column in Table 1.

- $g(O_c)$, is the criterion of evaluation of offers by the manager agent. We assume as evaluation criterion the availability of time of each agent to perform the task.
- Messages for the success (M_s):

Table 5: Message for the successful of each task DTi_j

T_j	A_c	M_p
DTi_1	A_{13}	$M_s = \{A_{13}, \{accept – offer\}\}$
DTi_2	A_{21}	$M_s = \{A_{21}, \{accept – offer \}\}$
DTi_3	A_{32}	$M_s = \{A_{32}, \{accept – offer \}\}$

- The vector RP of each task DTi_j

Table 6: Parcials report of each task TDi_j

Task	Report
DTi_1	$rp_{3,1}$ $rp_{3,2}$
DTi_2	$rp_{1,1}$ $rp_{1,2}$ $rp_{1,3}$
DTi_3	$rp_{2,1}$ $rp_{2,2}$

Each report has two indexs, the first is the number of the agent and the second the number the reports.

- h(RP, k) is the function that evaluate performance of each contractor agent in each task for which it was accepted the offer, such that each vector RP coresponds to the task TDi_j in evaluation

In summary:

- The number of agents offering to each one of the tasks (according to the function f(DTi)

Where:

For DTi_1 = 3 agents (Table 2), DTi_2 = 3 agents (Table 3), DTi_3 = 3 agents (Table 4), they are diagnoses.

- Table 5 show that each manager evaluates the offers receives, leaving as winner: Agent 3 for DTi_1, agent 1 for DTi_2 and agent 2 for DTi_3, which are assumed to have immediate availability.
- For vectors of partial reports (Table 6), the agent 1 report more reliable in its accepted the offers (3 time)

Finally, they get the final report with the diagnosis of each one of the equipment requested in the task.

5. CONCLUSION

In the effort to implement and evaluate the proposed agent-based message-oriented middleware, the agent's framework analyzed and designed using standard agent-based modeling and simulation methods. There are several agent-based modeling methods available to model multi-agent systems such as ABM (agent-based modeling), ABS (agent-based systems or simulation), and IBM (individual-based modeling). However, Agent-Based Modeling and Simulation (ABMS) is the well know and widely used modeling method to analyze and develop a complex application that is comprised of intelligent and flexible interaction. ABMS is also considered as a modeling approach that can be exploited to represent some system attributes that are not clearly described as attributes or functionalities in a complex application.

The suggested multi-agent framework is elaborating by representing it mathematically using Coordination mechanisms. This process enables to evaluate and validate the proposed solution mathematically. This step is important as it can proof explicitly the validity and completeness of the suggested framework. This paper starts by introducing a formal method and mathematical model and highlighting the importance of mathematical modeling and why it is widely used in software engineering. Follow by the design and development of the proposed framework. Then, introducing the components of the Communications Net Protocol informal method and evaluation section. The definition of the operations in the proposed solution that modeled and then identified their rules that have to be validated. This formal definition allows us to validate the system mathematically. The importance of this paper relies on the value of mathematical validation of any software system before the prototype implementation. The validation process shows the stability and validity of the system in all the situations.

REFERENCES

1. Hassan, M.H., Jaafar, J. & Hassan, M.F (2014). Monitoring web services' quality of services: a literature review, Artificial Intelligent Review journal, issue 4, p. 835-850, Springer Netherlands.
2. WaiShiang, C., Nissom. S., YeeWai, S., & Sharbini, H. (2017). *Validating agent oriented methodology (AOM) for netlogo modelling and simulation*. AIP Conference Proceedings 1891, 020035.

3. Parhi, M., Pattanayak, B. K. & Patra, M. R. (2014). A Multi-agent-Based Framework for Cloud Service Description and Discovery Using Ontology, book Intelligent Computing, Communication and Devices: Proceedings of ICCD 2014, Volume 1 (pp.337-348).

4. Hamzah, M. H. I., Baharo,m F. & Mohd, H. (2017) A conceptual model for service-oriented architecture adoption maturity MODEL, the 6 th International Conference on Computing and Informatics, ICOCI 2017, Kuala Lumpur, 25-27April.

5 Menasce & Almeida (2000). Scaling for E-Business: Technologies, Models, Performance, and Capacity Planning.

6. M. Ibrahim, N., Hassan, M.F., & Balfagih, Z. (2011). Agent-based MOM framework for Interoperability cross-platform communications of SOA Systems, in SHUSER2011, IEEE: Kuala lumpur.

7. Ni, X., & (Jian) Sun, D *(2017)*. *Agent-Based Modelling and Simulation to Assess the Impact of Parking Reservation System*, in *Journal of Advanced Transportation, Article ID 2576094, 10 pages*.

8. Klein, M. (*2017)*. *Models in Models –On Agent-Based Modelling and Simulation in Energy Systems Analysis*, in *High Performance Computing Center (HLRS) On Computer Simulation Methods Stuttgart, 28th September*.

9. M.Ibrahim, N., & Hassan, M.F. (2014). Enhancement of Message Oriented Middleware for multiple type of SOA system, special edition of the Int. Journal of Science, Lahore (ISSN 1013-5316).

10. Simon J. E. Taylor (2014). *A domain-expert oriented methodology for Introducing agent-based modeling and simulation.* Part of the The OR Essentials series book series (ORESS).

11. Garro, A., & Russo, W. (2009). *Exploiting the easyABMS methodology in the logistics domain*.

12. M.Ibrahim, N., & Hassan, M.F (2016). *Cross-platform Communications Model for different SOA Applications*, in *ICCOINS2016*, IEEE: Kuala Lumpur, KLCC.

13. Belghiat, A., Kerkouche, E., Chaoui, A. & Beldjehem, M. (*2016). Mobile Agent-Based Software Systems Modeling Approaches:A Comparative Study*, in *Journal of Computing and Information Technology, Vol. 24, No. 2, 149–163, June*.

14. Campean. F. & Yildirim, U. (*2017). Enhanced Sequence Diagram for Function Modelling of Complex Systems*, in *Procedia CIRP Volume 60, Pages 273-278*.

15. Nguyen, X.T., & Kowalczyk, R. (2007). *WS2JADE: Integrating Web Service with Jade Agents*, in *Verlag Berlin Heidelberg 2007*, Springer: Berlin.

16. Laftah Al-Yaseen, W., Ali Othman, Z., & Ahmad Nazri, M. Z. (2016). *A Large Data Exchange Method for Multi-agent in Java Agent Development Framework*. Special Issue for "International Conference on Applied Science and Technology (ICAST), Malaysia.

17. Bennajeh, A. & Hachicha, H. (2015). Multi-agent Coordination for a Composite Web Service, Part of the Advances in Intelligent Systems and Computing book series (AISC, volume 372), Springer International Publishing Switzerland.

18. Terán, J., Aguilar, José L., & Cerrada, M. (2013). Mathematical Models of Coordination Mechanisms in Multi-Agent Systems. CLEI electronic journal, volume 16, number 2, paper 5, august 2013.