A Survey on Multi Agent Application Engineering Methodologies

Dr. Mallikarjun M. Math and Vinita Tapaskar

Abstract—Many normal methodologies for developing agent-based systems have been proposed. Nevertheless, their application continues to be confined because of their lack of maturity. All these methodologies have not been accepted within the enterprise on significant scale. On this paper we are evaluating Multi agent Software Engineering (MaSE) development methodology and Agent Unified Modeling Language (AUML). The software development phases followed by each methodology to improve multi agent software development are compared and analyzed.

Keywords--- AOSE, AUML, Agent, MaSE, Multi Agent.

I. INTRODUCTION

A Software agent is a piece of software that works as an agent for a consumer or yet another program, working autonomously and regularly in a distinct environment. It's inhibited through other procedures and agents, however can also be competent to gain knowledge from its experience while functioning in an given environment over a long interval of time [3].

An agent is an entity that acts on behalf of others in an self sufficient manner. Performs its actions with some degree of proactive and reactive exhibits. The agent has some key attributes such as studying, co-operation, and mobility. Primary elements or features of agents are shown in Figure 1.

3. Social potential: A couple of agents participate and communicate with each other. They collaborate with each other to complete the given task to them.
4. Reactivity: As the external environment changes, the agent will react to that. They respond according to the needs of changing environment.

There are different classifications of agents as depicted in Figure 2. Agents are classified based on the functions they perform and the environment they participate into [4][5].

![Types of Agents](image)

Figure 2: Types of Agents

Interface agent provides help to the user by participating with the user in his work area.

Interface agents can know about the user in four different ways and provide assistance to him based on need. Those four ways are:

1. By observing the user
2. Through receiving positive and negative feedback from user
3. By receiving clear cut instructions from the user
4. By receiving information from other agents
5. Collaborative agents: Be in contact with different other agents to participate in functionality in the distributed environment. Here a couple of agents collaborate to have interaction and they solve the common issue.

Information Agent: These agents come into existence due to heavy growth of information on the World Wide Web. The work of these agents is to collect, manage and manipulate the information spread across the web in different formats and structure. These agents are also called as Internet agents and their aim is to manage big data.

Mobile Agent: These agents are software piece of code which moves from one host to another host in a heterogeneous network environment to perform its task.

Reactive agent: These agents act and respond within the environment wherein they are embedded. These agents are embedded into the environment and respond at the state of the art.
Hybrid Agents: These are the agents which are constituted with the combination of two or more agents which are discussed above. These agents can be interface, mobile, information or collaborative agents. The main aim of creating hybrid agents is to get the benefits of all agents in a single unit [6][7].

II. MULTI AGENT SYSTEM

A multi-agent system (MAS) is a computerized system consisting of many interacting, intelligent agents within a given user environment. Multi-agent techniques can be utilized to solve issues which are complicated or difficult for a single agent or a monolithic process to solve.

The agents of the multi-agent system have important and different attributes and qualities which are discussed below:

1. Autonomy: These agents are independent or partially independent, self-aware, and have capability of taking decisions on behalf of the user.
2. Local views: Every agent is provided a local view of a system in which they are functioning. The agent need not get global view of the system as it is too complex and they cannot make use of the global view of system in their functioning.
3. Decentralization: In multiagent system each agent is independent and no master slave relationship. There is no controlling agent which monitors or directs other agents.[2]

A Multi-Agent System has set of agents which operate in common environment. That environment may be the real world or the virtual world.

As per artificial intelligence concept the multiagent system is a set of individual agents that interact with each other to share knowledge and communicate with each other to achieve their individual goals or a common goal set.

But from the software engineering point of view the multi-agent system is software made up of multiple independent agents which interact with each other in a particular application.

Agent Oriented Software Engineering (AOSE) is Methodology used for development of complex and distributed system with the use of agents which have intelligence of their own. There are various methodologies available for developing Agent Oriented Software for example TroPos, Prometheus and so on. To develop multi agent system the popular methodology used is MaSE,AUML[1][2].

III. MULTI AGENT SOFTWARE ENGINEERING

MaSE has framework and also include the entire lifecycle methodology for developing the multi-agent system. The framework includes the steps of analyzing, designing, and developing the heterogeneous multi-agent system as shown in figure 3.

Figure 3: MaSE Methodology Phases

MaSE has an additional abstraction level as compare to object-oriented software development methodology. The agents are created with the high level of abstraction as compare to object.

MaSE methodology is not suitable for development of dynamic system where agents are created, destroyed and transferred from one place to another place during execution, rather this methodology is useful in developing closed system where agent participates in system communication and includes all external interfaces.

Inter-agent conversations are assumed to be one-to-one, versus multicast. Methods designed with MaSE should not vary huge; the goal is 10 or less software agent classes.

A. MaSE Methodology

MaSE is stimulated from traditional object-oriented software engineering. MaSE is a goal based methodology.

This methodology specifies analysis and design step however don’t include features needed to design complex system.

The testing and deployment procedure are also not specified [9][10].

a) Analysis

In this stage the system goals get modified to roles. At the beginning phase the initial system specification is captured and later that specification is analysed and converted into set of system requirements where the requirements are specified in the form of Goal and Goal Hierarchy Diagram is drawn. This representation of goals is used in design phase for designing agent’s functionality. Based on the importance of a goal the Goal Hierarchy Diagram is organized.

1. Roles are the building blocks used to define agent’s classes and seize system goals during the design phase.
2. A role is describing entities functions in an abstract way and specifies system goals that agent is assigned the responsibility of fulfilling
3. The final stage is transformation of goals to roles is one-to-one format: each and every goal maps to a role.
b) Design

Using the roles the agent classes can be identified. At this stage the designer creates an Agent Class Diagram where he identifies the agents required for implementation of functionality and the communication required between them is also shown in this diagram.

The rectangular box represents the agent class containing the name of the agent class and the role assigned to that agent. The lines with solid arrow used to show the communication between the agent classes.

It's viable to mix multiple roles into a single agent classification. It's good to combine two or more roles into one agent class but while doing so the coupling and cohesion should be kept in mind. While choosing the roles to combine the size of agent class and the frequency of communication are to be considered.

Multiple agents are assembled by establishing communication between them. A MaSE Communication defines a coordination protocol between two agents. A conversation between agent classes is shown through communication class diagrams one for initiator and one for responder. A communication class diagram is pair of finite state automation and describes the communication state of two agent classes that participate.

Later agent classes are assembled for building internal structure of agent classes. A designer may design internal structure of agents from scratch or by using on-the-shelf existing design components.

The last step of MaSE is to take the system design and create the actual agent classes. Then the deployment diagram is drawn to show where which agents will be located in the system architecture[1][2].

IV. AGENT UNIFIED MODELING LANGUAGE (AUML)

The Unified Modeling Language is broadly used within the software engineering field for object oriented software analysis and design. The extension of that UML to accommodate Agent notion in comparisons with classes and object is proposed in the form of AUML.

A. AUML Methodology

AUML is an extension of UML which adds specialized agent class, a new idea of role and agent interaction diagram.

The software development stages proposed in the UML are Requirement Gathering, System Analysis, System Design and Implementation.

At each stage there is different type of diagram to be created. The diagram at one stage acts like an input for the next stage. These stages are shown in the Figure 4.

This methodology does not include the phases of testing and deployment. This method also does not provide any strong concept of developing multi agent complex applications[9][10].

Figure 4: AUML Methodology Phases

AUML introduces Agent interaction Protocol (AIP) diagrams. AIPs are design patterns introduced as agent classes and describe the solution to the problem which are frequently occurring the multi-agent system and these are reusable solution. These design patterns in actual fact provide communication mechanism between the agents and message passing between them. Agent Interaction Diagram is the main extension which is defined in AUML and does not exist in the UML.

In AUML an agent role means a set of agents satisfying the distinguished properties, interfaces, service description or having a distinguished behavior. Class diagrams in AUML seem identical with the ones in UML, with the change that they describe agent roles or agent classes. Class diagrams symbolize the knowledge of the agent and its relationship with other agents.

An individual with object-oriented knowledge can readily develop the software with agent. AUML is just not used for a specific language or architecture. AUML method of developing software can be used to develop computation object oriented software or knowledge based software.

V. CONCLUSION AND FUTURE ENHANCEMENT

MaSE methodology is sufficient for creating new software, reengineering and designing techniques. It covers the entire lifecycle besides for testing. The output at every stage is well defined. Concerning useful implementation MaSE is just not tied with any architecture or programming language, so it's general-purpose methodology for designing multi-agent system. AUML is extension of UML so features of UML are widely used in AUML. But AUML is not widely recognized as language in software engineering terms.

In order to develop multi-agent software which are complex there are certain key issues which are not included in these methodologies forming future research scope.

1. There is a need to identify essential concepts in multi-agent software engineering that can be used in the analysis and design of emerging complex systems. This concept should provide supportive methodologies needed to analyze, specify, design and develop multi-agent systems and applications.
2. There is even the need for the verification techniques that can be used to verify and validate the correctness of designed systems.
3. There is a need to identify efficient development platforms and testing techniques that can be used in...
implementing, testing and deploying multi-agent systems.

REFERENCES


