The Mobile Agent Enhanced Thin Client Approach to Network Management

Akhil Sahai, Christine Morin
INRIA-IRISA, Campus de Beaulieu, 35042, Rennes CEDEX, France
{asahai,cmorin}@irisa.fr

The network management architectures used in practice lack distribution, decentralization and flexibility. In addition they have not kept pace with the latest technologies. Some approaches have been proposed to introduce new technologies in the domain. Recently there have been proposals to have a thin client approach to network management. Other suggestions have been to use mobile code for delegation. Our approach effectively combines the two seemingly different approaches. We also take the mobile code approach to logical conclusion by utilizing mobile agents for network management. A mobile agent is a mobile code which is also autonomous, reactive, proactive and communicative. The mobile agents are furnished by MAGENTA (Mobile AGENT environment for distributed Applications) which we have designed and implemented in this regard. We have designed and implemented a distributed and dynamic environment for network management based on our approach. The environment uses client-server or mobile-agent technology depending on the functionalities implemented.

1 Introduction

The growing importance of the network in the organizations and their increasing complexity dictate the availability of network management systems which can ensure proper performance of the networks. These network management system monitor and control the networks. They also configure the network, obtain and archive the information, monitor the performance, monitor and manage faults and perform security control. However, the prevalent network management system have certain limitations. They are usually tied to a particular system and are centralized in nature. Of late, some efforts have been made to address these issues. The thin client approach advocates remote access to network management functions through a client which consists of graphical display and applications[2]. While other approaches have called for distribution and decentralization of network management systems by using mobile code[1]. The mobile code in this kind of approach is used to execute network management code closer to the network elements being managed.

The Astrolog[a] approach effectively combines two different approaches and benefits from them. Mobile agents are used to supplement the basic network

[a]This work has been partially funded by Dyade, R& D joint venture between BULL and INRIA.

management functionalities and are also used to perform some of the basic network management functionalities like network element monitoring. We also compare the applicability of mobile agent approach by comparing it with the client-server approach. The rest of the paper is structured as follows. In the subsequent section an overview of network management is presented. In section 2 the thin client approach is elaborated. Section 3 presents our approach. The following section details the implementation, functioning and the evaluation of the approach.

2 Overview of Network Management

A network comprises of an interconnected set of network elements e.g routers, bridges, workstations, printers etc. Network management systems essentially have the functionality of network monitoring and network control of the network elements. Network monitoring is concerned with monitoring the state of the network without interference, while network control involves active participation and interference in the state of the network.

A typical network management system as shown in Figure 1 comprises of four types of components: Network Management Stations (NMSs), management agents running on the network elements[b], protocols, and management information. A NMS is a computer system, typically a workstation, which executes network management applications. It provides the basic functionality of network management for a variety of different devices. The components of a typical NMS are:

- a graphical user interface (GUI)
- management applications
- a NMS kernel
- a database management system (DBMS)
- information handling mechanisms
- management protocols

The GUI gives users easier access to the features of the NMS. The GUI is provided at the workstations's console(s). The NMS uses standard methods to query devices as it must be capable of gathering information from many different vendor components. The information obtained from the devices is stored in

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

the DBMS. A DBMS helps in many network management tasks. Management applications use it for information storage and to build relationships between data items, which help in network diagnosis and maintenance. The NMS uses management protocols 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 protocols in use are simple network management protocol (SNMP)[7], common management information protocol (CMIP), systems network architecture (SNA) etc. Amongst the protocols in use, the SNMP is the most commonly used protocol because of its simplicity of installation and use. Astrolog utilizes SNMP as the representative protocol for implementing the network management functionalities. The SNMP protocol includes the following capabilities mainly,

Get : enables the management station to retrieve the value of objects from the agent

Set : enables the management station to set the value of objects at the agent

Trap : enables an agent to notify the management station of significant events

The prevalent network management system architectures are centralized in nature which leads to scalability problems. A bottleneck results at the NMS as all the network data is brought to it while the peripheric devices perform minimal management activities. They also utilize the client server mechanism for interaction with the management agents which can cause lot of bandwidth usage. They are usually tied to particular operating systems and architectures and the NMS is not available over a variety of platforms.

3 The Thin Client Approach

The thin client approach has been proposed to provide portability and load distribution lacking in the traditional network management systems. This approach separates the graphical and application part of the NMS from the kernel and data collection part[2 3 9 10]. In this case the applications are usually written in architecturally neutral language and are system independent. They are usually light-weight and multiple applications can be executed simultaneously, sometimes through web[11] which access the network management server and/or SNMP agents. This approach does not call for a return to main-frame computer because the processing is partly done at the application site. The bottleneck at the NMS is removed to an extent as the processing part is shifted to the client located on a separate machine.

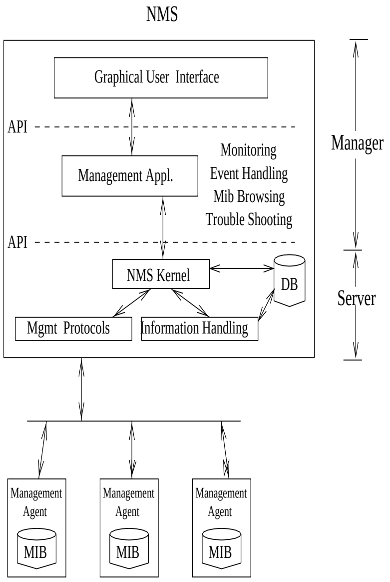

Figure 1: A typical Network Management System

4 The Mobile Code Approach

The manager sends mobile code to the network elements being managed [1]. The mobile code is incorporated and executed by the static program located at the network element. The mobile code approach brings the management code near the SNMP agents and network elements being managed. It provides for dynamic changing of network management policies as new code can be easily written to change the policies, if so required. They also distribute the management and thus enable further load balancing and removal of bottleneck at the manager. However, this approach does not provide for a portable manager and the separation of the application and the kernel part of the network management system.

The mobile code however provides just a subset of the possible activities of mobile agents. The utilization of mobile code is limited to one destination jump and thus restricts them from being utilized for decentralization of other important network management functionalities like software upgrade and installation, long term monitoring of network elements, audit of network, discovery etc, which needs the code to retain its data and execution state and to move over multiple machines to achieve its goal. Mobile agents [5] are in fact mobile code which are capable of moving from machine to machine preserving their data state and exhibiting goal-directed behaviour. In addition, they demonstrate autonomy, react to changes in the environment, seize initiative and communicate with other agents and resources. Recently, there have been some proposals to use mobile agents for management [4,8].

5 Our Approach

The architecture incorporates the thin client approach and the mobile agent approach to network management. It thus benefits from the advantages obtained by two seemingly different approaches. The proposed architecture is highly modular in nature. It comprises of managers, server and management agents as shown in Figure 2. The manager consists of the GUI and the application part of a typical NMS, while the server consists of the NMS kernel, the DBMS, the protocol and mechanisms for communicating with the management agents. There is thus a separation between the application and the kernel part of the NMS amongst the manager and server in our architecture. The network management load is thus distributed amongst the managers and the server and since the managers and servers typically execute at different sites there is a distribution of load over multiple machines. The managers are architecturally neutral and light-weight in nature. The managers can obtain the network management information either from the server or from the management agents thus there is no bottleneck at the server and not all requests and data are channelized through the servers. The servers obtain and store only the information which needs to be archived and comprises of various daemons which continuously collect information about the configuration changes in the network and perform network discovery. The network management system provides the possibility of adding multiple servers for larger networks and the number of servers is determined by the size of the network. In such a case the DBs needs to be replicated amongst the servers and data consistency mechanisms need to be utilized.

The managers, server and the management agents are equipped with communication mechanisms to interact amongst themselves. Each manager is assigned a server however it can connect to other servers in case the assigned server fails (if other servers are available). This communication mechanism consists of client-server as well as mobile agent paradigm. The communication mechanism is chosen depending on the location of the manager and the suitability of the communication mechanism for the functionality being implemented. The architecture retains the client-server mechanism for interaction between the components. As is the case in traditional network management systems the managers can communicate with the management agents through primitives like get, get-next, set. They also interact with the servers through client-server mechanism. The managers, servers and the management agents are integrated with the MAGENTA environment. The MAGENTA environment provides them the capability of sending, receiving and storing mobile agents.

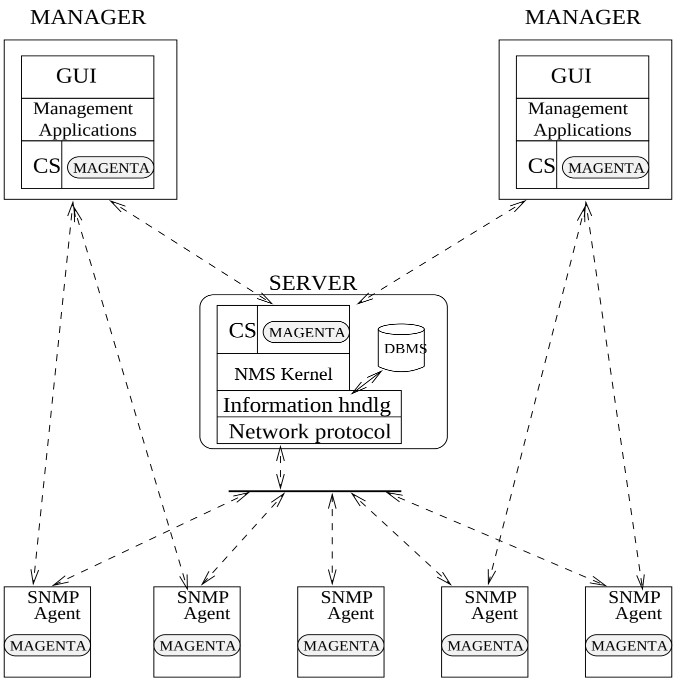

Figure 2: The network management architecture

The managers being light-weight and portable can operate either on a static resourceful computer or on a resource constrained portable computer. The manager uses client server or mobile agents technologies depending on the information requested and the functionality implemented. The architecture is rendered more distributed and decentralized by mobile agents which distribute management code to remote sites thus involving the peripheric devices in management.

6 Implementation, Functioning and Evaluation

6.1 Implementation issues

Environment

The research activity Astrolog intends to design a network management system for our local distributed system Astrolab. Astrolab is a heterogeneous distributed system comprising of PCs (executing WinNT, Win95, Linux, NetBSD etc), SUN Sparcs and AIX machines connected by a LAN. The managers have been implemented in Java and the server utilizes the facilities offered by ISM/OpenMaster. The MAGENTA environment has also been written in Java.

MAGENTA

The architecture of MAGENTA comprises of *lieus* and *agents*. A lieu is a location where an agent can originate, reside, execute and interact with the system as well as with other agents. An agent performs a goal-directed behaviour and utilizes the lieu to perform it's functions. A schematic view of MAGENTA environment is presented in Figure. 3.

The agents in MAGENTA exhibit autonomy, reactivity, proactivity, and social ability. An agent is said to be autonomous if it can operate without direct human intervention and demonstrate some kind of control over its actions and internal state. In MAGENTA each agent is provided with a *purpose* and the agents behave autonomously to achieve the purpose. An agent is said to be reactive, if it perceives its environment and modifies its behaviour according to the changes in its environment. In MAGENTA the agents can alter their itineraries depending on the changes in the environment and due to disappearances of lieus. An agent is said to be proactive when it shows goal-directed behaviour. In our case, each agent acts according to its purpose and *perform* a given task. The task to be performed differs from application to application. An agent exhibits social ability, if it interacts with other agents. In MAGENTA we provide the *meet* primitive to the agents. The agents can meet at a lieu and exchange *notes*. Every agent is provided with a unique name. This name is obtained by adding the symbolic name of the agent with the name of the lieu at which it originates. As each lieu ensures that the agents originating on it have unique names, the agents as a result have unique global names. Every agent is provided with a *folder* which it carries with it. The folder is utilized for storing the outcomes of the tasks performed by the agent. It also maintains the information about the originating lieu and other existing lieus. Each agent is provided with a *history* which stores the information about the agent, its itinerary so far and tasks performed by it.

A lieu is a static program and has a unique names. A lieu provides the execution environment for the incoming agents. The incoming agents are first authenticated to determine their suitability for execution. After the agent passes the security requirements of the lieu, the agent is allowed to access the services available at the lieu as shown in Figure.3 . The lieu facilitates the agents to communicate with other agents and to move to other sites. They also allow the agents to reside. This becomes important in the case of meet operation where an agent might have to wait at a lieu to meet another agent. Each lieu has a repository of all the agents created and dispatched by it and so it keeps the trace of the agents. A lieu also maintains information about all the executing agents. They maintain the backup copies of agents absent locally as long as they do not get the instruction from another lieu to delete the copy.

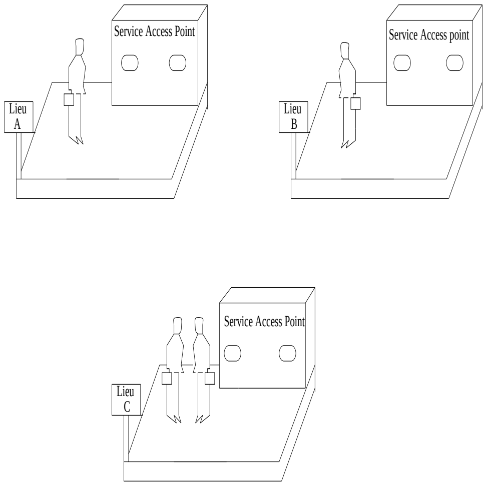

Figure 3: MAGENTA: lieus and agents

6.2 Functioning of the architecture

The functionalities implemented by the managers are either implemented by utilizing client server or by using mobile agents. The table1 demonstrates the technology utilized to implement the functionalities. In addition to the traditional functionalities implemented using client-server, the mobile agent are used to implement *Performance*, *Install*, *audit*, *retrieve*, *Network health* etc and are explained in detail as follows:

Functionality	Client-Server	Mobile Agent
Monitoring	x	
Performance	x	x
Alarms	x	
Telnet	x	
Install		x
Audit		x
Network Health		x
Security		x
FTP	x	x
Fault-management		x

Table 1: Implementation of network management functionalities

Performance

The agent is given a number of samples, multiple oids (Object identifiers) and a network component identifier. The agent moves to the network component studies the oid values for a given number of samples and comes back to the manager 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 manager 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.

Network Health

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 manager. This is useful to test the status of the health of the network.

FTP The agent is sent to one or more remote sites to obtain one or more files. The file retrieval is asynchronous in this case as compared to the normal file transfer.

6.3 Evaluation of the approach

We did a quantitative evaluation of our approach. Our approach advocates usage of different techniques for implementation of different network management functionalities. In order to verify the efficacity of our approach we compared the client-server and mobile agent techniques for performing simple network management tasks like retrieval of SNMP variables. The manager was executed on a variety of machines to test its portability. The manager was also executed on a portable computer to verify the light-weightedness and the veracity of "thin client". Thus two series of tests were conducted. In the first scenario the manager was executed on a static computer and in the second scenario on a portable computer.

Scenario 1

The manager was executed on *Wallace*, a Sun Sparc 5 machine executing Solaris 2.5 and connected on a 10 Mbps ethernet. The SNMP agent was located on the machine *Astrolog*, an IBM Power PC executing AIX operating system and connected on the same Ethernet. In the case of client-server, the response times were noted by sending SNMP get requests from Wallace to Astrolog and the responses were obtained at Wallace. The total time taken for the given number of sample queries were noted. In the case of mobile agent, the agent was sent from Wallace to Astrolog and the agent performed the get request locally and brought back the values in its folder. The response time in this case was noted. The comparison between the response times are depicted in 4. 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 5. 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. As shown in Figure 4 in terms of response times the client-server is faster only for collection of few samples after which it becomes relatively costly to utilise it. However in terms of bandwidth utilization client-server is beneficial to a threshold of 300 samples after which the mobile agent technique becomes more effective. Thus mobile agent approach is suitable for long term monitoring of network components.

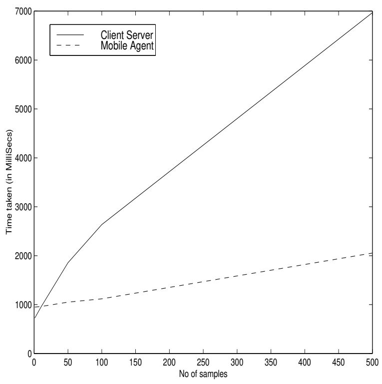

Figure 4:Comparison of response times for varying no of samples of a single SNMP variable

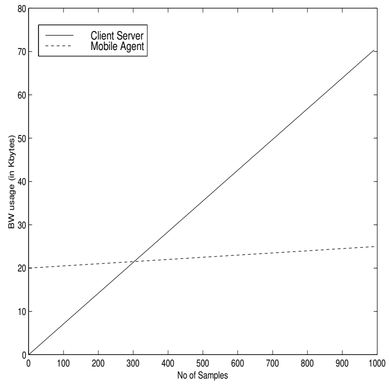

Figure 5:Comparison of bandwidth utilization while obtaining samples of a single SNMP variable

Scenario 2

The manager was executed on *Cezembre* an IBM Thinkpad 760 EL machine connected onto a 10 Mbps Ethernet and an SNMP agent was located on Astrolog connected on the same Ethernet. The response times for sending SNMP queries from the manager to the SNMP agent at Astrolog and obtaining the response is compared with the time taken for sending an agent, obtaining the results locally at Astrolog and the receipt of the agent at Cezembre in Figure 6. The bandwidth utilization is also shown in Figure 7. In this case values of 5 SNMP variables were simultaneously obtained from the SNMP agent. The bandwidth estimation is done by measuring the packet sizes of SNMP get request and the packet sizes of the response which were found to be 40 bytes and 31 bytes respectively. While the mobile agent size after initialization was measured to be 10 K bytes.

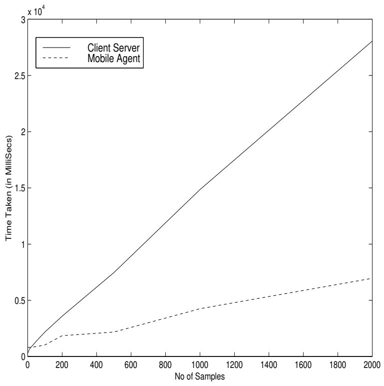

Figure 6:Comparison of Response times for varying number of samples of a single SNMP variable from a mobile computer

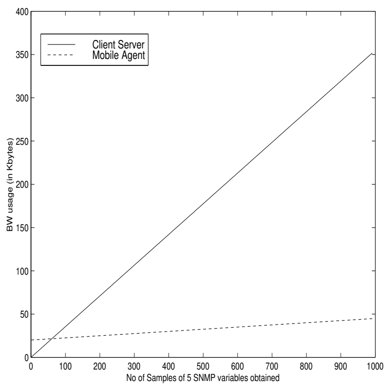

Figure 7:Comparison of bandwidth utilization while obtaining samples of 5 SNMP variables

As shown in Figure 6 the response time of mobile agent approach is better except for collection of few samples. In the case of bandwidth utilization as shown in Figure 7 as values of 5 SNMP variables are obtained from the SNMP agent the threshold is much lower in this case as the data obtained is much larger compared to scenario 1. The client server approach thus utilizes less of bandwidth upto 60 samples each of 5 SNMP variables from the SNMP agent. The mobile agent approach performs better after the threshold. The results thus are similar in nature irrespective of the location of the manager.

7 Conclusion

In this paper we have presented the design, implementation and quantitative evaluation of our approach. Our approach combines two seemingly different approaches to network management drawing on strengths of these approaches and benefiting from them. We also utilize the latest technology of mobile agents and Java to improve upon the prevalent centralized and rigidly defined network management systems. The mobile agents are used to perform certain basic network management functionalities and to supplement the existing network management functionalities. In basic network management functionality implementations the mobile agents are found to outperform the client-server approach in terms of bandwidth usage and response times beyond a threshold.

Acknowledgments

We would like to thank Dr. Michel Banatre for his encouragement and support to the project. We would also like to thank Tiziana Signorelli and Janusz Wisniewski for their contributions to the implementation of the environment. Acknowledgments are also due to Stéphane Billiart and Serge Lassabe for their suggestions and feedbacks.

References

1. Goldzmith G. and Yemini, Y. Decentralizing Control and Intelligence in Network Management. *In the Proc. of 4th International Symposium on Integrated Network Management, Santa Barbara, CA, USA, 1995.*
2. Java Management API (JMAPI). *Java Management Home page at Sun Microsystems*
http://java.sun.com/products/JavaManagement/index.html.
3. AdventNet Monitor. *Fabricated by Advent Network Management Inc. http://www.adventnet.com/.*
4. T. Magedanz, K. Rothermel, S. Krause. Intelligent Agents: An Emerging Technology for Next Generation Telecommunications? *INFOCOM' 96, USA, March 24-28, 1996*
5. Wooldridge M. and Jennings R. N. Agent Theories, Architectures, and Languages: A Survey. *Springer-Verlag Lecture Notes in Artificial Intelligence 890, pp 1-39*
6. Sahai A., Morin C. Towards Distributed and Dynamic Network Management *In the proceedings of IEEE/IFIP Network Operations and Management Symposium (NOMS), New Orleans, USA, Feb 15-20, 1998.*
7. W. Stallings. SNMP, SNMPv2 and CMIP: The practical guide to network management standards. *Addison-wesley publication, 1994.*
8. Mario Baldi, Silvano Gai, and Gian Pietro Picco. Exploiting Code Mobility in Decentralized and Flexible Network Management. *In the proceedings of Mobile Agents 97 (MA97)*
9. Product Tengah at Weblogic. *http://www.weblogic.com/*
10. Product jsnmp at OutBack Resource Group, Inc. *http://www.outbackinc.com/.*
11. Webbin CMIP at IBM *http://www.alphaWorks.ibm.com/.*