Understanding Multi-Agent Capability in Solving Complex Problem like Traffic Management

P.Felcy Judith
Associate professor, T John College, Bangalore

***Abstract*- For resolving traffic management problems it requires a coordinative control solutions where controls like traffic signals, vehicles progressing direction, roads capacity and many other parameters needs to be carefully analyzed and decisions to be made individually and collectively. Multi-agents systems have the potential and framework to define several agents to monitor various parameters across various locations and help co-ordinate agents to resolve individual decision for every single problem and collectively decisions could be made from the available individual decisions. This paper focuses on the various capabilities available in Multi-agents framework to resolve complex issues like traffic management system.**

Index Terms- Multi-agent System, Road Traffic Management

I. INTRODUCTION

The Road traffic management actions get influenced by various parameters depending on the local situations. The interrelations between the traffic controls could directly or indirectly affect the traffic situations in the neighboring areas. Solving one could influence the neighboring situation positively or negatively. Different resources related to the traffic management could be managed by different bodies or organizations. This initiates the need for co-operation between different organizations. In order to achieve the goal various organization, resource parameters needs to be individually operated and collective decisions needs to be made. Multi-agent system is known for its team work and collective decision making. Multi-agent system evolved from distributed artificial intelligence. This paper discuss on various capability of Multi-agent system in resolving complex problems.

II. MULTI AGENT SYSTEM

Multi Agent Systems (MAS) is based on distributed artificial intelligence computing, where the aim is to split a complex problem into several subtasks and distribute the management of these tasks to individual software entities [2]. This allows system intelligence to be distributed across the system components rather than being concentrated on a single point. “An agent is a computer system, situated in some environment that is capable of flexible autonomous action in order to meet its design objectives [1]” this is the commonly accepted definition for agents. An agent controls the environment through sensors and actuators and possess well defined boundary. Agents are flexible by exhibiting the following characteristics [4].

Reactive – Agents were able to perceive their environment and respond to the changes occur in their environment in order to satisfy their design behavior

Pro-Active – Agents exhibit goal-oriented behavior by taking initiative in order to satisfy their design behavior

Being Social – Agents communicate with other agents in order to satisfy their design behavior

Few multi-agent systems are intentional systems implementing practical reasoning, by process of deciding step by step [8][9]. The intentional model of agency originates from Michael Bratman’s [10] theory of human rational choice and action. He posts a more complex interplay of informational and motivational aspects which constitutes together a belief-desire-intention (BDI) model of rational agency.

Agent beliefs are the information it possess about the environment and other agents. The agent desires is the state the agent chooses to take. The desire could also be called as goal. Intentions in the other hand could not be converted into beliefs and desires [10]; they are subset of goals which the agent focuses at a point of time. This way agent creates a long term decision process called deliberation. In the process of deliberation the agents decide what state of affairs they want to achieve based on the interaction of their beliefs, goals and intentions. In the case of planning agents intention realize complex plan. This phase introduces agent's commitment, directly leading to action.

III AGENT ORIENTED METHODOLOGIES

Agent Oriented Methodology role is to assist in all the phases of the life cycle of the agent-based application, including its management and that extends some known existing development solutions [48]. Extension should not be trivial and alters the existing solution with a different method and finds itself coherent with the earlier solution. The extensions are also called as recycled methodologies [7].

Generally in various agent oriented methodologies, the methodology tries to build the notation and vocabulary to address some specific development phases. In most other methodologies, methodology embeds the notation and vocabulary in a tool that permits to move from analysis and design to the implementation. Recent methodologies provides the mixture of both, it supports notation and vocabulary to support some specific development phases and also provides tool support to help translate the design into implementation.

IV AGENT BASED MODELING TOOLS

It is a decentralized and individual centric design of a model. The modeler finds out the active agent (actor)and define their behavior (states) and then put them in an environment by making a connection between them . The task of agent base modeling is to simulate a project. Then the global modeling behavior works with the individual behavior. There are many tools and software used for this modeling. Agent base modeling platform has a framework which is a set of concepts for designing and describing and library for software implementation. The first one used for this was Swarm with object oriented modeling. MASON was developed as a new Java platform for modeling. Meta Agent based modeling is another platform which gives meta-level visual design studio for agent based modeling.

AgentSheets is the most proprietary agent based modeling tool which uses spreadsheet approach. Each cell is considered as an agent it was developed for non-programmers and it is very simple. It uses visual and graphical interfaces, dragging and dropping which makes user friendly. Each agent are created in a gallery and associated with their behavior with a set of rules. This is a quicker tool for simulations.

AnyLogic is a package of agent based modeling which has a wide range of functionality. Models can dynamically read and write data to database at simulation time. It can create model in Microsoft operating system but it can run java system once it is complied.

Ascape this framework is used for developing and analyzing agent based models. It has some idea of Swarm, which makes the end user to create complex simulations by providing end user tools. Ascape is a java platform framework with object oriented idea.

Flexible Agent Modeling Environment (FLAME) was developed to simulate agent and non-agent models. It has a framework where modeler can create a model and there is a software tools that are compatible with other agent. New Models can be created adhering to the specification and can easily incorporate to the existing. It defines an agent as X-machines with finite state machines as addition of memory for communication. It allows modeling for parallel architecture.

Java Agent Development Framework (JADE) is a framework fully works on Java. With a middleware it targets users and provides agent services it complies with FIPA specifications and simplifies the implementation of multi-agent systems. JADE was developed by Telecom Italia. It is mainly used in Mobile applications, Internet applications, corporate applications and machine-to- machine applications. It acts as a tool to support debugging phase and designed for scalability. The goal of JADE is to make development more simplified by providing a comprehensive set of system services. It has some special features to achieve this goal: FIPA a compliant Agent Platform, a distributed agent platform which split into several hosts. JADE schedules threads more efficiently than java virtual machine .It hides FIPA from the programmer. It is not needed to implement the agent platform Message transport, Parsing and agent services. It is automatically inbuilt framework. Interaction protocol is extended by handle methods.

V AGENT COMMUNICATION

Knowledge Query and Manipulation Language (KQML) is a high level, message oriented communication language which is independent to the content syntax ontology of the content and transport mechanism. KQML is based on speech-act theory and performatives. It was developed by APRA supported Knowledge Sharing Effort (KSE). Its aim is to developed techniques and methodology for large scale knowledge base system which can be sharable and reuse able. KQML is both a message format and a message handling protocol at run time knowledge sharing among agents. It can be used in application program to communicate with other agents by knowledge sharing. A software agent transmits messages composed in its own representation language, wrapped in a KQML message. Ithas a special class of agents called communication facilitators which has a registry of service names, forwarding message to named services, routing messages based on content. KQML has set off per formatives which are the core language and it determines the kind of interaction one can have with KQML speaking agent. The semantics of KQML are provided in terms of preconditions, post conditions and completion conditions for each per formative.

Each KQML message represents a single Speech act (e.g., ask, tell, achieve,) with an associated semantics and protocol. As a communication language KQML works on both distributed system and distributed AI which provides high level of abstraction in both of them. In distributed software system KQML is an abstraction of process as information agents known as knowledge base system which is similar to database management system hypertext system and service oriented system. A software agent transmits content of message wrapped with KQML. The content of message can be of any representation languages or written even in ASCII strings or in any one of the binary notations.

The Foundation for Intelligent Physical Agents (FIPA) is an IEEE computer society standard organization that promotes agent-based technology and the interoperability of its standards with other technologies. The most widely used standards are the Agent Management System and Agent Communication Language (ACL) specifications. FIPA ACL is open agent architecture. Agent Communication language (ACL) is a language which is used as a means of transforming information and knowledge between agents .which is similar to other type of exchanges between applications such as RPC to CORBA. The main objective of ACL is to model a framework that allows heterogeneous communication. But ACL is distinguished from the others by two reasons

1. ACL uses propositions rules and actions instead of objects without semantics associated with them.
2. ACL messages are described in declarative language, rather than a procedure or method.

It covers the entire spectrum of what the application what to exchange. ACL is in logic layer above the transport layer which deals with intentional and social level. Its structure is composed of different sub languages that specify the message content, its interpretation parameter between the sender and the receiver.

KQML and FIPA-ACL both are agent based communication languages which are designed to work with content languages and any ontology specification approach. But FIPA-ACL and KQML have philosophical differences in terms of what is done in the wrapper language and what is delegated by the wrapper language to the content language.

VI. CONCLUSION

This paper discusses the various capabilities of Multi Agent System (MAS) in resolving complex problems like Road Traffic Management. It discusses the nature and properties of Multi Agent Systems. It briefs the Agent oriented methodologies without getting into the details of various methodologies. It briefly describes some of the modelling tools available. Later it discusses on communication between agents in order to co-ordinate and collective decisions.

REFERENCES

[1] Ferber J. 'Multi-agent systems: An introduction to distributed artificial intelligence', 1999

Published by: Addison-Wesley, ISBN: 0201360489

[2] Trichakis, Pavlos (2009) Multi Agent Systems for the Active Management of Electrical Distribution Networks, Durham theses, Durham University.

[3] Teamwork in Multi-Agent Systems: A Formal Approach Barbara Dunin-K ¸eplicz and RinekeVerbrugge, 2010 John Wiley & Sons, Ltd

[4] Multiagent Systems, A Modern Approach to Distributed Modern Approach to Artificial Intelligence, 1999 Massachusetts Institute of Technology

[5] Wooldridge, M., Jennings, N. R. and Kinny, D. (2000) the Gaia Methodology for Agent-Oriented Analysis and Design. Journal of Autonomous Agents and Multi-Agent Systems, 3, (3), 285-312.

[6] [IEE90] IEEE standard glossary of software engineering terminology. IEEE Std 610.12-1990, 1990.

[7] [M97]H. Jurgen Muller. Towards agent systems engineering. Data Knowl. Eng., 23:217–245, September 1997.

[8] [Anscombe,1957] Anscombe, G. (1957). Intention.Cornell University Press, Ithaca, NY, USA.

[9] [Velleman,2000] Velleman, D. (2000). The Possibility of Practical Reason. Oxford University Press, Oxford, UK.

[10] [Bratman, 1987] Bratman, M. (1987). Intention, Plans, and Practical Reason.Harvard University Press, ambridge, MA, USA.

[11] [Dennett 1987] Dennett, D. (1987). The Intentional Stance.MIT Press, Cambridge, MA, USA.

[12] [Cohen and Levesque, 1990] Cohen, P. and Levesque, H. (1990). Intention is choice with commitment. Artificial Intelligence, 42: 213–261.

[13] [Rao and Georgeff,1991] Rao, A. and Georgeff, M. (1991). Modeling rational agents within a BDI-architecture.InFikes, R. and Sandewall,E. (Eds), Proceedings of the Second Conference on Knowledge Representation and Reasoning, pages473–484.Morgan Kaufman, San Francisco, CA, USA.

[14] [Rao and Georgeff, 1995a] Rao, A. and Georgeff, M. (1995a). BDI agents: From theory to practice. In Lesser, V. (Ed.), Proceedings of the First International Conference on Multi-Agent Systems, pages 312–319, San Francisco, CA, USA.AAAI-Press, Menlo Park, CA, USA and MIT Press, Cambridge, MA, USA.

[15] [Jennings and Bussmann, 2003] Jennings, N. R. and Bussmann, S. (2003). Agent-based control systems: Why are they suited to engineering complex systems?