University of Nairobi

SCHOOL OF COMPUTING AND INFORMATICS

AN AGENT-BASED MODEL OF A RESPONSIVE NETWORK MONITORING SYSTEM FOR INTERNET SERVICE PROVIDERS

BY

FELIX AGIK OSONGI P58/75939/2012

MASTER OF SCIENCE IN COMPUTER SCIENCE

SUPERVISOR: Prof. William Okello Odongo

This Research Project is submitted in partial fulfillment of the requirements of the Master of Science in Computer Science of University of Nairobi.

DECLARATION

I Felix Osongi do declare that this project, as presented in this report is my own original work and has not been presented anywhere for the purpose of an academic award.

Signature: ______________________________________ Date:_____/_____/________

Felix Osongi

P58/75939/2012

I Prof.William Okello Odongo of the University of Nairobi supervisor do confirm that this project proposal was presented for evaluation by the above mentioned candidate and it satisfied the project requirements for the Master of Science in Computer Science of the University of Nairobi.

Prof.William Okello Odongo

Signature: ______________________________ Date: ______/______/______

ACKNOWLEDGEMENT

I would like to gratefully take this opportunity to thank God for his care, grace and good health. He provided for me throughout the master course. During this research project proposal I was aided by many people directly and indirectly through feedback, support and motivation.

I like to express my appreciation to Prof.William Okello and Dr. Elisha Opiyo for their support. They have given me important directions on writing this project. Their advices and comments are always very professional, and concise. Under their supervisory, this project writing became easier and successful.

Abstract

The purpose of this project was to develop a responsive Agent based network monitoring system designed to reduce the time taken by network administrators to initiate monitoring of network nodes and making any changes and updates on the monitoring parameters.

Due to development of various technologies that interconnect most of today's organizations, may highly rely on the Internet and Intranets for their day to day activities and thus lay a huge reliance on it. A few hours of network breakdown would incur huge losses to the organization.

The widely used Simple Network Management Protocol (SNMP) is today particularly limiting due to the client-server centralized and static paradigm. The approach is Ad-hoc, centralized and relies on a limited set of capabilities at network nodes. In this paradigm, the system does not auto respond to changes and modifications in the network as hard coding is required to make the necessary changes.

Our approach used a Multi Agent System (MAS) that provides a solution that works by reducing the time taken to initiate and make any changes in the monitoring parameters by introducing intelligence to the system. In this project, we used Prometheus methodology in the system specification, design and implementation. The Prometheus methodology is a detailed process for specifying, designing, and implementing intelligent agent systems. Our goal in using the Prometheus methodology was to have a process with defined deliverables.

The results of the study showed that the MAS system optimizes responsiveness by ensuring changes done on the database are automatically updated to the agent's database which makes the relevant changes on the monitored parameters.

Table of Contents

DECLARATION ii

ACKNOWLEDGEMENT iii

Abstract iv

LIST OF TERMS VIII

LIST of Figures ix

CHAPTER ONE: INTRODUCTION 1

1.0 BACKGROUND OF THE STUDY 1

1.1 PROBLEM STATEMENT 3

1.2 RESEARCH OBJECTIVES 3

1.2.1 General Objective: 3

1.2.2 Specific Objectives: 3

1.3 WHY SIMULATION 3

Some of the reasons for using simulation in this project include: 4

1.4 Benefits of simulation: 4

1.5 SIGNIFICANCE OF THE STUDY 5

1.6 LIMITATIONS OF THE STUDY: 5

1.7 SCOPE OF THE STUDY: 6

1.8 ASSUMPTIONS: 6

CHAPTER TWO: LITERATURE REVIEW 7

2.0 LITERATURE REVIEW 7

2.1 Theoretical Review 7

2.1.1 SNMP 13

2.1.2 Multi Agent Systems (MAS) 14

2.1.3 Agent Characteristics: 15

2.2 Multi -Agent based systems technology 16

2.4 Multi-Agent System (MAS) Methodologies 17

2.4.1 FEATURES OF AGENT METHODOLOGY 18

2.4.2 Agent Genealogies - (James Odell (2005)) 18

CHAPTER THREE: METHODOLOGY 19

3.0 RESEARCH METHODOLOGY 19

3.1 METHODOLOGY 19

3.2 METHODOLOGY PHASES: 20
3.3 CONCEPTUAL FRAMEWORK: 21
3.4 RESEARCH DESIGN 23
3.5 PROBLEM DESIGN 23
3.6 TOOLS DESIGN 23
3.7 TOOLS AND SKILLS REQUIRED: 23
3.8 DATA COLLECTION METHOD AND PROCEDURES 24
3.9 SYSTEMS DESIGN 24
3.9.1 Why Promethius Methodology? 24
3.10 EVALUATION AND RECOMMENDATION: 24
3.10.2 Data collection/Requirement gathering: 25
3.10.3 Design of the MAS Network monitoring model: 25
3.11 THE PROMETHEUS METHODOLOGY 26
CHAPTER FOUR: DESIGN 27
4.0 ANALYSIS AND DESIGN 27
4.1 SNMP ANALYSIS 27
4.1.1 Requirements for SNMP monitoring tool: 27
4.1.2 Requirements for Nagios SolarWinds: 27
4.2 A NALYSIS OF PROPOSED MAS SYSTEM 38
4.3 ANALYSIS USING PROMETHEUS CONCEPTS 39
4.4 USER INTERFACES 41
4.5 MAS ANALYSIS MODEL AND VIEW 41
4.5.1 SYSTEM SPECIFICATION 42
4.5.2 Architectural Design: 48
CHAPTER FIVE: IMPLEMENTATION 53
5.0 IMPLEMENTATION AND RESULTS 53
5.1 IMPLEMENTATION TOOLS 53
5.2 SYSTEM TESTING AND EVALUATION 54
5.2.1 Component testing 54
5.2.2 Integrating Testing 55
5.2.3 System Testing 55
5.2.4 Acceptance Testing 56

5.2.5 Performance Testing.. 56

5.2.6 Regression Testing 56

5.2.7 Evaluation Testing.. 56

5.3 TESTS FOR THE DEVELOPED SYSTEMS.. 57

5.4 EVALUATION OF THE MULTI AGENT BASED NETWORK MONITORING SYSTEM:.................... 62

5.5 DISCUSSION OF RESULTS: 63

CHAPTER SIX: CONCLUSION...65

6.0 CONCLUSION AND FEATURE WORK.. 65

6.1 RECOMMENDATIONS FOR FUTURE WORK: 65

6.2 CHALLENGES 65

REFERENCES:.. 67

List Of Terms

ATM: Automated Teller Machine

CDP: Cisco Discovery Protocol

JADE: Java Agent Development Environment

ISP: Internet Service Provider

MAS: Multi Agent System

MPLS: Multi Protocol Label Switching

NOC: Network Operations Center

NMS: Network Monitoring System

IP: Internet Protocol

POP: Point of Presence

QOS: Quality OF Service

RADIUS: Remote Access Dial In User Service

RMON: Remote Monitoring

SLA: Service Level Agreement

SNMP: Simple Network Monitoring Protocol

VPN: Virtual Private Network

VOIP: Voice Over Internet Protocol

WWW: World Wide Web

LIST of Figures

Figure 1: Architecture of A Typical ISP Monitoring System: 2
Figure 2: Typical Data Center Network Monitoring Architecture. 2
Figure 3: The MOMENT system Architecture: 8
Figure 4: Cisco Network Management System: 9
Figure 5: Architecture for cloud monitoring: 12
Figure 6: influences of Object Oriented Methodologies on Agent Oriented Methods 18
Figure 7: The phases of the Prometheus methodology 19
Figure 8: Conceptual Framework. 22
Figure 9: Creating SNMP server community string 29
Figure 10: Describing an Interface in a Cisco Router 30
Figure 11: Successful Logon to Monitoring system. 30
Figure 12:Device Specifications: 31
Figure 13: Configuring SNMP parameters; 32
Figure 14: Choosing SNMP Template; 33
Figure 15: SNMP device Status: 33
Figure 16: Viewing Available Device Interfaces: 34
Figure 17: Choosing an Interface to be Polled by SNMP: 34
Figure 18: Creating the Graph: 35
Figure 19: Preview of the Graph Created: 35
Figure 20: SNMP collecting Usage Data from the Device: 36
Figure 21: Logon to SolarWinds: 37
Figure 22: Adding Device to be monitored: 37
Figure 23: Confirming Device Status: 37
Figure 24: Promethius Methodology System Specifications: 42
Figure 25: Analysis Design: 43
Figure 26: System Goals: 44
Figure 27: Use Case Scenario One: 45
Figure 28: Use Case Scenario Two: 45
Figure 29: Use Case Scenario Three: 45
Figure 30: Creating New Agent 46
Figure 31: Creating New Region/POP: 47
Figure 32: Creating New Client: 47
Figure 33: Updating Client Details: 48
Figure 34: Agent Interaction Diagram: 49
Figure 35: Agent Role Grouping Diagram: 50
Figure 36: Agent Acquaintance Diagram: 50
Figure 37: System Overview Diagram 51
Figure 38: System Access: 59
Figure 39:New Client Details: 59
Figure 40: Package One Client Usage Statistics: 60
Figure 41: package Two Client Usage Statistics: 61
Figure 42: Agent Interaction Protocol: 62

CHAPTER ONE: INTRODUCTION

1.0 Background of the Study

Network monitoring and maintenance was traditionally not taken so seriously but with new developments and technologies, organizations are requiring strict Service Level Agreements (SLA’s) that prompts the need for advanced monitoring management systems by the providers.

Network users are increasingly having high expectations of reliability and quality of service (QoS) that the network offers. Most of today’s organizations rely on the Internet for their day to day activities and thus lay a huge reliance on it. A few hours of network breakdown would incur huge losses to the organization.

Architectures and algorithms for solving monitoring problems have been devised; different monitoring technologies have been proposed and standardized. From the protocol-based Simple Network Management Protocol (SNMP) version 1 to distributed object-based approaches like the Common Object Request Broker Architecture (CORBA) and later by Java’s Remote Method Invocation (Java-RMI).

With the current developments in different sectors example the banking sector and the deploying Automated Teller Machines (ATMs), Internet Banking and Electronic Funds transfer (ETF), the mobile companies delivering SMS services, money transfer services; Voice Over IP (VOIP), Virtual Private Networks (VPN), Video conferencing, E-mail, Internet (World Wide Web), Social networking and all other ways of information flow is highly growing; the paradigm of moving management logic close to the data it requires is becoming highly desirable in monitoring architectures. Delegation in the context of general distributed object frameworks is achieved through object mobility.

In the work described in this paper we are trying to evaluate, using the JADE platform, the use of mobile agents for a responsive network monitoring, assuming a constrained mobility paradigm in which a mobile agent is sent to execute and monitor information within a network element.

The figures (Fig 1 and 2) below show how most of the Internet Service Provider (ISP) monitoring systems are designed. The NMS has to communicate with all nodes in the network in order for it to report the required parameters.

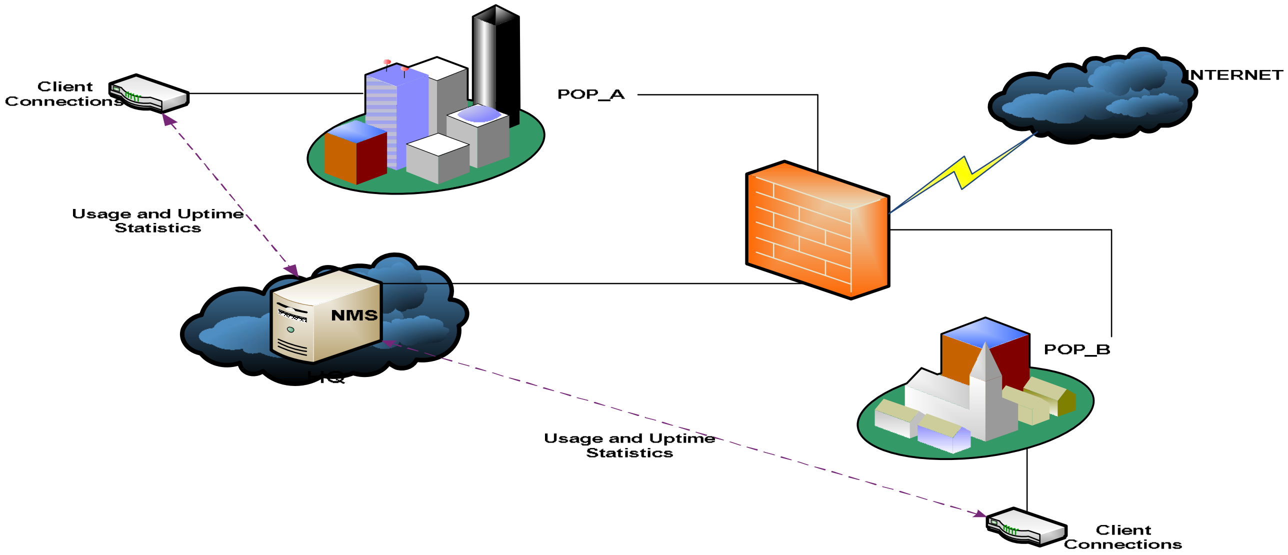


Figure 1: Architecture of A Typical ISP Monitoring System:

POP: Point of Presence;   **NMS**: Network Monitoring System

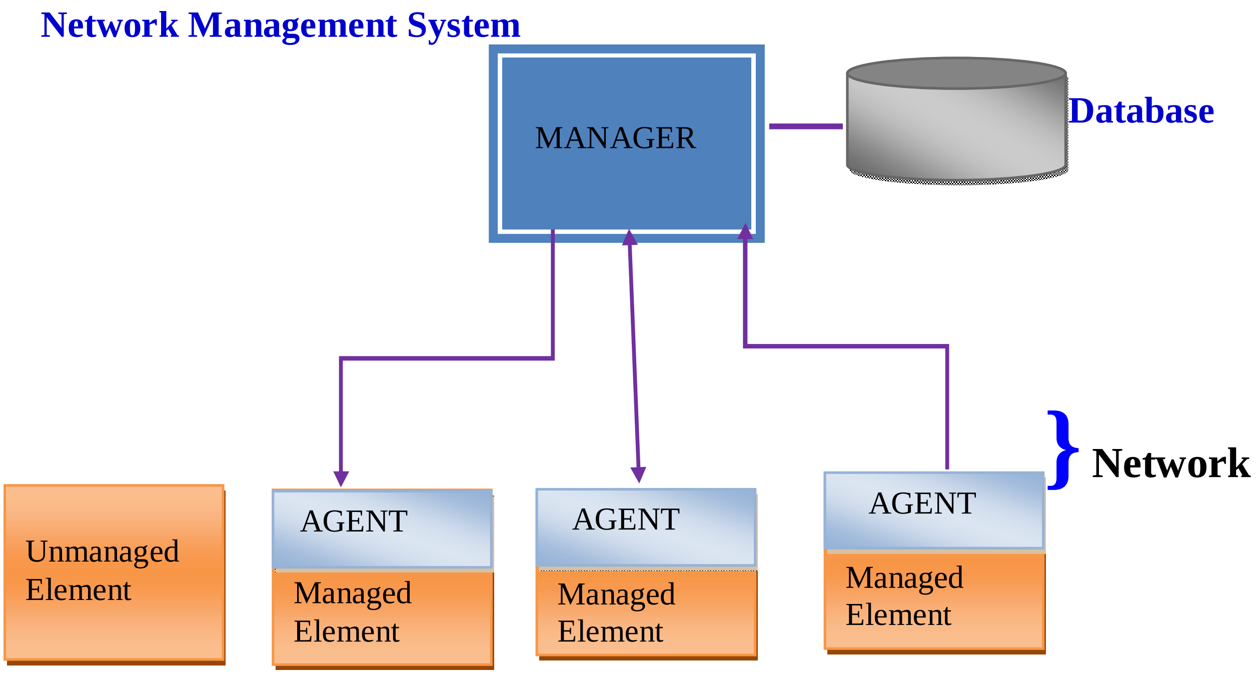


Figure 2: Typical Data Center Network Monitoring Architecture.

Source: Author 2014

1.1 Problem Statement

The widely used Simple Network Management Protocol (SNMP) is today particularly limiting due to the client-server centralized and static paradigm. The approach is centralized as it relies on a limited set of capabilities at network nodes while management processing has to be performed at the network management station. This requires transferring of large amounts of management data between the manager and the managed nodes.

In this paradigm, the system does not auto respond to changes and modifications in the network as hard coding is required to make the necessary changes.

Multi Agent System (MAS) provides a solution that works by reducing the time taken to initiate and make any changes in the monitoring parameters by introducing intelligence to the system. This research discusses a MAS solution to certain real world requirements of any network administrators in an Internet Service Provider domain.

Other solutions like Remote Monitoring (RMON), Cisco Discovery Protocol (CDP), Cisco Net flow Accounting and Monitoring by Delegation (MbD) have been put forward and some of them from are in use though these protocols are not used to compare against the results of the proposed MAS to access the necessity of the proposed system.

1.2 Research Objectives

1.2.1 General Objective:

The objective of this project is to demonstrate how Agent Based System can be used to realize a more responsive Network Monitoring system.

1.2.2 Specific Objectives:

1. To investigate how software agents can be used to implement Network Monitoring.
2. Simulate a MAS Model of a responsive Network Monitoring system.
3. Evaluate the performance of an Agent Based Network monitoring system against other currently used systems.

1.3 Why Simulation

Based on Stewart Robinson (2004), there are several reasons why simulation is necessary and benefits for using simulation.

Some of the reasons for using simulation in this project include:

a) To gain the insight necessary for making some decisions: in this simulation of network monitoring we model how Agents can pick details from a client data base and proceed to initiate monitoring of customer links. This helps in deciding which parameters to enter into the data base that are required by the Agents.

b) To make use of models in understanding, changing, managing and controlling reality. In particular, this involves understanding and/or identifying ways of improving a system.

c) Inform decision-making on the real system regarding the future items.

d) To enable the prediction of the performance of an operations system under a specific set of inputs. For example, in our system, one can predict the average time taken to initiate monitoring once all parameters are given and the client link activated.

e) To allow a ‘‘what-if’’ analysis as the user enters a scenario and the model predicts the outcome. The alternative scenarios may be explored until the experimenter has obtained sufficient understanding or identified how to improve the real system. It thus acts as a decision support system (Robinson, 2004).

f) To handle a problem that is too complicated to solve analytically.

g) To handle tractable problem (Easily solved or worked) whose level of detail provided by the analytical answers is insufficient for the required needs. To put a new concept into practice on an experimental basis and see if it produces the desired results;

1.4 Benefits of simulation:

a) Simulation is less costly, while experimentation with the real system can be very expensive.

b) Simulation takes less time, while an experiment with a real system may take many weeks or months before a true reflection of the performance of the system can be obtained. With simulation the, results on system performance can be obtained in a matter of seconds, minutes, may be hours. The faster experimentation also enables many ideas to be explored in a short time frame.

c) Simulation enables easier control of the experimental conditions, which is useful in comparing alternatives. This can be very difficult when experimenting with the real system. For example consider the difficulty of controlling the arrival of patients at a hospital.

d) Simulation can be used even where the real system does not exist, such as the case of a new yet to be built school, football stadium or hospital.

e) Multi Agent systems simulations use virtual time, time and environment are controlled by the modeler.

1.5 Significance of the Study

This study was intended to demonstrate using a model, the use of a Multi Agent System (MAS) that provides several benefits to network management including; distribution of management work load, adaptability, responsiveness, flexibility, programmability and customization using the JADE Agent platform.

The study proposes a framework that helps to enhance Performance monitoring within the ISP Networks and enhances QoS management.

- For **Systems and Network administrators**, the study improves the quality and efficiency of their work by eliminating the ad-hoc methods used to initiate monitoring and management of nodes.
- For **customers**, the study improves quality of service delivered by improving the methods used in capturing and accuracy of reports as agent based systems does not allow for errors.
- For **managers**, the study improves data collection, reporting and facilitates statistical and economic analysis of service provision.

1.6 Limitations of the Study:

The researcher faced difficulties due to fear of information leakage to the competitors by the service providers. To deal with this the researcher ensured the staffs giving information were informed that the information given is for research and no leakage was to occur to the competitors.

The researcher had to present an official letter from the University to the persons giving the information as an assurance of the data being used only for academic purposes.

Technical understanding of issues to deal with Network Monitoring was a major area of concern since some did not understand the technical terms used with the Network Monitoring systems.

This forced the researcher to use simpler terms that could not hold the real meaning of the issue being addressed. To deal with this the researcher drew diagrams to explain some of the systems that can be put in place for Network Monitoring purposes.

Time was a major constraint as the design work required more time to fully capture most of the requirements of a fully functional, highly responsive monitoring system. To deal with this the scope had been limited to ensure as much is done within the available time.

Budgetary constraint was also a hindrance to the design of the proposed solution as some components were expensive to be procured under the study. The study used some free software available and a model that does not need to run on live systems that was unaffordable under the scope of the study.

1.7 Scope of the Study:

This study limited itself to corporate links that are serviced by Internet Service Providers and are provisioned on agreed Service Level Agreements (SLAs) hence require strict monitoring to ensure the SLA and Quality of Service (QOS) are met.

The study was limited to connections that are on Point to Point and are statically connected with Internet Protocol (IP) address block.

Nevertheless, the study also appreciated that different services may be provided to the stakeholders of any service provider. The privileges provided to the subscribers are not the same as that of the network administrators. The users' authentication on different networks is done by different methods including RADIUS, LDAP, Active Directory services, so as to enable universal login to all the users of the network within the service provider network.

1.8 Assumptions:

The study took into consideration certain assumptions to be able to design the desired model. These included:

1. The service providers have more than one Point of Presence (POP).
2. Internet and inter branch connectivity is provided to the users of the network using different technologies like Static routing, MPLS VPN, GRE etc.
3. There is an existing Network Operations Center (NoC) for Network Monitoring and management.
4. The available Internet Protocol (IP) block assigned to the Service Provider is subnetted to the different POPs appropriately
5. There is an existing Database of Corporate clients with the required parameters to initiate polling.
6. There is an existing monitoring tool used by the service providers against which the Agent based design will be evaluated.

CHAPTER TWO: LITERATURE REVIEW

2.0 LITERATURE REVIEW

Mobile Agents as an approach has been investigated by many researchers and this paper analyses substantive findings, as well as theoretical and methodological contributions to Network performance monitoring.

This chapter dealt with theoretical review of how the ISPs have been carrying out Network monitoring with the available tools and how they implement QoS so as to achieve the Service Level Agreements (SLAs).

2.1 Theoretical Review

This section identified theories related to network monitoring and how they are linked to the dependent and the independent variables.

Rao Sathya (January 2008 – June 2010); makes bold contributions about the things we know to make sure that network researchers will gain a better understanding of current networks as well as of The Network of the Future. By bringing together existing pan-European network monitoring infrastructures, MOMENT (**Monitoring and Measurement in the Next generation Technologies)** tends to have mobilized the European key stakeholders who can make a difference and enhance our understanding of the network.

The MOMENT project was meant to integrate the existing measurement and monitoring infrastructures towards a common and open, pan-European platform. The project was to achieve a semantic representation and retrieval of measurement and monitoring information. It was also to develop and demonstrate a set of tools and applications for the future Internet taking advantage of the integrated approach.

The key issues addressed by the project can be summarized as follows:

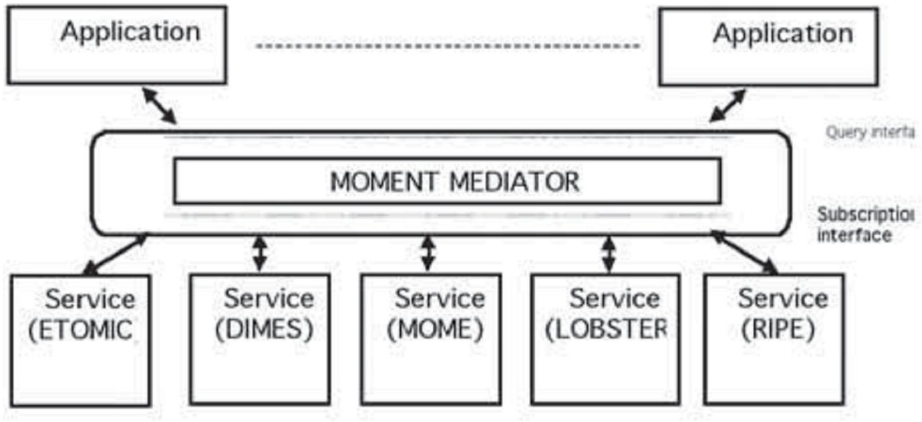
i. The protocol that serves for the applications to perform semantic queries to the mediation engine through the query interface using web services.

ii. The monitoring services to subscribe through subscription interface

iii. The interface or wrapper that the measurement infrastructures should use in order to register the offered service and data. For example, XML could be used to specify the data.

iv. Configuration interface for communicating with management tasks

Figure 3: The MOMENT system Architecture:



Source: R.Sathya (2008-2010)

- **ETOMIC:** ETOMIC provides both a database infrastructure and tools for real-time measurements.
- **DIMES:** DIMES provides both a database infrastructure and a tool for real-time measurements.
- **LOBSTER:** LOBSTER provides a database infrastructure.
- **RIPE:** RIPE provides a database infrastructure.
- **BART:** BART is a tool for real-time measurements.

The challenge with the MOMENT architecture was the introduction of the mediator which is a central point of failure in the system as the Applications can only communicate to the services via the MOMENT mediator.

Cisco Network Management System; Best Practices White Paper analyzes The International Organization for Standardization (ISO) network management model. It defines five functional areas of network management. The document describes the functional areas and the overall purpose of the document is to provide practical recommendations on each functional area to increase the overall effectiveness of current management tools and practices. It also provides design guidelines for future implementation of network management tools and technologies.

The ISO network management model's five functional areas are listed below.
Fault Management—Detect, isolate, notify, and correct faults encountered in the network.
Configuration Management—Configuration aspects of network devices such as configuration file management, inventory management, and software management.

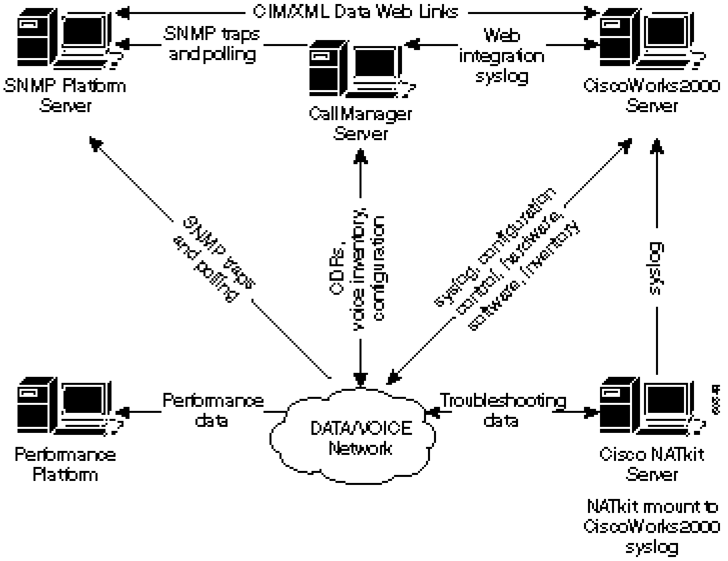
Performance Management—Monitor and measure various aspects of performance so that overall performance can be maintained at an acceptable level.

Security Management—Provide access to network devices and corporate resources to authorized individuals.

Accounting Management—Usage information of network resources.

The following diagram shows a reference architecture that Cisco Systems believes should be the minimal solution for managing a data network.

Figure 4: Cisco Network Management System:



The network management architecture includes the following:

Simple Network Management Protocol **(SNMP)** platform for fault management

Performance monitoring platform for long term performance management and trending

CiscoWorks2000 server for configuration management, syslog collection, and hardware and software inventory management

The architecture is designed on Client Server architecture with SNMP server sending traps to the clients. A central point of failure is created if the server malfunction rendering monitoring impossible in the whole network. With critical services in mind like the call server, it’s desirable to have a more responsive and mobile system of monitoring the Network. In this paper we design a Mobile Agent based system to evaluate against such systems and make recommendations based on the results.

G. Aceto et al (2013) discusses Monitoring of Cloud is a task of paramount importance for both Providers and Consumers. On the one side, it is a key tool for controlling and managing hardware and software infrastructures; on the other side, it provides information and Key Performance Indicators (KPIs) for both platforms and applications. The continuous monitoring of the Cloud and of its SLAs (for example, in terms of availability, delay, etc.) supplies both the Providers and the Consumers with information such as the workload generated by the latter and the performance and QoS offered through the Cloud, also allowing to implement mechanisms to prevent or recover violations (for both the Provider and Consumers).

Monitoring is clearly instrumental for all the activities covered by the role of Cloud Auditor. In more general terms, Cloud Computing involves many activities for which monitoring is an essential task. In his paper Aceto mentions the roles of monitoring in cloud computing as below:

Capacity and resource planning

One of the most challenging tasks for application and service developers, before the large scale adoption of Cloud Computing, has always been resource and capacity planning (e.g. Web Services. In order to guarantee the performance required by applications and services, developers have to:

(i) Quantify capacity and resources (e.g. CPU, memory, storage, etc.) to be purchased, depending on how such applications and services are designed and implemented.

(ii) Determine the estimated workload. However, while estimation can be obtained through static analysis, testing and monitoring, the real values are unpredictable and highly variable. Cloud Service Providers usually offer guarantees in terms of QoS and thus of resources and capacity for their services as specified in SLAs and they are in charge of their resource and capacity planning so that service and application developers do not have to worry about them. To this end, monitoring becomes essential for Cloud Service Providers to predict and keep track of the evolution of all the parameters involved in the process of QoS assurance in order to properly plan their infrastructure and resources for respecting the SLAs.

Capacity and resource management

The first step to manage a complex system like a Cloud consists in having a monitoring system able to accurately capture its state. Over the years, virtualization has become a key component to implement Cloud Computing. Hiding the high heterogeneity of resources of the physical infrastructure, virtualization technologies introduced another complexity level for the infrastructure provider, which has to manage both physical and virtualized resources. Virtualized resources may migrate from a physical machine to another at any time. Hence, in Cloud Computing scenarios (especially in mobile ones monitoring is necessary to cope with volatility of resources and fast-changing network conditions (which may lead to faults).

SLA management

The unprecedented responsiveness in terms of resource management provided by Cloud Computing calls for new programming models in which Cloud applications can take advantage of such new feature, whose underlying premise is monitoring. Moreover, monitoring is mandatory and instrumental in certifying SLA compliance when auditing activities are performed to respect regulation (e.g. when government data or services are involved). Finally, monitoring may allow Cloud Providers to formulate more realistic and dynamic SLAs and better pricing models by exploiting the knowledge of user-perceived performance.

Billing

One of the essential characteristics of Cloud Computing is the offer of ‘‘measured services’’, allowing the Consumer to pay proportionally to the use of the service with different metrics and different granularity, according to the type of service and the price model adopted. Examples of billing criteria are: for SaaS, the number of contemporary users, or the total user base, or application-specific performance levels and functions; in PaaS services, the CPU utilization, or the task completion time; for IaaS, the number of VMs, possibly varying with different CPU/Memory setups.

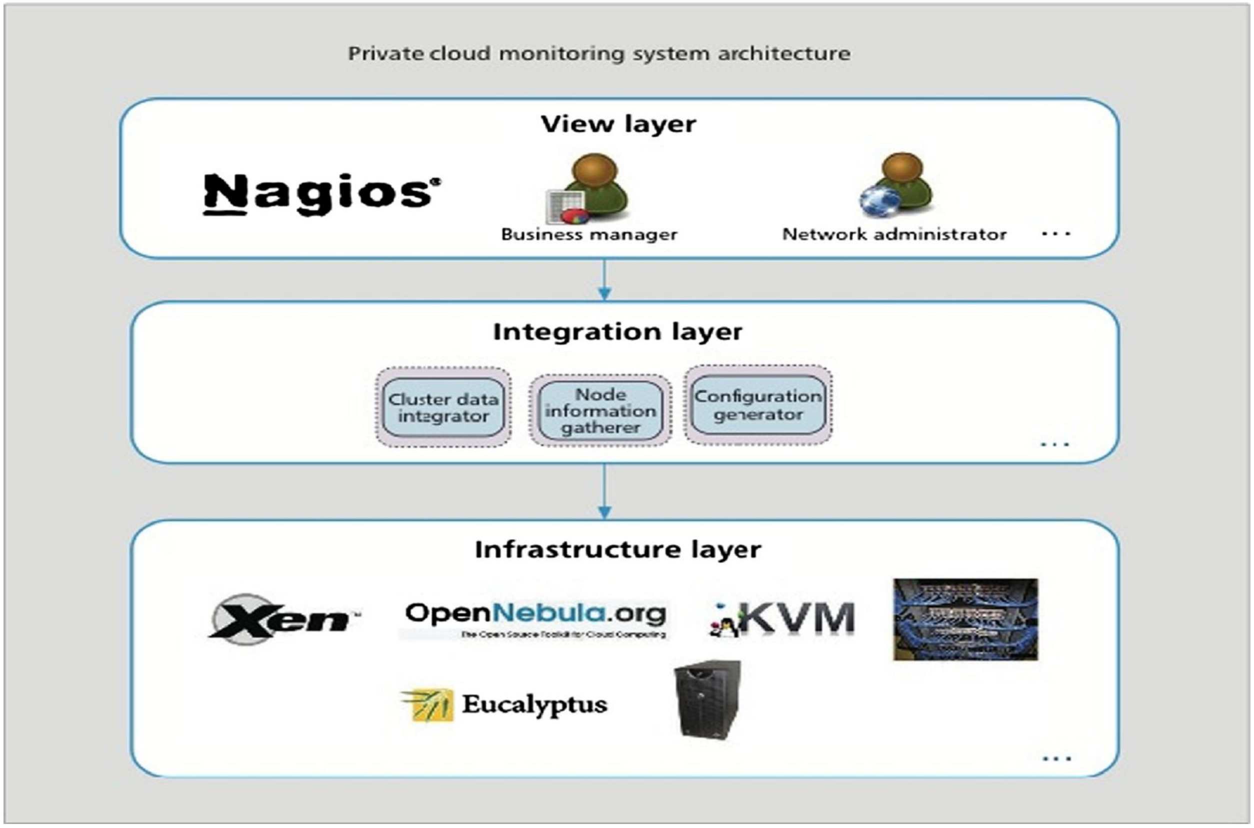
Troubleshooting

The complex infrastructure of a Cloud represents a big challenge for troubleshooting (e.g. root cause analysis), as the cause of the problem has to be searched in several possible components (e.g. network, host, etc.), each of them made of several layers (e.g. real and virtual hardware, host and guest OS, etc.). A comprehensive, reliable and timely monitoring platform is therefore needed for Providers to understand where to locate the problem inside their complex infrastructure and for Consumers to understand if any occurring performance issue or failure is caused by the Provider, network infrastructure, or by the application itself.

Security management

Cloud security is very important for a number of reasons. Security is considered as one of the most significant obstacles to the spread of Cloud Computing, especially considering certain kinds of applications (e.g. business critical ones) and Consumers (e.g. governments). Different works in literature have provided reviews and recommendations for Cloud security and the references therein. For managing the security in Cloud infrastructures and services, proper monitoring systems are needed. Moreover, for hosting critical services for public agencies, Clouds have to satisfy strict regulations and prove it. And this can be done through a monitoring system that enables auditing (e.g. to certify the compliance to regulations and obligations, such as keeping data of a user inside country borders).

Figure 5: Architecture for cloud monitoring:



Source: C.Bohoris

It’s due to the above mentioned roles that MAS becomes of importance in a cloud environment as it addresses most of the concerns effectively than may be compared to other means of achieving the same goals. For example, MAS would address security concerns of the cloud environment better than the use of agentless systems like SNMP that has been compromised due to the vulnerabilities with the protocol.

SNMP (Simple Network Management Protocol) is the common language of network monitoring–it is integrated into most network infrastructure devices today, and many network management tools include the ability to pull and receive SNMP information.

RFC 1157 defines the Simple Network Management Protocol (SNMP) as a standard application layer protocol that allows a management station (the software that collects SNMP information) to poll agents running on network devices for specific pieces of information.

What the agents report is dependent on the device. For example, if the agent is running on a server, it might report the server’s processor utilization and memory usage. If the agent is running on a router, it could report statistics such as interface utilization, priority queue levels, congestion notifications, environmental factors (i.e. fans are running, heat is acceptable), and interface status.

2.1.1 SNMP

All SNMP-compliant devices include a specific text file called a Management Information Base (MIB). A MIB is a collection of hierarchically organized information that defines what specific data can be collected from that particular device. SNMP is the protocol used to access the information on the device the MIB describes. MIB compilers convert these text-based MIB modules into a format usable by SNMP management stations. With this information, the SNMP management station queries the device using different commands to obtain device-specific information. There are three principal commands that an SNMP management station uses to obtain information from an SNMP agent:

a) The get command collects statistics on SNMP devices.

b) The set command changes the values of variables stored within the device.

c) The trap command reports on unusual events that occur on the SNMP device.

2.1.2 Multi Agent Systems (MAS)

Back in 1998, emerging agent paradigms and enabling technologies were considered a key for the implementation of highly responsive and scalable solutions that add a degree of openness to the telecommunications industry. In retrospect, agent technology suffered from different terminology and heterogeneity of technical approaches due to the lack of standards. Furthermore the broad application of agent technologies to specific environments such as network management was still in its infancy; C. Bohoris (2003).

A. Liotta (July 2001), discusses "One necessary step towards the realization of active distributed monitoring is its implementation and experimentation on prototype networks or on a real networked system. Real measurements will enlighten the actual behavior of the proposed approach with regard to overheads, stability and complexity.

Measurements on the actual agent development time as a function of various scale factors will strengthen even further the motivation for integrating agents in monitoring systems".

Software agents are often more clearly understood through their attributes and behavior. It is commonly agreed among researchers that every agent exhibits several (but rarely all) of the following characteristics; M. Wooldridge (1995).

- **Autonomous:** Agents should operate without the intervention of external entities. They typically have control over their actions and internal state.

- **Adaptable:** Agents are characterized by their ability to set-up their own goals and strategy to achieve them. They typically acquire and process information on their environment both spatially and temporarily and use this information to influence their future behavior.

- **Goal oriented**: Agents should exhibit goal-oriented behavior such that their performed actions cause beneficial changes to the environment. In most cases an agent terminates after the completion of its goal.
- **Communicative/collaborative:** No agent has a complete picture of the overall system within which it operates. Each agent is an expert in a specific task and it has to collaborate with other agents in order to solve a given problem.
- **Pro-active/Active:** Often agents are required to anticipate future situations along with responding to changes within their environment.

Service Level Agreements (SLAs) have been used for many years in IT organizations and departments to identify the support requirements for internal and external customers of IT services. In this context, an SLA sets the expectations of both the IT service consumer and provider. It is common for IT service providers to deliver services at different levels of quality based on the cost paid for a service. An SLA is valuable for helping all parties understand the tradeoffs inherent between cost, schedule, and quality because their relationship is stated explicitly.

As with any type of contract, the existence of an SLA cannot guarantee that all promises will be kept, but it does define what will happen if those promises are not kept. “An SLA cannot guarantee that you will get the service it describes, any more than a warranty can guarantee that your car will never break down. In particular, an SLA cannot make a good service out of a bad one. At the same time, an SLA can mitigate the risk of choosing a bad service”, Allen (2006). A “good” service is one that meets the needs of the service customer in terms of both quality and suitability.

IT SLAs are enforced by service management processes and procedures. An example framework for IT service management (ITSM) is the Information Technology Infrastructure Library (ITIL), which has been widely used since the mid 1990s. The ITIL focuses on the activities involved with the deployment, operation, support, and optimization of applications. Its main purpose is to ensure that deployed applications can meet their defined service levels; Allen (2006).

Russell et al., (1995) defines agents as anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors.

Chang (1998) describes intelligent agents as software entities that carry out some set of operations on behalf of the user or another program with some degree of independence or autonomy and employ some knowledge or representation of the user's goals or desires. Autonomous agents are described as computational systems that inhibit some complex dynamic environment, can sense and act autonomously in its environment and realize the designed goals and tasks.

2.1.3 Agent Characteristics:

Franklin et al., (1996) describes the following agent characteristics:

a) Agent responds in a timely fashion to changes in the environment.

b) Agent exercises control over its own actions.

c) Agent does not act in response to the environment.

d) Agent is a continuous running process.

e) Agents communicate to other agents and systems.

f) Agent can change its behaviors based on previous experience.

g) Agent can migrate from one machine to another.

h) Agent actions may not be scripted.

Sunsted et al., (1998) describes reasons to adopt mobile agent:

i. In a client server architecture continues handshake for each request/response requires a complete round trip across the network. This uses the useful network bandwidth available. Mobile agents reduce this bandwidth utilization problem by the client migrating to the server's machine and making requests on the server directly.

ii. Mobile agents can work off-line and communicate their results when the application is back on-line, thus solving the problems in client server architecture created by intermittent or unreliable network connections.

iii. Clear idea on what problem is to be solved is a mandatory requirement for any application designed in a client server architectural model. Scope for modifications in an application is very limited or costly in this model. Problems in network monitoring are dynamic and ever changing and thus the application needs a more agile development model. Agent based model for development is most applicable since the administrator of the network can decide what an agent should do on a visit, to a managed node.

iv. Networks of today work on heterogeneous environments and thus the management tool used must be able to work on such a network. Agents are well adapted to work in any heterogeneous environments. All these offer compelling reasons to adopt agent architectural model for developing a tool for network management tasks.

2.2 Multi -Agent based systems technology

This project used agent-based systems technology which is a new paradigm for conceptualizing, designing, and implementing software systems. This technology is an attractive technology for building software systems for environments that are distributed, open and complex. In the recent decades the agent systems were developed and built with one agent. As complexity of these systems increased and agent technology matured incorporation of several agents has been adopted to counter the limitations of a single agent namely, limitation of computational resources and risk of failure. Multi-agent system is a system of agents which interact with one another through cooperation, competition, coordination or negotiation [usually to accomplish some goal]; Wooldridge (2002). According to Sycara (1998), Multi-agent system is a system of several agents (implied), and according to Georgini et al (2001), MAS is an organization of coordinated autonomous agents that interact in order to achieve common goals.

2.3 Characteristics of MAS

Agents act on behalf of the user with different goals and motivation. For agents to reliably interact, they require the ability to cooperate, coordinate, and negotiate with each other, much as people do. In comparison to a single agent or a centralized system, MAS has several advantages which include the following:

i. MAS lead to the realization of increased speed and efficiency.

ii. Robustness and reliability- no single point of failure; ability to solve problems that are too large for a single agent. Implementation of many agents increases robustness and reliability. If one agent fails, another one can take over.

iii. Allows for the interconnection and interoperation of multiple existing legacy systems. Legacy systems increasingly fail to interoperate due to differences in syntax and semantics. In order to keep them relevant and useful, MAS technology can be used to incorporate them in to an agent community where they can interact with other systems. This can be achieved by building wrappers, which inject code in a program to allow it to communicate; Genesereth and Ketchpel, (1994).

iv. Scalability and flexibility- the agents can enter or exit.

v). Reusability and cost – single agent developed, roles assigned; replicable, MAS is also suitable for distributed environment.

2.4 Multi-Agent System (MAS) Methodologies

A methodology is a body of methods employed by a discipline; it is also a method or a procedure for attaining something; Paolo Girogini (2005). Cases where agent solution is appropriate include the following:-

i. Open, dynamic, uncertain or complex environment.

ii. Agents are natural metaphors- such as in organizations with distributed functions, intelligent interfaces.

iii. Data control or expertise is distributed- such as database systems with different autonomous ownership.

iv. Legacy systems- where interfaces to old systems are important.

A MAS Methodology generally encompasses a set of concepts, notations for modeling aspects of the software and processes that follow in order to build the software. The agent-oriented methodologies have multiple roots; this is shown in the Figure below

2.4.1 Features of Agent Methodology

i. Should provide sufficient abstractions to fully model and support agents and MASs arguably.

ii. Should focus on an organized society of agents playing roles within an environment

iii. Support MAS, where agents interact according to protocols determined by the agents' roles

iv. Should be agent-oriented in that it is geared towards the creation of agent-based software

2.4.2 Agent Genealogies - (James Odell (2005))

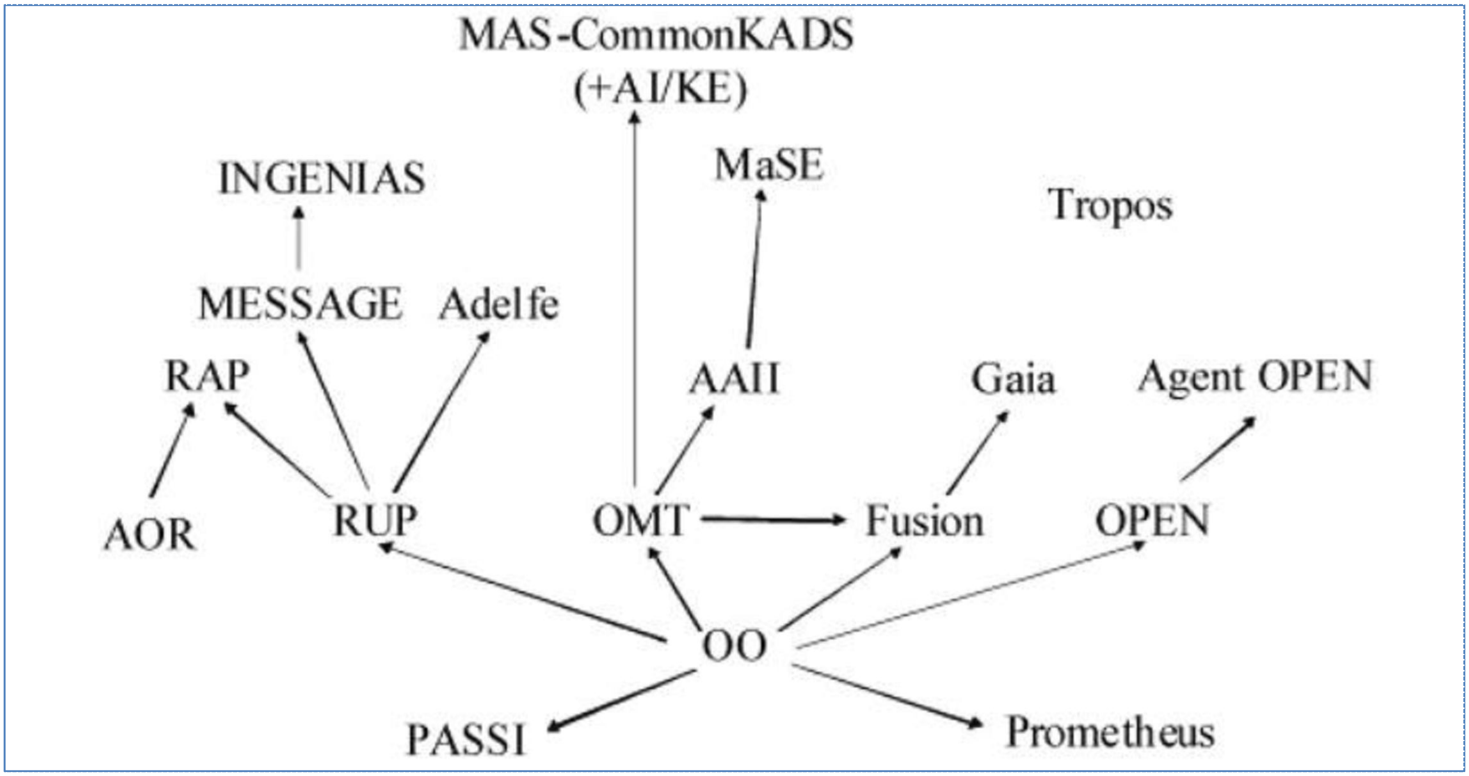


Figure 6: influences of Object Oriented Methodologies on Agent Oriented Methods

The above figure depicts the influences of Object Oriented Methodologies on Agent Oriented Methods, Henderson- Sellers & Giorgini, (2005)

CHAPTER THREE: METHODOLOGY

3.0 RESEARCH METHODOLOGY

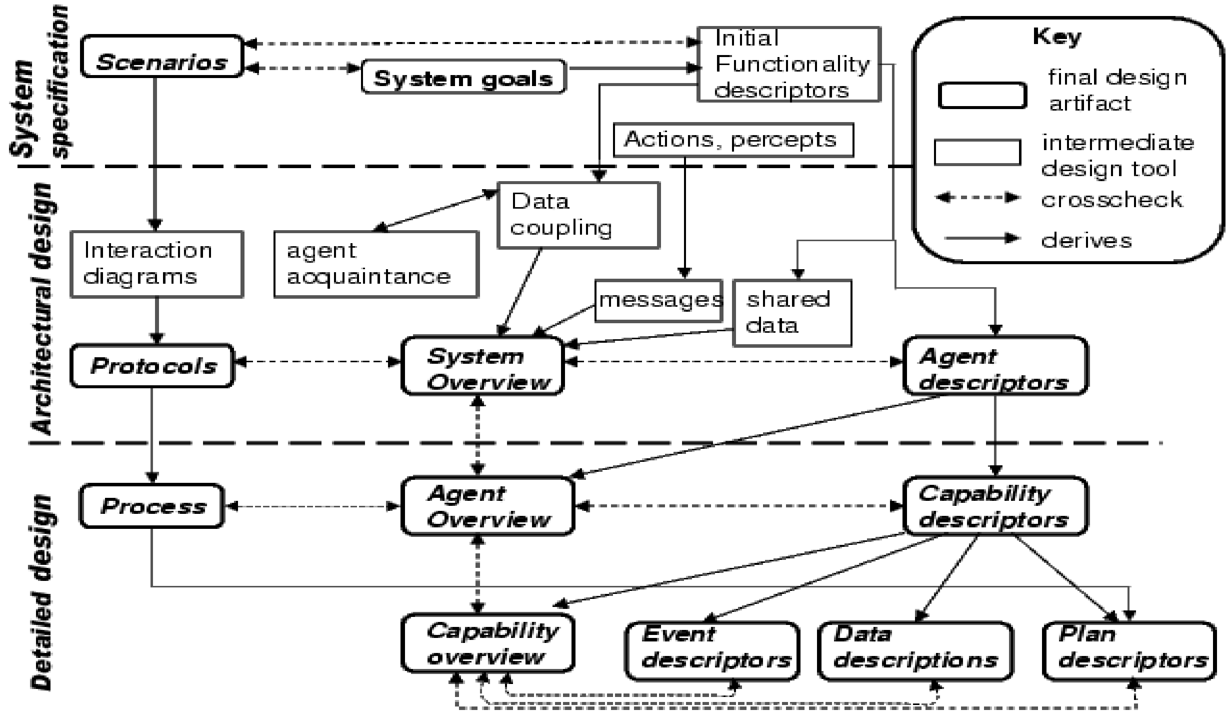
This chapter contained the steps necessary in the execution of the study to fulfill the study objectives. These are the research Design, Target population, the sampling frame, Sampling Techniques, Sample size, Data collection methods, research procedures and methods of Data analysis.

3.1 Methodology

In this project, Prometheus methodology was used in the system specification, design and implementation.

The Prometheus methodology is a detailed process for specifying, designing, and implementing intelligent agent systems. Our goal in using the Prometheus methodology was to have a process with defined deliverables.

Figure 7: The phases of the Prometheus methodology



Prometheus distinguishes itself from other methodologies by supporting the development of *intelligent* agents, providing "start-to-end" support, having evolved out of practical industrial and pedagogical experience, having been used in both industry and academia, and, above all, in being *detailed* and *complete*. Prometheus is also amenable to tool support and provides scope for cross checking between designs.

3.2 Methodology Phases:

The methodology consists of three phases:

System specification consists of the following activities:

- Identify system goals and sub-goals
- Develop use case scenarios
- Identify the agent system's interface to the environment in terms of actions, percepts, and external data
- Identify functionalities
- Identify data read and written by functionalities
- Prepare functionality schemas (name, description, actions, percepts, data used/produced, interaction (with other functionalities), and goals)

Architectural Design consists of the following activities:

- Group functionalities to determine agent types using data coupling and agent acquaintance diagrams to assess alternative groupings
- Define agent types (also define the number and life-cycle of the agent types) and develop agent descriptors
- Produce a system level overview diagram describing the overall structure of the system
- Develop interaction protocols from use case scenarios (via interaction diagrams).

Detailed Design consists of the following activities:

- Develop process diagrams
- Produce agent overview diagrams showing the internal workings of agents in terms of capabilities, events, data and plans
- Refine capability internals (add included capabilities and interactions)
- Introduce plans to handle events
- Define details of events (external, between agents, between capabilities and within agents)
- Define details of plans (relevance, context, sub goals).
- Define details of beliefs/data.

3.3 Conceptual Framework:

The proposed multi-agent based system enabled responsiveness and flexibility in monitoring client nodes within the network. The framework will have the following five main agents.

i. NOC Agent
ii. POP agent
iii. Persistence Agent
iv. External System Agents
v. Client Agent

Conceptual Framework:

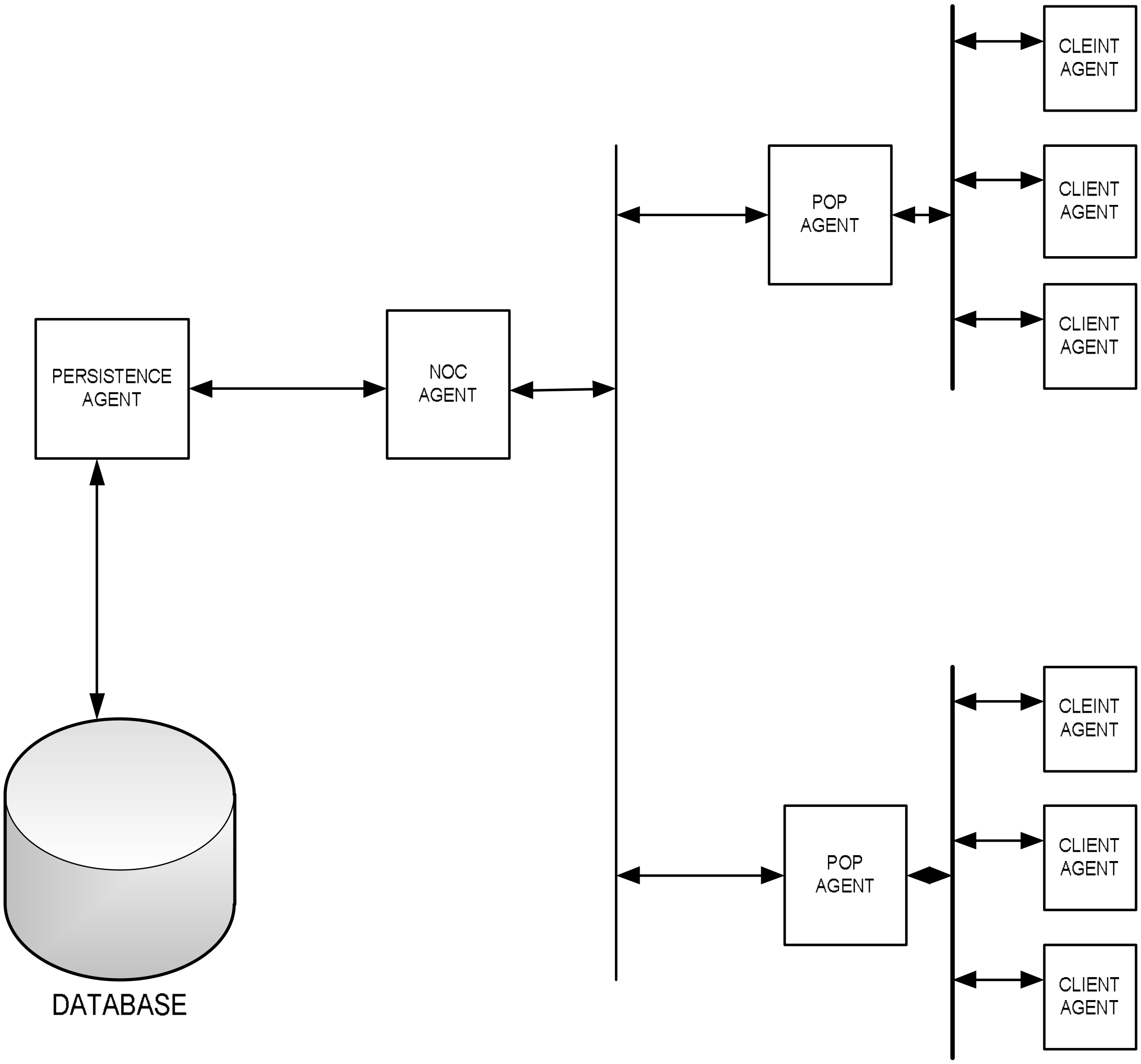


Figure 8: Conceptual Framework.

POP AGENT: Initiates Communication with the client Agents collecting status of the Client Terminals

NOC AGENT: Communicates with the POP Agents and gathers information of client status.

Persistence Agent: Handles Database Operations.

DATABASE: Server from where data is saved for analysis.

3.4 Research Design

This section ensured the evidence gotten was used to optimally answer the objectives of this project. Chauhan D(1997) and Kothari C.R(2004) describes research design as an arrangement of conditions for collection and analysis of data in a way that aims to coalesce relevance to the research purpose with t he environment in mind. Here we looked at the problem design, tools design, procedures of data collection and implementation.

3.5 Problem Design

The approach began with work already done in the field of network monitoring. A number of works have been reviewed and related models have been reviewed and their shortcomings highlighted. The model was used to initiate and manage network monitoring without having to do manual configurations on the client's appliances.

We observed (1) Agents being generated automatically (2) Agents getting back the observed data from the clients and the graphs generated to be used in the monitoring process.

3.6 Tools Design

We integrated Netbeans with JADE so that we use java functionalities in building the model we intended to use. We also used MySQL for developing client data base. All these software are freely available hence made the simulation affordable.

3.7 TOOLS AND SKILLS REQUIRED:

The following tools were used to realize the goals of this project.

a) JRE and JDE

b) MYSQL

c) Netbeans

d) Laptop Computer

e) Agent Oriented programming skills

f) Java Programming skills

g) Systems analysis and design skills

h) Reference Materials

i) JADE

j) Apache services

3.8 Data Collection Method and Procedures

In our project, the design type was simulation and method of data collection was majorly observation.

To obtain information about current systems used, we used some questionnaires that were filled and collected to guide in analyzing the current systems used by the different service providers.

We got client usage data from different ISPs used for our result analysis. Observation was majorly used as it is cheaper and less involving as compared to other methods.

3.9 Systems Design

This focused on developing the agents and showing how they would work based on the methodology. For this project we used Promethius methodology.

3.9.1 Why Promethius Methodology?

The Prometheus methodology is a detailed process for specifying, designing, and implementing intelligent agent systems. Our goal in using the Prometheus methodology was to have a process with defined deliverables.

3.10 Evaluation and Recommendation:

Once data was analyzed, results from other techniques used were used to compare against the results of the MAS system and give recommendations.

In order to realize the success of the proposed study, the following steps were taken into consideration

i. Data Collection/Requirements gathering

ii. Design of Network monitoring system.

iii. Implementation

iv. Testing and evaluation of the platform

v. Documentation and submission of the final report

3.10.1 Methodology for Evaluation:

In this project we did an evaluation to determine the performance of the Agent based system against SNMP. Our evaluation approach was to analyze SNMP as a tool for monitoring and thereafter analyze the Agent based system. In chapter four, we discuss the design and implementations for the two methods and used live examples to show how each is done to be able to come to a conclusion. Our design approach took a step by step procedure of deploying both SNMP and the AGENT system and the results were noted in terms of time taken to use either system as recorded in Table one in chapter five of this document.

3.10.2 Data collection/Requirement gathering:

The design type was experiment and data was collected through observation. We got monitoring data from the Internet Service Providers to be used for our result analysis. Observation is cheaper, faster and less involving as suggested by Chauhan (1997).

To have some sample of the observed data, some screen shots of the data observed was taken and used in the evaluation stage.

3.10.3 Design of the MAS Network monitoring model:

Analysis and design of this platform was done using multi-agent system (MAS) methodology. In this case PROMETHEUS was specifically used. The details of analysis and design of this platform are covered in next chapter.

System Implementation:

In order to implement this platform, the Java Agent Development (JADE) framework was used. JADE has been used in Java development kit (JDK) version 6.1 environment

Testing

Due to the sensitivity of client Internet usage data, it was not possible to use live data for testing purposes. The simulated data was therefore used to test the operation of the developed system.

Evaluation of the multi-agent based system

Test results determined the behavior of the system when subjected to load. The results were analyzed and interpreted to establish the correct operation of the system. Based on the interpretation of the results, further study is recommended on the proposed system. The evaluation of the multi-agent based system was done with the users in various Internet Service Provider firms so as to get comparative views of the existing system and the multi-agent based system. Evaluation involved data collection and data analysis.

Documentation (Report Writing)

This step started from the beginning of the project up to the end of the project. All pieces of documentation had to be refined to come up with a report for submission to the panel of examiners. The MS office 2007 was used for documentation and report writing.

3.11 The Prometheus methodology

Prometheus methodology was used in the system specification, system design and implementation. The Prometheus methodology is a detailed process for specifying, designing, and implementing intelligent agent systems. This methodology was used in this study because it distinguishes itself from other methodologies by supporting the development of intelligent agents, providing start-to-end support, having evolved out of practical industrial and pedagogical experience, having been used in both industry and academia, and, above all, in being detailed and complete. Prometheus is also amenable to tool support and provides scope for cross checking between designs.

It was preferred over other MAS methodologies in this study due to the fact that it is a complete methodology that is from start to end. It has detailed and elaborate process for system specification, implementation, testing and debugging. It is also supported by Prometheus Design Tool that allows the user to design overview diagrams.

Further, it has examples which help to better understand what is required in each stage of development.

The methodology consists of three phases: system specification, architectural design, and detailed design. An overview of the methodology, including its phases, deliverables, and intermediate products, is depicted below. Although the phases are described in a sequential fashion it is acknowledged that like most Software Engineering methodologies, practice involves revisiting earlier phases as one works out the details.

CHAPTER FOUR: DESIGN

4.0 ANALYSIS AND DESIGN

The system has been developed for Network Monitoring in Internet Service Provider environment. It has been developed using multi-agent system methodology. The ISP can provide various services requiring Internet connectivity; this system allows for a more responsive monitoring of the corporate links from various Points of Presence (POP) without having to initiate and/or change the process manually by the systems administrator.

We started by analyzing the current mostly used monitoring tools, SNMP and Nagios NPM and then analyzed the MAS monitoring system before drawing any conclusions.

4.1 SNMP ANALYSIS

4.1.1 Requirements for SNMP monitoring tool:

There are requirements needed in the current system to enable an administrator initiate monitoring of any link, these are as below:

(i) SNMP server Software, This can be licensed of free software.

(ii) Web server for monitoring client links from the web browser.

(iii) SNMP enabled client premise Equipment (CPE)

(iv) SNMP string

4.1.2 Requirements for Nagios SolarWinds:

(i) Nagios NPM software from Nagios.

(ii) ICMP enabled CPE

(iii) Server computer to install the NPM software

SNMP Server software:

For SNMP to be used for monitoring, a server has to be dedicated to be used to install and run the server software that communicates with the clients. Some of the software used are; Cacti, WhatsUpGold, MRTG, Nagios among others. Cacti being a free available tool on the Internet, most providers have adopted it and are using it for network monitoring. In this project, we have analyzed how Cacti is used to initiate monitoring in most of the service provider networks.

Web server:

Web servers are used to enable users to be able to view usage data from the web browser through a URL provided by the administrator.

SNMP enabled Client:

The Client Premise Equipment (CPE) has to be SNMP ready so as to be able to communicate with the server. For this project, we used Cisco 7600s as a CPE to be able to initiate client SNMP configurations and run SNMP walk from the server to be able to query the CPE.

In our test we configured the router and initiated monitoring to be able to pick some usage graph.

SNMP string:

This is a combination of characters configured on both the client and server for it to be able to query and recognize the CPE and poll the described interfaces. In our project we used the string <learn> configured on both server and client. Once the string is configured the version has to be the same at both ends for the server and client to agree on the parameter. We used SNMP version one for this project. SNMP version one was used as its basic and simple to initiate than version two or three which need configuration of many parameters like authentication which requires more services to be involved.

This project also analyzed Nagios NPM as the mostly used uptime monitoring software. In its function it requires;

a) Nagios Software:

This is commercial software developed by SolarWinds which is used to initiate UP/Down time monitoring of nodes in a network. The software is installed in a Windows or Linux server computer and functions using ICMP (ping) protocol by sending requests to the CPE in a configured time interval.

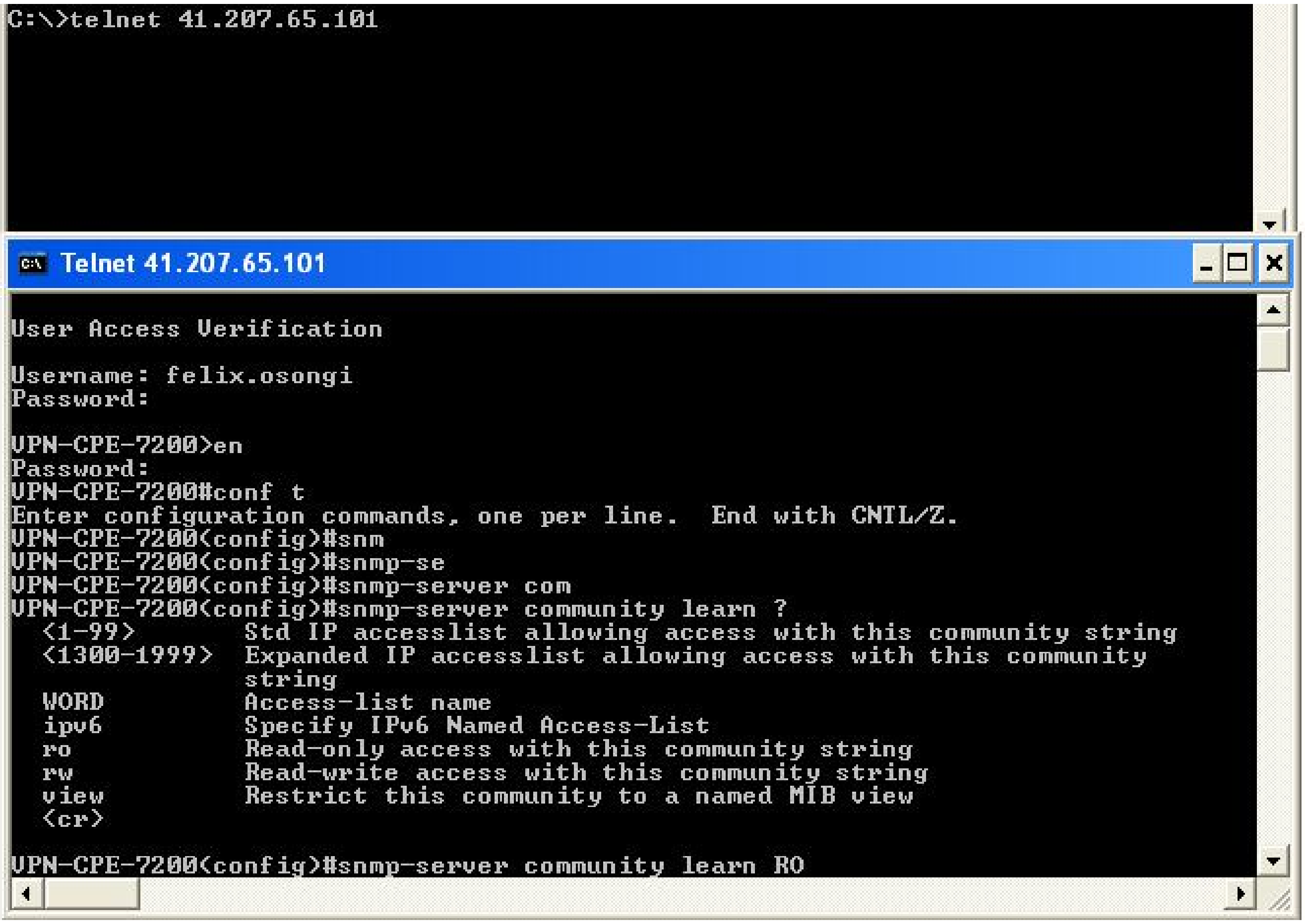
This project takes a look at the step by step procedure of initiating usage monitoring bandwidth usage using Cacti and uptime statistics using Nagios.

4.1.3 SNMP CONFIGURATIONS:

CPE preparation For SNMP:

The person to initiate monitoring for a Client Premise Equipment (CPE) has to have administrator logon credentials so as to activate SNMP and configure the SNMP string that will be queried by the server.

We first telnet into the router and provide the logon details as provided by the administrator;

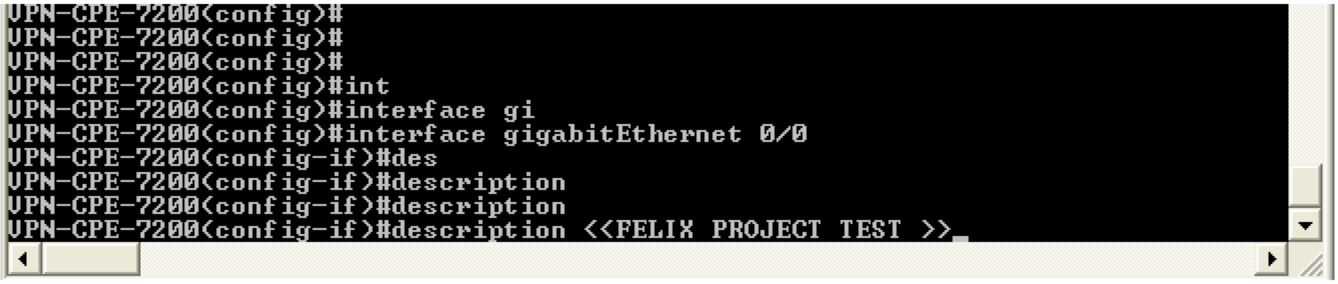


```
C:\>telnet 41.207.65.101
```

```
User Access Verification

Username: felix.osongi
Password:

VPN-CPE-7200>en
Password:
VPN-CPE-7200#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
VPN-CPE-7200(config)#snm
VPN-CPE-7200(config)#snmp-se
VPN-CPE-7200(config)#snmp-server com
VPN-CPE-7200(config)#snmp-server community learn ?
  <1-99>       Std IP accesslist allowing access with this community string
  <1300-1999>  Expanded IP accesslist allowing access with this community
               string
  WORD         Access-list name
  ipv6         Specify IPv6 Named Access-List
  ro           Read-only access with this community string
  rw           Read-write access with this community string
  view         Restrict this community to a named MIB view
  <cr>

VPN-CPE-7200(config)#snmp-server community learn RO
```

Figure 9: Creating SNMP server community string

We configured the SNMP server community string and set the access rights to Read only (RO). The interfaces to be monitored were then described so as to be able to be visible from the SNMP server once an SNMP walk is done.

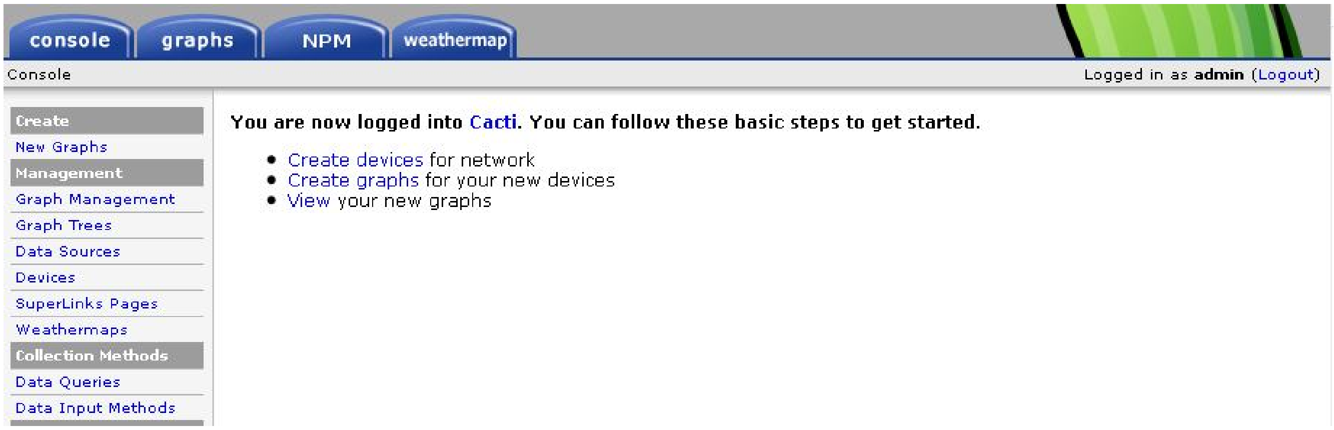


```
VPN-CPE-7200(config)#
VPN-CPE-7200(config)#
VPN-CPE-7200(config)#
VPN-CPE-7200(config)#int
VPN-CPE-7200(config)#interface gi
VPN-CPE-7200(config)#interface gigabitEthernet 0/0
VPN-CPE-7200(config-if)#des
VPN-CPE-7200(config-if)#description
VPN-CPE-7200(config-if)#description
VPN-CPE-7200(config-if)#description <<FELIX PROJECT TEST >>_
```

Figure 10: Describing an Interface in a Cisco Router

The figure above shows how a Cisco interface is described to be able to initiate polling.

Once the above is accomplished, the admin logs onto cacti and follows the following steps:



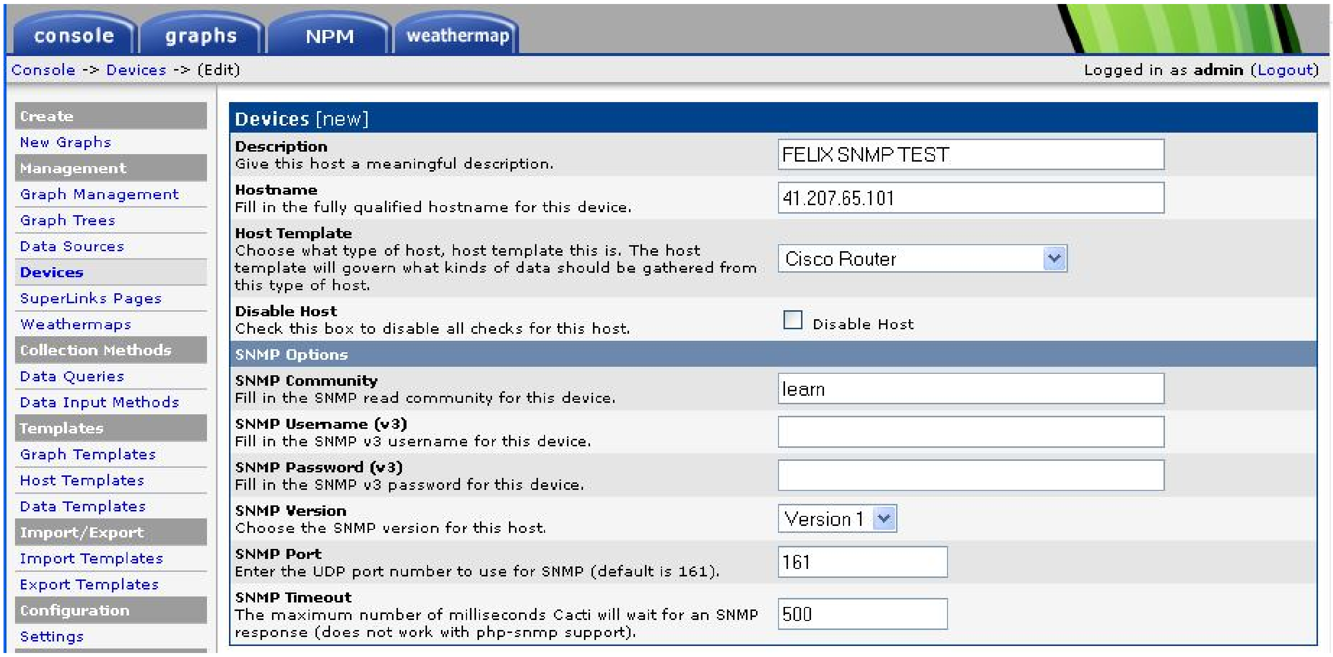
You are now logged into Cacti. You can follow these basic steps to get started.

- Create devices for network
- Create graphs for your new devices
- View your new graphs

Figure 11: Successful Logon to Monitoring system.

The figure above shows the default page once an administrator logs onto Cacti Network monitoring system.

The administrator then adds the Client Device to be monitored and saves as below:



Devices [new]

Description
Give this host a meaningful description.
FELIX SNMP TEST

Hostname
Fill in the fully qualified hostname for this device.
41.207.65.101

Host Template
Choose what type of host, host template this is. The host template will govern what kinds of data should be gathered from this type of host.
Cisco Router

Disable Host
Check this box to disable all checks for this host.
□ Disable Host

SNMP Options

SNMP Community
Fill in the SNMP read community for this device.
learn

SNMP Username (v3)
Fill in the SNMP v3 username for this device.

SNMP Password (v3)
Fill in the SNMP v3 password for this device.

SNMP Version
Choose the SNMP version for this host.
Version 1

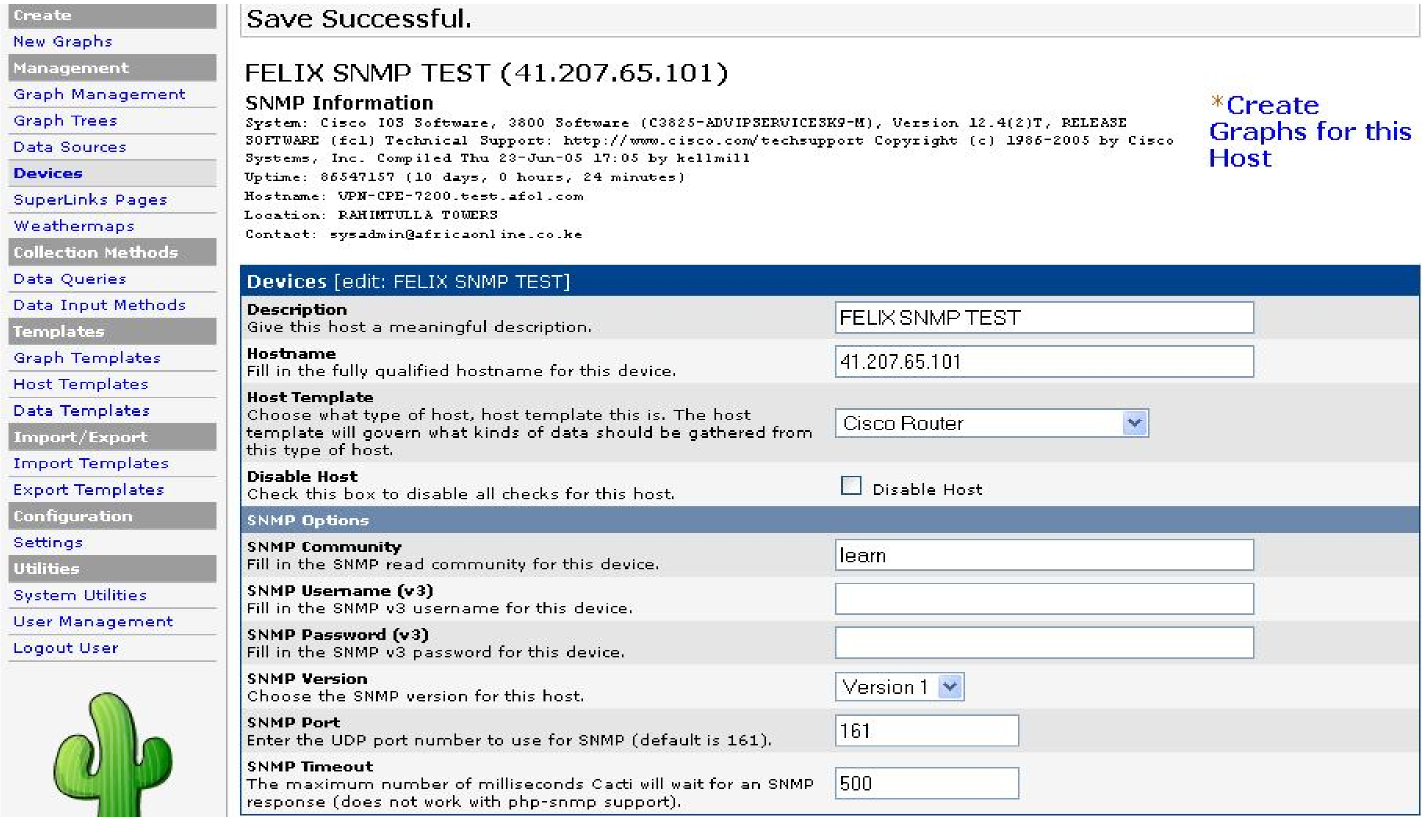
SNMP Port
Enter the UDP port number to use for SNMP (default is 161).
161

SNMP Timeout
The maximum number of milliseconds Cacti will wait for an SNMP response (does not work with php-snmp support).
500

Figure 12:Device Specifications:

Upon successful query of the CPE, the server sends the community string to the CPE which verifies the string and communicates its readiness to the server as below.

If the CPE does not respond or there is a mismatch in the parameters, the server generates an error and the information is not displayed.



Save Successful.

FELIX SNMP TEST (41.207.65.101)

SNMP Information

```
System: Cisco IOS Software, 3800 Software (C3825-ADVIPSERVICESK9-M), Version 12.4(2)T, RELEASE
SOFTWARE (fc1) Technical Support: http://www.cisco.com/techsupport Copyright (c) 1986-2005 by Cisco
Systems, Inc. Compiled Thu 23-Jun-05 17:05 by kellmill
Uptime: 86547157 (10 days, 0 hours, 24 minutes)
Hostname: VPN-CPE-7200.test.afol.com
Location: RAHIMTULLA TOWERS
Contact: sysadmin@africaonline.co.ke
```

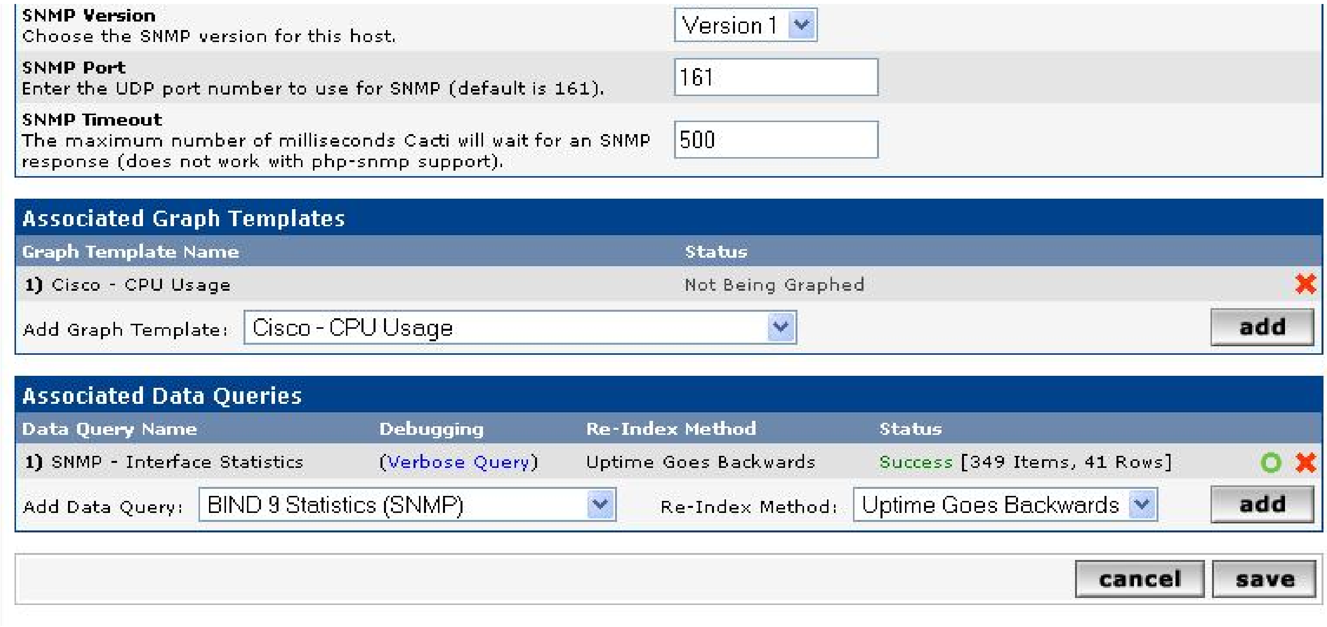
Devices [edit: FELIX SNMP TEST]

Description Give this host a meaningful description.	FELIX SNMP TEST
Hostname Fill in the fully qualified hostname for this device.	41.207.65.101
Host Template Choose what type of host, host template this is. The host template will govern what kinds of data should be gathered from this type of host.	Cisco Router
Disable Host Check this box to disable all checks for this host.	□ Disable Host
SNMP Options	
SNMP Community Fill in the SNMP read community for this device.	learn
SNMP Username (v3) Fill in the SNMP v3 username for this device.	
SNMP Password (v3) Fill in the SNMP v3 password for this device.	
SNMP Version Choose the SNMP version for this host.	Version 1
SNMP Port Enter the UDP port number to use for SNMP (default is 161).	161
SNMP Timeout The maximum number of milliseconds Cacti will wait for an SNMP response (does not work with php-snmp support).	500

Figure 13: Configuring SNMP parameters;

The figure above indicates a successful SNMP query information meaning appliance is ready for polling.

The administrator scrolls down and chooses the graph template to use and associates t he Queries to SNMP as below:



SNMP Version Choose the SNMP version for this host.	Version 1
SNMP Port Enter the UDP port number to use for SNMP (default is 161).	161
SNMP Timeout The maximum number of milliseconds Cacti will wait for an SNMP response (does not work with php-snmp support).	500

Associated Graph Templates

Graph Template Name	Status	
1) Cisco - CPU Usage	Not Being Graphed	✕
Add Graph Template: Cisco - CPU Usage		add

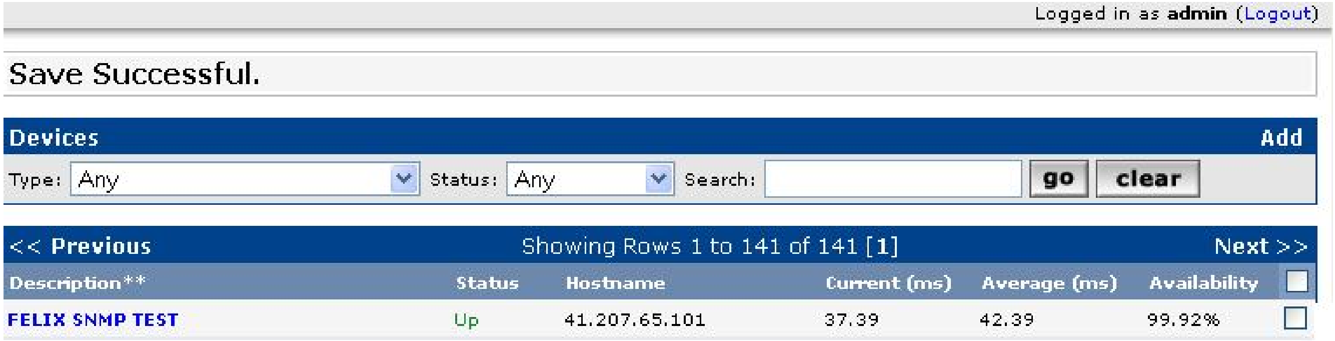
Associated Data Queries

Data Query Name	Debugging	Re-Index Method	Status	
1) SNMP - Interface Statistics	(Verbose Query)	Uptime Goes Backwards	Success [349 Items, 41 Rows]	○ ✕
Add Data Query: BIND 9 Statistics (SNMP)		Re-Index Method: Uptime Goes Backwards		add

Figure 14: Choosing SNMP Template;

The figure above shows the choices to be made by the admin so as to initiate polling; the correct version and graph template have to be chosen before proceeding to poll the interface.

The device status is noted as UP and the administrator can now check on the device to initiate polling for the described interfaces:



Logged in as **admin** (Logout)

Save Successful.

Devices Add

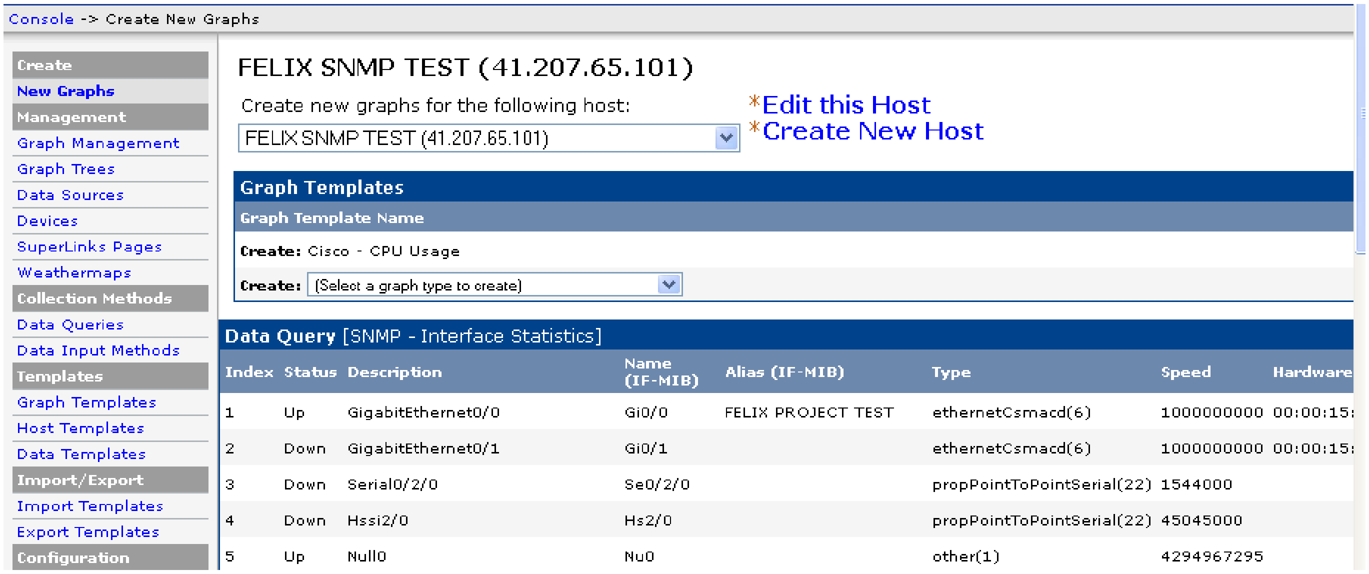
Type: Any Status: Any Search:

<< Previous Showing Rows 1 to 141 of 141 [1] Next >>

Description**	Status	Hostname	Current (ms)	Average (ms)	Availability	□
FELIX SNMP TEST	Up	41.207.65.101	37.39	42.39	99.92%	□

Figure 15: SNMP device Status:

The figure above shows the device status as "UP" meaning it's ready for polling and the administrator clicks at the device and selects create graph, which prompts the administrator to select the interface to graph as below:



FELIX SNMP TEST (41.207.65.101)

Create new graphs for the following host: FELIX SNMP TEST (41.207.65.101)

Graph Templates

Graph Template Name
Create: Cisco - CPU Usage
Create: (Select a graph type to create)

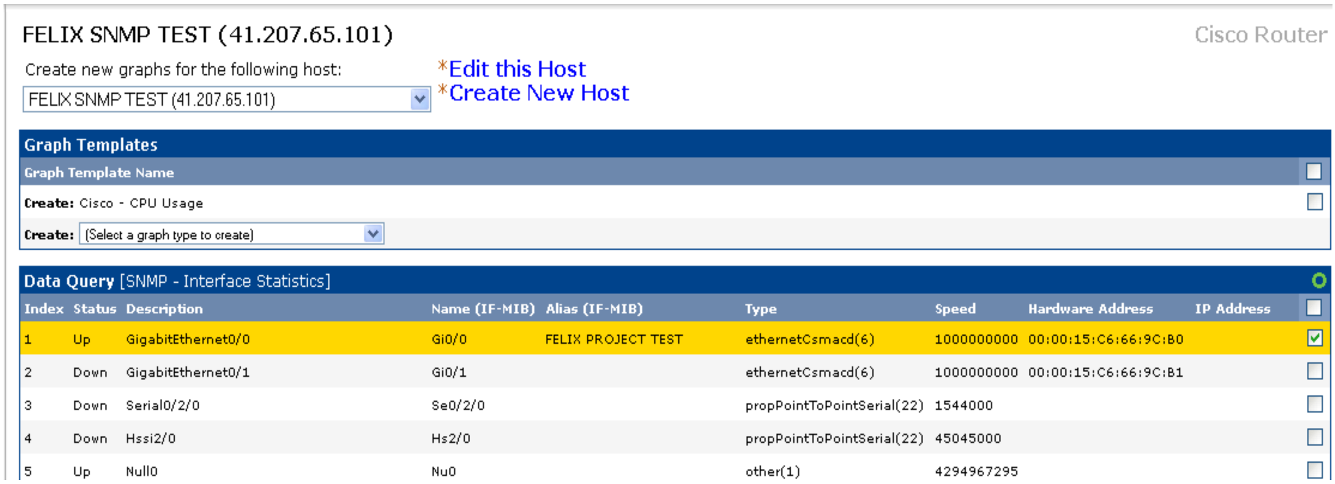
Data Query [SNMP - Interface Statistics]

Index	Status	Description	Name (IF-MIB)	Alias (IF-MIB)	Type	Speed	Hardware
1	Up	GigabitEthernet0/0	Gi0/0	FELIX PROJECT TEST	ethernetCsmacd(6)	1000000000	00:00:15:
2	Down	GigabitEthernet0/1	Gi0/1		ethernetCsmacd(6)	1000000000	00:00:15:
3	Down	Serial0/2/0	Se0/2/0		propPointToPointSerial(22)	1544000	
4	Down	Hssi2/0	Hs2/0		propPointToPointSerial(22)	45045000	
5	Up	Null0	Nu0		other(1)	4294967295	

Figure 16: Viewing Available Device Interfaces:

The figure shows the interfaces ready for polling as picked by SNMP.

For this project we selected the appropriate interface to be graphed as below:



FELIX SNMP TEST (41.207.65.101)

Create new graphs for the following host: FELIX SNMP TEST (41.207.65.101)

Graph Templates

Graph Template Name	□
Create: Cisco - CPU Usage	□
Create: (Select a graph type to create)	

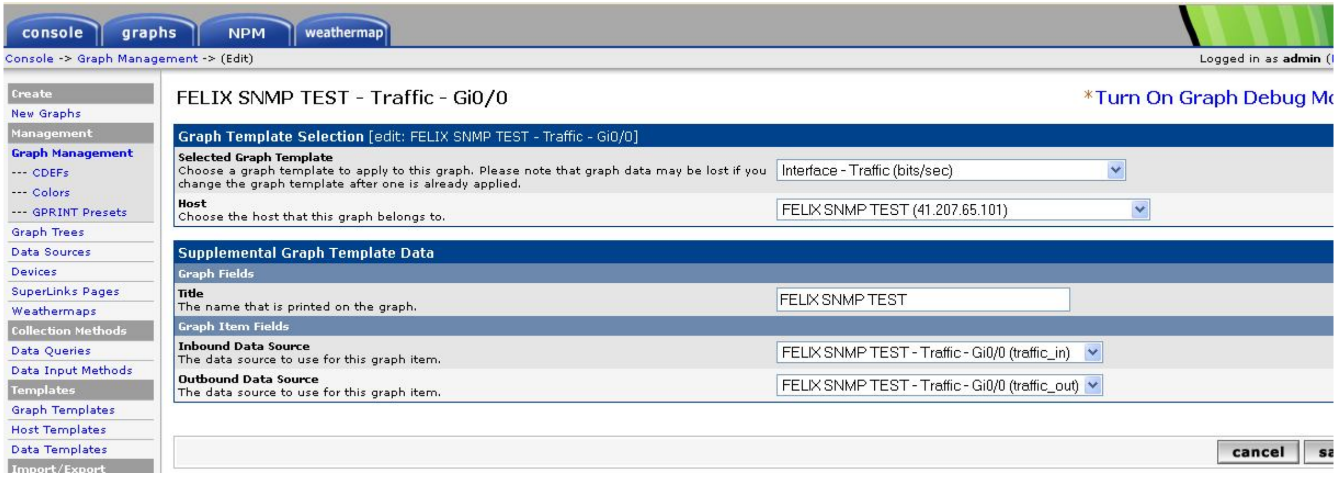
Data Query [SNMP - Interface Statistics]

Index	Status	Description	Name (IF-MIB)	Alias (IF-MIB)	Type	Speed	Hardware Address	IP Address	□
1	Up	GigabitEthernet0/0	Gi0/0	FELIX PROJECT TEST	ethernetCsmacd(6)	1000000000	00:00:15:C6:66:9C:B0		☑
2	Down	GigabitEthernet0/1	Gi0/1		ethernetCsmacd(6)	1000000000	00:00:15:C6:66:9C:B1		□
3	Down	Serial0/2/0	Se0/2/0		propPointToPointSerial(22)	1544000			□
4	Down	Hssi2/0	Hs2/0		propPointToPointSerial(22)	45045000			□
5	Up	Null0	Nu0		other(1)	4294967295			□

Figure 17: Choosing an Interface to be Polled by SNMP:

The figure above shows a selection made o Gigabit Ethernet 0/0 which describes the link being polled.

Under graph management, the administrator selects the node and gives the description of the graph to be displayed on the web browser and clicks at save.



FELIX SNMP TEST - Traffic - Gi0/0

Graph Template Selection [edit: FELIX SNMP TEST - Traffic - Gi0/0]

Selected Graph Template
Choose a graph template to apply to this graph. Please note that graph data may be lost if you change the graph template after one is already applied. — Interface - Traffic (bits/sec)

Host
Choose the host that this graph belongs to. — FELIX SNMP TEST (41.207.65.101)

Supplemental Graph Template Data

Graph Fields

Title
The name that is printed on the graph. — FELIX SNMP TEST

Graph Item Fields

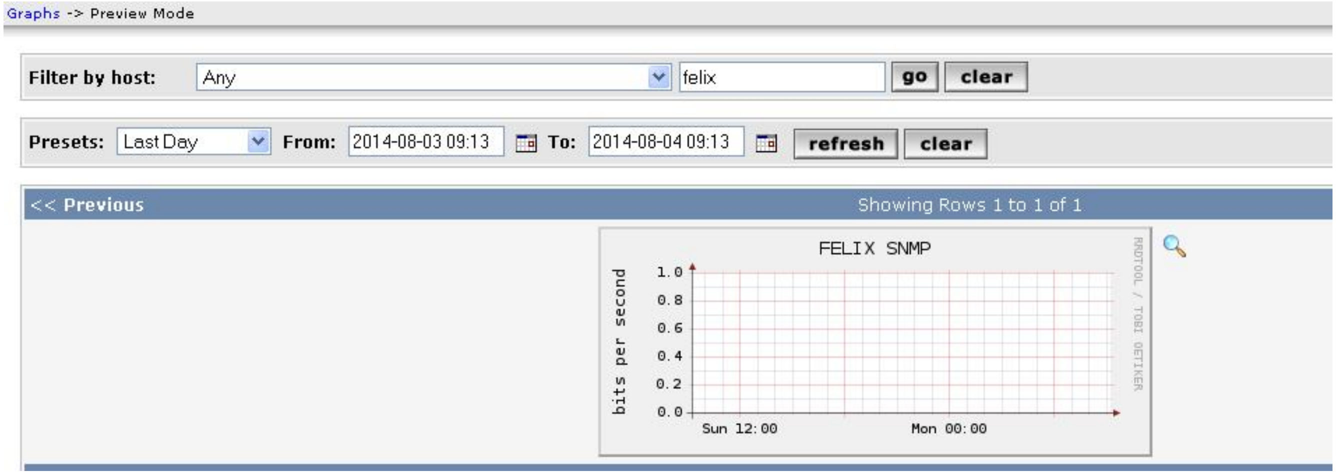
Inbound Data Source
The data source to use for this graph item. — FELIX SNMP TEST - Traffic - Gi0/0 (traffic_in)

Outbound Data Source
The data source to use for this graph item. — FELIX SNMP TEST - Traffic - Gi0/0 (traffic_out)

Figure 18: Creating the Graph:

The figure above shows the final steps of initiating polling by describing the graph and shoosing the interfaces for both inbound and outbound traffic.

Once the above process is successfully followed, the graph for the usage of the CPE interface Gigabit Ethernet 0/0 is created with the given described name, the graph is blank until traffic starts to be noted on the given interface.



Filter by host: Any — felix

Presets: Last Day — From: 2014-08-03 09:13 — To: 2014-08-04 09:13

Figure 19: Preview of the Graph Created:

The figure above shows usage graph builds based on usage on the interface being monitored.

Below is the usage of the interface after thirty (30) minutes of monitoring.

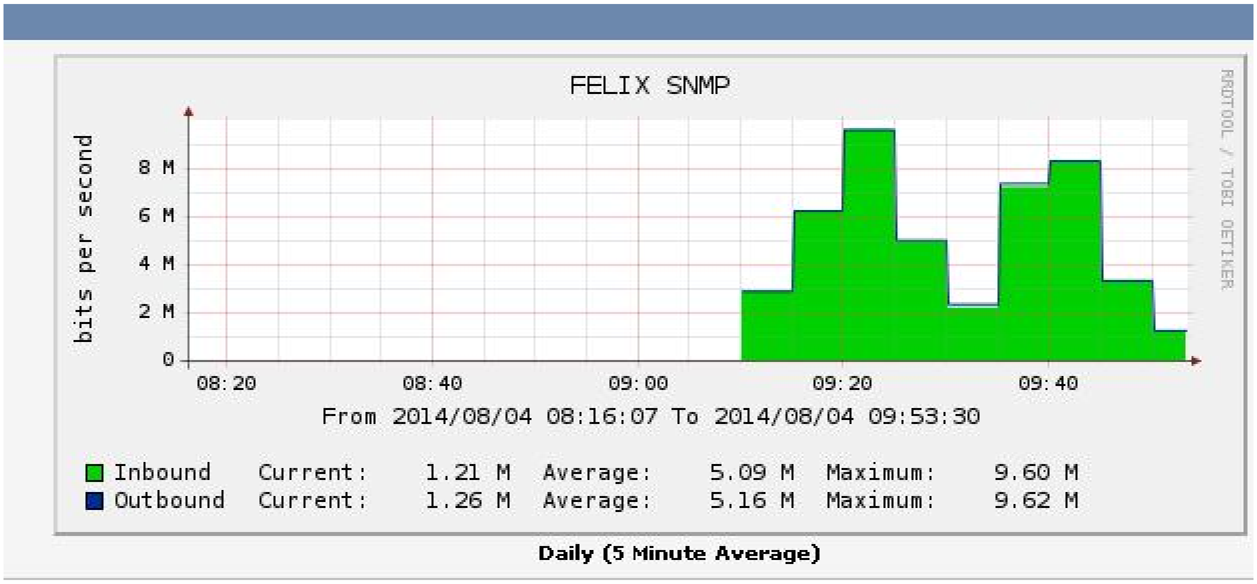


Figure 20: SNMP collecting Usage Data from the Device:

The above explains how SNMP is initiated and usage graphs presented to the user. It’s noted that any change in monitoring parameters of the CPE needs configuration change of parameters to be monitored from the server and hence more time is consumed making changes in order to keep to date monitoring statistics. A change in the Client CPE, IP address and/or SNMP community string means configuration change at both ends as the protocol has to be re initialized afresh.

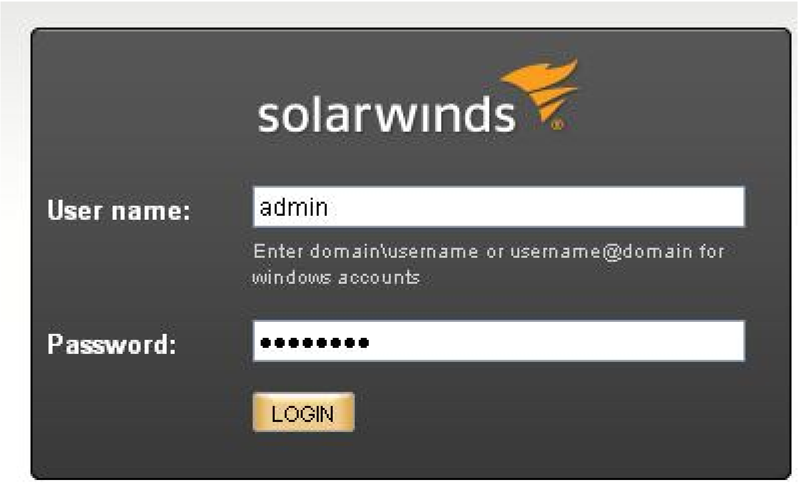
4.1.4 NAGIOS CONFIGURATIONS:

This project also took a look at how Nagios NPM is used to monitor Up time of the links.

For Nagios, it works with ICMP(Ping) as the server sends requests to the client and if the client is available and ping requests are enabled, then the server records uptime of the node else the server records the node to be down.

Below is a sample configuration of the same node used for capturing usage graph. In this case we monitor uptime status

The administrator logs onto the system with the administrator credentials.



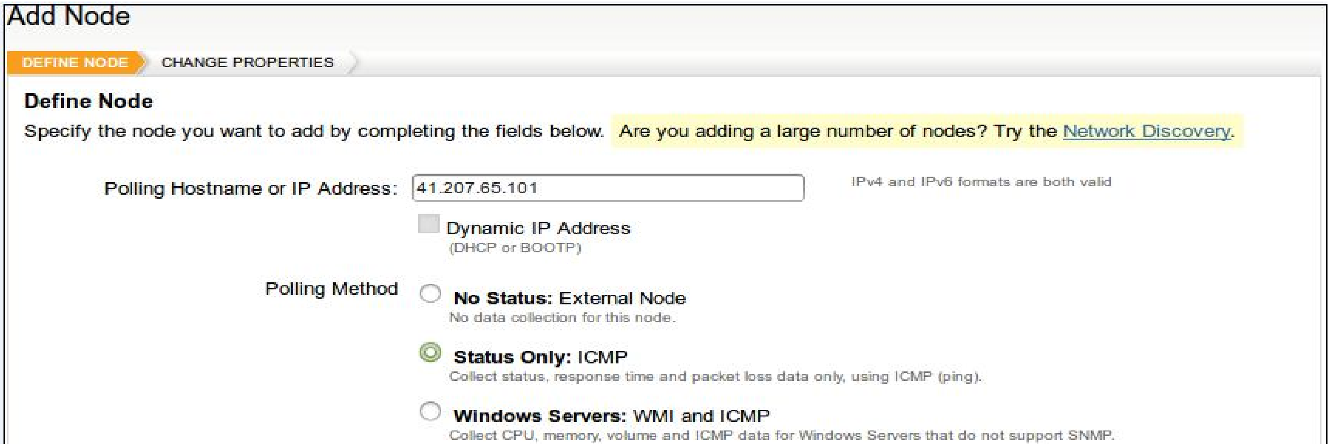
User name: admin

Enter domain\username or username@domain for windows accounts

Password: •••••••••

Figure 21: Logon to SolarWinds:

The IP of the node to be monitored is added and saved.



Add Node

Define Node

Specify the node you want to add by completing the fields below. Are you adding a large number of nodes? Try the Network Discovery.

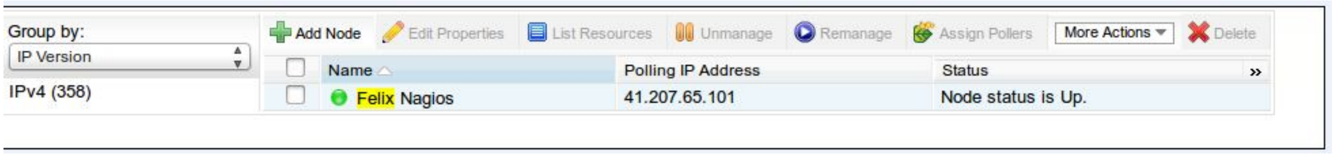
Polling Hostname or IP Address: 41.207.65.101 IPv4 and IPv6 formats are both valid

□ Dynamic IP Address
(DHCP or BOOTP)

Polling Method
○ **No Status:** External Node
No data collection for this node.
● **Status Only:** ICMP
Collect status, response time and packet loss data only, using ICMP (ping).
○ **Windows Servers:** WMI and ICMP
Collect CPU, memory, volume and ICMP data for Windows Servers that do not support SNMP.

Figure 22: Adding Device to be monitored:

Once saved, the node is monitored UP or Down once the CPE is reachable on ping. Below is a capture of the node status upon initializing monitoring.



Group by: IP Version

IPv4 (358)

	Name	Polling IP Address	Status
□	● Felix Nagios	41.207.65.101	Node status is Up.

Figure 23: Confirming Device Status:

Once the nodes are monitored, any change to the IP and/or name of the node has to be done from the server so as to reflect from the monitoring web page.

4.2 A NALYSIS OF PROPOSED MAS SYSTEM

4.2.1 User requirements

User requirements specify what the user wants to achieve from the proposed system. These are the problems which the proposed system should solve. The following are the user requirements

i. Interaction with the System Administrator and users using the GUI interface.
ii. Ability to pick necessary details from an existing Data base
iii. Regular polling intervals
iv. Ability to pick new details and update existing automatically if a change is done on the Database.
v. Ability to store monitored data over time for reference purposes.
vi. Ability to present the results in a graphical manner.
vii. Ability to indicate UP time and Down time of all clients on a GUI interface.

4.2.2 System requirements

System requirements (specifications) include the description of

i. The inputs to the process and output.
ii. The operations the system performs for each input.
iii. The output obtained for the corresponding input

Agent-based Network Monitoring System has been developed using the following agent programs

i. POP Agents
ii. Persistence agent
iii. Client Agent
iv. NOC Agent
v. Platform Monitor Agent
vi. RMA, DF and AMS Agents

POP Agent

This agent picks receives communications from the NOC agent about details of clients related to the particular region of operation. The Agent keeps details of the Clients within its area and keeps receiving updates from the NOC Agent about new clients, client status, modifications and/or deletion of client records. The Agent is also responsible for communicating the information to the clients. It in reverse receives Usage and uptime status from the clients that is then communicated to the NOC Agent.

Client Agent

This agent is located at the clients end and is created after details are entered to the Database and sent to the relevant POP Agent via the NOC agent. The client agent monitors the Uptime and bandwidth usage of the client link and sends the information to the POP which is then communicated to the NOC and presented in a graphical format.

NOC Agent

This agent receives information from the client database from the persistence agent and relays to the relevant POP agent. It's the Agent responsible for the whole

RMA, DF and AMS agents

These are system agents, shipped with the JADE distribution; HostMonitor can monitor remote networks using Remote Monitoring Agents (RMA). RMA is small application that accepts requests from HostMonitor, performs test (or action) and provides information about test result back to HostMonitor.

Persistence Agent

This handles database operations for the platform.

Platform Monitor Agent

This agent initializes the platform and agents. It creates a link to the platform database and it ensures that all the required parts of the platform are running, these may include database and all the required agents.

4.3 Analysis using PROMETHEUS concepts

The analysis of this system was done using multi-agent system development known as PROMETHEUS which was used for designing intelligent software agents and agent systems. The methodology provides detailed guidance in terms of processes as well as notations. The analysis models for the system are represented using the concept and symbols for PROMETHEUS methodology. The following is a list of various agents which constitute the system, the service/tasks to be performed, goals to be achieved and the collaborators.

a) External system Agents

Tasks/services

- Register and edit clients on the Client Database

- provide access to the external systems registered onto the platform

- perform search for requested data

Main Goal

-Put data into one standard format understood by all external systems.

Collaborators

1. NOC Agents
2. Persistence Agents

b) NOC Agent

Tasks

- Handles registration to and deregistration from the platform by external system via the Persistence Agent.

- Communicates to the POP Agents.

c) Platform Monitor Agent

Tasks/Services

- Initializes the platform and agents.
- Creates a link to the platform database

Goal

- Ensures that all the required parts of the platform are running, these may include database and all the required agents

Collaborators

- NOC Agent
- POP Agents
- Persistence Agent
- Client Agents

d) Persistence Agent

Task/Service and Goal

- Handles database operations for the platform

4.4 User interfaces

Each agent in the system should know the identity of other agents to interact with in pursuit of its goals. For this Network monitoring system, the identities of other agents must be configured during system installation or modification. Configuration of the agent identities will ensure that the right information is delivered and received by the right agent. For example, if a new external system agent joins the existing service, the existing agents will have to include its identity to their databases.

All these configurations and modifications are done by the systems Administrator. In order to facilitate system configuration and modification, each agent program has at least a configuration interface. This interface is either command-based or graphical user interface.

Apart from system configuration and modification done by system administrators, other system users would also wish to obtain some information. The interaction between the system users and the system can also be provided by using command-based or GUI. For ease of user interaction, user interfaces are provided using GUIs. The data received from the Client Agents about Usage and Uptime status is also represented in a graphical user interface.

4.5 MAS ANALYSIS MODEL AND VIEW

Analysis was carried out to produce system specification. The specification consisted of a collection of views of the system to be developed and its environment. It facilitated communication among the people involved in the system development. The Agent-based Network monitoring system has been analyzed accordingly to the modal views defined in the PROMETHEUS using and developing subsets of entity and relationship concepts. Prometheus defines a proper detailed process to specify, implement, test/debug agent-oriented software systems.

It offers a set of detailed guidelines that includes examples and heuristics, which help better understanding what is required in each step of the development. This process incorporates three phases.

The system specification phase identifies the basic goals and functionalities of the system, develops the use case scenarios that illustrate the functioning of the Prometheus system, and specifies which are the inputs (percepts) and outputs (actions). It obtains the scenarios diagram, goal overview diagram, and system roles diagram.

The architectural design phase uses the outputs produced in the previous phase to determine the agent types that exist in the system and how they interact. It obtains the data coupling diagram, agent-role diagram, agent acquaintance diagram, and system overview diagram.

The detailed design phase centers on developing the internal structure of each agent and how each agent will perform his tasks within the global system. It obtains agent overview and capability overview diagrams. Finally, Prometheus details how the entities obtained during the design are transformed into the concepts used in a specific agent-oriented programming language.

4.5.1 SYSTEM SPECIFICATION

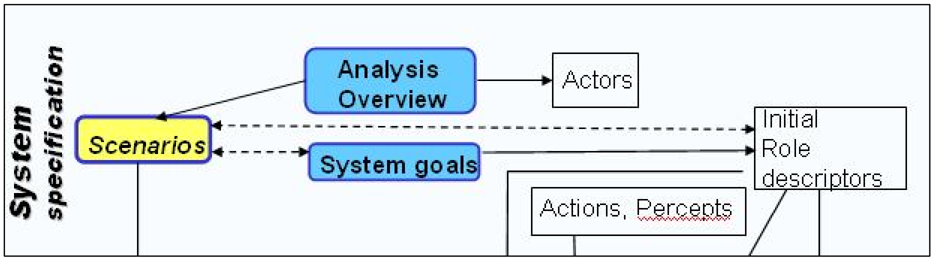
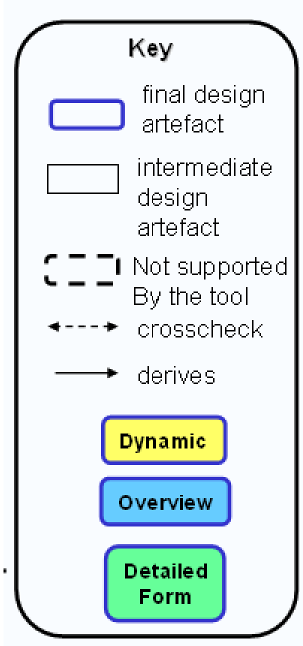


Figure 24: Promethius Methodology System Specifications:

This is the initial phase of Prometheus methodology for building MAS. The phase involved specifying the system requirements using goals and scenarios and identifying the actions and percepts.

Use of case scenarios were developed together with the agents interface to the environment.

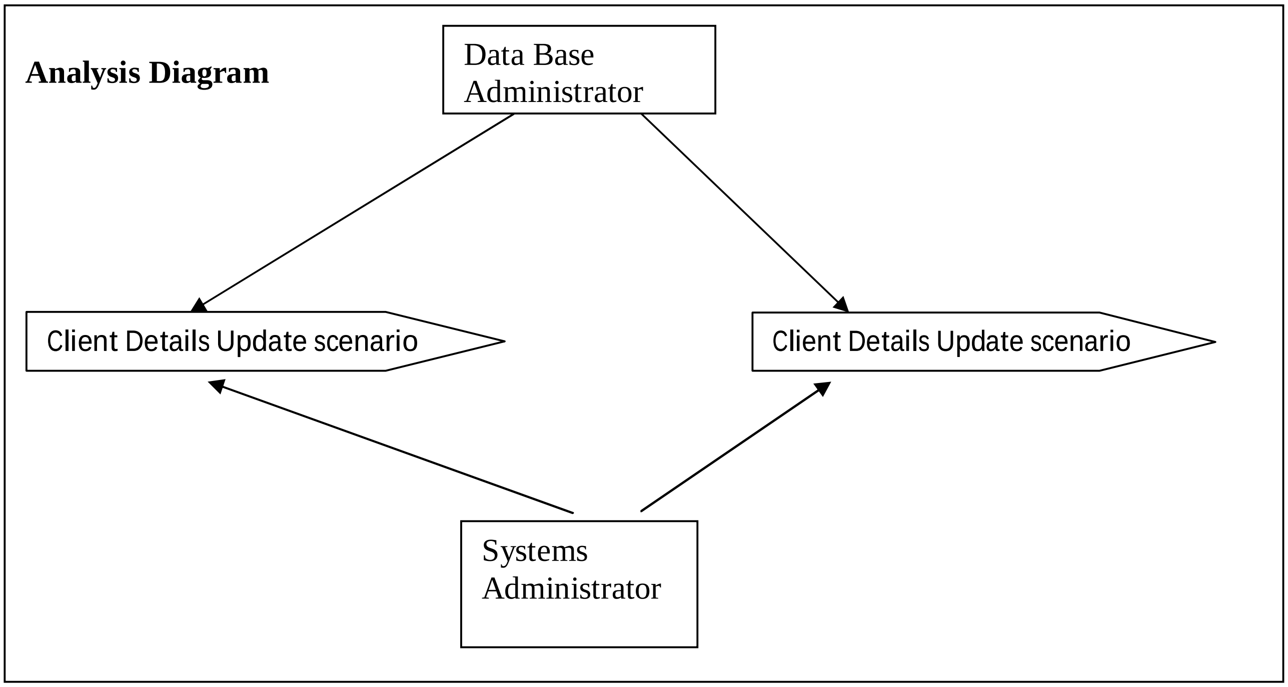


During the research study this phase was first phase of Prometheus methodology which was considered as the pre-design stage, the diagram shows the overview diagram of System specification and its design artifacts. It is not unusual for the initial ideas for a system to be captured very briefly, possibly in a few paragraphs. During System Specification this description must be elaborated and explored, to provide a sound basis for system design and development. In this research, an Agent-based Network Monitoring was described as a system with four distinct agents which makes the platform to function

Typically, using Prometheus, the development of the System Specification began with identifying the external entities which were referred to as *actors* that used or interacted in some way with the system, and the key *scenarios* around which interaction occurred.

In the figure below the following Systems and data Base administrators were identified as the entities that would interact with the system. They were associated with some main scenarios which corresponded to the main functionality of the system

Figure 25: Analysis Design:


Analysis Diagram
Data Base Administrator
Client Details Update scenario
Client Details Update scenario
Systems Administrator

After coming up with the above diagram, percepts as input to each scenario were identified and the actions produced by the system for each scenario linking them to the appropriate actors were also identified. For example, a systems administrator registers a new client on the system as a percept (input) to the system and the system performs an action of sending information to the relevant POP.

The analysis diagram thus defines the *interface* to the system in terms of the percepts (inputs) and actions (outputs).

The next step was to specify the details of the scenarios that we identified in the analysis overview diagram. A scenario is a sequence of *structured steps* where each step can be one of: *goal, action, percept*, or *(sub) scenario*. Each step also allows the designer to indicate the *roles* associated with that step, the *data* accessed, and a *description* of the step. These preliminary goals, roles and data that are identified are used to automatically propagate information into other aspects of the design. As steps are defined, the relevant entities are created if they do not yet exist.

System Description:

Number of cooperate customers get signed up for service provision in different regions of the country and are to be registered in a central data base for billing and management purposes. The goals of the system and use cases were used to analyze the requirements of the system.

4.5.1.1 System Goals

This describes what is the system is build for.

Overall goal

The overall goal of the system is to ensure all clients links are monitored from the NOC in a more responsive manner. Once client details are entered or updated into the database, the NOC agent informs the relevant POP AGENT which then communicates with the client Agents and relays the required data to the NOC in a timely manner. The system is to eliminate manual intervention of the systems administrators in initiating polling for any corporate client links in the system.

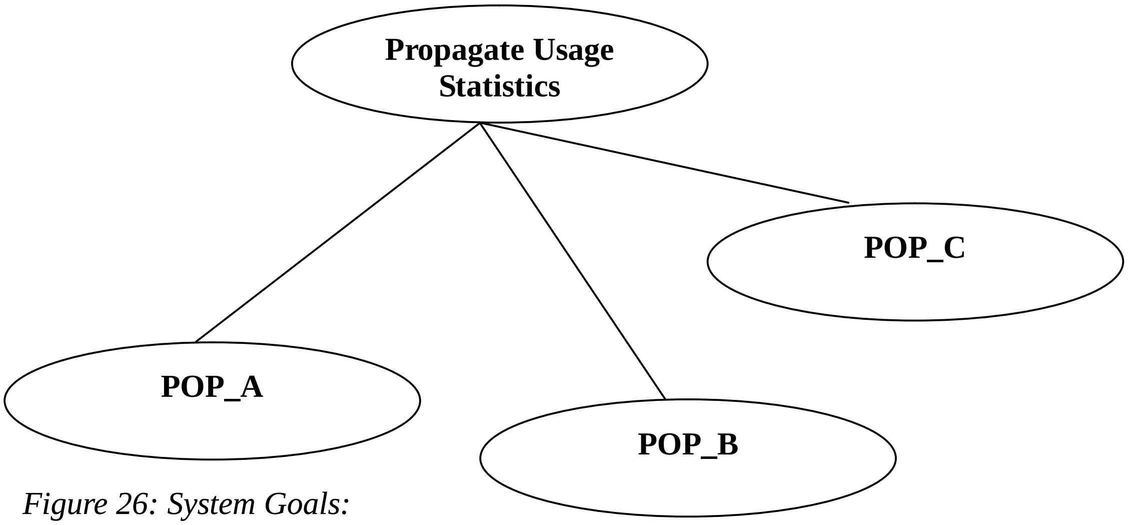


Figure 26: System Goals:

4.5.1.2 Use case scenarios:

A use case is a view of the functionality (or use) of a system from the user’s perspective. Use case scenarios represent a particular instance of the system without branching. Use case analysis has been used for requirements specification of the multi-agent system and the behavior specification.

Use case 1: The systems administrator or Database administrator logs on to the system and proceeds to register a client into the client database. The administrator activates the service after verifying any physically signed job completion forms from the installers and saves the information.

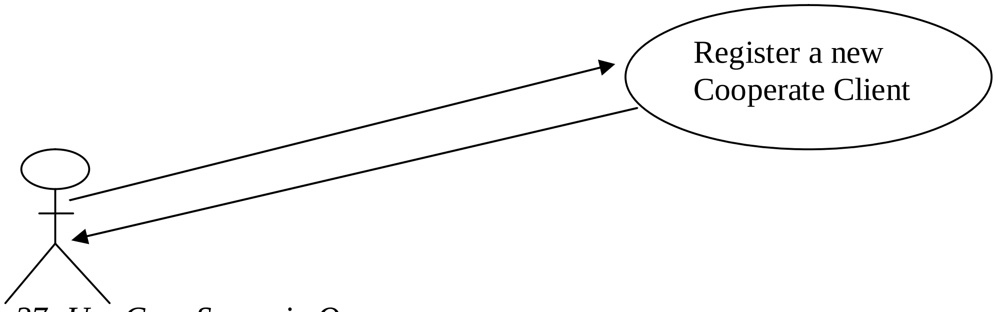


Figure 27: Use Case Scenario One:

Use case2: Updating: the entities are updated on the database and reflected to the client. The clients get the information from the respective POPs and adjust appropriately.

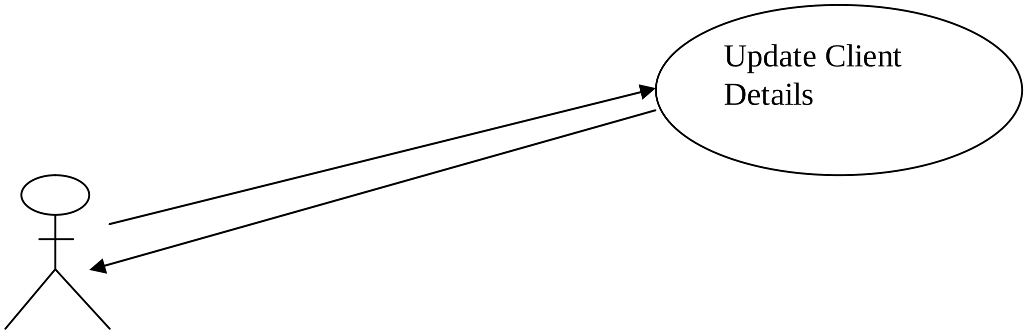


Figure 28: Use Case Scenario Two:

Use case 3: A system user can want to view usage and uptime status of a specific Client from the management tool.

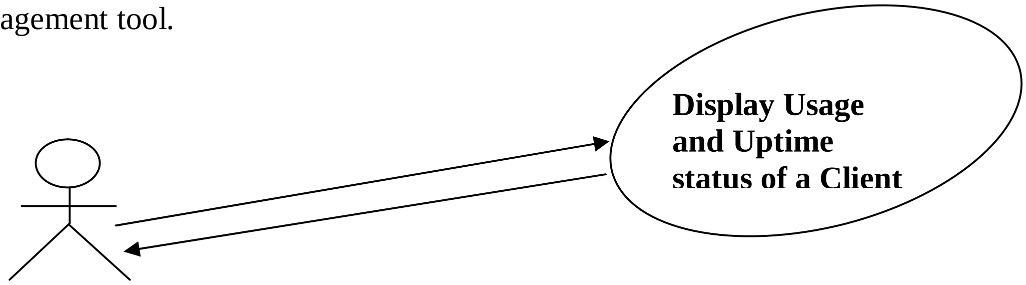


Figure 29: Use Case Scenario Three:

4.5.1.3 Actors

Main actors in the system are:

i. Systems and Database administrators

Once an account manager from the organizations Point of Presence gathers information about a prospect and follows the lead to closure of the account, the details entered into the system manually by the administrators or through a ticketing system. The information captured includes Name of the Client, Contact details, Region, Bandwidth required and any other relevant information.

Upon successful completion of the job, the administrator assigns an IP address and activates the Link so as to initialize monitoring.

ii. Users

The users of the system will query the system for usage and uptime status of the specific clients on demand. The users will have logon credentials to be able to view the information

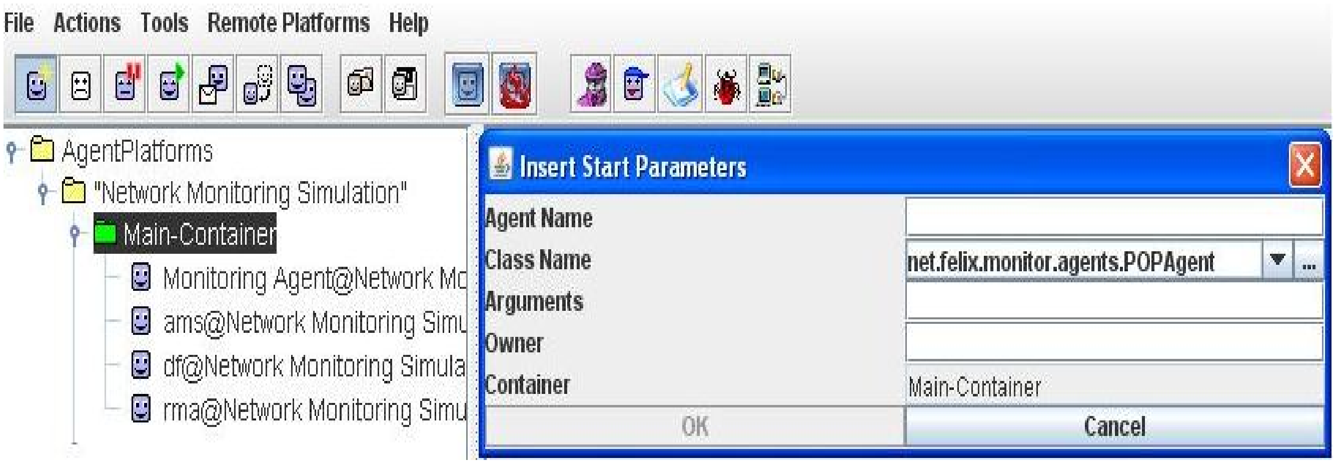
iii. Percepts and Actions

The multi-agent based system for network monitoring include the system admin receiving a new sign up form with job completion and entering it into the agent system, users searching the system for usage and uptime information from the system. Actions include sending information to the relevant POPs and receiving usage and Uptime information from the clients.

System Functionality

The system functionalities are identified as: Creation of new POPs assigning the IP subnet, Sending information to the relevant POP about a newly created or updated client and receiving usage and uptime statistics from the clients and updating in the system for users to be able to view.

Figure 30: Creating New Agent



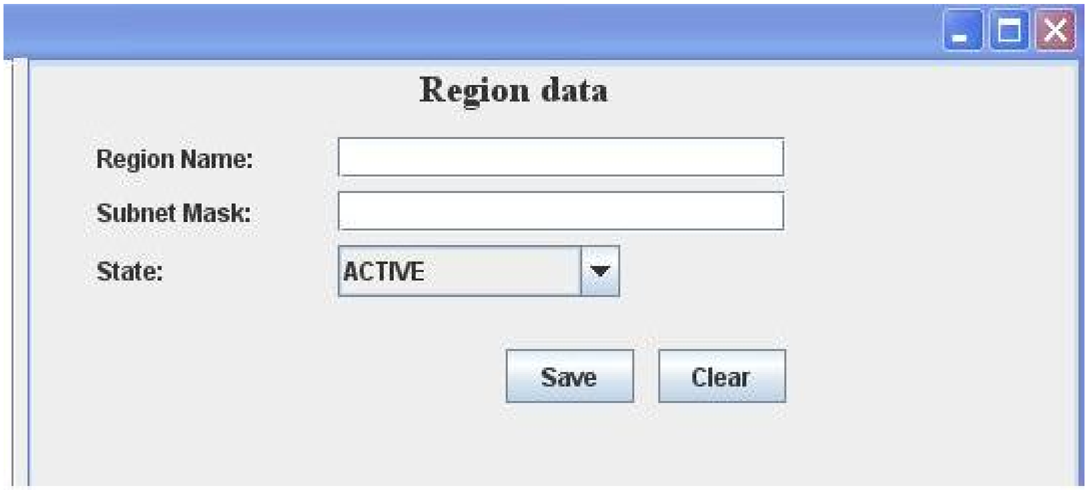
Insert Start Parameters

Agent Name:
Class Name: net.felix.monitor.agents.POPAgent
Arguments:
Owner:
Container: Main-Container

File
Actions
Tools
Remote Platforms
Help
AgentPlatforms
"Network Monitoring Simulation"
Main-Container
Monitoring Agent@Network Mo
ams@Network Monitoring Simu
df@Network Monitoring Simula
rma@Network Monitoring Simu
OK
Cancel

The figure above shows how a new Agent is created in the Agent management platform and placed in a specific Agent container

Figure 31: Creating New Region/POP:



Region data

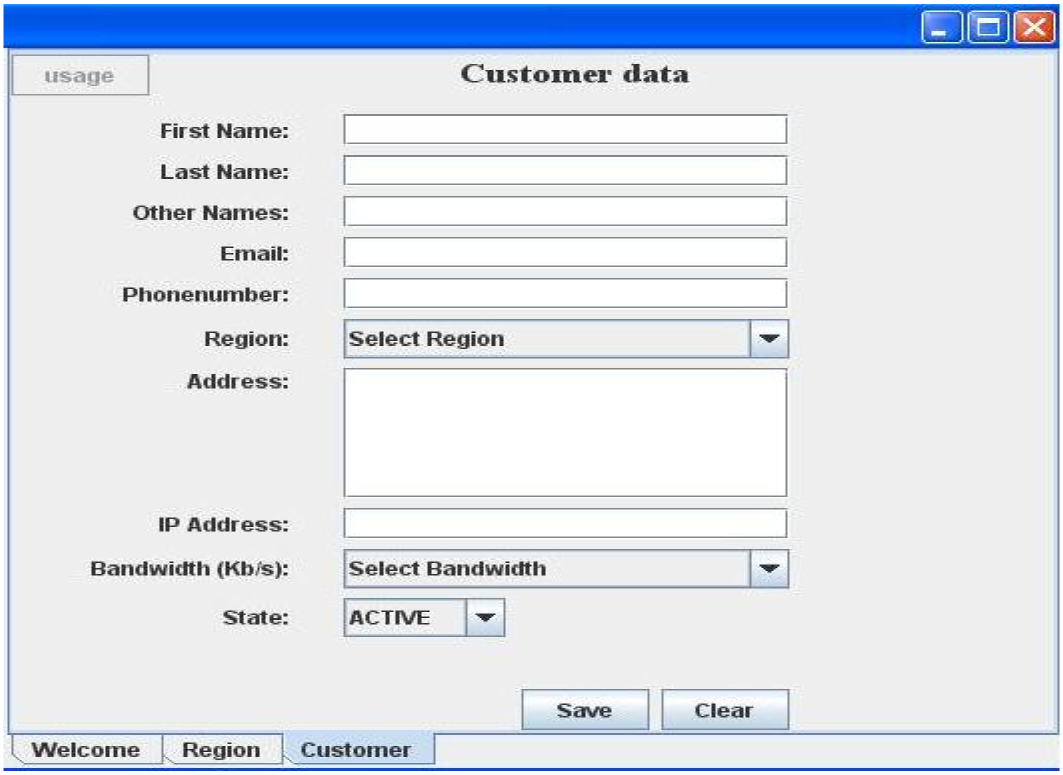
Region Name:

Subnet Mask:

State: ACTIVE

The figure above shows how a new region/Point of Presence (PoP) is created in the system.

Figure 32: Creating New Client:



Customer data

First Name:

Last Name:

Other Names:

Email:

Phonenumber:

Region: Select Region

Address:

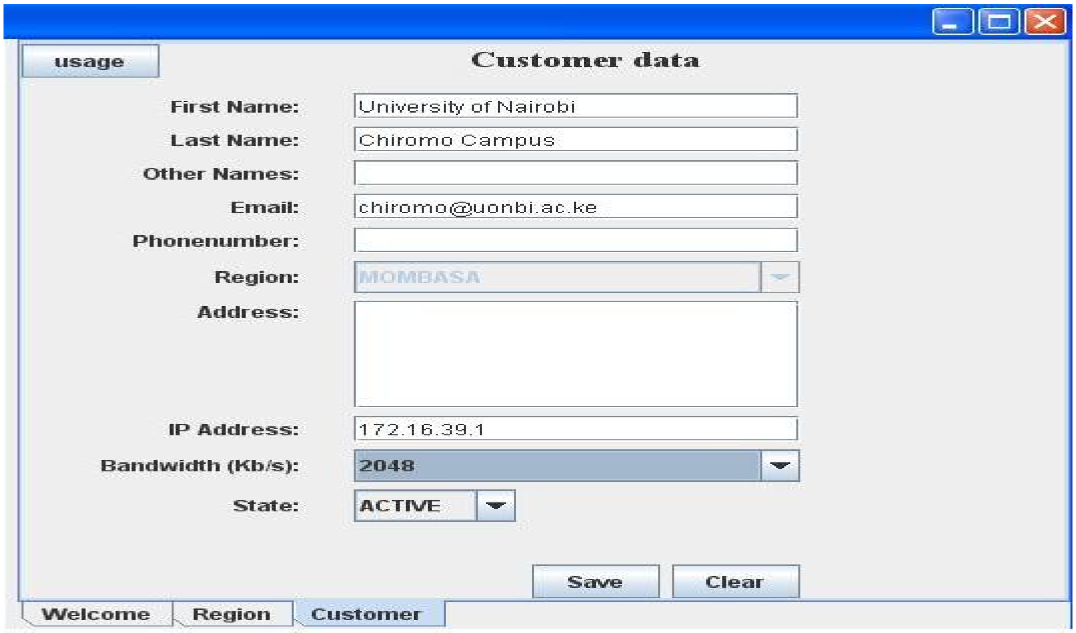
IP Address:

Bandwidth (Kb/s): Select Bandwidth

State: ACTIVE

The figure above shows how a new Client is created and associated to a Point of Presence (PoP). Controls are used to ensure details do not overlap

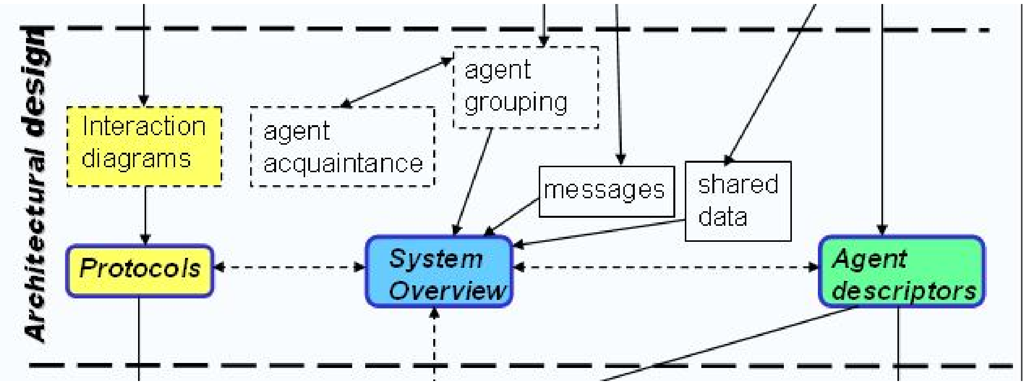
Figure 33: Updating Client Details:



Customer data

First Name: University of Nairobi
Last Name: Chiromo Campus
Other Names:
Email: chiromo@uonbi.ac.ke
Phonenumber:
Region: MOMBASA
Address:
IP Address: 172.16.39.1
Bandwidth (Kb/s): 2048
State: ACTIVE

4.5.2 Architectural Design:

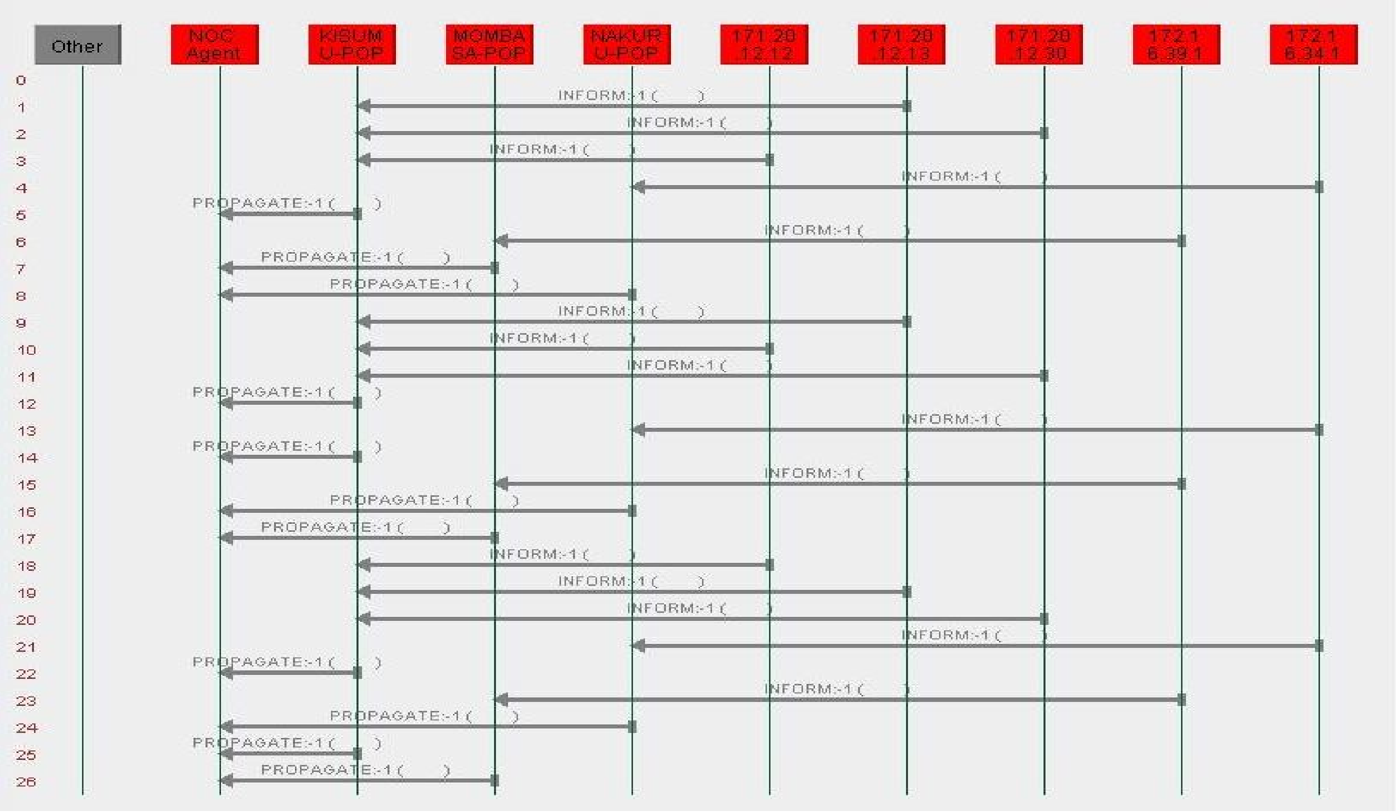


4.5.2.1 Agent interactions

In this design phase we focus on describing the interactions between agents using interaction diagrams.

It describes how Agents within the system communicate in order to achieve their goals.

Figure 34: Agent Interaction Diagram:



The diagram above shows how agents interact to accomplish their goals. Their roles are explained as:

NOC Agent -This agent handles new sign ups and updates to and deregistration from the platform by external system. It also receives propagated results from the POP agents.

POP Agent -This agent receives information about new clients and updates from the NOC agent. It receives the usage and uptime information from the clients and propagates the results to the NOC.

CLIENT Agents -These agents are at the client end and are responsible for communicating usage and uptime statistics to the POP agent.

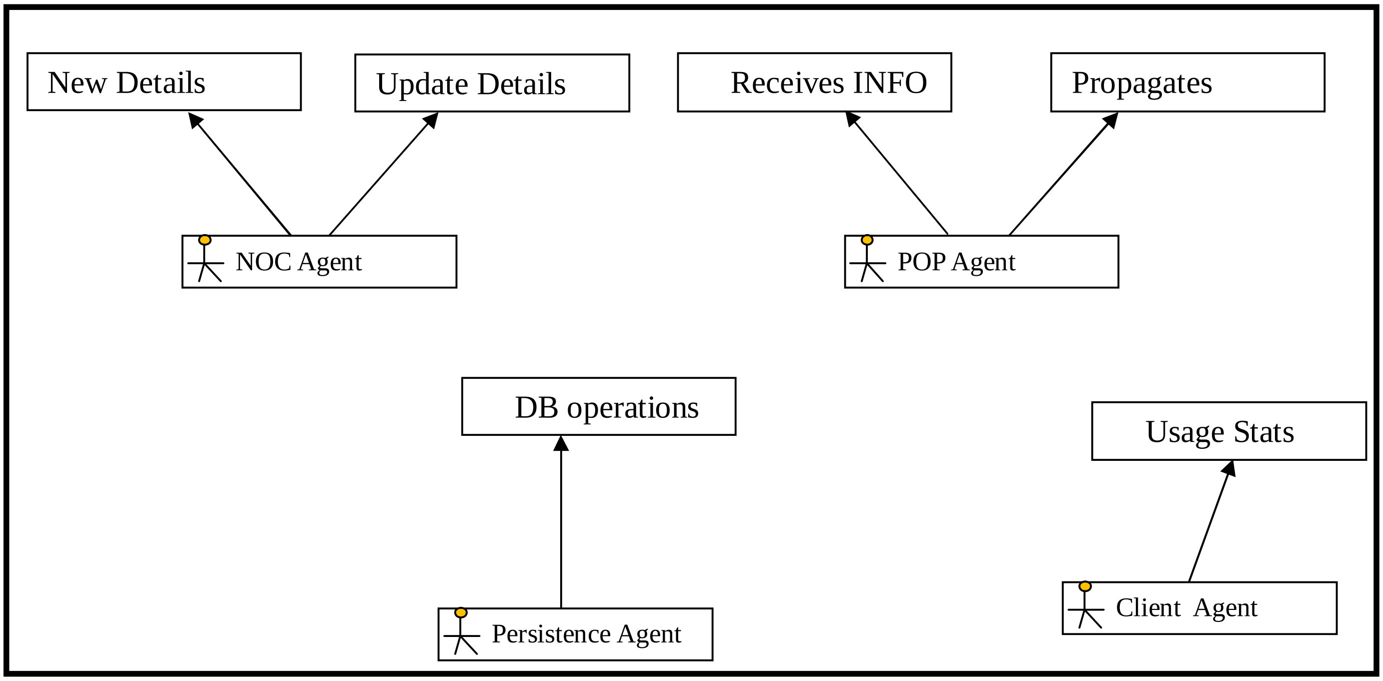
RMA, DF and AMS agents -These are system agents, shipped with the JADE distribution; HostMonitor can monitor remote networks using Remote Monitoring Agents (RMA). RMA is small application that accepts requests from HostMonitor, performs test (or action) and provides information about test result back to HostMonitor.

Persistence Agent -This handles database operations for the platform.

Agent Role Grouping:

This is grouping of Agents based on role similarities as well as analysis of data usage. Below diagram describes the group of roles that are performed by an agent.

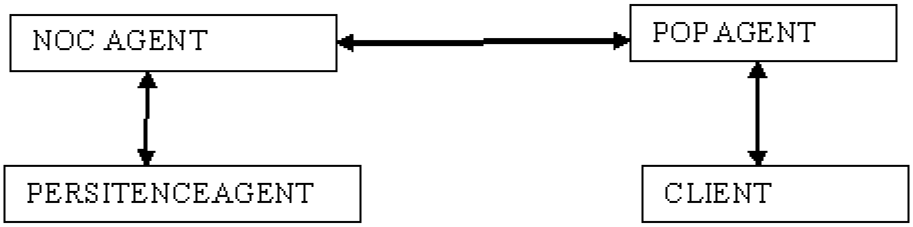
*Figure 35: Agent Role Grouping Diagram***:**



Agent Acquaintances

The agent acquaintance diagram indicates how the agents within the system interact. An agent acquaintance diagram contains only one type of entity; an agent.

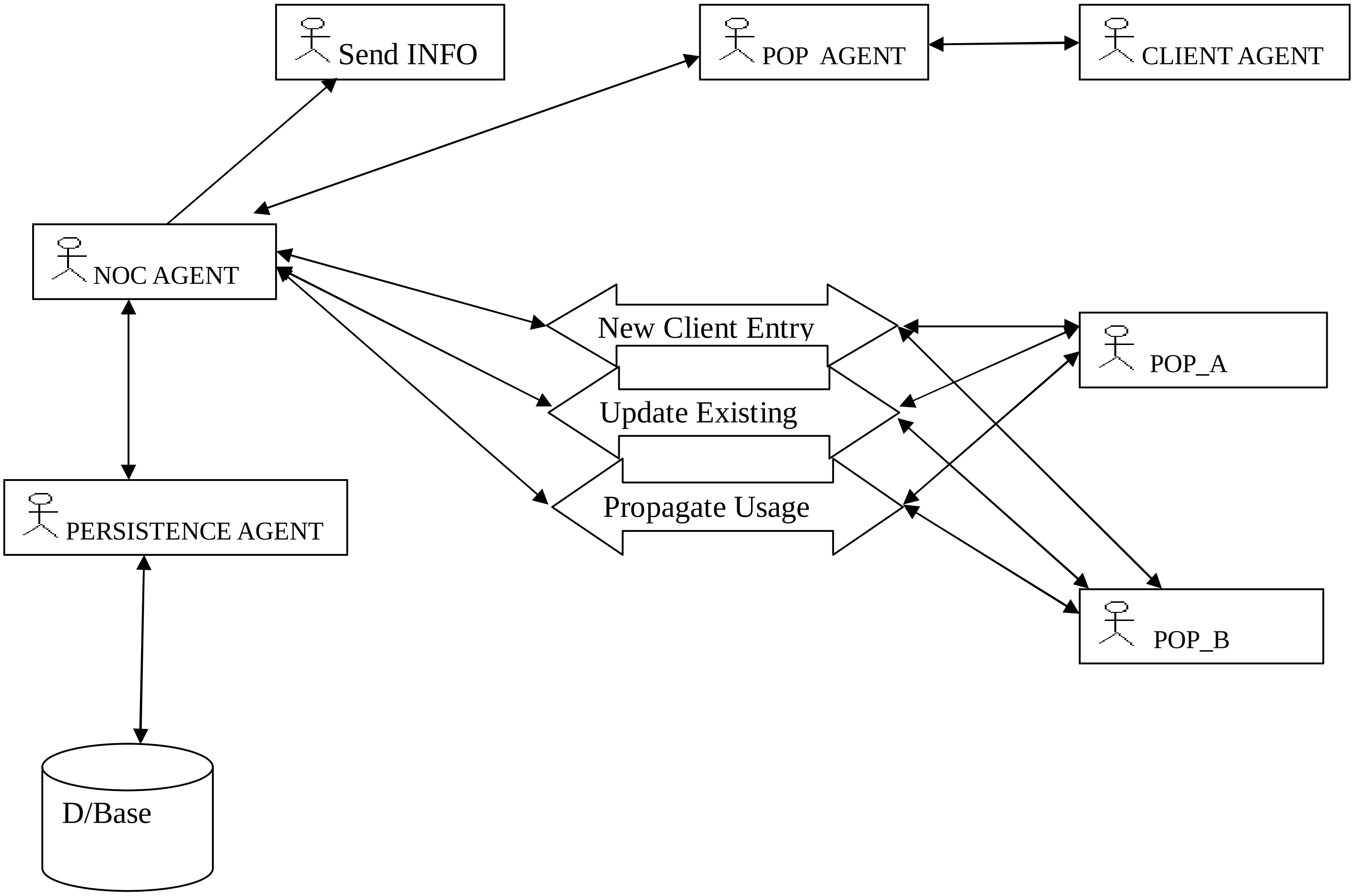
Figure 36: Agent Acquaintance Diagram:



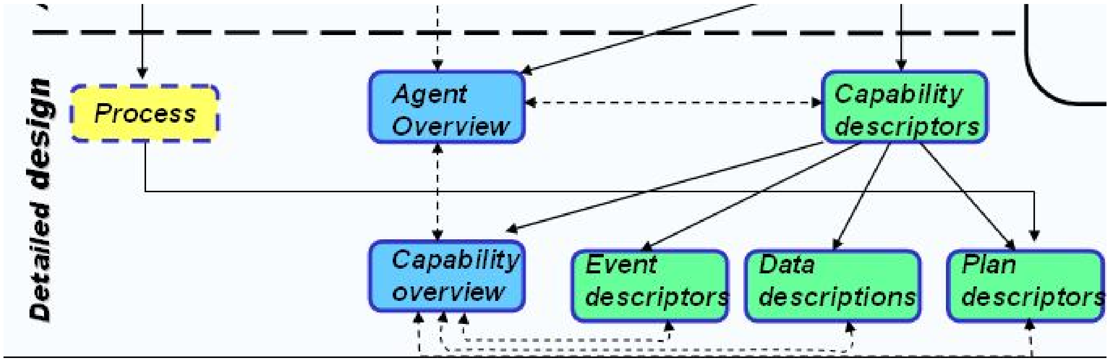
Systems Overview

This describes Agent interfaces and the interactions. It contains the agents, data, message, actions, protocols and Percept.

Figure 37: System Overview Diagram



4.5.3 Detailed Design:



This design phase focuses on developing the internal structure of each of the agents and the behavior of the agents. This phase also consisted of developing agents in terms of capabilities. This was done using agent overview diagrams and capability descriptors.

Detailed capabilities was done using capability overview diagrams and various descriptors in identifying the capabilities that each agent type contained, the study considered capability for each functionality that was grouped in the agent type.

Agent Capabilities

Agent	Capabilities
NOC Agent	This agent handles signups and updates from the platform by external system. It sends information to the relevant POPs concerning the client details. It also receives propagated usage and uptime statistics from the POPs.
POP Agent	This Agent receives Information from the NOC and initiates monitoring of the Client. It also propagates the usage and uptime statistics to the NOC upon receipt from the Client.
Persistence Agent	This handles database operations for the platform. Ensure data is persisted into the DB and initiate communication with POPs.
Platform Monitor Agent	This agent initializes the platform and agents
CLIENT Agent	This agent sits at the Client node and communicates with the POP agent. It sends usage and uptime statistics to the POP to be propagated to the NOC.
RMA **(Remote Monitoring Agent)**	Is a system Agent shipped with the JADE distribution; It's a tool to monitor and administer the status of all the components of the distributed platform, including agents and containers
DF **(Directory Facilitator Agent)**	Is a system Agent that provides the **Yellow Pages** service where agents can publish the services they provide and find other agents providing the services they need. Allows to register/deregister/modify/search for agents and services
AMS	Is a system Agent shipped with JADE, it represents the authority in the platform and is the only agent able to perform platform management actions such as starting and killing agents or shutting down the whole platform
Sniffer Agent	It's a tool to sniff message exchange between agents. This tool is useful to debug a conversation between agents. It allows also to save the conversation to a file and load from a file.

Table 1: Agent Capabilities

CHAPTER FIVE: IMPLEMENTATION

5.0 IMPLEMENTATION AND RESULTS

5.1 Implementation tools

Multi Agent system used in Network monitoring can be developed using a number of analysis and development tools. In this project the following were used to accomplish the goals.

MySQL Server 5.1:

This is an open-source relational database management system (RDBMS). The SQL phrase stands for Structured Query Language. MySQL provides a suite of tools for developing and managing MySQL-based business critical applications on Windows.

SQLyog Community – MySQL GUI ver 11.31 (32bit):

It provides a powerful means to manage MySQL databases. It Runs on all Windows version from Win XP to Win 8.x (desktop systems) as well as "Windows Server" systems of same generations (Windows Server 2003 and higher). It allows Create/Drop/Alter Tables, Stored Procedures, Functions, Views, Triggers and Events.

Netbeans IDE 7.2.1:

This is an integrated development environment (IDE) for developing primarily with Java, but also with other languages, in particular PHP, C/C++, and HTML5. It is also an application platform framework for Java desktop applications and others. The NetBeans IDE is written in Java and can run on Windows, OS X, Linux, Solaris and other platforms supporting a compatible JVM.

JADE 4.3.1: For building and managing Agents

JADE (Java Agent Development Framework) is a software framework fully implemented in Java language. It simplifies the implementation of multi-agent systems through a middle-ware that claims to comply with the FIPA specifications and through a set of tools that supports the debugging and deployment phase.

Java JDK:

This is an implementation of either one of the Java SE, Java EE or Java ME platforms released by Oracle Corporation in the form of a binary product aimed at Java developers on Solaris, Linux, Mac OS X or Windows. Since the introduction of the Java platform, it has been by far the most widely used Software Development Kit (SDK).

JAVA JRE:

This is a Run Time Environment for Java applications.

5.2 System testing and evaluation

System testing is a process of detecting errors in a software program. The errors are then corrected to ensure that the software meets its intended purpose. In this project dynamic and static testing were done. Static testing involved reading and checking documentation concerning requirements, reading and checking of code without running the system.

Dynamic testing involved running the code to check its output. The dynamic testing included black box testing. The black box testing treated the system as a black box that received inputs and produce outputs; it was based on assessment of requirements and functionality of the system.

The system was tested with random signup of clients into the system. Due to the sensitivity of client usage data, it was not possible to use live data for testing purposes. The simulated data was therefore used to test the operation of the developed system. The system was able to propagate usage and Uptime status data to the NOC. The study used a test plan that had the following contents:

i. The identity of the test to be done and the component to be tested.

ii. The purpose of the test

iii. Condition for carrying out the test

iv. Test data to be used (correct and incorrect data)

v. Expected results

5.2.1 Component testing

Under this study each module was tested with correct and incorrect data to reveal loopholes in the code. This was done in parallel with coding to provide intermediate results that could reveal logic, functionality and error handling of various units in the system.

This testing was done at the source or code level for language, specifically for identifying programming errors such as bad syntax, logic errors or to test particular functions or code modules.

5.2.2 Integrating Testing

In this project specific functional user requirements were tested to ensure that the system met these requirements and ensured that the right product is developed. Each user specific function was thoroughly tested to ensure that they perform their tasks properly. The test tried to find errors within the inputs and outputs of these functions. The results displayed and format would be closely examined to make sure that it meets user requirements. The tests included the following:-

i. The interfaces to ensure that they were functioning well by checking if the interface components were properly integrated and behaving as expected especially verifying that the appropriate action was associated with correct interaction method of designed mouse click events, key strokes, etc.

ii. The interaction of the system's graphical user interface and the backend database. In this study, data would be inserted from the front end using data entry forms and check if each entry corresponds to the correct database table fields, also check if data inserted is in the desired format.

iii. The form interaction design was also tested to ensure that the information presentation to the user is in a manner that the users can easily find and/or perform the operation they want. The design should be easy to understand and adopt for use.

5.2.3 System Testing

The system was properly tested to verify its functionality. This was achieved by ensuring that relevant tests were performed such as integration tests, unit tests, functional tests and acceptance tests. This test was to be used as the overall testing of the system after the system had fully been developed, where the users were to perform acceptance testing with the developer and raise any adjustments they would like to be included at different system levels. This type of testing was designed to ensure that the system meets all functional and non functional user requirements. The testing was also focusing on the behavior of the system once it is subjected to different inputs. It also focused on how different components of the system interact with each other for optimal performance of the system. Different user demands were tested against the system which included the error message testing and several screen mappings.

5.2.4 Acceptance Testing

This was done by the system user after all other types of testing had been performed. The system was taken to an ISP Technical Department to verify if it optimally delivers a more responsive network monitoring functions and then to the NOC users to test whether the system generates the correct results as per the System requirements Specification Document. The users at the NOC also used black box testing to evaluate the system functionality.

Upon successful acceptance testing, the system was then proposed for implementation; a process that would see the developed system is in place and working as intended. This can be achieved by running the system in parallel to the existing system and allow for transition.

5.2.5 Performance Testing

This type of testing relates to the expected level of responsiveness especially when there are many new signups and changes in the existing network.

5.2.6 Regression Testing

This type of testing involves ensuring that the system corrections do not result in the same or other errors as corrections are being made.

5.2.7 Evaluation Testing

For accurate results on testing and to guarantee quality of the system non technical users were assigned to carry out a beta test. This included users from management, customer relations, legal, and finance teams. This was done to guarantee software quality since these users do not understand much of the system structure. Therefore, they used different test data which helped in discovering more errors. Their recommendations were considered and errors corrected accordingly. Later the system was released to the end user. For the purposes of software maintenance and documentation, all the test records and test logs were recorded and documented for future reference.

5.3 Tests for the Developed Systems

ID.	Component Tested	Purpose	Condition	Test Data	Expected Results
1	User Authentication	Correct Login to display main GUI	Login GUI	Username and Password	Main menu form display
2	New Region Creation	Successful Creation of a POP	Region Data GUI	Correct Name and Subnet Mask	New POP created
3	New Customer Creation	Successful creation of new Customer	Customer Data GUI	Region ID, IP address	Creation of New Client
4	Update Client	Successful update of client detail	Customer Data GUI	IP address, Bandwidth	Successful Update of Selected Client
5	Usage and Uptime Viewing	View usage and Uptime Statistics for a client Link	Customer Data GUI	Bandwidth and Client Status on Database	Usage Graph
6	Uptime Status	Viewing Uptime Status of a Client	Customer Data GUI	IP address	Uptime statistics

Table 2: Development testing

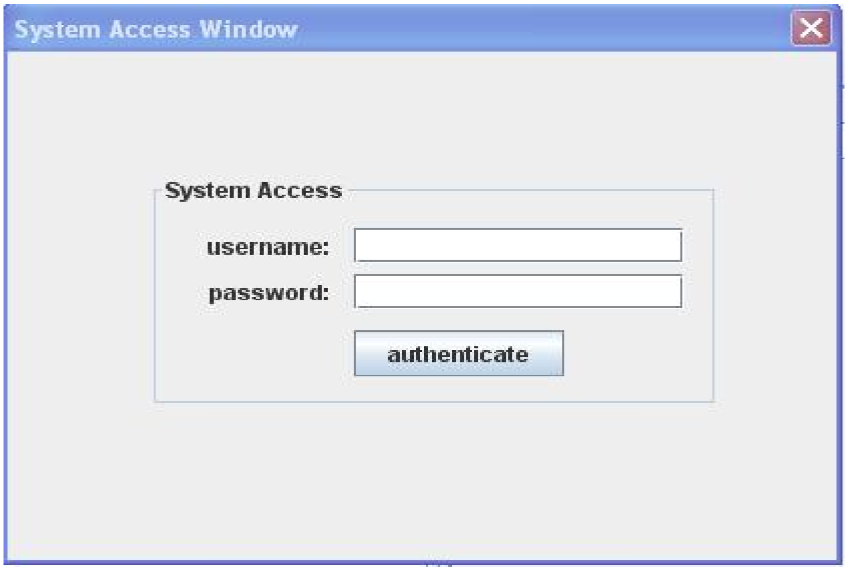
The table above describes the components of the system that were tested, the purpose of the test, the conditions the test were subjected to, the data used to test and the expected results from the test. This helped ensure the components meet the expected performance requirements once the system is fully developed.

5.3.1 Network monitoring MAS

We first run the Multi-Agent System so that the registered and active agents would be displayed, the successful test run is as shown below:-

```
------------------------------------------------------------------
Agent container Main-Container@10.176.0.153 is ready.
-------------------------------------------------------------------
 Agent container headquarters@10.176.0.153 is ready.
------------------------------------------------------------------
Agent container regions@10.176.0.153 is ready.
-------------------------------------------------------------------
INFO: --- Node <regions> ALIVE ---
[EL Info]: 2014-08-02 08:25:53.546--ServerSession(2737316)--EclipseLink, version: Eclipse
Persistence Services - 2.5.1.v20130918-f2b9fc5
[EL Info]: 2014-08-02 08:25:53.734--ServerSession(17491341)--EclipseLink, version: Eclipse
Persistence Services - 2.5.1.v20130918-f2b9fc5
[EL Info]: connection: 2014-08-02 08:25:56.296--ServerSession(2737316)--file:/C:/Documents and
Settings/Admin/My Documents/NetBeansProjects/Network
Monitoring/build/classes/_Network_MonitoringPU2_url=jdbc:derby:./databases/nakuru login
successful
[EL Info]: connection: 2014-08-02 08:25:56.296--ServerSession(17491341)--file:/C:/Documents and
Settings/Admin/My Documents/NetBeansProjects/Network
Monitoring/build/classes/_Network_MonitoringPU2_url=jdbc:derby:./databases/mombasa login
successful
NAKURU-POP Active 2 2
MOMBASA-POP Active 2 2
```

Figure 38: System Access:



username:

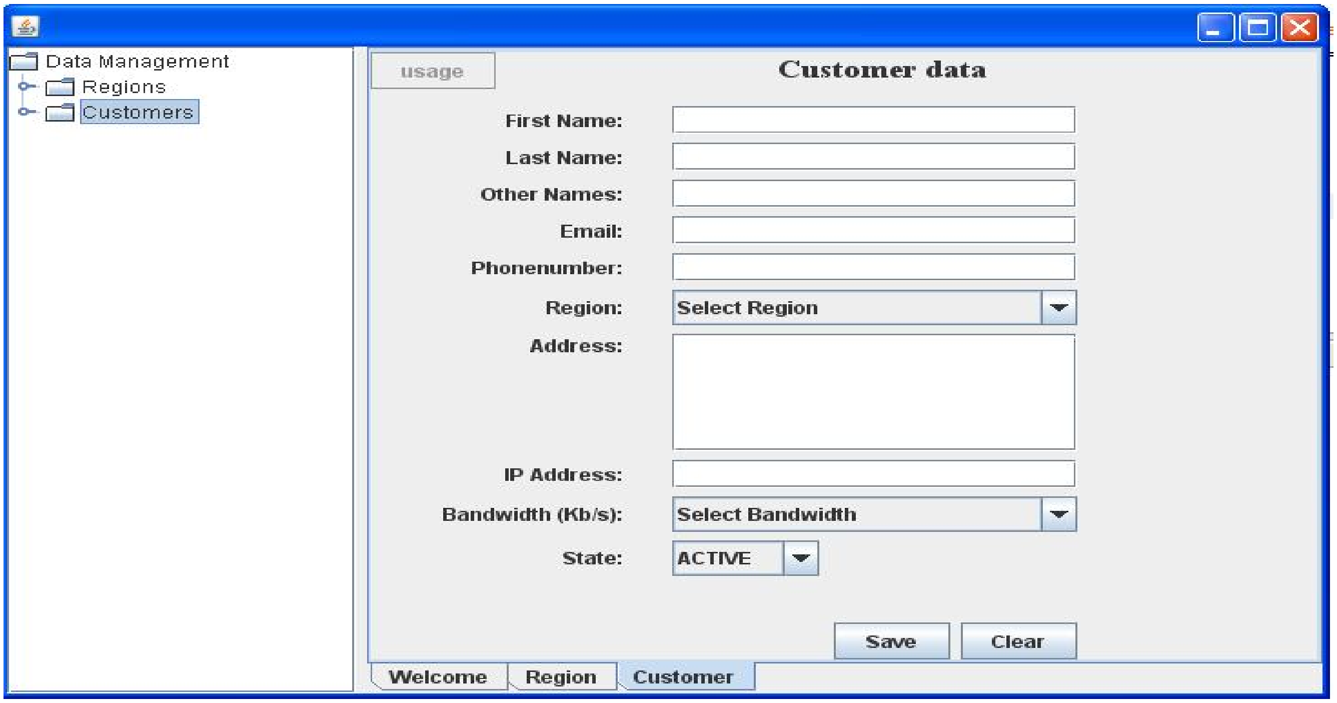
password:

The figure above shows how to Login to the Client management System Access Window

This is a web login portal from where all external logons are done in order to interact with t he system.

The system users must provide a valid User name and password to be allowed to access the system.

Upon successful login, the following interface is provided from where the user is able to perform tasks.



Customer data

First Name:

Last Name:

Other Names:

Email:

Phonenumber:

Region: Select Region

Address:

IP Address:

Bandwidth (Kb/s): Select Bandwidth

State: ACTIVE

Figure 39:New Client Details:

The figure above shows the widow used to enter details of a new client into the monitoring system. Once a user selects a customer or a POP/region, the person clicks at <USAGE> to be able to view client usage. Example is given below:

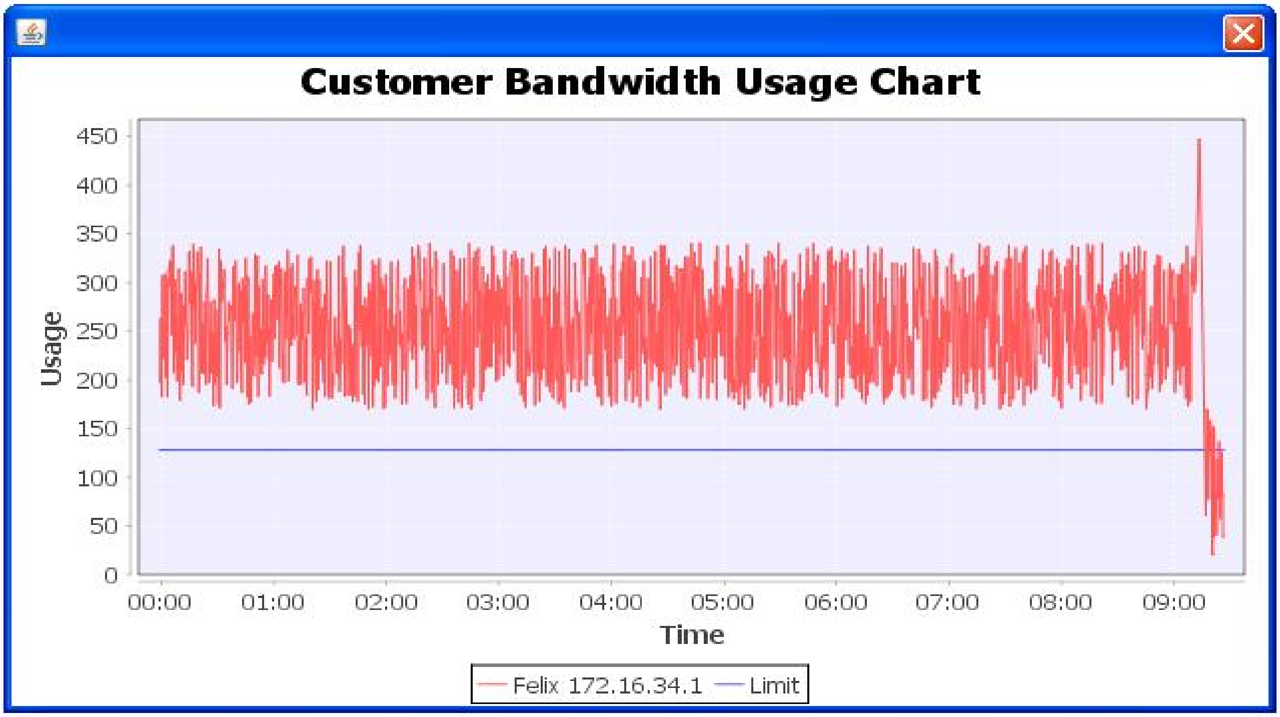


Figure 40: Package One Client Usage Statistics:

The figure above shows how to view Customer usage Statistics which is shown in a graphical format.

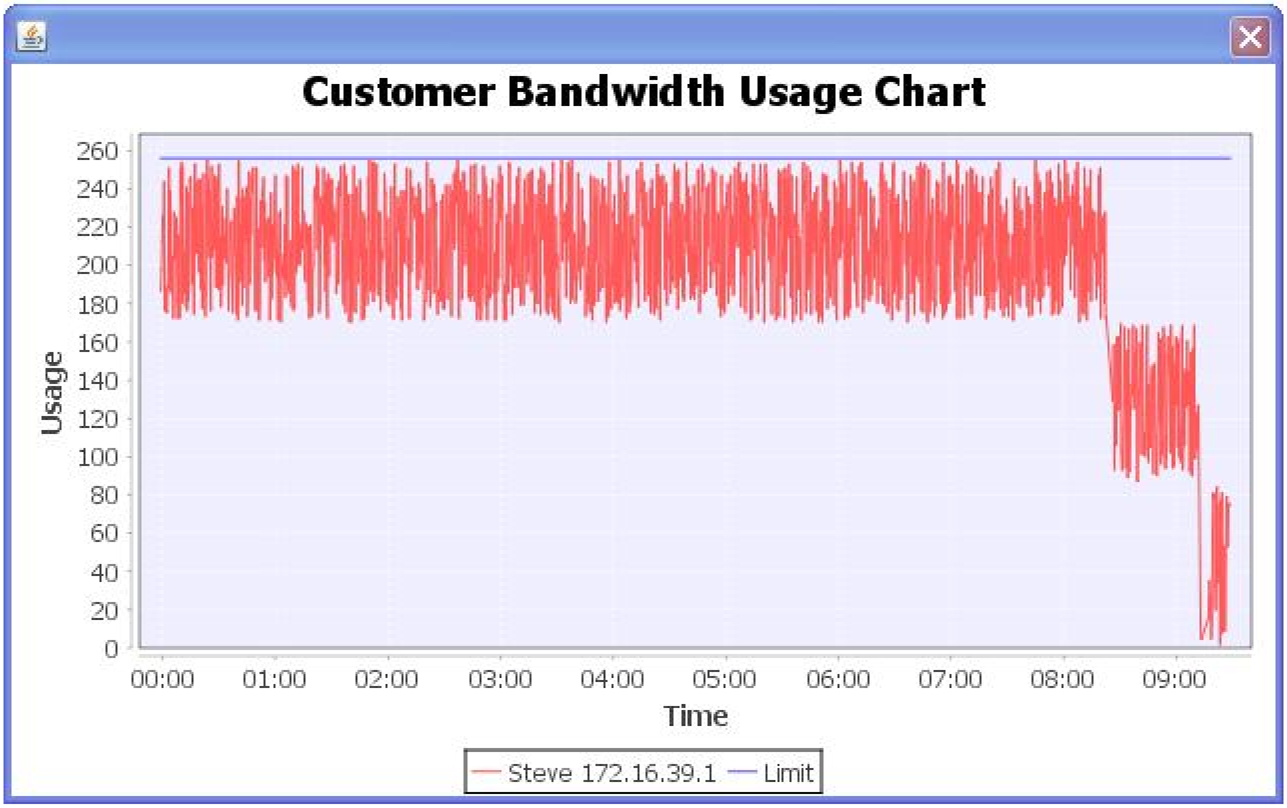


Figure 41: package Two Client Usage Statistics:

From the above usage data captured for two different clients; client 1 was provisioned for 512kbps but was later downgraded to 128kbps. This is used by the NOC team to check if the changes made take effect and if there is any degradation on the link due to such changes. The graph however is not recreated due to such a change.

Client 2 is not maxing the link but is almost utilizing contracted bandwidth; the graph also shows a deep in usage hence the NOC may want to find out the cause of the sudden change in usage pattern.

Agent Interaction Protocol:

During propagation of client usage and uptime statistics, the Agents exchange messages, informing and propagating as shown below:

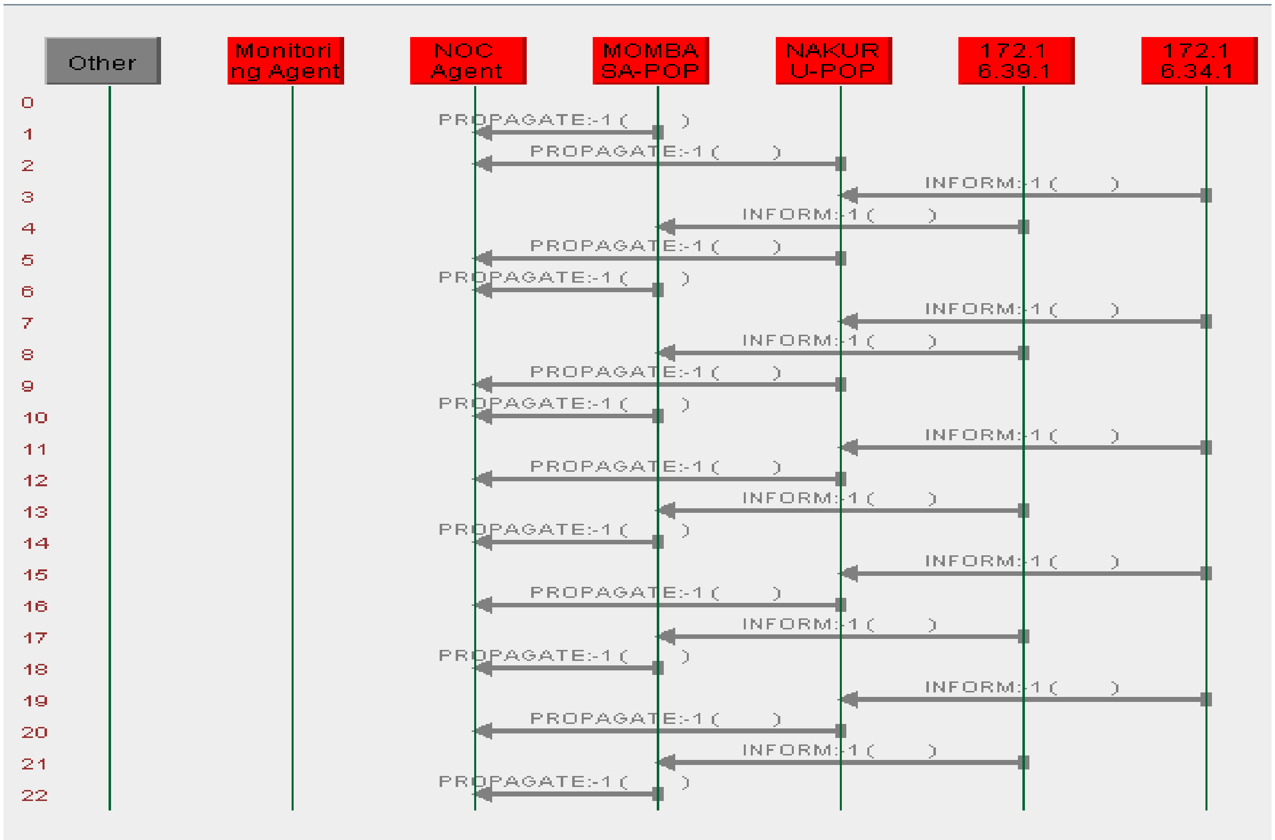


Figure 42: Agent Interaction Protocol:

The figure above shows how Agents interact to accomplish their goals.

5.4 Evaluation of the Multi Agent Based Network Monitoring system:

Test results determined the behavior of the system when it’s subjected to changes in the network and when client equipment is not accessible to the service provider for configuration purposes.

The evaluation of the multi-agent based system was done with the system administrators of Internet service providers so as to get comparative views against the existing system. Evaluation consisted data collection and data analysis.

Testing initiation of monitoring: The administrators enter details into the database and monitoring is initiated by the agents once the information is received by the relevant POP. There is no manual intervention of the administrator. Testing Data Capture: Usage and uptime data statistics was captured and displayed in graphical format and up/down status on a live GUI interface.

System evaluation was carried out to ensure the system meets the user requirement and the system specifications. After evaluation, the system was able to achieve the following:

i. System administrators were able to interact with the system using the GUI interface.

ii. Agents were able to pick necessary details from an existing Data base and use for monitoring.

iii. Regular polling intervals that was set at 20seconds but can be changed appropriately.

iv. Able to pick new client details and update existing automatically if a change is done on the Database.

v. Able to store monitored data over time for reference purposes.

vi. The results were presented in a graphical format

vii. UP time and down time of all clients are indicated on a GUI interface.

5.5 Discussion of Results:

Many Internet service providers who deal with corporate clients such as universities, banks, insurance firms etc and are located in different regions in the country often has a central location for monitoring and are initiated from a central data base. They constantly add new clients and occasionally make changes to the existing links in order to keep monitoring and adhere to the Service Level Agreements (SLAs). This project focused on an agent-based system that allows more responsive configurations to be done without having to involve much of manual setups.

It reduces the time taken to initialize and update parameters on the system and releases time for the administrators that can be used in other productive activities. It also eliminates mistakes that could be done by the administrators as the system has controls that ensure the agents only use the exact information as provided from the database.

The system optimizes responsiveness by ensuring changes done on the database are automatically updated to the agents database which makes the relevant changes on the graphs and monitored parameters of uptime.

Below table indicates the tabulated results of SNMP against MAS system.

	Av.Time to Initiate (mins)	**Av.Time to Modify** (Mins)	**Netwok Mode**	**Presentation**	**OS required**
SNMP(Cacti)	15	10	Client-Server	Graphical	Windows/Linux
MAS	3	1	Distributed	Graphical	Windows
Nagios	5	2	Client-Server	Graphical	Windows

Table 3: Results

From the analysis done, it was noted that it takes less time to initiate usage and uptime monitoring and to update monitoring parameters using MAS than using SNMP tools like cacti and Nagios respectively.

It was also noted that the MAS is applicable in distributed environments as its not based on Client-Server architecture as SNMP hence can be more applicable in most environments.

CHAPTER SIX: CONCLUSION

6.0 CONCLUSION AND FEATURE WORK

The Internet service providers are key to enhancing connectivity and ICT development in the country.

Corporate clients and businesses are growing in their requests for strict SLAs with negligible downtime required at any given time of business transactions.

Highly responsive monitoring systems are hence critical to ensure engineers sited at the network operation center (NOC) are able to accurately monitor links and hence able to diagnose faults and resolve connection issues appropriately.

This study has seen that multi-agent systems can be used to provide more responsive monitoring solutions to service providers. The study also shows it's easy to model agent interactions and communication. Changes to the client parameters are propagated immediately they are done on the database and monitoring is kept continuous without manual interventions.

6.1 RECOMMENDATIONS FOR FUTURE WORK:

Based on the results, further study is recommended on the developed system to be able to capture not only availability of Client to NOC statistics but also Client to Internet break out point statistics. This will help determine availability of subscribed bandwidth from the User to the Internet and not only the Point to Point between the user and the provider.

6.2 CHALLENGES

In this research project, many challenges were encountered which included the following:

i. Difficulty in getting information from service providers as most information is guarded confidential

ii. Difficulty in getting access to the Network monitoring systems and getting to do live tests on current system.

iii. During testing, it was not easy to get full attention of the engineers as some thought implementation of the new system renders them unproductive as the system could perform their functions.

iv. Programming and coding was a major challenge given the limited time allocated for the project as we had to learn new skills within the shortest time period. These included learning Java, JADE, Netbeans, SQL in order to perform system implementation.

v. Implementation of Multi Agents remains a complex task due to:

- Lack of adequate specialized tools
- Lack of adequate skills needed from analysis to design stages
- Lack of maturity in both methodologies and programming tools.

REFERENCES:

i. *Allen, Paul (2006) Service Level Agreements.*

ii. *A.Liotta, C. Bohoris, G. Pavlou (October 2002); "A Hybrid approach to Network Performance.*

iii. *Bournemouth University (2014); Guide to Citation & Referencing in the Harvard Style. Version updated September 2014 for the 2014/15 academic year*

iv. *Monitoring based on Mobile Agents and CORBA", In the Proceedings of the 4thInternational Workshop on Mobile Agents for Telecommunication Applications.*

v. *(MATA '02), Barcelona, Spain, A. Karmouch, T. Magedanz, J. Delgado, eds., pp 151-162, Springer.*

vi. *Bianco Phillip (20008); Software Engineering Institute ; Service Level Agreements in a service-Oriented Architecture Environment.*

vii. *Brian Henderson- Sellers, Paolo Girogini (2005). Agent oriented methodologies. Idea Group Publishing)):*

viii. *Chang'(1998); Mobile Agents in a Distributed Network Environment.*

ix. *D.Gavalas (July 2001) ; Mobile Agents for Network Monitoring and performance management; PHD thesis University of Essex,*

x. *Dr.John Murphy (August 2004); A framework for Adaptive Monitoring and Performance Management of a Component-Based Enterprise Applications*

xi. *Rao Sathya (January 2008 – June 2010) Project Coordinator MOMENT - Monitoring and Measurement in the Next generation Technologies, Switzerland.*

xii. *Franklin Et Al (1996); Multi-Agent Based Network Monitoring and Management using Jade*

xiii. *G. Aceto et al ((2013); Cloud monitoring: A survey, Computer Networks*

xiv. *Kothari C.R (2004); Research Methodology; Methods and techniques, 2nd Revised Edition.*

xv. *M. Wooldridge, N. Jennings (1995) "Intelligent Agents: Theory and Practice"; The Knowledge Engineering Review, vol. 10(2), pp. 115-152.*

xvi. *M. Fedor, M. Schoffstall , J. Davin (May 1990) A Simple Network Management Protocol (SNMP) research based on RFC 1157.*

xvii. *Odell James(2005); Agent Geneologies.*

xviii. *Stewart Robbinson (2004); The practice of Model development and use.*

xix. *Sunsted et al, (1998); Mobile Agents Approach to Congestion control in Heterogeneous Networks.*

xx. *Rusell Et Al,(1996); Multi Agent System Architecture Oriented Promethius Methodology.*

xxi. *Wooldridge Michael (2002); An Introduction to Multi Agent Systems published by John Willey & Sons June 12th 2002.*

xxii. *Wooldridge, Jennings, et al.(1999), A methodology for agent-oriented analysis and design.*