Illustration of a Multi-agent Systems Concept as User's Collaborative Software System

Ebrahim AlHashel and Masoud Mohammadian
Faculty of Information Science and Engineering
University of Canberra - Australia
Ebrahim.al.hashel@canberra.edu.au Masoud.Mohammadian@canberra.edu.au

Abstract

To introduce the multi-agent systems (MaS) as a new technology in software engineering and making it acceptable by the industries requires adequate clarification to show the advantages and the improvements of this technology over the existing ones. A substantial survey in the field of multi-agent systems research literature reveals that the majority of the system developers have not accurately identified the purpose behind applying the multi-agent system from the human computer interaction (HCI) perspective, as a user's cooperative system.

The aim of this paper is to illustrate how the MaS concept is centred on establishing a user collaborative system and explain why this concept is essential to the modern software engineering strategies. The research also identifies the multi-agent systems problem domains and its emphasis on the existing Web services user coordination problem. It then proposes high level semantic web architecture. The research concludes by a listing of the advantages of applying multi-agent systems, from the usability and quality perspectives.

Keywords: Software engineering, Multi-agent systems, Agent-oriented system, Web services, Human computer interaction.

1. Introduction

The multi-agent systems is characterised by its cooperation process which provides an ability for a collection of agents to operate in a cooperative mode. The cooperation process in the multi-agent systems can be explained as the potential to dynamically create a team of agents to achieve a common goal. The dynamic team formation process (DTFP) is the fundamental value of the multi-agent systems over all the other computer software approaches. Without it MaS could lose its potential advantage.

The reason for incorporating the dynamic team formation process within MaS is to equip the system with a mechanism to solve a **distributed coordination problem** or a problem that has networking characteristics. In fact, the multi-agent systems is the modern approach that emerges from distributed artificial intelligence (DAI) [1]. The Cooperative Distributed Problem-Solver (CDPS) technique [2] is modified by autonomous agents.

This paper investigates the multi-agent systems approach and finds that the cooperation strategies that the MaS possess can be utilises to develop user based cooperative systems. The vision in this regard is that a user has a goal (set of tasks) and that the goal needs several subsystems to be achieved; at the same time, the user does not want to be involved in the coordination of those subsystems. In this context the relationship between the user's goal and those subsystems coincides with a **distributed coordination problem that is well-matched with the concept of the multi-agent systems approach**.

The scenario of the user goal that requires several subsystems to be achieved can be examined in the current relationship between the user and the Web services (Ws). When the user needs to achieve his goal through the web services, he must tour inside the Internet and activate the applications using the computer mouse and menus which were invented in the early 1970s [3]. In this case, the user is the coordinator between the Web services. The user has to guide the system as to what to do. The system can not work alone. In other words, **the system driven by the user; it does not work on behalf of the user**.

Although the Web services are managed by one standard, that is W3C and run on the same global network environment (Internet), so far there is no operational integration between the web services. The Web services are not able to understand each other's functionality, capability, and services. This dilemma is situated in the centre of modern software engineering. The number of available Web services on the World Wide Web (WWW) is estimated to be 1.8 billion with a monthly growth of 3.9 million as per the survey conducted by NetCraft [4]. As the internet applications increase the user involvement along with the associated coordination between the Web services also increases.

The multi-agent systems' problem solving strategy is an efficient approach to apply in solving the user and the Web services coordination problem [5]. Envisage that each Web service is extended by an agent, and on the other hand, every user's goal breaks into tasks and possibly sub-tasks. Each task or sub-task is pointed to the related Web services' agent. The number of the Web services' agents forming a team where each member of the team is performing specific services (job). Then the services are reassembled in a form that achieves the desired goal and then sent back to the user. As the user keeps changing his goal (set of tasks) the system will be able to recreate a different set of agents' capabilities according to the new goal's requirements (team formation process). In this regard, the user will interact with the system based on his goal; furthermore, the user will be relief from the problem systems' coordination. From this perspective, the multi-agent systems' team formation process is ideal techniques to develop a user cooperative software system, a system that can change its structure dynamically for the purpose of perform the user goal.

Unfortunately there is an ambiguity in understanding the concept of multi-agent systems as a collaborative system working on the user's behalf [6]. In fact the MaS was invented to serve three types of problem domains; distribution, coordination, and human concepts simulation. The majority of the developments have not been able to define the role of a multi-agent system as an efficient software technique to build a distributed coordination user-based collaborative system, further more, the application domains is mismatch with the MaS approach.

The aim of this paper is to illustrate how the MaS concept is centred on establishing a user collaborative system that the HCI encouraged and explain why the MaS concept is essential to modern software engineering strategies that are necessary to solve distributed, coordination problems type, such as the Web services user's goal coordination problem. The research lays emphasis on the existing Web services disintegration dilemma, then reveals that the MaS concept is a very effective technique to implement in the case of this modern software critical problem, subsequently, provide a high level solution for building the semantic web.

The rest of this paper is structured as follows: Section 2 explains the current software problem and identifies it in the absence of the functional integration between the Web services; Section 3 clarifies the multi-agent systems' main feature and describes why we need the agent team formation process; Section 4 illustrates the MaS problem domains that matched and coincided with the approach; Section 5 introduces the advantages of applying the multi-agent systems approach in the software engineering, from usability and quality perspectives. The article concludes with section 6 which, summarises the paper and recommends future work.

2. The Problem of the Modern Software Engineering

The Internet is the universal network that is connecting the Web services using standard technologies such as SOAP [7], WSDL [8] and UDDI [9], thus becoming a global software environment. Although the Web services run under one standard environment, they are still not operationally integrated. The problem is located in the Web services architectures that have been designed without a semantic to their services, and further more, without a mechanism to specify their functionality.

The purpose for designing a common semantic for all Web services is to integrate these services and to make them available to each other when needed. In other words, the purpose is to create a common understanding between the Ws, who should know each other's services (functionality). The services' integrity will enable the Ws to communicate and coordinate the use of each other's services in order to achieve a particular goal. The Web services' integrity will allow the creation of a cooperative user-based system, a system that can achieve user goals with minimum user intervention.

The absence of services integration is creating a rigid software environment, which is reflected on the system's usability. The Web services applications do not know about each other's services, capabilities or functionalities; they are stand-alone services; a hyper link line waiting to be activated by a user. The applications do not have the capability to work together by themselves; they need the user as coordinator. The user is the task owner; s/he uses the PC mouse and menus, which were invented in the 1970s, to drive the system in order to achieve her/his task. The user has to guide the system. The system can not work alone; in other words, the system is driven by the user; it does not work in behalf of the user.

A survey conducted by NetCraft [10] estimates that 1.8 billions Web services are available on the WWW, with a monthly growth of 3.9 million services. As the number of Web services increase, users will be spending more time and effort touring inside the Internet in order to achieve their tasks.

There is substantial research aiming to resolve dilemma of disintegrated Web services. The problem is defined by Web services' communities as Web services composition (WSC). The community suggested several solutions. Sheila McIlraith [10] proposes a markup language (Markup of Web services in the DAML family) to enable a wide variety of agent technologies for automated Web services. Biplay Srivastava et al. [11], suggest the use of RDF/DAML-S along with Golog/Planing to provide Web services composition. Vikas [12], proposes a stage composition approach. Many other suggestions and proposals have been developed, for further illustration refer to Schahram & Wolfgang [13]. However, each of those approaches has its strength and weakness, in addition, each solution has a suitable technique to address a particular problem domain [14].

Developing a semantic web services mechanism to integrate the Web services is very complex task, and has to take into account the following points:

- The web services discovery process should be able to perform effectively and efficiently and mainly dynamically (at runtime).
- A sharing resources' mechanism that can works across dynamic, heterogeneous, and multi-institutional organization should be provided.
- The Web services are already established and have become the largest distributed heterogeneous applications built independently of each other at different locations by different people.

We have studied and examined the Web services problem then concludes that the existing software engineering challenges centre around how to build a cooperative system, a system that works for the user, or works on the user's behalf, isolating the user from coordination of the Web services. This objective can not be achieved without creating a common understanding between the existing Web services and make them integrated with each other's based on their functionalities and services.

We have also deeply studied both the multi-agent system concepts, and the cooperative system specification. The study found that building a collaborative system needs to follow a cooperative approach that can make the system cooperative at both internal and external levels. The internal level is within one system components and the external level between collections of systems of one environment. Both of these approaches and specifications are situated at the heart of the multi-agent systems concept.

3. The Multi-agent Systems Concept

The multi-agent systems is a society of autonomous, intelligent agents that can interact and coordinate with each other to reach goals that are difficult to achieve by an individual agent or monolithic system. Sycara [15], describes MaS as an agent system with the following characteristics:(1) each agent has incomplete information or capabilities for solving the problem and, thus, has a limited viewpoint; (2) there is no global system control; (3) data are decentralised; and (4) computation is asynchronous.

The reason behind incorporating the dynamic team formation process within MaS is to equip the system with a mechanism to solve a distributed problem or a problem that has networking characteristics. In fact the multi-agent systems is a modern approach that has emerged from distributed artificial intelligence (DAI). The Cooperative Distributed Problem-Solver (CDPS) techniques used in DAI has been replaced by an autonomous agent and a dynamic team formation process to create an automated resilience system that can reconfigure itself according to changes in the environment, and in addition, to cope with changes of the user's goals.

Given that agents are autonomous in their behaviour, the relationship between agents or agent-to-agent interaction also becomes autonomous. This creates a social relationship between agents. Social behaviour recognizes that agents interact in a request form (accept or reject) to take on a problem solving role. In other words, agents achieve a common goal through a cooperative method [16]. Solving a common problem using cooperative techniques is the key element of the multi-agent system approach. In order to establish the cooperation process, a dynamic team formation process is essential. Without a dynamic team formation process, the multi-agent system will lose its fundamental value (cooperation), which will result in poorly designed systems [17].

The cooperative techniques in the multi-agent systems can be seen as the potential to dynamically form a team of agents for achieving a common goal. The potential of dynamically forming a team of agents is an internal cooperation structure that sustains the idea of designing a collaborative system. The team formation process has been designed by adopting different approaches, including agent motivation, execution plan, organization structure, built-in goal [18], [19], [20], [21].

But why do we need the dynamic agent team formation process for achieving a common goal? There are two primary reasons for the need of team formation process in MaS. First, because MaS was invented to solve problems that are loosely couples distributed in their nature, run in a dynamic environment. For such characteristics the networking solution strategy is the most appropriate approach [10]. The user goal and Web services coordination problem, mentioned the previous section, is an ideal problem domain in need of a team formation process.

Secondly, because we want to implement a technique that has the potential to map the user goal's achievement plan with the problem solving team (agent team). The operation is in dynamic mode: the user will keep asking the system to perform different goals and the system should be able to change the agent team according to the goal achievements plan. For example, if a user, who wants to book a holiday, has three goals G1, G2 and G3, the system will need flight booking agent and hotel booking agent to achieve G1; conference agent and calendar agent to achieve G2, and may need a taxi booking agent and flight booking agent to achieve G3. The cooperative system is supposed to cope with the user request. As the user changes her/his goal, the system should search to find the proper agent team (or form a new team) that can achieve the new goal. This is the main purpose of the agent cooperation process which is unfortunately not clear to many multi-agent system developers.

The multi-agent system approach is a typical cooperative system designed to work on the user's behalf, it is a new concept that aims to isolate the user from the system coordination, mainly when the problem domain is situated in a heterogeneous, open, and distributed environment.

4. The Multi-agent Systems Problem Domains

In the last decade or so, hundreds and thousand of research papers have been published in the field of agent and multi-agent systems. The majority of research papers which rely on the application development misinterpret the problem domains compatible with the MaS approach [22], [6], [23]. Unfortunately, the developers have generally designed applications supported by cooperative user-interface, or object-based applications under the name of agent-oriented or multi-agent systems. According to this study multi-agent system is appropriate in three problem domains:

1. To model and simulate human concepts
2. To develop a software solution for coordination strategies
3. To Solve a problem that has a distributed nature using Cooperative Distributed Problem-Solving (CDPS) techniques.

In all three problem domains, MaS operates in a cooperative style directly or indirectly trying to achieve the task on behalf of the user. However, this research focuses on the coordination strategies and states that the Web services coordination problem perfectly coincides with the MaS approach, base on the following concepts:

1. Each Web service is extended by an agent in the form of a one to one relationship. The idea is to avoid restructuring the Web services, by extending the Web service with an agent that encompasses information containing services and attributes that have the ability to create a semantic web.
2. The Multi-agent Systems exist between the user and the Web services (see figure 1) with the ability to:
    a. accept the user's goal then analyse it into sub-goals (tasks)
    b. create a goal's plan (set of tasks)
    c. form a team of agents to support the planned goals
    d. compose the Web services base on the Ws semantic agent
    e. execute the plan.
3. Update the repository for the future experience use.
4. When the user interacts with the system again with a new goal, the system will replay the above steps with a new appropriate team of agents.

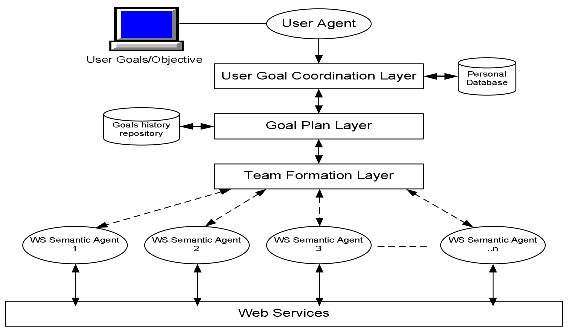

Figure 1. High level multi-agent systems for Web services' problem domain.

The operation of team formation have complex task and most of the difficulties are located in the interface between the current Web services and the Ws semantic agent proposed by this research. Further study of the Ws technologies (WSDL, SOAP, RDF, URL, UDDI, and URL) been started in parallel with the development tools. The next stage in the plan of this project is to deal with the implementation aspects relating to the future work.

5. The Advantages of the Multi-agent Systems

Human computer interaction (HCI) and from the usability perspective always pushed forward on the frontiers of software engineering to develop collaborative computer systems that have the potential to be user problem solving partners. Defining the collaborative system as high quality user interface system using sophisticated media, for example, Microsoft agents, voice recognition, or animations will improve the system-user interaction only. In fact, designing an efficient collaborative user interface system will be achievable if the system has been developed base on collaboration concepts. The collaboration concepts must be used as a foundation to develop the system on; the system should be collaborative on two levels, first is the internal structure of the system, second, is the external system behaviour's. The internal level describes the system internal structure (team formation process), and the external level describes the cooperation between the systems of one environment (operational integrity). These specifications will guide to develop a native collaborative system that has integration between its internal functionality as cooperative based and the user interface. These specifications are well-matched with the multi-agent system concepts which designed base on internal and external collaborative techniques. However, there are significant advantages out of applying MaS approach over the software from the usability perspective and the perspective of quality software engineering, worth to be mention.

5.1. Usability Perspectives

As it has been stated earlier the MaS use a built-in cooperative concept (team formation). This concept is presented in the agent cooperation process as a problem solving strategy. Applying multi-agent system collaborative concept in software engineering will improve the system usability as following:

a. The user will be isolated from the problem solving coordination.
b. The system will process and solve the problem alone.
c. The user needs less skill and training to use the systems.
d. The systems allow increased usage and form a synergy between the users and the systems.
e. The user will be more productive being released to carry out more important tasks, for example, the doctor will treat more patients.

5.2. Quality Perspectives

f. Maximise software reliability by reducing the dependence between the programs at the system level.
g. Produce robust software by illuminating the hardwired linkage, by converting the linkage from an invocation to a request type of call.
h. Reduce the development costs. The system can be developed at any time in any place by any body.
i. Will be able to solve a new type of problem using software, for example, biological relationships in the healthcare domains.
j. Improve the software functionality in reacting to changes which occur in the operational domain and generate update results.
k. Integrate with the available software services on the internet, and utilise those services effectively to benefit the user and achieve the system goals.
l. Improve the system design by implementing highly cohesive and low coupling techniques.

6. Conclusion

The multi-agent system is a modern software approach characterised by agent cooperation using dynamic team formation techniques for problem solving strategies. The multi-agent system without a team formation process is discrete and loses its fundamental concept.

Unfortunately there is ambiguity in understanding the concept of multi-agent systems as effective in building distributed, coordination applications that form a collaborative system working on the user's behalf.

The multi-agent systems approach is efficient in tackling three major types of problem domains; coordination between various independent entities, problems of a distributed nature, and the simulation human concepts.

The dynamic team formation process techniques must be able to achieve the user needs by accessing the vast open, distributed, heterogeneous environment which is identical to Web services applications. This research found that the multi-agent system techniques and concepts are efficient, and satisfy Web services problem of user goals coordination. Significant advantages also arise from applying MaS to software engineering.

The next step in this research is to investigate and studied Web services technologies to establish the best practice in extending the existing Web services for the development of semantic web agents. The result will be documented and forwarded at the next research opportunity.

Reference

[1] E. b. G. Weiss, Multiagent System: A Modern Approach to Distributed Artificial Intelligence, page 83. Cambridge, Massaachusetts London, England: The MIT Press, 1999.

[2] E. H.Durfee, V. R. Lesser, and D. D. Corkill, "Trends in Cooperative Distributed Problem Solving", IEEE Transactions on Knowledge and Data Engineering, vol. 1, pp. 63-83, 1989.

[3] B. J. Grosz, "Beyond Mice and Menus", in American Philosophical Society, vol. 149, 2005, pp. 529 - 543.
[4] NetCraft, "June 2008 Web Server Survey", in http://news.netcraft.com/archives/2008/06/22/june_2008_web_server_survey.html.

[5] M. Luck, P. McBurney, and C. Preist, Agent Technology: Enabling Next Generation Computing, vol. 1.0: AgentLink, 2003.

[6] L. N. Foner, "What's An Agent, Anyway", presented at First International Conference on Autonomous Agents, MA, USA, 1997.

[7] World Wide Web Consortium (W3C), "Simple Object Access Protocol (SOAP)", in http://www.w3.org/2000/xp/Group/.

[8] World Wide Web Consortium (W3C), "Web Services Description Language (WSDL) ", in http://www.w3.org/2002/ws/desc/.

[9] OASIS (Organization of the Advancement of Structured Information Standards), "Universal Description, Discovery and Integration of Web Services(UDDI)", in http://uddi.xml.org/.

[10] Sheila A. McIlraith, Tran Cao Son, and Honglei Zeng, " vol. 16, no. 2, pp. 46-53, "Semantic Web Services", IEEE Intelligent Systems, vol. 16, pp. 46 - 53, Mar/Apr, 2001.

[11] Biplay Srivastava and J. Koehler, "Web Service Composition - Current Solutions and Open Problems", http://www.zurich.ibm.com/pdf/ebizz/icaps-ws.pdf, Ed.: IBM Research Laboratory, 2003.

[12] A. Vikas, D. Koustuv, K. Neeran, K. Arun, K. Ashish, M. Sumit, and S. Biplav, "A service creation environment based on end to end composition of Web services", in Proceedings of the 14th international conference on World Wide Web. Chiba, Japan: ACM, 2005.

[13] D. Schahram and S. Wolfgang, "A survey on web services composition", Int. J. Web Grid Serv., vol. 1, pp. 1-30, 2005.

[14] A. Vikas, C. Girish, M. Sumit, and S. Biplav, "Understanding approaches for web service composition and execution", in Proceedings of the 1st Bangalore annual Compute conference. Bangalore, India: ACM, 2008.

[15] K. P. Sycara, "Multiagent Systems", American Association for Artificial Intelligence, 1998.

[16] L. M. Li Changhong, Kou Jisong, "Cooperation Structure of Multi-Agent and Algorithms", presented at IEEE International Conference on Artificial Intelligence Systems (ICAIS'02), China, 2002.

[17] J. E. Doran, S. Franklin, N. R. Jennings, and T. J. Norman, "On Cooperation in Multi-Agent Systems", presented at Foundations of Multi-Agent Systems, University of Warwick UK, 1996.

[18] K. P. Sycara and G. Sukthankar, "Literature Review of Teamwork Models", Robotics Institute, Carnegie Mellon University 31, Pittsburgh, Pennsylvania, Technical CMU-RI-TR-06-50, November 2006.

[19] J. G. Barbara and K. Sarit, "Collaborative plans for complex group action", Artif. Intell., vol. 86, pp. 269-357, 1996.

[20] D. Kinny, M. Ljungberg, A. Rao, G. Tidhar, E. Werner, and E. Sonenberg, "Planed Team Activity", presented at 4th. Workshop on Modelling Autonomous Agents in a Multi-Agent World MAAMAW '92, Rome, Italy, 1992.
[21] X. Fan and J. Yen, "Modeling and Simulating Human Teamwork Behaviors Using Intelligent Agents", in Physic of

Life Reviews, vol. 1 Issue 3. Pennsylvania, USA: Elsevier B.V. 173-201, 2004.

[22] M. Wooldridge and N. Jennings, "Pitfalls of Agent-Oriented Development", presented at 2nd. International Conference on Autonomous Agents, Minneapolis, Minnesota, United States, 1998.

[23] F. Stan and G. Art, "Is it an Agent, or Just a Program? A Taxonomy for Autonomous Agents", in Proceedings of the Workshop on Intelligent Agents III, Agent Theories, Architectures, and Languages: Springer-Verlag, 1997.