Enabling a Mobile Network Manager (MNM) through mobile agents

Akhil Sahai, Christine Morin

IRISA-INRIA,
Campus Universitaire de Beaulieu
35042 Rennes Cedex (France)
{asahai, cmorin }@irisa.fr

Abstract. Network management comprises of monitoring and control of a network. Ideally a network management system should be able to provide the facility of managing the network from any site and from any system and network. However the present network management systems are centralized and tied to particular systems. We introduce the unique concept of a Mobile Network Manager (MNM) to provide network administrators a system and location independent network manager which will be useful under a variety of circumstances. In order to enable the MNM mobile agents have been used. The mobile agents are furnished by the mobile agent environment MAGENTA (Mobile AGENT environment for distributed Applications) which provides autonomous, reactive, proactive and communicative mobile agents. MAGENTA has been designed and implemented to support mobile user aware applications. The mobile agents are used not only to facilitate the disconnected mode of functioning of the MNM but also to decentralize network management functionalities. The applicability and usefulness of mobile agents in implementing the functionalities of MNM has been compared with the client-server mechanism.

1 Introduction

Network management plays an important role in the configuration, proper functioning, and maintenance of a network. The task of network management relies heavily on the competence of the administrator to discern the large amount of available data. It is thus necessary for the administrator to be available in the proximity of a Network Management Station (NMS). However, the nature of network faults being random it is often not the case. In case of an unforeseen and untimely emergency an administrator away from office can utilize the MNM from the present location without wasting precious time. In the existent network management systems the administrator receives a cryptic pager message in case of an emergency and one of the recourse left to the administrator is to rush to the management station. Administrators absent from their office (due to work meetings, travelling etc) may also intend to perform some amount of their duties and may avail of the MNM. The MNM would also be useful when mobility of the administrator comes as a part of job description. In case of administrators managing large intranets the administrator may be obliged to move from one site to another(e.g, because of some network upgrade being carried out). The administrator however should be able to perform his/her duties normally in spite of being mobile. The usage of MNM can be envisaged during installation and upgrade of networks. The field personnel installing a network component may wish to study the components behaviour in the context of larger network and may wish to utilize an MNM to continuously study and modify the component characteristics. The MNM has thus been designed to be a system and location independent manager which can be utilized by the administrator from whichever system or site he/she is present on. The MNM executes on a portable computer and thus is constrained by the minimal resource available to it and also by the fallible and costly link which the mobile computer utilizes to connect to the wired network. In order to overcome the resource constraints and also to provide for disconnected computing the MNM is enabled through mobile agents.

The agent-based computing has been found to be more effective than the client-server paradigm in certain application domains [6] and specially in the case of partially connected computing. Of late mobile agents have evoked lot of interest [2][5] [4][3]. The mobile agents used by the MNM are furnished by MAGENTA. The MAGENTA (Mobile AGENT environment for Administration) agents are autonomous, they perform according to predefined goals, react to changes in the environment and communicate with other mobile agents and thus satisfy the weaker notion of agency [1]. MAGENTA provides for extensible and modifiable functionalities of the agents, fault-tolerance, dynamic adaptability to changes in the environment, flexibility in choice of location of agent code, simultaneous and fast execution of agents, true system independence along with access to system functions and the facility of remote execution.

In this paper, we present the motivations, design and implementation of the MNM. We also present the MAGENTA environment and describe, how it is utilized to enable the functioning of the MNM. The rest of the paper is structured as follows. The subsequent section provides a background on mobile agents followed by a brief overview of network management systems. Section 4 presents the design of MNM. The MAGENTA environment is then presented in Section 5. The next section describes the functioning of MNM followed by performance measures of the MNM before concluding.

2 Overview of Network Management

A typical network management system contains four types of components: Network Management Stations (NMSs), agents[1] running on managed nodes, management protocols, and management information. A typical NMS is depicted in

[1] These are network management programs executing at the network elements, not to be confused with mobile agents

Figure 1. A NMS uses the management protocol to communicate with agents running on the managed nodes. The information communicated between the NMS and agents is defined by a Management Information Base (MIB). The management standards that have emerged are the Simple Network Management Protocol (SNMP)[8] and the OSI management system which utilizes the Common Management Information Protocol (CMIP)[8]. The SNMP protocol provides *get, set, trap* primitives for interaction between the NMS and SNMP agents.

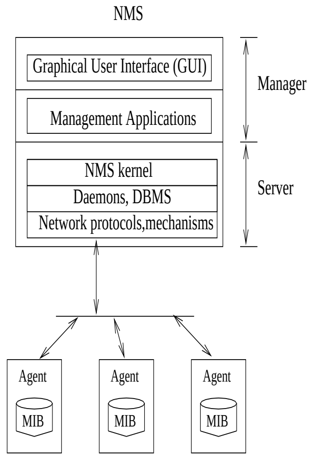

Fig. 1. A typical Network Management System

Each resource to be managed at the SNMP agent is represented by an object and MIB is a structured collection of such objects. The MIB is defined in terms of a tree structure. Each object is assigned an object identifier (oid). These objects are placed under different groups. Some of the important groups in the commonly used MIB-2 are system, interfaces, at, ip, icmp, tcp, udp, egp etc. These groups consists of large number of variables. The group system for example, comprises of *sysDescr*, *sysObjectID*, *sysUpTime*, *sysContact*, *sysName*, *sysLocation* and *sysServices*. These variables describe a system, for example, sysUpTime describes the time since the network management portion of the system was last reinitialized and sysName refers to an administratively assigned name for the managed node. The NMS requests the SNMP agent from time to time the values of these variables by referring to them by their oids. The SNMP agents send a trap to the NMS in case there is an alarming change in the SNMP variables. The NMS also typically utilizes client-server mechanism to access the SNMP values.

3 Design of the MNM

In order to implement the MNM we made some design choices. These design choices were guided by the special nature of the environment of execution of the MNM and also by the nature of tasks to be performed by it. Following are some of the design choices we made.

3.1 Design choices

Distribution of the network management system

We intended to build a light-weight and portable network manager, however the existent Network Management Stations (NMS) require a large amount of resource are limited in number and centralized in nature which leads to scalability problem and network bottlenecks. We thus needed to separate the management applications part of the NMS from the heavier kernel part. This enabled the possibility of having multiple managers existing at the same time and thus the possibility of distributing the network management system. We thus came up with an architecture as shown in Figure2. In the architecture there are multiple managers which communicate with a server. The manager comprises of the GUI and the applications as well as the communication mechanism to access the server and SNMP agents and thus are light-weight in nature.

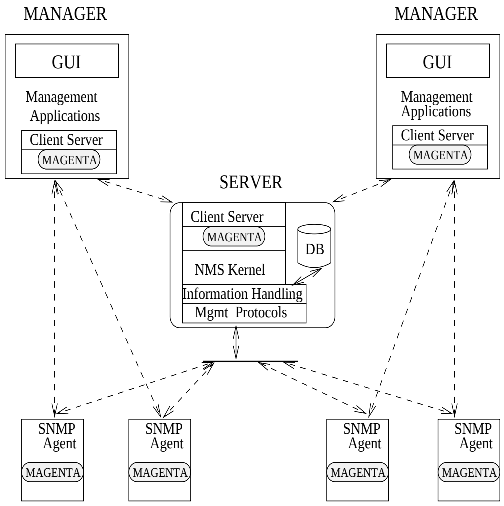

Fig. 2. The utilized network management architecture

Location independence

We also intended to have a location independent manager i.e MNM needed to operate from any site. In case of absence of wired network, the MNM is obliged to use the telephone line or use the wireless link. In the network management system we have provided mechanisms to the managers so as to access the server and SNMP agents from a remote site using different types of connections and possibly simultaneously, irrespective of the location of the site.

System independence

The existent network management systems are normally tied to a particular system and have dedicated machines for their operation. We distributed the network management system in order to provide for a uniform and simple interface, irrespective of system. Java being architecturally neutral provides the facility of a uniform interface on different platforms. Our network management systems being written in Java is also cost-effective.

Overcoming resource constraints

The MNM was to execute from a portable computer operating either in the tethered mode (using telephone lines) or in the wireless mode. Both types of connections are characterized by a limited bandwidth availability, high cost and fallibility. In addition the portable computer has limited battery power and resources. Thus, it could not be connected for a long time to the network. In order to get around these problems, mobile agent technology has been used to enable the functioning of the MNM.

3.2 Agent based network management system

Our network management system comprises of managers, a server and SNMP agents. The managers are dual mode in nature. In static mode the manager executes on a wired computer while in the mobile mode it operates on a mobile computer. The manager communicates with the server and the SNMP agents using either the client-server mechanism or the mobile agent mechanism. The mobile agents are furnished by the MAGENTA (Mobile AGENT environment for distributed Applications) environment. All the management components namely, managers, server and SNMP agents as shown in Figure 2 are integrated with the MAGENTA environment. The managers also are provided with the client-server mechanism and can perform get, get-next and set operations on the SNMP agents directly.

4 MAGENTA Environment

MAGENTA is a mobile agent environment available over a large range of systems as it is written in Java. The MAGENTA environment essentially comprises of the *lieus* and *agents*. A lieu is a program that implements support services for reception, execution, storage and moving of incoming agents, and services to maintain information about other lieus and agents in the environment. The lieu also facilitates communication between agents allowing the agents to reside on it. Every lieu has a globally unique name and can appear and disappear dynamically. The

MAGENTA agents are programs that move between the lieus, following a travel itinerary and performing a predefined task. Each agent is provided with a *purpose* which defines its type behaves autonomously to achieve its purpose. The agent executes a *perform* primitive which comprises of code of execution. Each agent is also provided with a *permit* in order to determine its suitability for execution on a lieu; an agent that arrives on a lieu is first authenticated and after, if it passes the security requirements, it is allowed to access the services available on the lieu. The agents also detect changes in the environment and store the information in their variable *knowledge*. In MAGENTA, the agents have the ability to operate autonomously, to react to changes in the environment by changing their itinerary, to *meet* other agents in order to communicate and exchange *notes* with them. MAGENTA in addition to general features provides mechanisms of support for mobile computing. Following are some of the important features of the environment:

4.1 Different types of agent mobility

MAGENTA provides for weak migration of agents i.e it provides for retention of the data state of the agent while on the move. However, it also provides for remote execution, remote evaluation and code on demand. In remote execution an agent can be initialized and sent to a remote site where it executes. Remote evaluation is more suitable for extremely resource constrained devices like PDA which can not either execute a lieu or do not support the Java environment. Such devices can request a remote lieu to initialize and launch an agent on its behalf and can collect the results of the computation once the agent's execution has terminated. Code on demand is a variant of agent mobility paradigm where a site requests another to send an agent to it with the desired code.

4.2 Naming Scheme

In MAGENTA each lieu has a globally unique name. This name can be either chosen at the time of commencement of the lieu or can be automatically assigned. For a lieu on a mobile computer the lieu name is so chosen so as to remain constant over different sessions. This renders the network transparent to the applications using MAGENTA. Even if the IP address changes from one session to another the lieu name remains constant. Once the lieu reconnects with a new address, the information is passed off to other lieus which update their references. The agents thus do not perceive any apparent change and can identify the correct lieu to return to.

4.3 Appearance and disappearance of lieus

The mobile computers are likely to connect intermittently. The disconnections could be preplanned or unplanned. MAGENTA provides for such likelihoods. In MAGENTA the lieus can appear dynamically and disappear dynamically. This disappearance can be foreseen or unforeseen. In case of planned disappearance a mechanism of *exit* is provided to the lieus. The lieu informs all the other lieus about its imminent disappearance and then quits gracefully. When the lieu reappears subsequently all the lieus in the environment are informed about the appearance of the new lieu. A distributed update of information is thus carried out. In case of abrupt disappearance of the lieu, fault-tolerance mechanisms are provided for detecting such disappearances.

The abrupt disappearances of lieus are detected by the following mechanism. Each agent, moving from a lieu to another avoids the unavailable lieus found during its travel, dynamically altering its itinerary and keeping track of the failed lieus in its variable *knowledge*. When the agent arrives at the last destination of its itinerary, it provides the lieu information about the unavailable lieus, if there are such failed lieus, then the destination lieu informs all other lieus about the disappeared lieus. On receiving this information, all other lieus update their local information. So the failure of a lieu running on a mobile computer which disconnects abruptly can be discovered by an agent not launched by it but having it in its itinerary.

A lieu executing on a portable computer might disappear abruptly after launching an agent. The agent after finishing its itinerary moves to the last destination lieu on the wired network. It tries to move to the mobile lieu but it fails in its attempt. It informs the lieu of residence about its failure which in turn informs all other present lieus about the abrupt disappearance of the mobile lieu. Thus a failed lieu is also discovered by the agents launched by it when they try to come back to it at the end of itinerary. The agent in this case waits on the last static lieu for the mobile lieu to connect back. Once the mobile computer lieu appears again, the other static lieus are informed and the agents waiting for the mobile lieu at different lieus come back to the mobile lieu.

4.4 Tolerating faults

The agents because of their very nature of execution are prone to falling prey to numerous accidents. The lieus are also liable to disappear because of abrupt disconnection of portable computers or system crashes. Fault tolerance schemes have been provided to minimize the overheads associated in detection and recovery from such site failures.

In MAGENTA we provide for disappearance and recreation of agents from a single fault at a time. We maintain backup copies of the agents in order to take care of abrupt disappearance of an agent. In order to reduce the number of backup copies and to reduce the communication which results in case of disappearance of an agent we employ a scheme to maintain judicious number of copies. A lieu before sending an agent to the next lieu saves a copy of it and takes the temporary responsibility of detecting the failure of the next lieu. The next lieu after sending the agent to another lieu informs the lieu with the backup to remove the backup. The lieu with the backup periodically performs checks on the next lieu until it receives a message from the next lieu to destroy the backup. In case such a message is not received by the lieu with the backup within timeout and if it detects that the next lieu has crashed it recreates the agent from the backup copy.

This is explained with an example as shown in Figure3. Before an agent moves from a lieu, lets call it a lieu A to the next lieu, say lieu B it makes a copy of itself on lieu A. Lieu B processes this agent and makes a copy of it before moving it to the next lieu, say lieu C. As soon as the agent is successfully moved to the lieu C, lieu B requests lieu A to delete the copy of the agent. In case lieu B discovers that it is not possible to move the agent to lieu C because of some problem, it tries to reach lieu D. In case this operation is successfully carried out it asks lieu A to delete the copy. Lieu B after successfully moving the agent to the next lieu waits for a message from the next lieu, say lieu C. In case, the lieu C fails to send message to lieu B, lieu B understands that lieu C has not been able to move the agent to the next site in the itinerary because of some problems. It thus first does its own check to find out the problem. In case lieu C has crashed the backup copy of the agent on lieu B is used to create the agent and is passed to the next alive lieu. This scheme thus takes care of single fault and regeneration of agent by keeping the messages to a bare minimum.

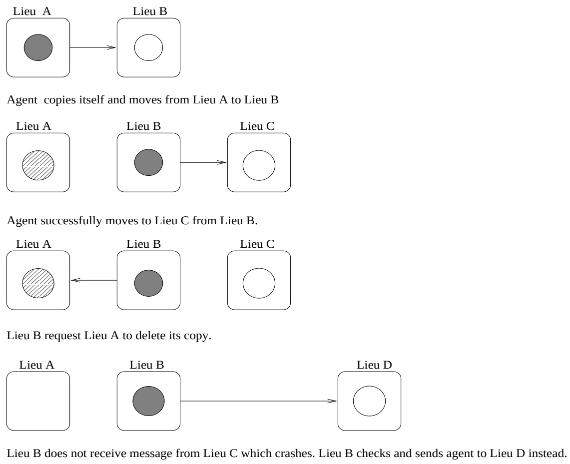

Fig. 3. The implementation of fault-tolerance scheme

4.5 Tracability of agents

In MAGENTA a directory service has been provided so as to trace the agents. Every lieu can request all the other lieu about the latest information about the agents present on them and launched by them. This comes in handy to any lieu to determine the status of execution of agents sent by it.

4.6 Flexibility of location of code

The agent can either bring the code to be executed along with it or can execute code present at a remote site. The choice of the location of the code to be executed thus can be made depending on the resources available and the frequency of usage of the code. If the code needs to be used over a long period of time it can be placed at the remote site however if the code is to evolve over time the code is brought by the agent.

4.7 System independence

MAGENTA is implemented using Java programming language, in order to have a system independent implementation. It was envisaged that for some applications for example MNM it was necessary to access system information. Different systems use a different path separator and also need different native method implementations. So an effort was made to satisfy the requirement and at the same time maintain the system independence of MAGENTA. Dynamic class loaders were written for a lieu. The Native method implementations are thus loaded on the fly in case they exist. In the absence of such implementations and error is provided to the agent. A dot-notation has also been used to take care of the different path separators available on different separator.

5 Functionalities of the MNM

The MNM has been provided a variety of mobile agent based functionalities. These functionalities range from performance study of network components to installation of software at multiple remote sites. Following are some of the functionalities.

Performance

The MNM can either obtain the values of the SNMP variables from a remote SNMP agent by using client server technique or can use mobile agents for the same. In case of client server mechanism the get, get-next, set request primitives are used to interact with the network components.

While using mobile agents, the agent is given a number of samples, multiple oids (Object identifiers) and component's lieu name. The agent moves to the network component studies the oid values for a given number of samples and comes back to the MNM carrying it's results in its folder. The studied values can be displayed in charts namely histograms, bar, pie and line charts. This is also useful for studying behaviour of network components over a long time.

Install

The agent is provided a software, the directory where to install the software and a list of network components. The agent moves from one component to another installing the software and returns with the results in its folder. MAGENTA uses a dot notation to overcome the problem of different path separators in different operating systems. These dots are replaced by local class path separators.

In case the desired directory is not found the mobile agent installs the software in a default directory provided by the lieu. The folder of the agent contains the information about which components have the software installed and on which of them the agent was unable to install the software. The same agent is also used for software upgrade.

Audit

The agent is provided a list of parameters to look into e.g disk usage, system and application configuration, users using the machine etc and a list of network components. The agent moves from one host to another finding all the required information and comes back to the MNM with all these results in its folder.

Retrieve

The agent is told to retrieve the desired information from one or more components. It is used to query the server for large quanta of informations, especially to study the archives.

Net checker

The agent moves to the lieu on the server and performs get on all the SNMP variables from a list of SNMP agents provided by the MNM. This is useful to test the status of the health of the network.

6 Performance Evaluation

In order to compare the usefulness and applicability of mobile agents in implementing the basic network management functionalities we have compared it with the client server mechanism. The MNM was executed on *Cezembre* IBM ThinkPad 760 EL, a Pentium PC, 133 MHz, having 16Mo memory, executing windows 95 connected onto a 10Mbps Ethernet. The server was run on *Astrolog* an IBM Power PC executing AIX 4.1 and connected on the same Ethernet. SNMP agents were run on a variety of machines. The results so obtained are depicted in the following figures.

The comparison of bandwidth utilization is done as follows. In the case of client-server mechanism a SNMP get requests are sent continuously from cezembre to Astrolog for the desired number of samples and the response packets are received continuously at Cezembre. In the case of mobile agents, the agent is sent from Cezembre to Astrolog where it collects locally the required number of samples and comes back to Cezembre. The packet sizes of the SNMP get query and that of the response were noted and the size of the agent sent and received were noted and the bandwidth utilization comparison is depicted in Figure 4.The comparison is pessimistic in approach because even a small amount of data as is the case in client-server might require a large UDP packet. The curve for the mobile agent starts at 20K bytes as the size of the agent sent is 10K bytes. The increase in size of the agent results from the data collected at Astrolog. The packet size of the SNMP get request was measured to be 40 bytes and the response packet size was measured to be 31 bytes at the application level. A similar comparison for retrieval of samples of 5 SNMP variables is shown in Figure 5. In terms of bandwidth utilization thus client-server is beneficial to a threshold of

300 samples after which the mobile agent technique becomes more effective. In the latter case this threshold reduces to 60 samples. Thus mobile agent approach is suitable for long term monitoring of network components.

In Figure 6 the comparison of bandwidth utilization, when a single agent is sent in a itinerary to retrieve 400 samples of a single SNMP variable from multiple machines and client-server technique is shown. The client-server mechanism is used from the manager machine to the SNMP agent machines separately and the requests are made in parallel. We observe that for this particular case an itinerary of no more than 23 machines is desirable. Thus after this threshold either the agent should return to the launching lieu or empty its load before moving on to new lieus which will reduce its size and thus the bandwidth usage. In Figure 7 the response times between client-server and the mobile-agent technology are compared. From the manager machine the time (in millisecs) it took for both the agents and the client-server technique to retrieve the given number of samples were noted and plotted as shown. The response time in retrieving the samples were initially found better in client-server technique. But for larger number of samples the response time was better in sending an agent to retrieve the values than using client-server technique to continuously query and obtain the values of the SNMP variable from the SNMP agent.

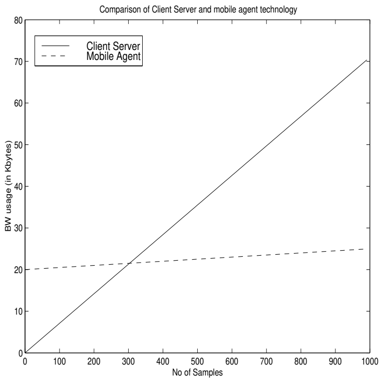

Figure 4:Comparison of bandwidth utilization

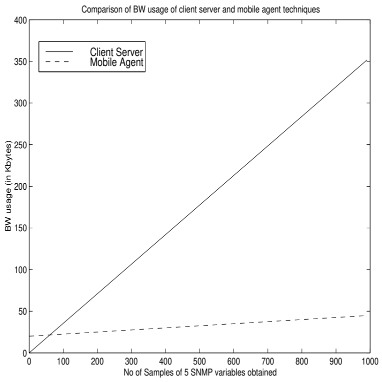

Figure 5:Comparison of bandwidth ut ilization

7 Conclusion

In this paper we have presented the design and implementations details of the unique idea of a Mobile Network Manager which we think would be of great importance in the realm of distributed system management. In this context we

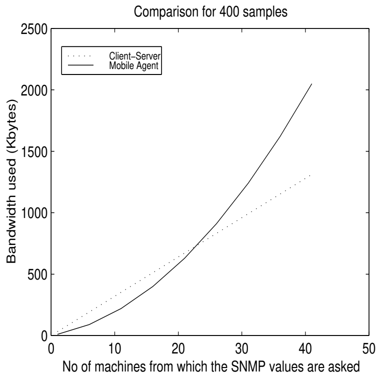

Figure 6:Comparison of bandwidth utilization

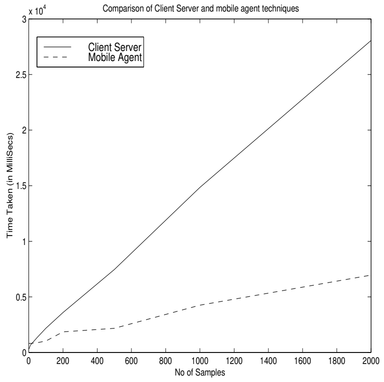

Figure 7:Comparison of response tim es

have designed and implemented the MAGENTA environment which is highly generic in nature. It provides autonomous, reactive, proactive and communicative mobile agents. We enable the MNM through the utilization of mobile agents which we use not only to overcome the constraints imposed by the portable computer and that of the fallible link but also to distribute network management functionalities.

References

1. Russell Beale and Andrew Wood. Agent Based Interaction *in People and Computers IX : Proceedings of HCI'94,Glasgow,UK, August 1994 , pp 239-245*
2. J. Bauman et al. Mole - A Java based Mobile Agent System. *ECOOP'96 Workshop on M obile Object Systems.*
3. H. Peine, T.Stolpmann. The Architecture of the Ara Platform for Mobile Agents. *First international Workshop on Mobile Agents, MA97*
4. D. Johansen, R. V. Renesse and F. B. Schneider. An Introduction to TACOMA Distributed System *Technical report CS-95-23, Department of Computer Science, Institute of Mathematical and Physical Sciences, University of Tromso. http://www.cs.uit.no/DOS/Tacoma/.*
5. D'Agents at Dartmouth. *Computer Science Deptt, Dartmouth college, USA, http://www.cs.dartmouth.edu/ãgent/.*
6. Colin G. and Harrison et al. Mobile Agents: Are they a good idea? *IBM T.J. Watson Research Center Technical Report, 1995.*
7. Sahai A. and Morin C. Towards Distributed and Dynamic Network Management *In the proceedings of IEEE/IFIP Network operations and Management Symposium (NOMS'98), New Orleans, USA, 16-20 February, 1998.*
8. W. Stallings. SNMP, SNMPv2 and CMIP: The practical guide to network management standards. *Addison-Wesley publication, 1994.*