Network Management using Multi-Agents System

Gustavo Isaza[a], Maria H. Mejía[a], Luis Castillo[a,b], Adriana Morales[a] ,Nestor Duque[b]

[a] Departamento de Sistemas e Informática, Universidad de Caldas

[b] Departamento de Computación y Administración, Universidad Nacional de Colombia Sede Manizales

KEYWORD

SNMP protocol
Multiagent System
Multiagent platform
Network management

ABSTRACT

This paper aims to present a multiagent system for network management. The models developed for the proposed system defines certain intelligent agents interact to achieve the objectives and requirements of the multiagent organization. These agents have the property of being adaptive, acquire knowledge and skills to make decisions according to the actual state of the network that is represented in the information base, MIB, SNMP devices. The ideal state of the network policy is defined by the end user entered, which contain the value that should have performance variables and other parameters such as the frequency with which these variables should be monitored.. An agent based architecture increase the integration, adaptability, cooperation, autonomy and the efficient operation in heterogeneous environment in the network supervision.

1 Introduction

The network management can be difficult when the network have a considerable number of network devices, because verify the correct functioning of all of them at the same time involves to have the enough human resource for its monitoring; is for that reason that necessarily certain tasks must be automatized and that these tasks generate results that at the moment of be consolidated can be consulted in any computer inside the network. The Simple Network Management Protocol (SNMP), allows manipulating certain variables of the network devices that support it, it is used to monitor and to control the behavior of those devices.

The intelligent agents are software entities with special properties that allow them to execute in an autonomous way and make decisions in agreement with predefined conditions of the environment in which they live. This article presents the development of a multiagent system (MAS) for the network management, supported with four agents that perform particular actions and that implement their communication with the messages passing. These messages are contextualized within a specific ontology for this system.

The agents involved in this system have the ability to extract information from network devices through the use of SNMP, analyze it and verify potential abnormalities that can occur; when these cases happens, the system notifies it through alarms or traps to the network managers so they can respond accordingly.

The construction of MAS allows shaping the environment for managing network distributed and autonomous, capable of adapting and transforming environmental conditions on which operates with minimal human intervention.

The rest of the material is organized as follows: The following paragraph contains some concepts of the protocol of network administration SNMP, then makes a specific reference to some features of the development methodologies of MAS that were promptly used in the project, the numeral 4 details the proposal phases in the analysis, design and construction, while paragraph 5 presents the results obtained, to finish with conclusions and future work.

2 SMP Protocol

The Simple Network Management Protocol (SNMP) is an application layer protocol that facilitates the exchange of management information between network devices. It is part of the TCP/IP protocol suite. SNMP enables network administrators to manage network performance, find and solve network problems, and plan for network growth. An SNMP managed network includes three main components: managed devices, network management systems (NMS) and agents [SUBRAMANYAN, R. *et al*. 2010], [CASE, J. *et al*. 2008]

A managed device is a network node that contains an SNMP agent and resides on a managed network. Managed devices collect and store management information and make this information available using SNMP. Managed devices, can be servers, switches, hosts, routers and printers, etc.

An agent is a software module that resides in a managed device. An agent has local knowledge of management information and translates that information in a compatible way with SNMP.

The Network Management System executes applications that monitor and control managed devices. NMS manage the processing and memory resources required for network management.

Management Information Base (MIB) is an information collection that is organized hierarchically. The MIB are accessed using a network management protocol such as SNMP. SNMP uses these structures to define the interface for a resource. A MIB consists of a set of object definitions, each of which exposes some property of the resource to be managed. These definitions must be extremely flexible, since SNMP is designed to support any resource type.

The Figure 1 presents an SNMP model and architecture using generic agents.

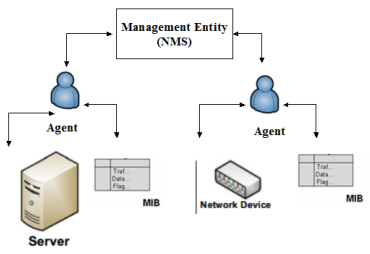

Fig. 1 SNMP Architecture

3 Multi-Agent System (MAS) Methodology

In [MEJIA, M. *et al*. 2007] is described the use of different methodologies to analyze the problem in discussion. In our case, we used GAIA, INGENIAS and MASE to compare the agent software engineer life cycle until the implementation.

A MAS is a system composed of multiple interacting intelligent agents, oriented to solve problems which are difficult for a single agent.

3.1 GAIA

GAIA is a methodology for agent-oriented analysis and design. The GAIA methodology is both general, in that it is applicable to a wide range of multi-agent systems, and comprehensive, in that it deals with both the macro-level (societal) and the micro-level (agent) aspects of systems. GAIA is founded on the view of a multi-agent system as a computational organization consisting of various interacting roles [WOOLDRIDGE, M, *et al*. 2000], [ZAMBONELLY, F., *et al*. 2003]

3.2 INGENIAS

The INGENIAS methodology, proposes a language of specification of Multiagent Systems, and its integration in the Software life cycle [PAVON, J. *et al*. 2007]. The specific language uses meta-models and natural language. The model Objectives/Tasks in the INGENIAS methodology allows the partial documentation of the task of supervising fulfillment of policies. This methodology reviews the dependencies between the different meta-models. In the case of the Objective/Tasks model, the objective is associated to the tasks by means of instances of *GTAfecta*.

3.3 Used Methodology

The different methodologies allowed making a comparative analysis of the problem and their possible process of software engineer based on agents studying different sequences and life cycles. In this case, we worked with INGENIAS, GAIA and MASE to determine the best solution to integrate Multiagent systems in networking management with SNMP.

4 Architecture and construction of Multiagent System

The multiagent platform was build on Java using the JADE framework [BELLIFEMINE F. *et al*. 2003] advantages that simplify the MAS construction and provides a set of tools for platform debug.

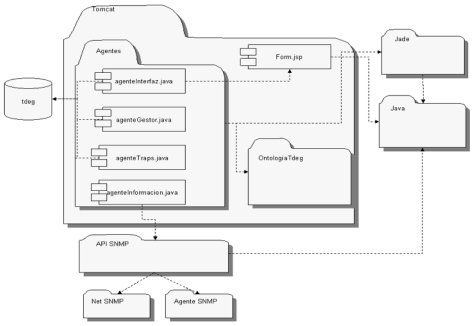

Fig. 2 Components diagram

The MAS consists of *Agente Interfaz*, *Agente Gestor*, *Agente de Información* and *Agente de Traps*. The agent's names are used in Spanish because in the design and implementation are defined in this language. The Figure 2 is the diagram of components, where agents are represented as Java Classes and the tools that support them.

The Figure 3 is the System Collaboration diagram where agents and interchanged messages with their own performative and the action implemented in the ontology is represented.

The system can be managed via Web, being necessary to integrate the multiagent platform and the server.

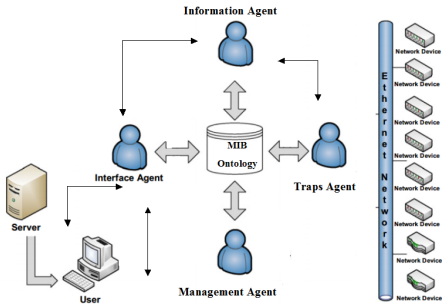

Fig. . 3. Collaboration diagram

4.1 Interfaz Agent

This agent is dedicated to receive IP directions and user specified OID, this information is needed to know the device and the SNMP characteristic subject of study, besides, this agent will present the information related to abnormal actions during the operation of the network, this information is show in traps and alarms.

The *Interfaz* Agent has 4 behaviors:

- *getValores*: It receive the IP and OID values previously entered by the user through the Web interface. It uses the REQUEST performative.
- *getReportes*: It receives the result of the user query to the MAS using the INFORM performative.
- *getAlarmaGestor*: It receives emitted alarms from *Gestor* Agent be shown to the user.
- *getTrapGestor*: It receives the emitted traps from *Gestor* Agent to be shown to the user.

4.2 Information Agent

This agent is devoted to interact with the *Traps* Agent, with the *Gestor* Agent and whit the SNMP API used for communication with network devices. This agent acts as intermediary between the MAS and the monitored network.

Información Agent has 4 behaviors to execute different tasks on the network. This agent rewrites the JADE Agent class methods *setup*() and *takedown*(). The behaviors are:

- *getRequest*: It receives an ACL message with Request performative and with the ontology *Buscar* action retrieves a MIB value from an specific IP. The response is sent to the *Gestor* Agent with a

message that includes an *Actualizar* action with the Inform performative.

- *getRequest1*: It is the same to the previous one, but it uses the Query_If and Inform_If performatives to receive and send the messages.
- *setRequest*: It is devoted to update a MIB value in an specific IP, it receives a message with the *Actualizar* action and Propose performative, but it doesn't respond with a message.
- *porcesaTrap*: Its function is to gather all the generated traps in the network and to send it to the *Gestor* Agent within a message with the Failure performative that consists of the ontology's *Buscar* action.

4.3 Traps Agent

This agent is devoted to receive the traps that are sent from *Información* Agent and to store the data (Date, source IP and Trap OID) into the database. It notifies to the *Gestor* Agent about the trap.

The only one behavior of this agent performs 3 tasks:

- It receives the messages originated form *Information* Agent with the Failure performative that has into its content the ontology's *Buscar* action.
- It stores in the database the information related to the trap.
- It sends to the *Gestor* Agent the same message received from the *Información* Agent.

4.4 Management Agent

It is the intermediary between the *Interfaz* Agent and the *Información* Agent. It receives the request of OIDs and sends it to the *Información* Agent; furthermore it receives the results and responds to the source agent (*Interfaz* Agent). This agent tracks several OIDs in some devices to know the behavior of the network to generate the appropriated alarms according to the defined user's threshold. The alarm will be stored in the database and a message will be sent to the *Interfaz* Agent informing about the problem, describing the failure, the device that generates the alarm and the altered OID.

This agent has 5 behaviors that define the tasks that must be accomplished depending on the received message.

- *getAgentInterfaz*. This CyclicBehaviour type behavior receives a message of Request type with the *Buscar* action and accomplish the forward function with the IP and the OID received from *Information* Agent with a Request type message.
- *getAgentInformacion*: This CyclicBehaviour type behavior receives an Inform type message with the device IP, and the requested OID and its value, it creates an Inform type message to be sent to *Interfaz* Agent that performed the original request.
- *getTrapTraps*: It is a CyclicBehaviour type behavior that receives a message from the *Traps* Agent indicating which was the device that generates the trap, it creates a new message with the Failure performative and send it to the *Interfaz* Agent to notify the received trap.
- *getTestRed*: It is a behavior that check several OIDs in specific devices, within a period of 10s to identify whether the status of the network is operating according to the established policy.
- *getAgentInformacionTestRed*: As a result of the query performed in the previous behavior about OIDs, the *Information* Agent sends an Inform_if type message including the OID's value. This value is confronted with the values defined in the policy, previously stored in a database to produce an alarm that will be stored in a database for historical purposes.

4.5 Ontology

The developed ontology consists of the concepts *IP*, *OID*, *Valor*, *Descripción* and of the actions *Buscar*, *Actualizar* and *Alarmar*.

The *IP* concept is a string that contains the IP address of a specific device. The *OID* concept is a string that contains the number of references for an OID variable. The *Valor* concept is a string that represents the MIB's variable value in a specific device in a specific time. The *Descripción* concept is a string that describes the cause of the generated alarm.

The *Buscar* action consists of concepts *IP* and *OID*, this action has several purposes: it is used to indicate to the *Información* Agent to search for a spe-

cific OID variable in a specific IP address, when the *Información* Agent receives a trap, sends it to the *Traps* Agent including the *Buscar* action whit the trap OID and the source IP. The *Buscar* action is used to pass the IP and OID values between the *Interfaz* Agent and the *Gestor* Agent.

The *Actualizar* action consists of concepts *IP*, *OID* and *Valor*; inside the ontology it has the function of communicate to the *Gestor* Agent about a specific value to a MIB's variable represented by the OID in the device with the corresponding IP to be assigned. The data is sent to the *Información* Agent.

The *Actualizar* action is used when the *Información* Agent responds to a query about a variable, then it sends inside of the message an *Actualizar* action that informs the actual OID value, related to the requested IP: it accomplish the same function of the message that goes from *Gestor* Agent to *Interfaz* Agent.

The *Alarmar* action consists of *OID* and *Descripción* concepts, this action is used to identify some abnormal situation presented during the network operation, it is, when a value defined by the policy is altered generating the abnormal situation.

5 Results

The system execution in a controlled environment gave the awaited results according to the limitations of the established policies and the defined thresholds. The effect of the execution of the Multiagent system to monitor the network can detect if the policy associated to the location of the device is being violated. The agents Behaviour and the detection among them through the message exchange allows to make the network scan; later the *Management* Agent generates an alarm that is stored in the database and is sent to the *Interfaz* Agent that is the one in charge of showing it to the users.

6 Conclusions

To specify completely a MAS in the analysis and design phases and their implementation, it is necessary to use different models using agent software engineering.

In this phase, the project increased the complete functionality of network supervision and reduced tuning the policies. It allows introducing other learning schemes that turn the platform in a solid network management system that becomes autonomous.

The Ontology integration for the MAS in SNMP is an important data representation to solve the heterogeneous problems in distributed network management systems and in a scalable semantic model find inferences using reasoning tools.

The emerging intelligent agent paradigm is an optimal solution for the distributed network management problem.

The proposed system could be compared against the traditional management technique in terms of response time and speedup using the prototype implemented using the performance management as the case study.

Also conflicts at the moment appeared at the same time for making the execution of all the agents, since these and all the used classes must be compiled with the same version of the Java Virtual Machine.

The agents have been proposed as a solution to the problem of the management of increasingly heterogeneous networks. This research extends an agent framework targeted at network management with an architecture and design for integration with an SNMP agent. The future network management solutions need to be adaptable, flexible, and intelligent without increasing the burden on network resources, the project addressed this requirement with a new management platform architecture based on multi-agents.

Acknowledgments

This work has been partially supported by joint call Universidad Nacional de Colombia and Universidad de Caldas.

References

[BELLIFEMINE F., et al. 2003] Bellifemine F., Caire G., Trucco T., Rimassa Giovanni, "Jade Programmer´s Guide", Jade Tutorial, Tilab (2003)

[CASE, J. *et al*. 2008] Case, J. Fedor, M. RFC 1157: Simple Network Man-agement Protocol (SNMP). MIT Laboratory for Computer Science. From URL: http://www.faqs.org/rfcs/rfc1157.html May 1990. Consulted March 2008

[DeLOACH, S. *et al*. 2006] DeLoach, Scott A. Engineering Organization-Based Multiagent Systems. SELMAS 2005, LNCS 3914, pp. 109 – 125, Springer-Verlag Berlin Heidelberg. (2006).

[MEJIA, M. *et al*, 2007] Mejía, M. Duque, N. D., Morales, A. Metodologías para Sistema Multiagente en el caso de estudio: Ges-tión de Redes basado en el Protocolo SNMP. In Tendencias en Ingeniería de Software e Inteligencia Artificial. Medellín. (2007)

[PAVON, J. *et al*, 2003] Pavón, J. and Gómez-Sanz, J. Agent Oriented Soft-ware Engineering with INGENIAS. In Proceedings of the Third International Central and Eastern Euro-pean Conference on Multi-Agent Systems (CEEMAS'03), Springer Verlag, 394-403, (2003).

[SUBRAMANYAN, R. *et al*. 2000] Subramanyan, R. Miguel-Alonso, J. and Fortes, J. A.B. Design and Evaluation of a SNMP-based Monitoring System for Heterogeneous, Distributed Computing, Technical Report, TR-ECE 00-11, School of Electrical and Computer Eng., Purdue University, (2000).

[WOOLDRIDGE, M. *et al*. 2000] Wooldridge, M. Jennings, N. Kinny, D. The Gaia Methodology for Agent-Oriented Analysis and De-sign. Autonomous Agents and Multi-Agent Systems, Kluwer Academic Publishers. Netherlands. (2000).

[ZAMBONELLY, F. *et al*. 2003] Zambonelly, F., Jennings N.R.and Wooldridge M., Developing Multiagent Systems: The Gaia Method-ology, ACM Transactions on Software Engineering and Methodology, Vol. 12, No. 3, Pages 317–370. (2003)