APPLYING WEB SERVICES WITH MOBILE AGENTS FOR COMPUTER NETWORK MANAGEMENT

Mydhili K.Nair[1] and V.Gopalakrishna[2]

[1] Department of Information Science and Engineering, M S Ramaiah Institute of Technology, Bangalore, India
mydhili.nair@gmail.com
[2] Integra Microsystems, Bangalore, India
v.gopi@integramicro.com

ABSTRACT

The exponential rise in complexity of the underlying network elements of a computer network makes its Management an intricate, multifaceted and complex problem to solve. With every passing decade, new technologies are developed to ease this problem of Network Management. The last decade of the pre-millennium era saw the peak of CORBA and Mobile Agent Based implementations, while the first decade of post millennium saw the emergence of Web Services. All of these technologies evolved as independent, self-contained implementation streams. There is a genuine dearth in finding authentic research outcomes where quantifiable, measureable benefits of convergence of these technologies applied to Network Management are put forth. This paper aims to fill this research gap. Here we put forth the experimental results obtained of a framework we developed in-house for Network Management that combined two seemingly divergent distributed computing technologies, namely, Web Services and Mobile Agents.

KEYWORDS

Web Services, Mobile Agents, Network Management, SNMP, Aglets, SOA.

1. INTRODUCTION

1.1. Motivating Factors for this Work

Computer Network Management is traditionally done by using a centralized ***NMS (Network Manager System)***.It constantly checks for ***F**ault, **C**onfiguration, **A**ccounting, **P**erformance* and ***S**ecurity*, called ***'FCAPS'*** monitoring, defined by ISO-OSI as the five pillars in the framework of Network Management [3]. ***SNMP*** (***S**imple **N**etwork **M**anagement **P**rotocol*) is the most popular protocol used in such a centralized NMS. It uses polling between the Manager and Managed nodes. It is a Client-Server approach where each poll is actually a ***RPC*** *(**R**emote **P**rocedure **C**all)* from the Manager Node to the remote SNMP Agent at the Managed Node.

Way back in 1996, James White [2] introduced *MA (Mobile Agents)* as a strategy for distributed applications. *MA* is a program, which migrates from host to host in a computer network. The interest in *MA* as a design paradigm for NMS seems to have dwindled over the past decade [1], with the number of research groups working on MA related research topics becoming smaller. There are a number of reasons [5] for this apparent decline in MA research activity, the most prominent being security concerns. As a stand-alone technology, MA seems to have lost out due to lack of trustworthiness, caused due to many NMS approaches that require *'Resident Agents'*[6] [7][8]which stay permanently at the Managed Node. In this paper, we address this concern in our framework by using a strategy called *'Do 'n' Die'* that *eliminates* the need for resident agents. Thus, the **focus** of this research work is to effectively project the need to revive the need to employ Intelligent Mobile Agent technology for managing distributed networks.

Today ***SOA(Service Oriented Architecture)*** is a fast emerging successor of the Object Oriented and Distributed Object Oriented paradigms. ***WS(Web Services)*** Technology, is an implementation of the SOA Model[9]. Since *'The Web'* today is omnipresent, '*Web Services*', which are *Services* offered on the *Web*, enable ubiquitous as well as distributed processing.

Both *Mobile Agents* and *Web Services* are Distributed Computing paradigms, and are well suited for a domain such as ***NM** (Network Management)* which is innately distributed in nature. Though there is a lot of published research material available, where Web Services and Mobile Agents are used independently for NM, there is very few research outcomes published which focuses on the ***convergence*** of these two technologies. There is a *scarcity* in published research work, where quantifiable, measureable benefits of this convergence, when applied to Network Management are shown. This paper, therefore *aims to fill that void*. Here, we propose a framework, where Mobile Agents are applied for Web Services Based Network Management.

1.2 Paper Contributions

This paper presents the architecture details of a framework we have developed called ***'Net Patrol'*** that uses an ***AWS**(Agent based Web Services)* Oriented technique for Network Management. Experiments were conducted to measure the **Response Time** and **Size of Data** of ***'Net Patrol',*** while using WSes with MAs as well as SNMP. The ***HF**(Health Functions)* computed were **Throughput** and **Bandwidth Utilization**. For all the *HFs*, we measured the time it took for *'Net Patrol's* WSes with plain SNMP and that with MA to retrieve the desired result. With these quantitative measurements, we argue that MA is a technology, which can be seamlessly integrated with Web Services and employed for Network Management. We propose that it needs to get back its deserved place as a technology which augments remote Network Monitoring and Management, thus enabling pervasive and distributed computing.

1.3 Paper Overview

There are a total of *10 Major Sections* in this paper, including this **Introduction** Section. The **Section 2** titled **A Journey into the NM Approaches of the past two decades**, shows the evolution of technologies for Network Management in the last 20 years. This is followed by the next *five* major parts of this paper that summarizes the design, implementation and results of ***'Net Patrol'*** in the Sections called **Architecture of 'Net Patrol', Framework Realization Phases, Tools Used, Experiments Performed, Results and Analysis.** These are followed by the **Related Work** Section that provide a comprehensive survey of the existing published research outcomes in the field of NMS since 1995, covering the *Mobile Agent* and *Web Services Based* approaches. This is followed by the **Conclusion, Future Work** Sections that summarizes the research contributions of ***'Net Patrol'*** and gives an insight into our research directions, respectively. Lastly, the **Acknowledgement** Section, gives an insight into the *'Net Patrol'* prototype demonstrations so far, thanking the people and organizations who have helped us.

2. A Journey Into The NM Approaches Of the Past Two Decades

This Section looks at the evolution of Network Management System(NMS) approaches, frameworks and technologies over the last 20 years. **Figure 1** below depicts the full taxonomy.

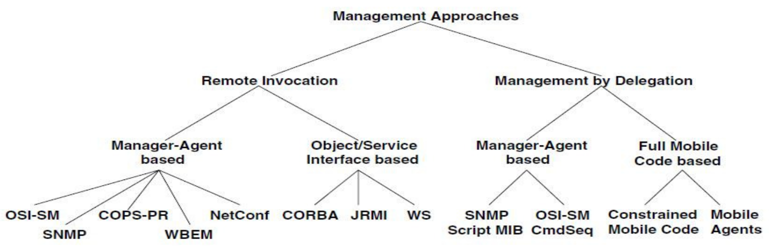


Figure 1. Network Management Approaches, Frameworks and Protocols[24]

2.1. Remote Invocation Approaches

There exist two different ways of performing remote invocations, namely:

a) Performing the invocations on managed objects via an Agent: *Manager–Agent Model*

b) Performing the invocations directly to the managed objects through *Distributed Object or Service Interface Model.*

The taxonomy for the Manager–Agent Model covers OSI-SM, SNMP, COPS-PR, WBEM and NetConf, while that of the Distributed Object/Service interface model include CORBA, JRMI and Web Services. The latter assumes that individual managed objects are modelled as distributed objects or service interfaces. The *Manager-Agent Remote Invocation* stream has:

- ***OSI-SM*** *(OSI System Management)*: This is a standardized solution that were deployed and used in the real world in telecommunications environments.
- ***COPS-PR*** (*Common Open Policy Service* for ***PR***ovisioning): This was conceived as an approach supporting configuration changes, the Achilles' heel of SNMP. However, it did not succeed as it had a rudimentary information model.
- ***WBEM***(*Web Based Enterprise Management*): This was the first XML-Web Based approach proposed by *DMTF(Distributed Management Task Force)*whose target applications were system management for desktop computers. Though it did quite well in that space, it failed in its adaptation for management over the HTTP Protocol.
- ***NetConf***(***Net***work ***Conf***iguration Protocol): The focus of this protocol was to overcome SNMP's shortcomings for configuration management, namely transaction support and security. It is evolved from Juniper's Junoscript and is used even today in its niche space.

2.1.1 Brief Over view of SNMP

Traditionally, to do Network Management, there is a centralized node (Manager) which polls all the network nodes under its administration. This polling is done as a Remote Invocation, through an *RPC(Remote Procedure Call)* by sending *SNMP commands* to *SNMP Agents* present at the Managed Nodes. The *SNMP Agents* in turn queries the ***MIB****(****M****anagement* ***I****nformation* ***Base****),* a *repository* containing the *state of the network parameters* like the Number of Incoming, Outgoing and Discarded Packets at the ***NIC*** *(****N****etwork* ***I****nterface* ***C****ard)* etc, as depicted in **Figure 2**.

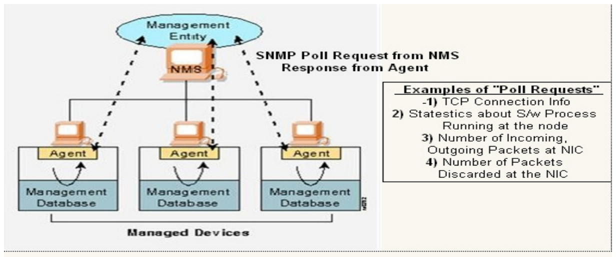

Management Entity
NMS
SNMP Poll Request from NMS
Response from Agent
Examples of "Poll Requests"
-1) TCP Connection Info
2) Statestics about S/w Process Running at the node
3) Number of Incoming, Outgoing Packets at NIC
4) Number of Packets Discarded at the NIC
Agent
Agent
Agent
Management Database
Management Database
Management Database
Managed Devices

Figure 2. Working of SNMP

2.1.2 Brief Overview of Web Services

SOA is popular acronym for *Service Oriented Architecture*. It is an architectural style, with the goal to achieve loose coupling among interacting software entities. The building blocks of SOA are 'Services'. When these services are deployed on the internet, they are called '***Web Services***'. Thus, *'Web Services(WS)'* are software applications accessible through a URL[10]. WS Clients who need to execute them, contacts the WS using XML-based protocols such as ***SOAP(****Simple Object Access Protocol)* which run over IP-based protocols like HTTP. Clients access a WS through its interface and bindings which are which are defined using an XML-format, called ***WSDL(****Web Services Definition Language).*

2.2. Management by Delegation Approaches

In ***MbD*** *(Management by Delegation)*, the simplest case is for logic to be uploaded to a managed device in order to operate close to its managed objects. This ***"One-Hop" Mobility*** can be supported by special ***MIB(Management Information Base)*** in the ***Manager–Agent Model*** and relevant standardization work has resulted in the *SNMP Script MIB* and the *OSI-SM Command Sequencer*. In the more general case, logic can move from device to device, through a predefined itinerary and this approach is known as ***Constrained Mobility***. It can also be done by autonomously sensing its environment, and this is the ***Full Mobile Agent Approach***. Thus, concisely, we can categorize Management by Delegation into One-hop Manager–Agent based approaches and Multiple hops Mobile Code based ones with Constrained and Full Mobility.

2.2.1 Brief Overview of Mobile Agents

Code Mobility is achieved by three approaches [25], namely,:

- Mobile Agents(MA)
- Remote Evaluation(REV)
- Code On Demand (COD)

In the ***Mobile Agent (MA)*** paradigm, a *Station* ***A***, the client, requires access to the resources that reside on a set of stations $\{B_1, B_2,...,B_j\}$. In order to complete the service, an MA is initiated on *Station* ***A*** which migrates to the j^{th} *server*, carrying its code, state, and data. Here the MA executes its tasks to completion and returns the results to *Station* ***A*** further processing.

The ***REV*** paradigm assumes that the *Station* ***A***, in order to run to completion, must access the resources on a *Host* ***B***. But, instead of migrating an agent to the server, *Station* ***A*** simply transmits the code, the set of instructions to perform an operation, and possible initial data. In REV, the notion of itinerary paths does not exist; intermediate results must be transmitted back to *Station* ***A*** before a decision for the next hop is made. A mobile agent α_i with an itinerary path $I_i = \{ N_0, B, N_0 \}$ can describe the *REV* paradigm.

In the ***COD*** paradigm, *Host* ***B*** wants to perform an operation but it does not know how. Thus, it contacts *Station* ***A*** and requests the associated code to be transmitted. While in the *MA* and *REV* paradigms, an operation is triggered by an entity external to the host, that is the *Station* ***A***, in *COD,* the operation is triggered by the server itself. *Station* ***A*** can be seen as a code repository. The COD paradigm can be encapsulated in the MA paradigm. For example, agent α_i is launched from *Station* ***A***, carrying minimal functionality, to visit a *set* ***B*** *of hosts*. Following its itinerary path, an event is triggered at *Host* B_k. The agent can either record the event and continue its route, or request the necessary code to handle the event.

The lifecycle of a MA includes creation, initialization, migration, monitoring, deletion, communication with other agents and termination. A *Mobile Agent System (MAS)* must be able to support all the above functionalities. A *Mobile Agent* is a software process that can autonomously migrate to another host for execution. The autonomy of MA is constrained by an itinerary path. *Itinerary* agents are assigned a routing schedule of the destination hosts they must visit before they return to their owner. An agent can also create its own itinerary path during execution. An agent can access a set of resources R that reside on a remote host. MAs are able to communicate with other agents that reside on the same host. If an agent resides on a different host, communication is established either by exchanging messages, or by using special purpose agents. The properties of a MA[25] are graphically illustrated in **Figure 3**.

MA α is sent for execution to host B_i where it interacts locally with another agent. Following its itinerary path, MA α migrates to host B_j where it exchanges messages with an agent that resides on platform C. Agent α completes its lifecycle by returning to its creation platform.

An agent α_i can be described by the triplet $\alpha_i = \{ C_i^{MA}, d_i, I_i \}$ where **[25]**

- C_i^{MA} is the binary code of the mobile agent, including its state
- d_i is the data (initial data or intermediate results) during its lifecycle, carried by the agent

- $I_i = \{ N_0, N_1, ..., N_k, N_0 \}$ is the itinerary path of the agent among a set of ***K*** hosts, starting and finishing at station N_0.

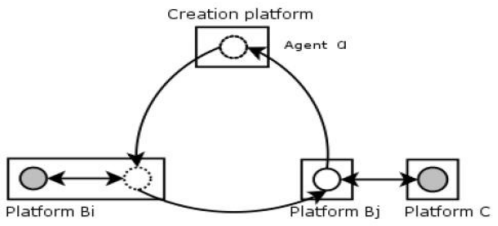


Figure 3. Properties of a MA (Mobile Agent) [25]

In our framework, ***'Net Patrol'***, we have used the *Management by Delegation, Constrained Mobile Code Approach* shown in Figure 1, essentially, the *REV* paradigm described above, with the itinerary path as $I_i = \{ N_0, B, N_0 \}$ where ***B*** is the host visited and N_0 the Manager Node.

3. Architecture of *'Net Patrol'*: Our Framework

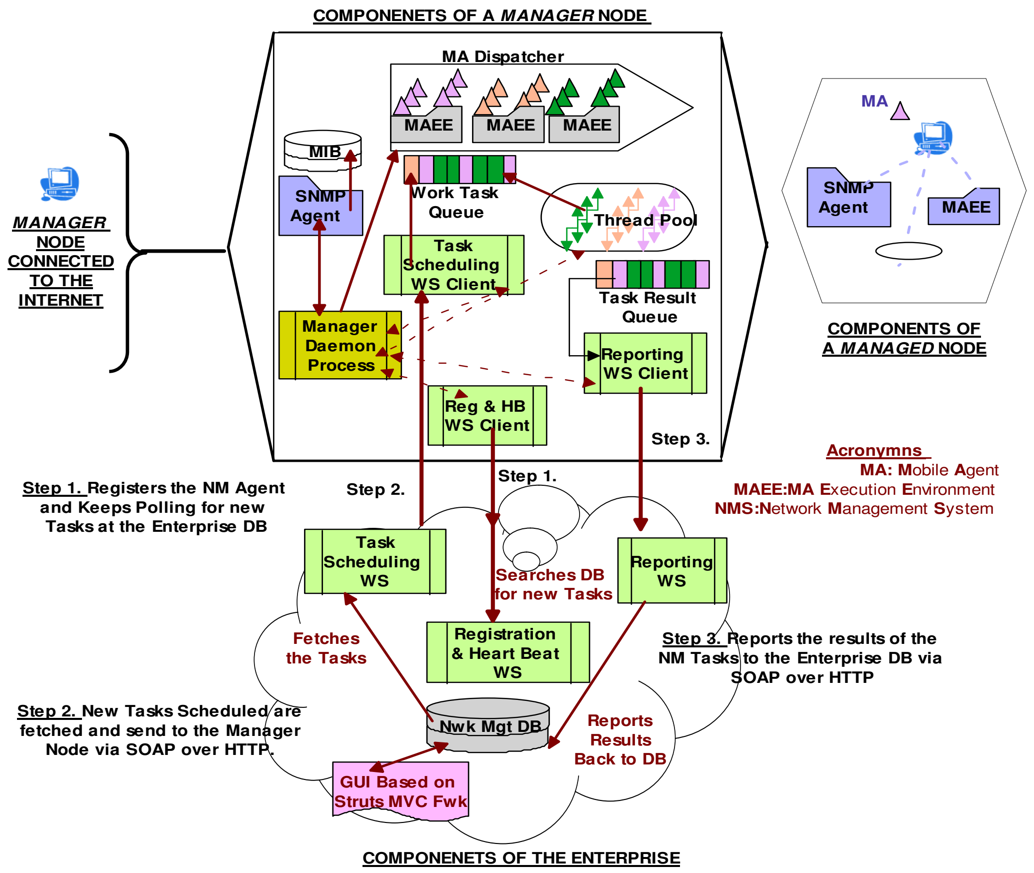

COMPONENETS OF A MANAGER NODE
MA Dispatcher
MAEE
MAEE
MAEE
MIB
SNMP Agent
Work Task Queue
Thread Pool
Task Scheduling WS Client
Task Result Queue
Manager Daemon Process
Reporting WS Client
Reg & HB WS Client
MANAGER NODE CONNECTED TO THE INTERNET
MA
SNMP Agent
MAEE
COMPONENTS OF A MANAGED NODE
Step 3.
Step 2.
Step 1.
Acronymns
MA: Mobile Agent
MAEE:MA Execution Environment
NMS:Network Management System
Step 1. Registers the NM Agent and Keeps Polling for new Tasks at the Enterprise DB
Task Scheduling WS
Searches DB for new Tasks
Reporting WS
Fetches the Tasks
Registration & Heart Beat WS
Step 3. Reports the results of the NM Tasks to the Enterprise DB via SOAP over HTTP
Step 2. New Tasks Scheduled are fetched and send to the Manager Node via SOAP over HTTP
Nwk Mgt DB
Reports Results Back to DB
GUI Based on Struts MVC Fwk
COMPONENETS OF THE ENTERPRISE

Figure 4. *'Net Patrol'* Architecture

3.1. Why the name – 'Net Patrol'?

As mentioned in the Introduction Section, our research work aims to fill the gap, caused due to paucity of research outcomes where convergence of WSes with MAs and SNMP is used. The inspiration for this name came from a related work done, way back in 1995, by Zapf et al [15], which they call *'Net Doctor'*, that uses MA and SNMP to manage networks. In this context, we gave the name ***'Net Patrol'*** to our framework, because, the Mobile Agents ***'patrols' / 'tours'*** the network by *migrating* to the network nodes scrutinizing its *Memory*, *CPU, Bandwidth Utilizations etc*. The network *Throughput* is watched and prospective *congestion* segments are tracked and reported by overseeing the *Packet Discard Rate* at the NIC.

As shown in **Figure 4** above, the framework broadly consists of three components namely:

a) The Enterprise
b) The Manager Node
c) The Managed Node

3.2. Manager Node

As shown in **Figure 5**, the Manager Node of 'Net Patrol' framework has a plethora of components working together. They are:

- A Daemon Process: It brings up the Agent.
- *MAEE* (*M*obile *A*gent *E*xecution *E*nvironment): *'Net Patrol'* used ***Aglets*** Framework.
- SNMP Agent: *'Net Patrol'* uses ***AdventNet*** SNMP API.
- *MIB* (*M*anagement *I*nformation *B*ase): Maintains state of network parameters of itself.
- Three SOAP Based WS Clients: Vital for Enterprise-Agent SOAP message exchange.
- A Pool of Threads: Made ready to execute the scheduled NM tasks, simultaneously.
- Result Queue: Maintains the task execution results until fetched by the Reporting WS
- Set of library files developed by us to ease manager & managed node communication.

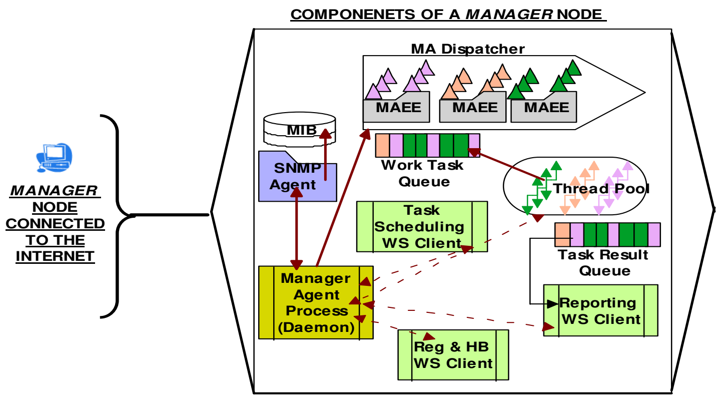
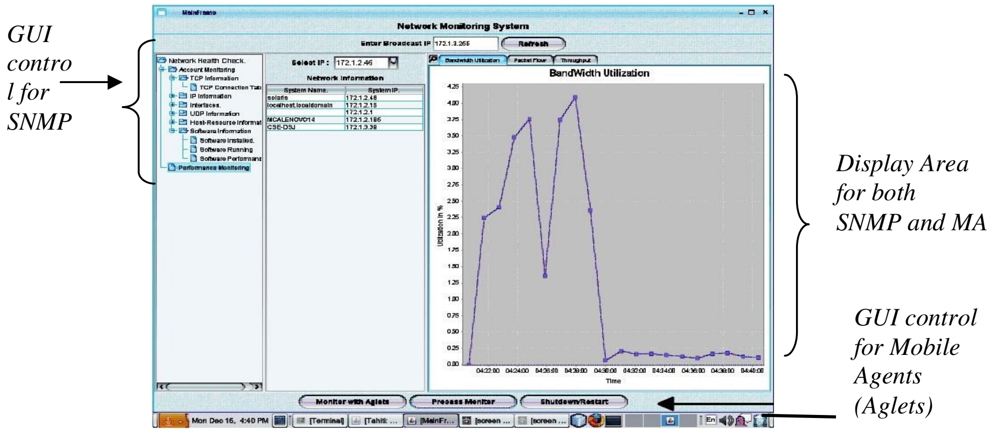

COMPONENETS OF A MANAGER NODE
MANAGER NODE CONNECTED TO THE INTERNET
MA Dispatcher
MAEE
MAEE
MAEE
MIB
SNMP Agent
Work Task Queue
Thread Pool
Task Scheduling WS Client
Task Result Queue
Manager Agent Process (Daemon)
Reporting WS Client
Reg & HB WS Client

Figure 5 Components of a Manager Node

As depicted in **Figure 6**, the framework *'Hybrid'* where the Sys-Admin has the option to choose either SNMP RPC Client-Server Polling mechanism or decentralized MA based NM.



System Name.	System IP.
solaris	172.1.2.46
localhost.localdomain	172.1.2.15
	172.1.2.1
MCALENOVO14	172.1.2.186
CSE-DSJ	172.1.3.39

Figure 6. Screenshot of *'Net Patrol'* showing the Hybrid GUI for Network Admin

As depicted in **Figure 5,** , the Manager creates *a pool MA ready for dispatch.* The Sys-Admin is given a GUI, having *hybrid controls* of using traditional SNMP and MA to do the NM tasks. In ***'Net Patrol'***, we have created *three types* of MAs, colour-coded as ***green***, ***orange*** and ***purple***, doing *three Network Management / Monitoring* activity, mentioned below:

- Account Monitoring Mobile Agents(▲): These MA monitor parameters like Software Processes installed and running at the node, IP and TCP Connection Information etc.
- Performance Monitoring Mobile Agents(▲): These MA gathers information at the NIC of the node, calculating the ***Health Functions***. In '***Net Patrol',*** we have calculated *Bandwidth Utilized, Packet Discard Rate* and *Throughput*.
- Root Cause Analysis Mobile Agents(▲): This is a set of MA dispatched to a node, only if the above two MA detects a potential problem at the node. The RCA MA is programmed to reside at the node and find out the cause / reason of the potential problem.

The set of library files to manage the network adheres to *FCAPS*[3] principles, namely *Fault*, *Configuration, Accounting, Performance* and *Security*. The Agent, acts as a 'glue'[12], interfacing with the NM Application and with the Enterprise Web Services. It is deployed on the Manager Node of the LAN monitored by the NM application. The Agent configured as a Daemon Process, is automatically started when this node boots up. It also has the Web Service Client components of each of the Enterprise WS, described in **Section 3.3**. The tasks to be executed are placed in the Work Queue of the Agent, which is implemented as a Thread-Pool Work Queue. This multi-threaded Agent execution ensures that many tasks can be concurrently executed. The Agent is described in detail in our work[12].

3.3. Enterprise

'Net Patrol' uses a *Service Oriented Enterprise*, with the Web Services deployed on the Internet, thus enabling pervasive monitoring and management of the computer network. As depicted in **Figure 7**, the framework has three Web Services, namely

a) Registration and Licensing Web Service
b) Task Scheduling Web Service
c) Result Reporting Web Service

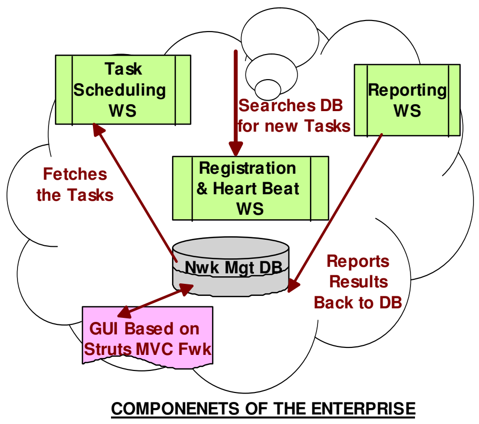

Task Scheduling WS
Searches DB for new Tasks
Reporting WS
Fetches the Tasks
Registration & Heart Beat WS
Nwk Mgt DB
Reports Results Back to DB
GUI Based on Struts MVC Fwk
COMPONENETS OF THE ENTERPRISE

Figure 7: Components of the Enterprise where the NM Web Services are deployed

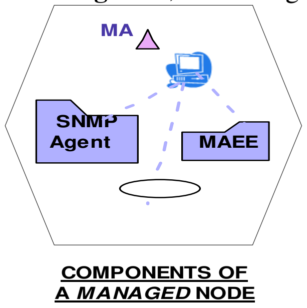
The first thing the Agent Daemon process at the Manager Node does, when it starts up, is to invoke the *Enterprise Registration and Licensing WS*, where it registers itself with a Network Monitoring ID, which is unique to the entire enterprise database and has an Encrypted License Key. It is imperative to complete this Registration Process in order to ensure that the Enterprise recognizes it as a valid Agent. Agent validity is determined by checking the application's contract expiry date with the Enterprise. If the contract has expired, the agent brings itself down, making the entire system un-usable, until a valid license is got.

We have designed the Enterprise User Interface so that the same can be deployed on a desktop web browser or a mobile device web browser. This is done, so that the Enterprise User who in this case is the network-admin, can access the backend NM application ubiquitously. To provide a unified view of the NM nodes and servers we provide an interface to the network-admin to schedule the routine NM tasks for the nodes of the LAN he is managing. Some of the typical tasks are Bandwidth Utilization, Throughput, Process Information, Memory Utilization etc.

The Agent constantly polls the *Enterprise Task Scheduler Web Service*, to check if there are any NM that are assigned to it. The polling interval of the Agent is configurable. If there are such tasks, they are fetched by the Task Scheduler WS and using our WS execution framework these tasks are sent to the Task Scheduler WS Client integrated with the Agent. The results of these tasks are sent back to the Enterprise through the *Enterprise Result Reporting Web Service*.

3.4. Managed Node

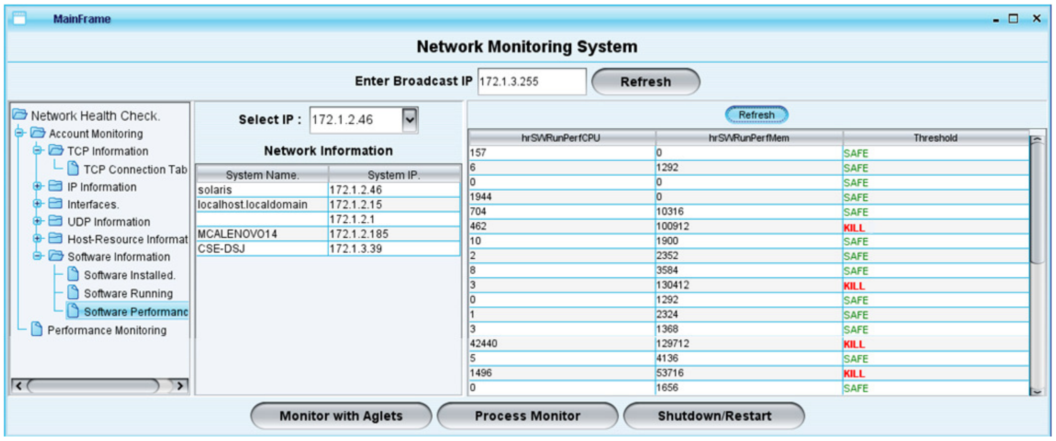
As in **Figure 8**, the Managed Node of ***'Net Patrol'*** has the following components installed in it:



- *MAEE* (*M*obile *A*gent *E*xecution *E*nvironment): *'Net Patrol'* used ***Aglets*** Framework.
- SNMP Agent: *'Net Patrol'* uses ***AdventNet*** SNMP API.
- *MIB* (*M*anagement *I*nformation *B*ase): Maintains state of network parameters of itself.
- Set of library files developed by us to ease manager & managed node communication.

Figure 8. Managed Node

The Account and Performance Monitoring MAs, described in **Section 3.2**, gather the data at a node and return to the Manager Node at the end of its polling duration. The Software Agent at the Manager Node *analyses* the data collected, before putting the result objects into the Result Reporting WS Queue. While analyzing, if it senses any potential problem at the node, the Manager Node dispatches the RCA MA to the Managed Node.



System Name.	System IP.
solaris	172.1.2.46
localhost.localdomain	172.1.2.15
	172.1.2.1
MCALENOVO14	172.1.2.185
CSE-DSJ	172.1.3.39

hrSWRunPerfCPU	hrSWRunPerfMem	Threshold
157	0	SAFE
6	1292	SAFE
0	0	SAFE
1944	0	SAFE
704	10316	SAFE
462	100912	KILL
10	1900	SAFE
2	2352	SAFE
8	3584	SAFE
3	130412	KILL
0	1292	SAFE
1	2324	SAFE
3	1368	SAFE
42440	129712	KILL
5	4136	SAFE
1496	53716	KILL
0	1656	SAFE

Figure 9 : Result of Root Cause Analysis and the resultant action taken

E.g. 1: If the analysis of Account Monitoring information gathered shows that a node is "sluggish", unable to execute the processes in it as expected, the RCA MA is programmed to check the process running table to check the use of 'torrents', audio and video downloads etc. If so, the MA goes ahead and kills those processes. Please refer **Figure 9** for the results. Here the threshold for memory utilization is set as 0.1MB. Anything above this value will be killed by the RCA MA (marked in ***red***). This is the default action of ***'Net Patrol'***.

E.g. 2: If the *Packet Discard Rate* at a node reveals that large number of packets are being lost, then the RCA MA is programmed to check the reason. It could be that the NIC is unable to accept any more packets due to heavy downloads or could even be that the NIC is faulty.

Thus, in our framework, the allotment and execution of tasks are done *asynchronously*. The results of the tasks scheduled are not visible by the network-admin immediately and are seen in the subsequent polling cycles of the Agent. This is the design principle we adopted to ensure that the framework is stable, with maximum uptime.

4. 'NET PATROL' FRAMEWORK REALIZATION PHASES

As with any framework that needs to be developed from scratch, the realization of '***Net Patrol'*** hybrid Network Management System was divided into *four phases*:

a) *Analysis Phase:* The feasibility of adopting an XML-Based WSes approach for NM was studied. The first reference of the need to use this approach is seen in the Minutes of Meeting of the 11th ***NMRG*** *(Network Management Research Group)*[16] meeting in 2002 after which there are several such references to the ***benefits*** *of using XML WS Based NMS* technique until the 2004 17th Minutes of Meeting[17][18][19]. The **Related Work Section**, presents a gist of this literature survey, where majority of research work used WSes in SNMP RPC-style implementations. To the best of our knowledge, the ***convergence*** of employing Mobile Agents along with SNMP in WS Based NM was not well documented. Thus, we set about designing ***'Net Patrol'*** as a first step to fill this research gap.

b) *Design Phase:* The initial design was made for creating a hybrid framework that used ***both*** *SNMP and MA*. The next step was to extend this for execution over the Internet via WSes. For this, we employed the *Enterprise-Agent Design Paradigm,* specifically, *Agent Based WSes*[9][20][21]. The Software Agent acted as a *'glue'* [12], interfacing with the *backend hybrid NMS* and the SOAP based *Web Service Clients*.

c) *Implementation Phase:* The biggest dilemma was to select the tools for implementing this slightly complex ***'Net Patrol'*** design, having a plethora of components. A thorough survey was done to study the various SNMP open-source packages, MA Frameworks, Network Packet Analyzers, MIB Browsers, WS Toolkits etc. In traditional SNMP, the ***ifTable****(Interface Table)*is the most frequently used table for gathering the NM parameters. Therefore, the XML Based SOAP Messages to carry this monitoring data was modelled.

d) *Testing Phase:* The performance of the backend hybrid NMS application comparing the usage of SNMP RPC-style polling and dispatching of MA was tested. This gave us insight into when to use SNMP and when to use MA. *Wireshark* was used to analyze the packets and *Netbeans Profiler* was used to probe into the CPU and Memory utilization at the node being monitored. The result of Testing Phase is detailed in the **RESULTS ANALYSIS.**

The research outcomes of completing each phase of realizing ***'Net Patrol'*** is published in a series of work [11][12][13][14] by the authors of this paper.

5. TOOLS USED

5.1. SNMP Tools

- ***Wireshark:*** This is a cross-platform free and open-source *packet analyzer*. It uses ***GTK+*** *widget toolkit* to implement its user interface, and ***pcap*** to capture packets. In '***Net Patrol'***, we have used it for network troubleshooting and packet data analysis.
- ***MIB Browser***: This is an auxiliary application for HostMonitor(NM for small enterprises). In '***NetPatrol'***, we have used Ver 1.14, downloaded separately. We can *view hierarchy of SNMP MIB variables* in the form of a tree and provides additional information about each node.

- ***AdventNet:*** This is a *Java SNMP library* that is a comprehensive toolkit for SNMP-based NMS, that needs to track network elements. It facilitates SNMP trap, table views and table handling. In '***Net Patrol'***, we have used AdventNet SNMP API Edition 4.
- ***Snmpd***: This is a *daemon* to respond to SNMP request packets. It is an SNMP agent that binds to a port and awaits requests from SNMP management software. Upon receiving a request, it processes the request(s), collects the requested information and/or performs the requested operation(s) and returns the information to the sender.

5.2. Mobile Agent Tools

- ***Aglets Framework***: In '***Net Patrol'***, the ***MAEE(Mobile Agent Execution Environment)*** was created using ***ASDK** 2.0.2(Aglets **S**oftware **D**evelopment **K**it).*
- ***Tahiti Server***: This is an application program that runs an MA Server. Tahiti provides a UI for monitoring, creating, dispatching, and disposing mobile agents as well as setting the access privileges for the agent server. This application works best with Aglets MA.

5.3. Development Tools

- ***Netbeans IDE Ver 6.5:*** This is an editor supporting a variety of languages like Java, C/C++, Ruby, XML, PHP, JSP etc. We have used it as an IDE with inherent Javadoc support, to develop ***'Net Patrol'***, that is primarily a Java application. We created the XML based WSDLs which was given to the JAX–WS SOAP Based WSes framework.
- ***Netbeans 6.5 Remote Profiler :*** This is an application running on the remote system to profile the CPU and memory usage of that system. In '***Net Patrol'***, we installed the Netbeans Profiler's remote pack on the managed node that needs to be profiled.
- ***Tomcat Web Server:*** This is an open source web server from Apache. In '***Net Patrol'***, we have used Tomcat Version 6.0.20, instead of *GlassFish*, which is the default implementation for Netbeans 6.5 because it has stability issues.
- ***Googlechartwrapper:*** This is an open-source Java Library for Google Charts API. It acts as a 'wrapper', that is, creates the URL request for the precise Google Charts API, at the same time hiding the details of the required URL parameters.
- ***JFreechart 1.0.9:*** This is an open-source Java chart library. In '***Net Patrol'***, we used this to create off-line charts(without connecting to internet).

5.4. Design Tools

- ***MySQL Workbench 5.2:*** This is a visual tool which we used for data modelling.
- ***Star UML 5.0:*** An Open-source UML Modelling tool for the *Windows platform.*
- ***Umbrello2.0 UML Modeler:*** UML Modelling tool for the Linux KDE platform(*KDE 4.0.0.).* In '***Net Patrol'***, we used it because interaction diagrams were not available in Star UML.

5.5. Database Tools

- ***MySQL:*** This is a RDBMS (Relational Data Base Management System) that runs as a server providing multi-user access to number of databases.
- ***phpMyAdmin***: This is an open source tool written in PHP which handles the administration of MySQL. In '***Net Patrol'***, we have used it to perform various tasks such as table manipulations as well as executing SQL statements and managing users and permissions.

5.6. Testing Platform

We have tested '***Net Patrol'*** in 10/100 Mbps Ethernet network, on,

- SUN Blade 4000 series machines running UNIX Solaris 10 version
- Intel X86 2.4 GHz, 512 MB RAM machines having Windows and Linux OS
- Number of Managed Nodes is 12.

6. Experiments Performed

6.1 Modus Operandi of *'Net Patrol'*

The focus of our *'Net Patrol'* Hybrid NM Framework is to examine the impact of combining the traditional Client-Server SNMP approach with that of the Remote Programming approach of using autonomous intelligent mobile agents to effectively manage distributed networks.

As shown in **Figure 6** above, the sys-admin is given a GUI to choose between ***two modes*** of execution, namely,

- ***Using SNMP***, we do network account and performance monitoring *on a continuous basis*. The managing node polls all nodes under its purview periodically. This poll RPC call returns raw values to the manager, which has to now, process this data to meaningful information.
- ***Using MAs***, we do performance monitoring of the network on an *on-demand basis*. The manager creates the MAs and keeps them ready for dispatch in an MA pool. The sys-admin dispatches Aglets [21] either to a particular machine or broadcasts to all managed nodes, where the MA creates an SNMP session through AdventNet[22], to talk to the snmpd, which in turn fetches the information from the MIB. This is shown in **Figure 10** below. .

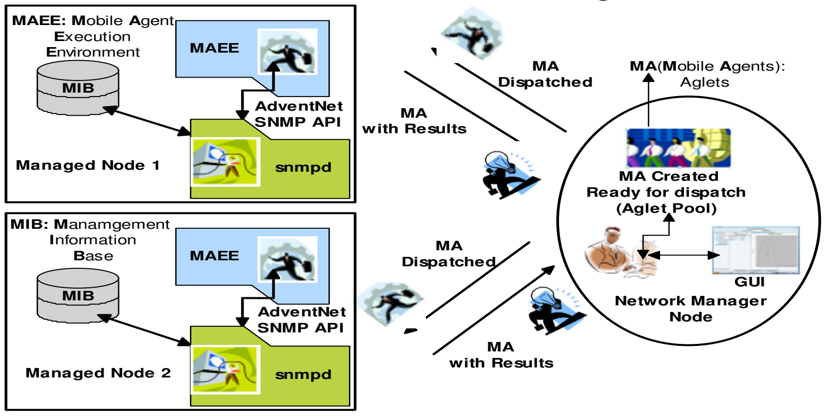


Figure 10. 'Net Patrol' Broadcast Model of Mobile Agents (Aglets)

6.2. Network Health Function Calculation

Network Health is a concept used in network management to indicate the robustness of a network operation. Some of the most popular functions to determine Network Health are

- Bandwidth Utilization
- Network Downtime
- Throughput
- Packet Discard Rate

An aggregation of variables is required to calculate the *Health Function*, which is a cumulative factor, indicating the state or efficiency of a managed node.

In *'Net Patrol'*, we used ***Throughput*** and ***Bandwidth Utilization*** Health Functions. In **Formula 1** and **Formula 2** above, *ifSpeed*, *ifInOctects* and *ifOutOctects* of the interfaces group in MIB are used to compute the percentage utilization at an interface over a time interval. Here, *ifSpeed* is the bandwidth of the interface, $ifInOctects_x$ is the bytes received at time x and $ifOutOctects_x$ is the bytes sent at time x and (y-x) is the polling interval.

$$U(t) = \frac{((\text{ifInOctects}_y - \text{ifInOctects}_x) + (\text{ifOutOctects}_y - \text{ifOutOctects}_x)) \times 8 \times 100}{(y - x) \times \text{ifSpeed}} \quad \text{--- (1)}$$

ifSpeed is the bandwidth of the interface
ifInOctects is the bytes received at time x and y
ifOutOctects is the bytes sent at time x and y
(y-x) is the polling interval.

Formula 1 : Bandwidth Utilization Health Function

$$Th(t) = \frac{((\text{ifOutOctects}/60 + \text{ifInOctects}/60) \times 8 \times 100}{60 \times \text{ifSpeed}} \quad \text{--- (2)}$$

ifSpeed: Bandwidth of the interface; ***ifOutOctects***: The bytes sent; ***ifInOctects:*** The bytes received

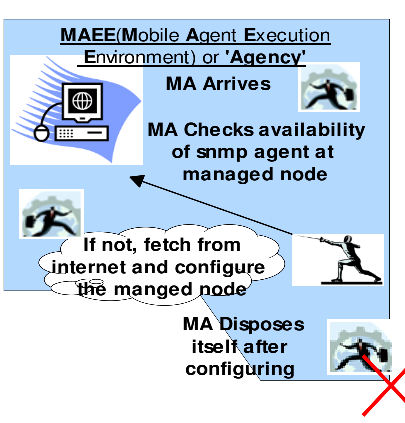
Formula 2: Calculation of Throughput

6.3. Configuring the Network Element

With this experiment, we attempt to take care of the *'C'* part of *FCAPS* Network Management, namely, *Configuration*.

It is essential to have **snmpd** running at the managed nodes for our framework to work. Therefore, we ***configure*** the managed nodes that do not have it, using Aglets [21]. In 'Net Patrol' we use a novel strategy called ***'Do 'n' Die'***. As depicted in **Figure 11**, the Aglets used are intelligent agents which first checks the availability of the SNMP agent at the managed node. If not present, they attempt to configure dynamically. Once they *'Do'* what they are programmed to do, they *'Die'*, that is they *dispose* themselves at the managed node.

The steps for configuration are depicted in **Figure 12**. Firstly, *'Auto-Discovery'* of the nodes managed is done by an ICMP broadcast. Then, the Aglets are dispatched to these nodes. The Aglets checks whether *snmpd* is already installed, but for some reason is not running. If so, it attempts to start it. If it fails to start, it will assume that the snmpd is corrupted and dynamically downloads it from the Internet to install in the default path */etc/snmp*. It will then edit the *snmpd.conf* file to make the *RO Community* for snmpd *'Public*'. This is done in order to ensure that the communication paradigm is set between the manager and its nodes.


MAEE(Mobile Agent Execution Environment) or 'Agency'
MA Arrives
MA Checks availability of snmp agent at managed node
If not, fetch from internet and configure the manged node
MA Disposes itself after configuring

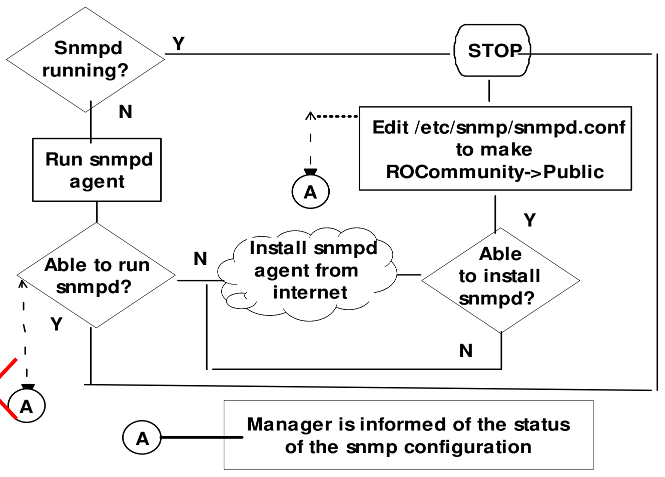

Snmpd running?
Y
STOP
N
Run snmpd agent
Edit /etc/snmp/snmpd.conf to make ROCommunity->Public
A
Able to run snmpd?
N
Install snmpd agent from internet
Y
Able to install snmpd?
N
Y
A
A
Manager is informed of the status of the snmp configuration

Fig 11'Do 'n' Die' by intelligent MA Fig 12. Part of Flow Chart: Dynamic snmp configuration

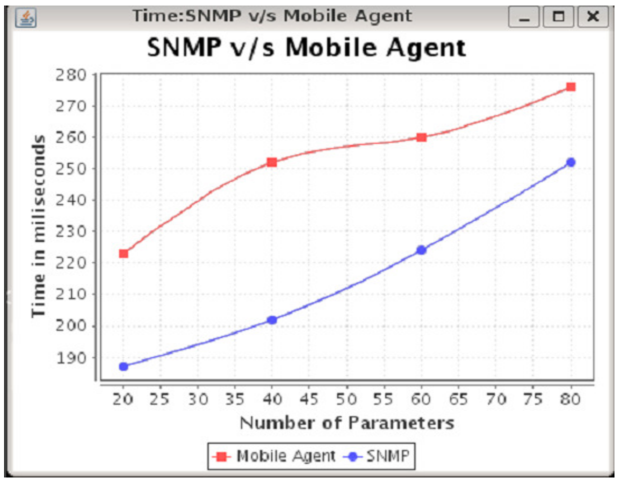
7. Result and Analysis

Here, the Aglets [21] collect and process the data, locally at the managed nodes. Only the calculated values are returned to the manager. Thus, the MAs remove processing load at the manager. In the same context, if conventional SNMP is used, the manager needs to send a *get-request* for *ifInOctects* and *ifOutOctects* at time x and repeat the same for time y and calculate the HF cumulative value. When we have to monitor a large number of machines, this is a huge processing overhead at the manager node.

In *'Net Patrol'* framework, we give the *flexibility of choosing* between SNMP and MA to the sys-admin. Nevertheless, we give him a few recommendations as to when to stick to using SNMP and when to use MA, so he can exploit the full potential of the hybrid nature of this framework imbibing both the strategies.

7.1. Response Time

The **Table** below shows the response times to do *Account Monitoring* operations for 20, 40, 60 and 80. parameters (*OIDs*). The timestamps are calculated using JDK 1.6 *System.currentTimemillis ()*.



Number of Parameters	Response Time	
	SNM	MA (Aglets)
20 Parameters	187	223
40 Parameters	202	252
60 Parameters	224	260
80 Parameters	252	276

Fig. 13. Comparison in Response Time (Milli Seconds) between Mobile Agents and Aglets

In **Figure 13**, we see that ***SNMP wins hand-down*** with respect to response time, because the Aglets require time to migrate to the managed nodes. On an average, it takes 200 milliseconds, while SNMP uses RPC to communicate remotely with the managed nodes, making the response time almost instantaneous.

7.2. Size of Data

Next, we used SNMP and MAs to calculate the cumulative value of the *Health Funtion* described in **Section 5.2**. Table 1, shows the amount of data transferred to and from the managed node, for which we monitor the values of *ifInOctects* and *ifOutOctects* at the NIC. We kept the *Polling Duration* constant for one set of values, changing the *Polling Interval*. The *Polling Duration* is the total duration during which we monitor the managed node while PI (*Polling Interval)* is the interval of time the MA interacts with the node's snmpd.

The experiment was conducted for a Total Duration of 1 hour and 2 hours. The number of parameters (MIB Objects) polled was kept constant as 80. We give below, the interpretation, for a sample set of values, which is the first row of the table below, **Table 1**.

- Polling Duration = 1 hour

- Polling Interval = 10 minutes
- Therefore, total number of polls made by the SNMP manager to the managing node = 6
- Number of remote requests from manager to managed node for SNMP using RPC = 12
- Number of RPC replies for SNMP Poll request from manager to managing node = 12
- Migration time for mobile agents to remote managed nodes = 200 ms(average)
- Number of RPC polls(request/reply) made by manager to managing node using MA = 0
- Number of local polls made by the mobile agent at the managed node = 6

Table 1. Comparison in Amount of Data Transferred over wire between SNMP and Mobile Agents

Polling Duration	**Polling Interval**	**#of Parameters Polled**	**Amount of data over wire (in Kbytes)**	
			SNMP	**Aglets**
1 hour	10(6 polls)	80	87.19	108.35
	5(12 polls)	80	170.29	112
	1(60 polls)	80	871.32	121.8
2 hour	10(12 polls)	80	178.56	112.15
	5(24 polls)	80	362.72	115.2
	1(120 polls)	80	1732.6	119.35

We represent the values of Table 1 graphically, in **Figure 14** below using *JFreeChart.*

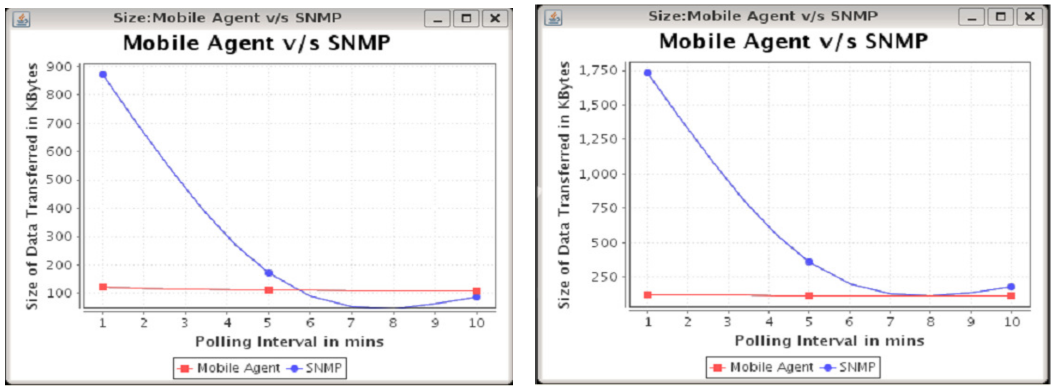


Fig. 14. Comparison of Size of MA and SNMP with Polling Durations of 1 and 2 hours respectively.

From **Table 1** and **Figure 14**, we observe that the amount of data transferred for MA remains *almost constant,* for fast and slow polling intervals. In contrast, when the *Polling Interval* is less, i.e the number of polls is more, the amount of data transferred for *conventional SNMP is more*. This is because, in SNMP, we make *many RPC* calls during the *Polling Duration* to fetch the data remotely. In SNMPv1, to fetch each parameter, we need to make a *get-request* with its *OID*. Even if we use *get-bulk* of SNMPv2C, the size of the SNMP PDU may exceed beyond the threshold value of *64 KB* which is usually set for a 10/100 Mbps Ethernet LAN.

Thus, in this case, the MA wins! This is because the MA calculates the cumulative value of the *HF* from the data it gathers *locally* from the SNMP agent at the managed node. This means, the mobile agent *does not* return with the raw data, but spends time at the managed node to locally process and calculate this raw data into meaningful *HF* value. In addition, there is *only one mobile agent* sent to the managed node, for every *Polling Duration,* which returns to the manager, with the calculated value of the *HF* after the *Polling Duration*.

8. RELATED WORK

8.1. Methodology: Using MA (Mobile Agents) in Network Management

Table 2: MA(Mobile Agent) Technology Experts Applying it to NMS between 1998 to 2005

Author/Year	Approach	Characteristics	Pitfalls
Morsy M Cheikhrouhou et al(1998)	Conducted an exhaustive ***Survey*** of the various *MA Properties* like mobility, intelligence, pro-activeness, as well as their autonomous, cooperative, collaborative, inherent nature. Presented a *Technical Report* focusing on the pros and cons of applying MA for NM.	Analyzed use of *Reactive* (event-reaction)*, Deliberate* (pro-active, learns from experience and acts) and *Hybrid* (combination of both) *Agents* in NM. Weighed the **two approaches,** namely, **a)**NM *Delegation Based on Cooperation and Communication* **b)***Autonomous Deliberate Self Sufficient Agents*	***Quantitative*** comparison aspects **not** provided. Mathematical analysis of the effect of using MA in the NM aspects, namely, *FCAPS(Fault, Configuration, Accounting, Performance, Security*) would have added a lot of credibility to this technical report
Zapf M et al(1999)	Built a *Hybrid* NM framework called *'NetDoctor'* using MA and SNMP, based on *AMETAS (Asynchronous Message Transfer Agent System),* which has a proprietary API set.	The **earliest** work which has shown the design, implementation as well as experimental results of the benefits of adopting the **hybrid** approach to NM, namely use of **both** SNMP and MA.	They claim to provide seamless integration with other Java SNMP package(*eg.AdventNet*), but have **not** shown any *proof of compatibility* or results of anything other than AMETAS APIs.
Antonio Puliafito et al(1999)	Designed and implemented a framework called *MAP(Mobile Agent Platform),* a **purely** MA based NMS. Used ***four types*** of Software Agents for NMS, namely Daemon, Browser, Messenger and Verifier. ***Verifier Agents*** do not perform actual NM. They are dispatched to monitor certain specific verification tasks such as software versions used, available disk space etc.	Actual NM is done by the other 3 Agents. ***Daemon Agents***, resides permanently on the managed node. They calculate its *Health Functions* (indicating the state or efficiency of a node). They give this calculated HF values to ***Messenger Agents*** dispatched to interact with them. Additionally, ***Browser Agents*** are dispatched to specific nodes by NMS GUI to interact with the MIB and gather data.	Though a very impressive strategy, this method suffers from the disadvantage that it has the potential to increase resource utilization at the managed node, with the Daemon Agents residing in the nodes. The architecture uses ***too many software agents***, some mobile, some stationary. It becomes difficult to monitor and manage these agents, leading to the possibility of *Agent-minions* (malicious code).
Gavalas et al(2000)	Focused on the ***known problem*** of centralized NMS having scalability limitations to transfer bulk NM data. This work also focused on *improving network performance management*. Proved through experimental results ***two MA migration strategies*** that can be effectively used to gather bulk NM data.	**a)***Get 'n'Go Strategy:* The MA collects the data from the managed node and migrates to the next node, in a sequential manner. **b)** *Go 'n' Stay Strategy*: This approach used resident MA to stay at the managed node permanently. They return the calculated HF data to the manager node either at pre-defined intervals or when requested.	The *Get 'n' Go Strategy*, though innovative, is still a sequential approach. This causes ***considerable round-trip delay*** to get the complete network health information. All the MA migration strategies have the same priority. *Time-critical network analysis* is **impossible** to achieve in this framework.
George	Published **two** research	**First work** which compares	Further research has proved

Pavlou et al(2002)	result in 2002 that focused on the different techniques to improve network performance management using MA, a well known ***challenging problem***.	the performance of the usage of *Grasshopper* MA Framework versus Java RMI and CORBA. ***Proved*** that MA is not a competitor technology with Distributed Objects such as CORBA and need to co-exist.	that *seamless integration* between MA and CORBA Frameworks is a ***next to impossible task*** to achieve due to the inherent problems CORBA has as a middleware, highlighted in the above section.
Manoj Kumar Kona et al(2002)	Leveraged on the earlier research work published by Zapf(1999) that proved that hybrid frameworks using SNMP and MA is the ideal approach to NM.	Worked on the pitfalls of *Zapf's NetDoctor* framework which used proprietary *AMETAS API* set. Here they used *AdventNet* a ***popular Java Based*** SNMP API package.	The research publication title and introduction gives the impression that the work focuses on *Network Management*, but **actually focuses** only on the *Monitoring* aspects.
Iwan Adhichandra et al(2003)	Presented a Simulation Approach using a Network ***Simulation Tool*** called *OPNET Modeler* to study the behavioral results of applying MA to NMS with parameters configured within the tool	Uses ***two*** Models: **a)***Transmit*:MA is sent to a specific managed node. **b)***Route*:MA is configured with a pre-determined 'route' of managed nodes to visit and collect data.	This work considered a very *idealistic network* where the links and nodes have *no load* and are *error free*. Such a situation is **practically impossible** to fathom in a real network scenario.
Robert Steele et al(2005)	Proposed an XML-Based MA architecture to address the problem of language specific MA implementation problems of inter-agent communication and migration restriction to that particular language's execution environment.	**First work** that used: **a)** *UDDI Registry* for Agent Registration **b)** *XML Web Service Calls* for MA inter-communications and migration. *Look-up and Discovery of Agents* in a Registry is also done for the **first time** here.	Though it avoids the earlier demand of a JVM at every managed node in Java-based implementations that provided agent inter-operability, it **does not clarify** the *MA migration strategies* that needs to be used for effective NMS.
Vipin Arora et al(2007)	Focused on the problems: **a)**how to provide ***Customizable*** *Health Functions* and *SNMP Table Filters* **b)**how to reduce the MA migration delay	Solutions are they used: **a)*****String*** *based expression building* to write HF. **b)*****MA cloning*** in an *Itinerary Partitioning* strategy to reduce the delay in MA migration.	**a)**String based code causes ***language dependancy***.XML would be a better choice. **b)**MA Cloning is **not** a desirable technique. If uncontrolled it *could spread malicious code.*

Discussion

In 2002, ***Torsten Klie et al,*** used **IETF Script MIB** to do Configuration Management. Management Scripts were used to delegate the NM logic to the managed nodes. To the best of our knowledge, this is the *only parallel effort* which we have found where the managed node downloads code / script from the manager to do NM functions, similar to what happens in the case of MA. Script MIB soon became ***obsolete*** due to inter-operability issues, making MA a predominant technology where superb synergy could be achieved by adopting a hybrid approach of using both SNMP and MA Based NMS. A year later, in 2003, the emergence of XML Based NMS and later WSes for NMS ***derailed*** the importance and spotlight given to MA Based NMS. As mentioned earlier, through this paper, we hope **to bring back that focus** and trigger exploring the possibility of developing sturdy NMS frameworks, similar to '***Net Patrol'***, which blend together to create a *hybrid Web Services Based NMS with both SNMP and MA*.

8.2. Methodology: Web Services Based Network Management

In 2000, ***J.P. Martin-Flatin's*** Ph.D Thesis proposed the *XML Translation Model* for the SNMP MIB. It had Model-level mapping and Meta model-level mapping. In *Model-level mapping*, the ***DTD(Document Type Definition)*** is specific to a particular MIB and XML elements and attributes in the DTD have the same names as the SNMP MIB variables. In *Metamodel-level mapping*, the DTD is generic and identical for all SNMP MIBs.

In the year 2002, ***Mi-Jung Choi et al*** extended this work by J.P Flatin and published two research papers focusing on overcoming the shortcomings of SNMP using XML-Based NMS. At that time XML was a popular technology known for its language neutral meta-data definition. But SOA, specifically Web Services was a very ***nascent technology.***. Thus, Mi-Jung Choi et al in their 2002 research publications presented the **earliest analysis of the effect of using** an *XML-Based NMS*.

They did not yet suggest the usage of *'Web Services'*, but proposed a framework to replace the traditional ways of communication between the manager and managed nodes via SNMP over UDP. They stressed the benefits of using XML technologies like DTDs, XSLT, DOM or SAX parsers, XML Schemas, XSLT, XPath, XQuery etc to ensure an ***open, vendor independent*** SNMP data communication. They envisioned the removal of SNMP from the shackles of its innate unreliable UDP and taking it to the next level of transport via HTTP over reliable TCP.

Table 3: Network Management Pundits Applying NMS Using SOA(Web Services)

Author/Year	Approach	Characteristics	Pitfalls
Aiko Pras et al(2003)	Presented the outcome of meetings conducted by IETF,IAB,IRTF to discuss the limitation of SNMP and choose the technologies to envision the ***future of an Internet Based NMS.*** Summarized the talks as **two** approaches: **a)**Evolutionary **b)**Revolutionary:	**a)** Evolutionary: IETF's NMRG proposed evolving SNMP for bulk data transfer. **b)**Revolutionary: *XML-Based NMS* approach, leveraging on the success story of IETF WG*(NetConf)* which has an XML-based interface for Configuration Management. In this paper, the **failure** of *Evolutionary* approach is spotlighted and the **need** for *Revolutionary* approach is proposed.	This work is a study focusing on the importance as well as benefits of using an Internet Based NMS in which ***Web Services could potentially*** be a major approach. Research in SOA, specifically Web Services had just begun at that time, therefore *no results are put forth.*
Mi-Jung Choi et al(2004)	Published **two** research papers spotlighting on *Configuration Management*: **1)**Presenting IETF's ***Netconf*** which was being standardized at the time of this work **2)**X-CONF, an XML based configuration management implementation	Applied X-CONF for a *real time*, distributed, Internet traffic and monitoring system called NG-MON. One of the highlights was the seamless automatic transfer of modified configuration information to related sub-systems. They **proved** that XML-based configurations are the way to go for Internet enabled pervasive network configuration management.	While Netconf was being standardized, these researchers *jumped the gun* and developed X-CONF **which became redundant** in later years when vendors began adapting to IETF's Configuration Management with Netconf XML standards. In the 17th NMRG Meeting Minutes it was clarified that X-CONF was developed as a *proof of concept* to *pave the way* for Netconf as a standard.
Aiko Pras et al(2004)	**First work** which presented a ***comparison*** of the *performance* of Web Services based NM versus that of SNMP.	Designed and implemented the **first** Web Services based NMS framework. Compared WS NMS with SNMP based on ***four*** network parameters: *Bandwidth Utilization, CPU Time, Round trip delay* and	Established that Compressed WS based NMS offer a very good performance compared to raw XML-based Web Services. **Did not** study the *effect of using Mobile Agents* based WS which should be capable of reducing

		memory requirements.	the NM data size.
Rajesh P et al(2006)	Leveraged on earlier work done by Aiko Pras in 2004, described above. ***Focused here*** on *Configuration Management*.	Used *Web Services* to: **a)** Give a unified, consistent configuration control across network devices **b)** Provide a consistent programmable interface for configuring network element	**Does not** provide any quantifiable results. **No** *mathematical modeling* or *experimental results* of the result of using WS Based NM for network configuration is shown.
Torsten Klie et al(2007)	Focused on the age old problem – how to *fully* ***automate*** *NM*. **First work**, which proposes ***Web Service Composition*** for routine NM tasks.	Compared different Composition techniques and proposed *aNeMaC(a Network Management Composition)* which is based on an OWL-S based NM Ontology.	Till date, governing bodies including OASIS, have **not formed a standard and generic network ontology** covering all aspects of networks. In that sense, this work jumps the gun, imbibing an immature domain.

Discussion

As is evident in **Table 3** above, a lot of papers have been published on the use and performance of WS based management. A vast majority of this work focuses on the use of a gateway between WS and SNMP. The gateway aims to make the *conversion* from a WS originated XML Request to pure SNMP Request and the associated Responses back. In our work, we focused on implementing Web Services Based Network Management ***without the use of a gateway.***

9. CONCLUSIONS

First, we confirm that the principle ***'MbD'*** (***M****anagement* ***b****y* ***D****elegation*) [23] is here to stay with mobile agents ruling the roost. ***Secondly***, in our work, the focal point was to give a prototype implementation, tested on real networks, allowing us to do network management by a seamless integration of traditional SNMP and Intelligent Mobile Agents. ***Thirdly***, we have given a bunch of useful tips to the system administrator, when to choose between SNMP and Aglet strategies. We have also demonstrated how *Remote Programming* can be used to build intelligent, autonomous mobile agents, by successfully testing our *Do 'n' Die strategy* [11] for dynamically configuring SNMP Agent on a managed node.

We have reinstated the fact that Mobile Agents are *autonomous* and capable of taking *intelligent decisions.* The major advantage of this is that Sys-Admin need not be physically present, *all the time*, within the premises of the LAN that he is administering. *Periodic* as well as *On-Demand Network Patrolling Reports* can be given to him. If the Software Agents deployed by the Manager takes any autonomous action to counter any untoward incident that hampers the seamless functioning of the network, the same is reported to the Sys-Admin. The manual interventions by the Sys-Admin, to handle routine network issues are avoided, due to the use of these intelligent software agents that ***'Patrol'*** the network elements.

The spotlight of our work was to demonstrate the research outcomes of using an approach for Network Management *combining two distributed computing paradigms,* namely, *Web Services* and *Mobile Agents*. But, we did not want to bring in a totally new approach, replacing the tried and tested SNMP, a stable, sturdy, simple Network Management Protocol prevalent for more than two decades. In our hybrid ***'Net Patrol'*** NMS framework, we have adhered to the wise old thought process that the charm of novelty should not obliterate the fact that it is unwise to change a working solution! Therefore, we employed a ***convergence*** of three techniques for Network Management, namely, SNMP, Mobile Agents as well as Web Services.

10. FUTURE WORK

This paper presents the first implementation of our *'Net Patrol'* Framework. Here, we have focused on the Fault, Configuration, Accounting and Performance aspects of FCAPS [14], and have missed out on the Security facet, which we plan to address next. We also hope to employ

advanced techniques like SNMP Table Filtering and explore the use of customized HF (Health Functions). We plan to do a comparative study of the difference in the usage of HTTP (Hypertext Transfer Protocol) with ATP (Aglet Transfer Protocol).We hope to explore other areas of management where Mobile Agents can be used like Network Tomography.
In fact, relevant research work *is expected to continue ad infinitum* as different networking environments emerge with new management needs, providing fertile soil for applying new problem solving techniques. There always seems to exist a permanent quest for the all encompassing next generation management technology, much like the *proverbial 'holy grail'!*

ACKNOWLEDGEMENTS

'Net Patrol' prototype was demonstrated in *Oracle (Sun) Tech Days* for two consecutive years, 2009 and 2010, in Hyderabad, India. The first author was also invited to give a tutorial session of this novel approach in 2010 Oracle (Sun) Tech Days. In the session, several developers and researchers appreciated the need to look into reviving the application of MA as a component for Web Services based NMS. We are thankful, therefore, to the *Oracle (erstwhile Sun Microsystems) SOA development team,* Bangalore, India, for being associated with us, as part of their University Relations Program for two years and conducting strict milestone reviews to ensure that our framework scales up to their expectations of being showcased in their Global Developer Tech Days Conference. We are equally indebted to *IBM – India Research Labs as well as IBM – India Software Labs*, Bangalore for taking this research work as part of their Remote Mentoring Project Program of University Relations and giving us immense technical guidance. We thank the Bangalore offices of *Intel and Oracle* for providing the *'Center of Excellence Lab'* infrastructures within our college premises, where we could conduct our experiments. Last but not the least, we are thankful to the management of our University to have laid down the framework of encouraging strong Industry-Academia Partnerships, which helped us take our research work to a whole new level.

REFERENCES

[1] Peter Braun, Ingo Mueller, R Kowalczyk, S Kern and Friedrich-Schiller(2005),"Attacking the Migration Bottleneck of Mobile Agents", Faculty of Information and Communication Technologies Centre for Intelligent Agents and Multi-Agent Systems, *Technical Report: SUTICT-TR2005.01.*

[2] J. E. White(1996), "Mobile agents," in Software Agents (J. Bradshaw, ed.), pp. 437–472, Menlo Park, CA: The MIT Press, 1996.

[3] Mani Subramanian,"Network Management: Principles and Practice", 2008 Edition, Pearson Education

[4] Yemini Y., Goldszmidt G., Yemini S.(1991), "Network Management by Delegation", *Proceedings of the 2nd International Symposium on Integrated Network Management,* pp. 95-107.

[5] David M. Chess, Colin G. Harrison, and Aaron Kershenbaum(1997). Mobile agents: Are they a good idea? In MOS'96: *Springer-Verlag, Selected Presentations, Invited Papers Second International Workshop on Mobile Object Systems -Towards the Programmable Internet*,pp 25–45,London, UK.

[6] Vipan Arora, Harpreet Kaur Bajaj(2007), Effective Network Monitoring Using Mobile Agents, *Proceedings of National Conference on Challenges & Opportunities in Information Technology (COIT-2007)*, Mandi Gobindgarh, India, March 23-24, 2007, 142-147

[7] Robert Steele, Tharam Dillon, Parth Pandya, Yuri Ventsov(2005), XML-based Mobile Agents, *Proceedings of International Conference on Information Technology Coding & Computing,* Las Vegas Nevada,April 04-06, 42-48.

[8] Aphrodite Tsalgatidou,George Athanasopoulos,Michael Pantazoglou(2008), Interoperability Among Heterogeneous Services: The Case of Integration of P2P Services with Web Services, *International Journal of Web Services Research*, Vol(5).No.4, 79-110

[9] Agostino Poggi, Michele Tomaiuolo, Paola Turci(2007), "An Agent-Based Service Oriented Architecture", WOA 2007, 8th Joint Workshop "From Objects to Agents": Agents and Industry: Technological Applications of Software Agents, Genova, Italy.

[10] Rajesh P, Ranjiith S., Soumya P.R, Karthik V, Datthathreya S(2006),Network management system using web services and service oriented architecture : A case study, *Proceedings of IEEE Network Operations and Management Symposium(NOMS'06)*, Vancouver B C, 3-7 April, 2006, 1-4

[11] Mydhili K.Nair, Chandan Bhosle, V. Gopalakrishna(2009), Net Mobile-Cop: A Hybrid 'Intelli-Agent' Framework to Manage Networks, *Proceedings of the IEEE International Conference on Intelligent and Multi- Agents(IAMA2009)*,Chennai, India, 1-8

[12] Mydhili K Nair, Shishir M Kakaraddi, Keerthi M Ramnarayan, V Gopalakrishna(2009), Agent with Rule Engine:The 'Glue' for Web Service Oriented Computing Applied to NMS , *Proceedings of IEEE Intl Conference on Services Computing(SCC 09)*,Bangalore, India, 528-531

[13] Mydhili K Nair, Gopalakrishna V(2009), Agent Based Web Services with RuleML for NMS, *IEEE Intl Conference on Networks and Communications Proceedings*, Chennai, India, 214-219.

[14] Mydhili K Nair, Gopalakrishna V(2009), Cloud-Cop: Putting Network Admin on Cloud Nine,*IEEE Intl Conference on Internet Multimedia Services Architecture & Applications*,Bangalore, India, 1-6

[15] Zapf M, Herrmann K, Geihs K(1999),Decentralized SNMP Management with Mobile Agents, *Proceedings of 6th Int. Symposium on Integrated Nwk Mgt*, Boston, MA, USA, 24-28 May, 623-635

[16] 11th NMRG Meeting,Germany,2002, http://svn.ibr.cs.tubs.de/projects/nmrg/minutes/minutes-011.txt

[17] 12th NMRG Meeting Colarado, 2003,http://svn.ibr.cs.tu-bs.de/projects/nmrg/minutes/minutes-012.txt

[18] 16th NMRG Meeting S Korea, 2004,http://svn.ibr.cs.tu-bs.de/projects/nmrg/minutes/minutes-016.txt

[19] 17th NMRG Meeting,California,2004,http://svn.ibr.cs.tu-bs.de/projects/nmrg/minutes/minutes-017.txt

[20] Chhabra, M and Hongen Lu Jones, "Towards Agent Based Web Service", 6th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2007), pp.93-99, 2007.

[21] Aglets Mobile Agents http://aglets.sourceforge.net

[22] The Advent Net Library. http://www.adventnet.com.

[23] C. Bohoris, G. Pavlou, H. Cruickshank,"Using Mobile Agents for Network Performance Management", Journal of Network and Systems Management, pp 147-169, 2006

[24] George Pavlou, Paris Flegkas, Stelios Gooveris, Antonio Liotta(2004), On management technologies and the potential of Web services, *IEEE Communications Magazine*, Vol(42), Issue 7, 58-66

[25] Alexander Loliousis, J Seventek(2007), "A Trustworthy Mobile Agent Infrastrcture for Network Management", *10th IFIP/IEEE Symposium on Integrated Network Management* pp 938-390.

Authors

Mydhili K Nair is working as an Assistant Professor in the Dept. of Information Science and Engg, M.S Ramaiah Institute of Technology, Bangalore, India. She holds a Masters Degree in Computer Science and Engg, and is currently pursuing her PhD in Computer Science and Engg. from Anna University, Chennai. Her work experience is a mixed bag of IT Industry plus Academics. Her IT tenure of 9 years was in the Software Development Industry in Multi-national companies donning various roles ranging from Technical Lead to Project Manager. Her ongoing academic profile of 6 years has seen her publish various papers in referred conferences and journals in the field of SOA, QoS Models, Ontology in Web Services, Network Management & Mobile Agents. Faculty Profile at http://msrit.edu/node/485

Dr.Gopalakrishna is one of the co-founders of Integra Micro Systems(www.integramicroservices.com) and the erstwhile Jataayu Software now called Comviva(www.comviva.com).He obtained his BE, ME and PhD degrees from Indian Institute of Science, Bangalore, India. He is a member of Computer Society of India (CSI); a Trustee of a non-profit agency called **F**oundation for **A**dvancement of **E**ducation and **R**esearch (FAER);President of IISc Alumni Association, Bangalore Chapter. He has a wide variety of interests ranging from Mobile and Internet Applications, SOA, Network Management, Software Agents, Device Drivers, Imaging, Embedded Systems etc.