THE INTERNATIONAL JOURNAL OF SCIENCE & TECHNOLEDGE

A New Mobile Agent Technology with Multi-Agent Systems for Distributed Traffic Detection and Management Systems

B. Subramani
Department of Information Technology
Dr. N.G.P. Arts and Science College, Coimbatore, India
M. Madhumitha
Research Scholar, Department of Information Technology
Dr. N.G.P. Arts and Science College, Coimbatore, India

Abstract:
Computing using agent is an important technology for the development of large scale distributed systems to deal with the uncertainty in a dynamic environment. A number of agent-based traffic control and management systems have been proposed and the multi-agent systems have been proposed, to the best of our knowledge, the mobile agent technology is not focused this field. The traffic and transportation system is well suited for an agent-based approach because transportation systems are usually geographically distributed in dynamic changing environments. In this paper mobile agent based security system is proposed using Node-based Intelligent Network with IEEE FIPA Architecture, Proposed-Security via AMS Shadowing and Protecting the AMS and the DF with prevention of attacks. The results are calculated for traffic and it proves its efficiency.

Keywords: *Mobile agent, FIPA, AMS Shadowing, DF, ACL Message*

1. Introduction

A software agent is defined as a part of autonomous or semi-autonomous proactive and reactive or computer software. It has the capability of making independent decisions and focus on internal goals based on computer environment. Agent architecture gives a high-level view of the subsystems that make up the agent system, their interactions and the flow of control and/or information among the subsystems. It is mainly used to solve the problems in large scale distribution systems. The multi-agent technology grown in this area where inherent distribution allows for a natural decomposition of the system into multiple agents. This communicates with each other to achieve a desired global goal (Hernandez et al., 2002) [1]. To enhance the design and analysis of a problem, multi-agent technology is used. The following three conditions (Adler and Blue, 2002) [2]: (1) the problem domain is geographically distributed. (2) The sub-systems exist in a dynamic environment. (3) Sub-systems need to interact with each other more flexibly. The application of traffic and transportation systems is much suited for an agent-based approach because of its geographically distributed and dynamic changing nature (Jennings et al., 1998) [3]. The domain of traffic and transportation systems is well suited to an agent-based approach because of its geographically distributed nature and its alternating busy-idle operating characteristics [4], [5]. From the traffic and transportation management perspective, the most appealing characteristics of agents are autonomy, collaboration, and reactivity. Agents can operate without the direct intervention of humans or others. This feature helps to implement automated traffic control and management systems. Agents are collaborative. In a Multi Agent System (MAS), agents communicate with other agents in a system to achieve a global goal. Agents can also perceive their environment and respond in a timely fashion to environmental changes. Agent-based transportation systems allow distributed subsystems collaborating with each other to perform traffic control and management based on real-time traffic conditions.

A distributed vehicle monitoring test bed presented in [14] is an early example of the distributed problem-solving network. Recently, more and more agent-based traffic and transportation applications have been reported. Our literature survey shows that the techniques and methods resulting from the field of agent system and MAS have been applied to many aspects of traffic and transportation systems, including modeling and simulation, intelligent traffic control and management, dynamic routing and congestion management, driver-infrastructure collaboration, and decision support. Although more and more studies have been reported to apply agent approaches to traffic and transportation systems, only few researches address the system architecture and the platform issues of agent-based traffic control and management systems. Garcia-Serrano et al. [6], Tomas and Garcia [7], and Chen et al. [8], [9] are three MASs that are designed for roadway traffic detection and management and are compliant with the IEEE Foundation for Intelligent Physical Agents (FIPA) standards, which is one of the major international agent standards. In [7], several Traffic Agent City for Knowledge-based Recommendation (TRACK-R) agents are designed to provide traffic route recommendation for humans or other agents. Each TRACK-R agent is responsible for a geographical area. Tomas and Garcia [7] propose MAS to help traffic operators determine the best traffic strategies for dealing with nonurban roadway meteorological incidents. The agents in these two systems are implemented using the Java Agent Development Framework (JADE) agent

platform [10], [11]. Chen et al. [8], [9], [12], [13] developed a mobile agent system called Mobile-C and designed an agent-based real-time traffic detection and management system based on Mobile-C. Mobile-C is an IEEE FIPA standard compliant multi agent platform for supporting C/C++ mobile agents in networked intelligent Mechatronic and embedded systems. Mobile-C was originally developed to enhance the distributed computing and information fusion capability for a laser-based highway vehicle-detection system [14], [15]. Although it is a general purpose multi agent platform, Mobile-C is specifically designed for real-time and resource-constrained applications with interface to hardware.

This paper is organized as follows. Section 2 describes the Related Works of this paper. Section 3 describes the Proposed Methodology using Load Control mechanisms, IEEE FIPA Architecture, Security via AMS Shadowing, Protecting the AMS and the DF, Protecting the ACL messages and Preventing Attacks due to Malformed ACL messages. Section 4 argues the Experimental results of this paper which is implemented in JAVA platform using JADE tool. Finally some conclusions are given in Section 5, followed by References.

2. Related Works

A Traffic control and management systems can be characterized as highly distributed and dynamic changing systems. Agent technology is a promising approach to solve problems in this domain. A distributed vehicle monitoring test-bed presented in Durfee (1988)[14] is an early example of distributed problem-solving network. Recently, a number of agent-based applications related to traffic control and management in different modes of transportation, including road (Hernandez et al., 2002[1], railway (Bel and van Stokkum, 1999[15]), and air transportation (Glover and Lygeros, 2004 [16], have been reported. This section examines agent applications in road traffic control and management systems.

Adler and Blue (2002)[17] proposed an agent-based approach for vehicle route choice and capacity allocation. The proposed framework, Cooperative Traffic Management and Route Guidance System (CTMRGS), extends the ITS National Architecture Market Package ATIS6 by adding a middleware layer that is comprised of agents and agent negotiation protocols. Agents in the system represent travelers (Agent-IRANS), information service providers (Agent-ISP), and system operators (Agent-TMC), respectively.

A principled negotiation is used to guide interactions between Agent-IRANS and Agent-ISP to make route choice and capacity allocation satisfying the objectives of both drivers and system operators. Logi and Ritchie (2002) [18] investigated the inter-jurisdictional traffic congestion management on freeway and surface street (arterial) networks. Their system is composed of two interacting real-time decision support agents, a freeway agent and an arterial agent, for analysis of congestion and generation of suitable responses. The freeway agent supports incident management operations for a freeway sub-network, and the arterial agent supports operation for the adjacent arterial network. Both agents interact with a human operator at their local Traffic Operation Center (TOC) and generate suitable local control plans. The system provides a dialog facility through a distributed user interface to allow operators at different TOCs to agree on the selection of a global solution.

Okamoto and Ishida (2007) [19] proposed a framework in which each computer generates diverse agents that are shared with other computers on the LAN. This sharing contributes to the diversity of the agents. User-specific agents are generated for every user on each computer; each agent has a unique profile that is presented by a parameter of the detection method.

3. Proposed Methodology

All paragraphs must be indented. All paragraphs must be justified, i.e. both left-justified and right-justified.

3.1. Node-based Intelligent Network load control mechanisms

Because of the central role played by the Service Control Points (SCP) the overall goal of most Intelligent Network load control mechanisms has been to protect SCP processors from overload, thereby providing customers with high service availability and acceptable network response times, even during periods of high network loading. In addition to this goal load control mechanisms have been designed to be:

- Efficient – Keeping SCP utilization as high as possible at all times;
- Scalable – Suited to all networks, irrespective of their size and topology;
- Responsive – Reacting quickly to changes in the network or offered traffic levels;
- Fair – Distributing system capacity among both the network users and service providers in a manner deemed ‘fair’ by the network operator.
- Stable – Avoiding fluctuations or oscillations in resources utilization;
- Simple – In terms of ease of implementation.

While flexible and adaptable network-based load control mechanisms could certainly be implemented using standard software engineering techniques, argue that there are many advantages to be gained through the adoption of an agent-based approach:

- **Methodology:** The agent paradigm encourages a knowledger information centered approach to application development, thus it may provide a useful methodology for the development of control mechanisms that require manipulation of large amounts of data collected throughout a network.
- **Agent Communication Languages:** Much work in the agent research community has focused on development of advanced communications languages that, for example, allow agents to negotiate in advance the semantics of future communications. This degree of flexibility is not present in traditional communications protocols and would be of use in realizing mechanisms that adapt to dynamic network environments where, for example, traffic patterns change due to introduction or withdrawal of services.

- **Adaptability:** Agents may have adaptive behavior allowing them to learn about the normal state of the network and better judge their choice of future actions. A substantial body of work relating to learning behavior in the context of agent systems exists and could be leveraged to incorporate learning behavior into a load control mechanism.
- **Openness**: The agent approach is very open, in that individual agents may exchange data and apply it in different ways to achieve a common goal. This means that equipment manufacturers could develop load control agents tailored to their own equipment, but which could still effectively communicate with agents residing in other equipment types.
- **Scalability**: The agent approach allows for increased scalability to larger networks. For example an agent associated with a recently introduced piece of equipment can easily incorporate itself into the agent community and learn from the other agents the range of parameters that it should use for its load control algorithms.
- **Robustness**: agents typically communicate asynchronously with each other and thus are not dependent on prompt delivery of inter-agent messages. The ability to act even during interrupted communications e.g., due to overload or network failures is a desirable attribute of a load control mechanism.

3.2. IEEE FIPA Architecture

FIPA provides an agent management reference model depicted in Figure 1 as a framework for agent creation, registration, location, communication, and transport between agent platforms. FIPA has specified an agent communication language as well as the protocols and infrastructure for MAS [24].

The agent management reference model consists of a set of entities, namely the software agent, directory facilitator, agent management system, message transport system, the agent platform, and non-agent software. These entities are shown in Figure 1, and each represents a set of capabilities. To be compliant with the FIPA model, each developer is expected to design the entities such that their capabilities comply with the specifications for those entities. The entities of the IEEE FIPA model are shown in Figure 1 and briefly explained in the following paragraphs.

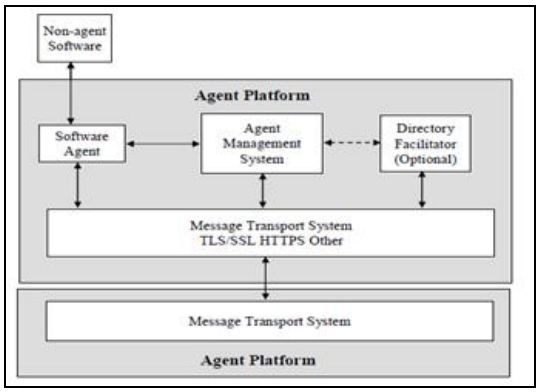

Figure 1: FIPA Agent Management Reference Model

A software agent is a computational entity (i.e., an abstraction of an object) [25] that are able to represent a state and change it by performing some actions on it and also to communicate with other agents through message passing. An agent is different from an object because of its ability to perform autonomous actions—that is, the agent is able to perform its tasks without reliance on human input. Agent Oriented Computing versus Object Oriented Computing Before we delve further into FIPA multi-agent management structure, it is appropriate to present some differences between agent- and object-oriented programming. Why can't we just apply the same approach used in object-oriented computing? While an agent is an abstraction of an object, the behavior of an agent and how one agent interacts with another agent is quite different from that of object to object interaction. In object oriented programming, when one object wishes to obtain a service from another object, it must send a message to the server object and tell the server object which method to use to perform the service. The interaction is usually synchronous and the communication link is torn down upon completion of the transaction. The process is different with agent-oriented programming where each intelligent agent is autonomous. To obtain service from a server agent, all it needs to do is to send a message in an agent communication language and simply tell the agent what it wants. It does not need to tell it how to provide the service. Furthermore, interactions between the agents are asynchronous thus permitting each agent to operate autonomously. The demand for asynchronous communications and autonomy of agents calls for the need to maintain a persistent connection and protect the connection from malicious agents. The structure and behavior of a multi-agent system are different from those of a multi-object system; consequently, the process of hardening the two systems is different. A Directory Facilitator (DF) is an optional requirement that provides an environment where agents can register their services, thus providing yellow pages services. The Agent Management System (AMS) is a mandatory requirement that controls agents' access and use of the agent platform. The AMS subsumes the functions of the DF when a DF is not implemented. The Message Transport Service (MTS) provides the agent communication protocol for exchange of messages between the agents. The Agent Platform (AP) is the environment for deploying the agents. The environment consists of the computer system, the operating system and other agent support software, the AMS, and software agents. All the agents do not need to reside on the same AP. FIPA does not have a standard for internal design of an AP, except that agents must be able to communicate within and between platforms in a FIPA compliant agent communication language (ACL). The standard permits developers to use non-agent software to support agent functions for new services such as new communication protocols, security protocols and algorithms.

3.2.1. Proposed-Security via AMS Shadowing

One approach for detecting malicious activities on the AMS agent is to implement an AMS mirror agent. It is feasible to implement the architecture similar to that of a typical Sebek [23] honey net deployment to monitor the activity of an intruder and covertly export the behavior of the AMS to a remote platform and possibly using a different operating system. The mirror agent can capture the state of the primary (AMS) agent and detect any changes in the state or behavior of the primary agent. Upon detection of a malicious activity, the mirror could take over the operations of the primary AMS, notify the administrator on the system status, or take any action as specified in its policy directives [24].

3.2.2. Protecting the AMS and the DF

T In order to protect the AMS and the DF against too many service requests, frequent and large service descriptions, and mal formed ACL messages we suggest the following steps:

- The use of policies on:
  - *Buffer sizes*
  - *Message acknowledgement,*
- The use of secure private channels for agent-to-agent interactions.

The limits on the buffer sizes will help eliminate the risk that a rogue agent will consume all the DF storage by writing frequent and large service descriptors. A policy that stops responses to frequent malformed ACL messages will help mitigate the resulting DoS attack. The use of the secure private communication channels will help eliminate the vulnerabilities due to impersonator rogue agents.

3.2.3. Protecting the ACL messages

As already described the use of secure private channels for agent to agent communications will help secure the ACL messages. In addition, it is feasible to add a few security parameters to the ACL message structure. This includes the following steps:

- *The invocation of security services within an ACL message.*
- *The use of unique agent identification.*

Both of these measures are enabled within the FIPA ACL message structure which employs the message envelope format. The envelope includes an agent service template and a message format. In this section we will illustrate how the service description can be implemented as a security service such as authentication, authorization, integrity, privacy, or non repudiation, etc. We will also illustrate how to craft a unique agent id for the agent-name-id in the message format. Let us illustrate bringing security into ACL messages using an example from the Victor MAS Message snippet taken from Victor MAS. It describes a request from the DSA to the network monitoring agent (NMA) to check the congestion for the link from Amsterdam (AMS) to Frankfurt. The message id (Msg-ID) in the ACL message is generated automatically and it consists of the name of the agent platform (i.e., host name), and the current date and time, plus a random number to provide a globally unique identification for the message. This ACL message does not include any security parameters.

3.2.4. Preventing Attacks due to Malformed ACL messages

Consider the incorporation of a determine-integrity action performing parameter, into the Performative service of the message. The integrity service is implemented in reference [22] based on hashing methods described in reference [21]. The approach demands that both the sender and receiver compute and compare the hash of all messages, and exchange encrypted hash values. If the hash values do not match, then the message has been tampered with and it should be rejected. The approach can be used to identify and reject spoofed messages issued by a rogue impersonating agent. The approach helps each agent to recognize the impersonators and use the action performing per formative to refuse service.

Alternatively, one can use the idea of a unique message identity whereby each agent appends a value (to the message identifier) generated from an algorithm with a seed, instead of the determine-integrity-PKI service. During the initial agent registration process, each agent would provide information on its skills and a unique identifier to aid message recognition. The information can be about a specific part of the message identifier, rather than the entire id. For example, the information could be on the random numbers that it plans to append to the ids in its messages to the AMS. The AMS would use the information as one of the steps to verify the source of the message. The AMS or the recipient agent can then use the Error Handling features of FIPA Communicative Acts to send a failure or not-understood as the Msg-Type parameter and a refuse or inform parameter for the action performing service Performative. The recipient may also use a policy to block the impersonator agent's buffers and stop accepting messages from it. Such policies can protect against DoS attacks launched via malformed messages, or messages that show rogue behavior.

4. Experimental Results

In this chapter, the proposed work is evaluated which is implemented in JAVA platform and JADE tool is used. The traffic image file freely available in internet is chosen for the evaluation. Initially agents are created and named as Agent Management System (AMS) in this work.

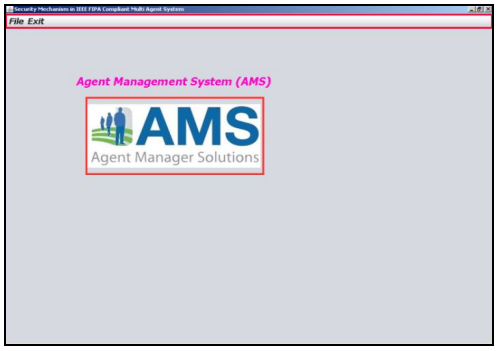

Figure 1: Agent Management System User Screen

The above figure 1 shows the User screen of mobile agent created for traffic image files.

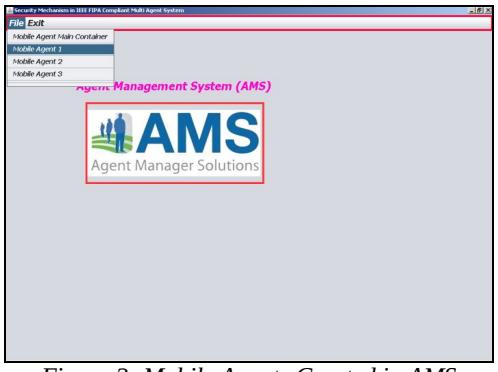

Figure 2: Mobile Agents Created in AMS

The above figure 2 shows the created of mobile agents. Here three mobile agents are created.

Figure 3: Mobile Agent Creation

This figure 3 shows the creation of mobile agents using host name and agents full path.

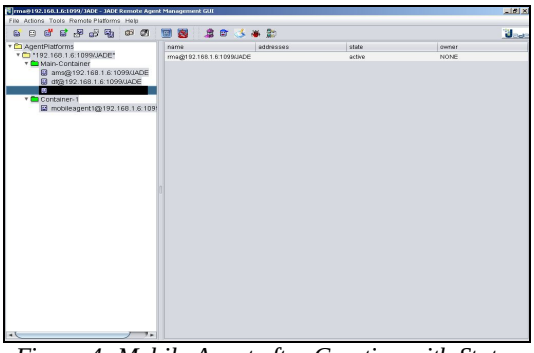

Figure 4: Mobile Agent after Creation with Status

The figure 4 shows the created sub agents with its status as active.

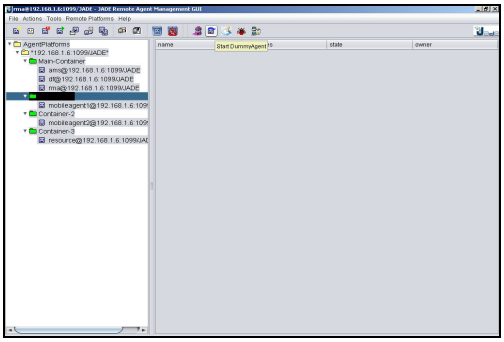

Figure 5: Creating Dummy Mobile Agent

The above figure 5 shows the dummy mobile agent for sending messages in the traffic. As the Agent will not communicate with another agent directly, they uses dummy agent.

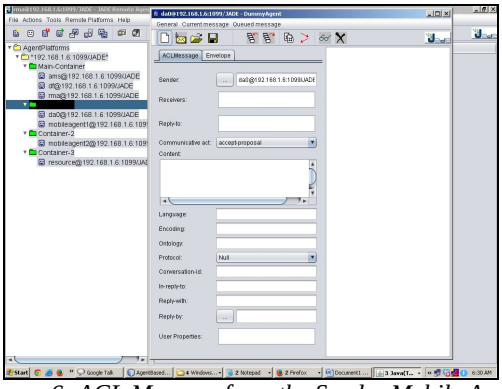

Figure 6: ACL Message from the Sender Mobile Agent

The above figure 6 shows the ACL message from the sender Mobile Agent Screen. This form is filled to send the message to another mobile agent in traffic.

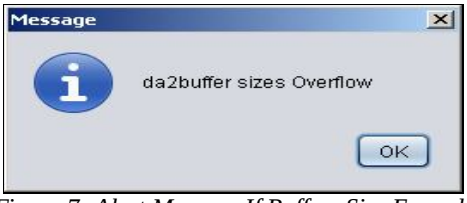

Figure 7: Alert Message If Buffers Size Exceeds

The above figure 7 shows the alert message screen if there is any overload occurs in the traffic while transmission of messages from the mobile agent.

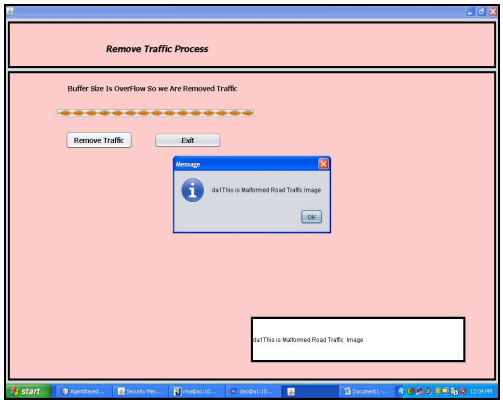

Figure 8: Alert Message of Malformed Attack in Road Traffic Image

The above figure 8 shows the alert message screen If there is a malformed attacked when transmission of message in the road traffic images.

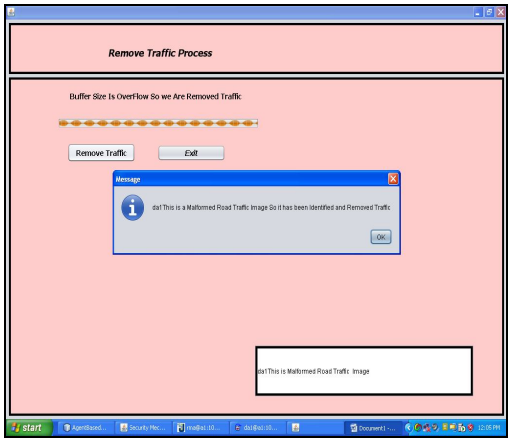

Figure 9: Malformed Road Traffic Image is Identified and Removed Traffic

This above figure 9 shows the Alert message Malformed Road Traffic Image is Identified and Removed Traffic.

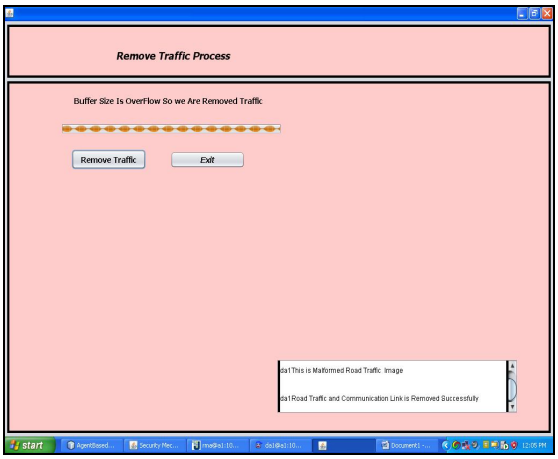

Figure 10: Road Traffic and Communication link Removed

This above figure 10 shows the process of Communication Link and the network traffic between the two mobile agents is removed.

Figure 11: Acknowledgement Message Received After Message Received By the Mobile Agent

The above figure 11 shows the Acknowledgement Message Received after Message Received by the Mobile Agent.

5. Conclusion

The mobile agent is a unique technique for making the system flexible and the ability of the system to deal with the uncertainty in a dynamic environment. The environment consists of the computer system, the operating system and other agent support software, the AMS, and software agents. In this work the AMS agent is secured by using AMS shadowing, AMS and DF are protected by using policies and secure private channels for agent-to-agent interactions. This technique reduces the attacks and prevents form it. The experimental results show how the agents are created and how the network is secured for traffic image files.

6. References

1. Hernandez, J.Z., Ossowski, S., Garcia-Serrano, A., 2002. “Multiagent architectures for intelligent traffic management systems”.Transportation Research Part C 10 (5–6), 473–506.
2. Adler, J.L., Blue, V.J., 2002. “A cooperative multi-agent transportation management and route guidance system”. Transportation Research Part C 10 (5–6), 433– 454.

3. Jennings, N.R., Sycara, K., Wooldridge, M.J., 1998. "A roadmap of agent research and development. International Journal of Autonomous Agents and Multi- Agent Systems" 1 (1), 7–38.
4. F. Y. Wang, "Agent-based control for networked traffic management systems," IEEE Intell. Syst., vol. 20, no. 5, pp. 92–96, Sep./Oct. 2005.
5. F. Y. Wang, "Toward a revolution in transportation operations: AI for complex systems," IEEE Intell. Syst., vol. 23, no. 6, pp. 8–13, Nov./Dec. 2008.
6. M. Garcia-Serrano, D. TeruelVioque, F. Carbone, and V. D. Mendez, "FIPA-compliant MAS development for road traffic management with a knowledge-based approach: The TRACK-R agents," in Proc. Challenges Open Agent Syst. Workshop, Melbourne, Australia, 2003.
7. V. R. Tomas and L. A. Garcia, "Agent-based management of non urban road meteorological incidents," in Proc. Multi-Agent Syst. Appl. IV, 2005, vol. 3690, pp. 213–222.
8. Chen, H. H. Cheng, and J. Palen, "Integrating mobile agent technology with multi-agent systems for distributed traffic detection and management systems," Transp. Res. Part C: Emerging Technol., vol. 17, no. 1, pp. 1–10, Feb. 2009.
9. Chen, H. H. Cheng, and J. Palen, "Agent-based real-time computing and its applications in traffic detection and management systems," in Proc. ASME 24th Comput. Inf. Eng. Conf., Salt Lake City, UT, 2004, pp. 543–552.
10. Java Agent Development Framework, [Online]. Available: http://jade.tilab.com/
11. F. L. Bellifemine, G. Caire, and D. Greenwood, Developing Multi-Agent Systems With JADE. Hoboken, NJ: Wiley, 2007.
12. B. Chen, H. H. Cheng, and J. Palen, "Mobile-C: A mobile agent platform formobile C/C++ agents," Softw.—Pract. Exp., vol. 36, no. 15, pp. 1711– 1733, Dec. 2006.
13. Mobile-C. [Online]. Available: www.mobilec.org.
14. Durfee, E.H., 1988. "Coordination of Distributed Problem Solvers". Kluwer Academic Publishers, Boston.
15. Bel, C., van Stokkum, W., 1999. "A model for distributed multi-agent traffic control,Multiple Approaches to Intelligent Systems", Proceedings 1611, 480–489.
16. Glover, W., Lygeros, J., 2004."A stochastic hybrid model for air traffic control simulation". Hybrid Systems, Computation and Control, Proceedings 2993, 372–386.
17. Adler, J.L., Satapathy, G., Manikonda, V., Bowles, B., Blue, V.J., 2005. "A multi-agent approach to cooperative traffic management and route guidance". Transportation" Research Part B 39 (4), 297–318.
18. Logi, F., Ritchie, S.G., 2002. "A multi-agent architecture for cooperative inter- jurisdictional traffic congestion management". Transportation Research Part C 10(5–6), 507–527.
19. Okamoto, T., Ishida,Y., 2007. "Framework of an immunity-based anomaly detection system for user behaviour". In: Proceedings of the International Conference on Knowledge-Based Intelligent Information and Engineering Systems, KES, 2007. Springer.
20. 20. Kaufman, C.; Perlman, R.; &Speciner, M. "Network Security: Private communication in a PUBLIC world". Prentice Hall, PTR, Upper Saddle River, New Jersey. 2002. ISBN 0-13-046019-2.
21. Foster, I., Globus Toolkit Version 4: Software for Service-Oriented Systems. Math and Computer Science Division, Argonne National Lab, Argonne, IL 60439, USA.
22. TheHoneynet Project: Know your enemy: "SEBEK—A kernel based data capture tool", 17 November 2003. Know your enemy: Honeynets. 31 May 2006.
23. Choudhary, A.R, Policy-Based Network Management, Bell Labs Technical Journal, 9, 19-29, 2004.
24. FIPA, FIPA agent management specification, Internet, http://www.fipa.org/specs/fipa00023/index.html, 2004.
25. N. Jennings, M. Wooldridge, Software agent, IEEE Personal Commun. Magazine (1996) 17-20