INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

Location Based Network Management System

Carlos Daniel Bento Mesquitela André

Dissertação para obtenção do Grau de Mestre em
Engenharia de Redes de Comunicações

Outubro 2009

Agradecimentos

Gostaria antes de mais de agradecer ao Professor Fernando Mira da Silva, orientador desta tese, pelo apoio, motivação e ideias para a concretização deste trabalho.

Gostaria também de agradecer:

- Ao meu colega António Marques, com o qual tive o prazer de partilhar algumas ideais sobre componente de monitorização da rede P2P
- Meus colegas de curso, pelo seu apoio e amizade que me proporcionaram durante todos estes anos.
- À minha família, a qual sempre acreditou em mim, fazendo todos os possíveis para que tivesse as melhores condições para efectuar os meus estudos.

Resumo

Hoje em dia, as empresas dependem cada vez mais das redes de computadores para suportar as suas operações de negócio. Em muitos dos casos a infraestrutura de rede é grande e complexa, e assegurar altos níveis de performance torna-se uma tarefa complexa. A solução adoptada consiste em recorrer a soluções de NMS (Network Management System) que realizam a monitorização e vigilância da rede.
A localização georeferenciada tem vindo a ser progressivamente incluída nestes sistemas, conferindo assim, novas capacidades de análise da rede, originando um novo tipo de sistemas, o LBNMS (Location Based Network Management System).
Este conceito foi explorado nesta tese, para desenvolver um NMS com suporte para georeferenciação baseado na integração do sistema Nagios com o Google Earth. O NMS desenvolvido foi aplicado à monitorização de redes IP com equipamentos fixos, e de uma rede P2P sobre MANET (Mobile Ad-hoc Network).
Palavras-chave: LBNMS, Nagios, Google Earth, KML, GWT, P2P sobre MANET, SNMP

Abstract

Today, companies increasingly rely on their computer networks to support their business operations, but in many cases the network infrastructure is large and complex. In order to achieve high performance, NMS (Network Management System) are installed to perform constant monitoring and surveillance of the network.
These systems increasingly use the georeferenced location of network elements in order to add new management capabilities, thus resulting in another type of systems, the LBNMS (Locations Based Network Management System).
This concept was explored in this thesis to develop a georeferenced NMS based on the integration of the Nagios system and Google Earth. The developed NMS was applied to the monitoring of IP networks with fixed equipment, and a P2P network over MANET (Mobile Ad-hoc Network).
Keywords: LBNMS, Nagios, Google Earth, KML, GWT, P2P over MANET, SNMP

Contents

List of Figures

List of Tables

List of Appendices

List of Acronyms

AP	Access Point
API	Application Programming Interface
BSC	Base Station Controller
CIIST	Centro Informática do Instituto Superior Técnico
CMIP	Common Management Information Protocol
COLLADA	COLLAborative Design Activity
CP	Collection Point
CPU	Central Processing Unit
DNS	Domain Name Server
DSLAM	Digital Subscriber Line Access Multiplexer
EMS	Element Managment System
EPE	Ekahau Positioning Engine
FTP	File Transfer Protocol
GE	Google Earth
GIS	Geographic Information System
GPS	Global Positioning System
GSM	Global System for Mobile Communications
GUI	Graphical User Interface
GWT	Google Web Toolkit
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IP	Internet Protocol
IR	Infrared
IST	Instituto Superior Técnico
ITU	International Telecommunication Union
KML	Keyhole Markup Language
LAN	Local Area Network
LBNMS	Location Based Network Management System
LBS	Location-based Service
LOS	Line Of Sight
MAC	Media Access Control
MAN	Metropolitan Area Network
MANET	Mobile Ad-Hoc Network

MIB	Management Information Base
MIT	Massachusetts Intitute of Techonology
NMS	Network Management System
OBEX	Object Exchange
OID	Object Identifier
OSI	Open Systems Interconnection
PAN	Personal Area Network
P2P	Peer to Peer
RF	Radio Frequency
RPC	Remote Procedure Call
RSS	Received Signal Strength
SNMP	Simple Network Management Protocol
SQL	Structured Query Language
SSH	Secure Shell
TTFF	Time to First Fix
VOIP	Voice over IP
WAN	Wide Area Network
WLAN	Wireless Local Area Network
XML	EXtensible Markup Language
YMU	YouMonitor.Us

Chapter 1

Introduction

In the current scenario, modern computer networks are large heterogeneous collections of computers, swiches, routers and other devices. The growth of such networks tends to be ad-hoc and based on the current and perceived future need of the users. In larger networks, the job of monitoring and managing becomes a complex task. Without any tools to deal with this complexity, network administrators cannot ensure the desired levels of performance and availability offered by the network. So, it is essential that companies invest in Network Management System (NMS) solutions, providing network adminstrators with tools to analyze, identify and solve problems efficiently. These solutions are becoming more important, because companies increasingly depend on computer networks to support their business operations. Today, a network downtime or performance degradation can result in large financial losses.

For these reasons, extensive research on the area of network management has been made in order to develop new tools and standards (e.g. SNMP [14] and CMIP [24]). These works and standards contribute to a more simplified management of the entire network infrastructure, covering scenarios from multiple areas (performance, fault, configuration, accounting and security) on the OSI network management model.

Over time, the NMS tools have undergone a process of specialization that seeks to deal with various types of network, such as WAN (Wide Area Network), WLAN (Wireless Local Area Network) and Mesh. This approach, allows NMS specialization, providing more appropriate features. A common trend in this tools, is the inclusion of the georeferenced information on the network elements, providing new management capabilities, and creating NMS based on location which are the core concept of the study developed in this thesis.

1.1 Motivation

The traditional NMS typically use network diagrams without any geographical information from their equipments, allowing only a topological view of the network and how the equipments are connected together. Therefore, we cannot know in detail where such equipment is located, for example, in which room, building or street.

With the addition of geographical information, the network analysis is improved, since it gets a clear view of how the connections between the equipments are physically made (see picture 1.1).

With this approach, it is possible to identify the dispersion of the network based on geographic distances and to improve the analysis of congestion problems and faults occurring in equipment and services. With the information shown in figure 1.1, for example, the administrator can check different points of the network, and observe a great activity in the building of router1, with yellow links between the AP and the router. From this geographical information, like the floors, GPS coordinates and some statistical data, the administrator may choise adding another AP to a specific building area.

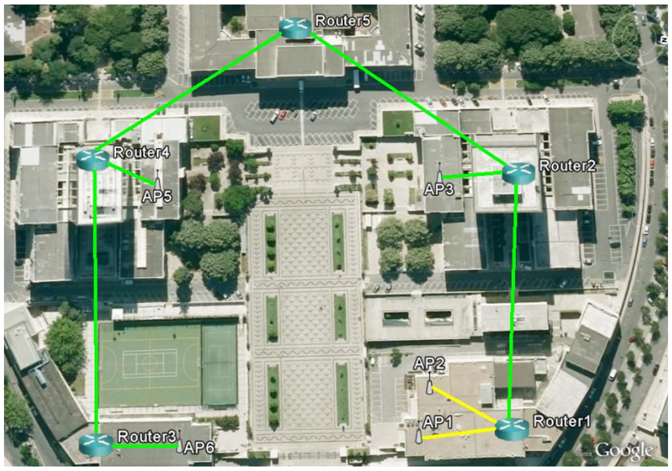


Figure 1.1: Example of a map with device locations and their connections

With this location concept, it is possible to provide a great flexibility on network maintenance, such as distribution of tasks to repair/assistance carried out by maintenance teams, based on geographical location(s) and the failure location. Some examples could be found at some Telecommunications Operator, when some BSC (Base Station Controller), DSLAM (Digital Subscriber Line Access Multiplexer) or router fail. Maintenance teams having a PDA running a NMS module are notified to move to the failure location, because they are the nearest or only available technical in that zone.

There are also advantages for network planning and design by combining the physical location of equipments and their performance statistics, helping to make better planning decisions. These are useful features, for outdoor environments, like celular, mesh and fixed networks. For the wireless networks, we can apply theoretical models of signal propagation and check the radio coverage in geographical maps. For complex models it is possible to use buildings and objects existent in that zone. For fixed networks, details of cabling can be represented in the map, like the real path taken by the fiber/copper through the cities and villages.

Another application, can be found in Ad-Hoc networks, updating in real-time the nodes location in the map. Ad-Hoc networks are very volatile, with node addition and deletion at any instance, making its monitoring a challenge. This type of network can be used in critical scenarios like rescue or even social environments. In this thesis, a P2P network over MANET (Mobile Ad-Hoc Network) was monitored.

1.2 Objectives

The goal of this thesis is the development of a location based NMS that uses the physical location of the network elements to provide additional information to manage networks. To accomplish this goal, the NMS will use Google Earth as a graphical interface, taking advantage of its technological potential, as the visualization layers, coverage of the globe with high image quality and also all features related to 2D/3D design.
The solution must be as possible agnostic to the network, regardless of its scale and settings. It is important to emphasize that the NMS will not be created from scratch, by selecting one with great popularity, online documentation and integration flexibility for Google Earth, easing the transition and conversion of data between these two worlds, i.e., NMS data to Google Earth. The NMS that fulfill these requirements was the open source Nagios [11].

The developed NMS was tested using the network data of IST (Instituto Superior Técnico) and a network mesh of 6 netbooks (Asus EeePC). The first environment was used to monitor a general fixed IP network, and the other environment to monitor a P2P network over MANET [18].

1.3 Organization

This thesis is structured in six chapters. The second chapter presents the state of art, providing the theoretical and practical support to this thesis. The third chapter presents the architecture solution, describes its main requirements and components description. The fourth chapter contains the implementation details. Chapter fifth describes the performed tests to validate the developed solution. Finally, the last chapter presents conclusions and future work.

Chapter 2

State of the Art

This chapter reviews the concepts and projects related to the area of network management including geographical information of network elements.

2.1 Georeferenced Systems

Georeferenced NMS systems can be classified in several application areas as, for example, the monitoring of WAN, Mesh and 802.11 wireless networks. Each of these types is discussed in each of the following sections.

2.1.1 WAN

In this class of applications, there are projects like the InterMapper and OpManager.

The project InterMapper is a proprietary software from the company Dartware, with great focus on developing software for monitoring networks and troubleshooting. This application provides a living view of the network, in a very interactive way, placing real-time information of the network into maps. This includes devices location, state of their performance, and points where the network traffic is low or high. This information is collected with SNMP (Simple Network Management Protocol).
This application, uses a mechanism of sub-maps to abstract each part of the network. A sub-map may represent a room, building or region. Using these approach, a complex infrastructure becomes more easy to manage, since we can navigate through this sub-maps, and focus into a certain section of the network, with the desired level of detail. This feature, makes this application able to adjust to different network scale, like LAN (Local Area Network), MAN (Metropolitan Area Network) and larger networks like WAN type. In figure 2.1, we have an illustration of this concept. As it shown in the the figure, there are several network abstractions for the regions of Boston, Chicago and Atlanta. When we focus on Boston network, we can see in more detail that it consists in a LAN with three PCs and one router.

This application support integration with Google Earth, by exporting all the information present in Intermapper. The mapping between these applications is not direct, since the user must specify the geographical information (latitude and longitude) for each network element.

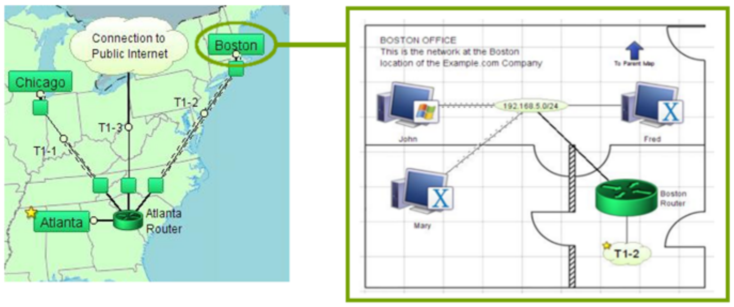


Figure 2.1: Example of navigation and viewing the network in Sub-Maps

On Google Earth it is possible to visualize the following:

- Devices, which are represented by their status badges (green, yellow, orange, red circle icons).
- Networks, represented as cloud icons.
- Links between devices are shown as lines.
- Status windows for each of these items are displayed when clicked.

Another solution for monitoring large scale networks, is Opmanager. It is very similar to Intermapper, since it also uses a background application to display network elements location with more detail. The Google Maps was the chosen application, providing a rich interactivity and image quality too, but only in 2D. Figure 2.2 shows the main NMS screen. As we see by this image, there are several equipments spread across North America with different status. Extra information about each network element can be displayed by clicking in the network element icon.

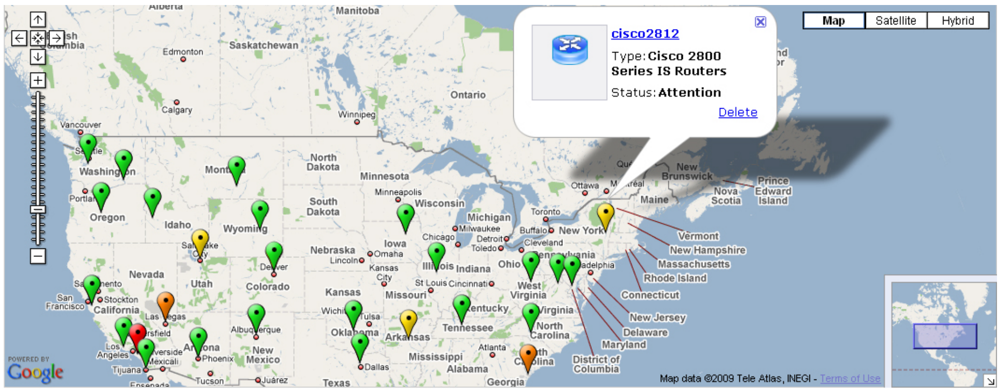

cisco2812
Type: Cisco 2800 Series IS Routers
Status: Attention
Delete
Map
Satellite
Hybrid
Map data ©2009 Tele Atlas, INEGI - Terms of Use
POWERED BY Google

Figure 2.2: Example of Opmanager using Google Maps

2.1.2 Mesh

Within these class of applications, there are projects like the Belview NMS, SkyControl and OnePointWireless, which will be described below.

The BelView Network Management System (BelView NMS) is a comprehensive software package that allows network operators to easily configure, monitor and manage a complete wireless mesh network built with BelAir equipments (multi-service wireless swich/routers). In this solution, network maps must be provided by the operator in order to map all the mesh network. This tool is designed to ensure an easy management of network growth, both in indoor and outdoors environments.
BelView NMS is a GPS-aware application, providing a auto-discovery service, reducing time on mapping the network to a geographical map. The auto-discovery module is used to automatically locate network wireless nodes and to place them in the appropriate location on the map. Once discovered and placed, the nodes are displayed in a topological view which also shows the connection links, links status and the number of clients that are connected to each node.
Concerning visualization features, it provides display filters, to show only a certain group of nodes by IP address, node type (access or core) and some custom fields. We can also retrieve node information, besides its geographical information, like some statistical information and software version. The following figure shows the BelView GUI environment. Analyzing the image, we get the view of the current mesh topology, composed by its access and core nodes, positioned throughout the city, showing the network coverage.

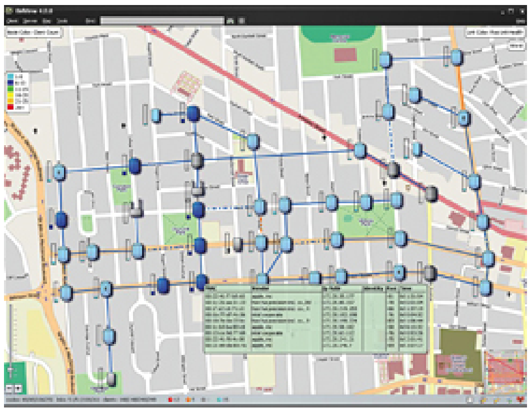


Figure 2.3: Example of BelView environment

The project SkyControl, that belongs to Skypilot, consists in an EMS (Element Management System) that provides to installers and network administrators a variety of tools for efficient deployment and ongoing management of SkyPilot wireless mesh networks. This mesh network supports many solutions, like extended Wi-Fi coverage or surveillance systems.
The SkyControl uses Google Earth PRO to represent the mesh network in real time, to take advantage of the GPS functionality embedded in SkyPilot infrastructure devices, retrieving GPS

coordinates from each node. With this approach, the SkyControl collects GPS and other statistical information from mesh nodes, and populates it into Google Earth PRO, providing an update map with vital information about the mesh network. Using this automatic feature, the network administrators are relieved from developing manual maps. This maps shows frequencies with different colors, links states between nodes (green indicates good connection, while yellow means a significant loss of performance), device type (signal repeaters, routing or access), and of course geographical position. The following figure shows an example of SkyControl.

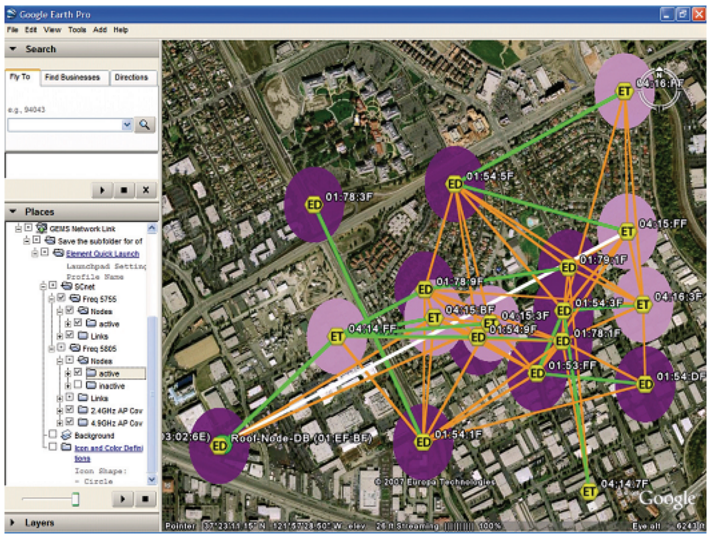


Figure 2.4: Example of SkyControl environment

Finally, OnePointWireless is an aplicational suite from Motorola, which allows network monitoring both in indoor and outdoor environments. Here, we with only consider the outdoor. OnePointWireless is very similar to Skycontrol in the way data is shown to user, providing link qualities, node state, signal range, and some statistics like uptime. However, this one uses Google Maps. Despite of not having many of the features of Google Earth, namely 3D features, it can also help operators to locate, diagnose and correct network issues, improving performance and availability. Because of this type of integration, it offers great visibility of the network topology, featuring zoom in/out for each desired part of the network.

2.1.3 Wireless 802.11

Within this class of applications, there are several projects such as iSPOTS [21] and FLUX* [16].

The iSPOTS project was developed at MIT (Massachusetts Institute of Techonology) in order to monitor and analyse the usage of the wireless network in the university campus, and determine community patterns by mobile terminals. In this project, the measures were obtained by over 1600 access points, recording in each AP the number of users and bytes transfered. With this information, it was created an on-line map server, which allows to observe real-time WiFi activity on a webpage.

In this project, active users were identified by their MAC address, which were associated to the respective movement patterns. After some days of test, it was possible to determine some of the preferred physical spaces to work in MIT community, network usage as function of time of the day, and also some typical move patterns. The figures 2.5 and 2.6 show some of this information.

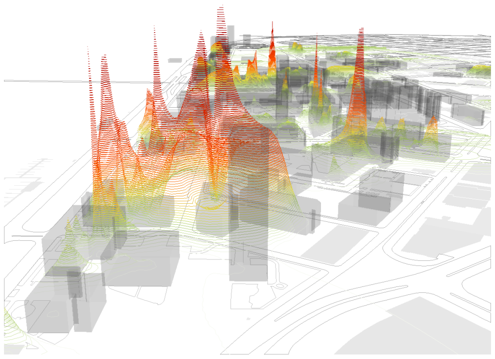


Figure 2.5: Network Usage

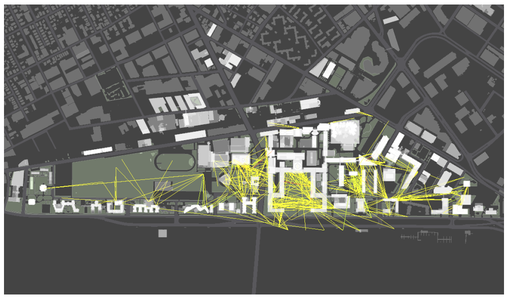


Figure 2.6: Roaming between APs

This study corresponds to a social analysis within the campus, indicating some students typical life routines. This shows some relevant issues about the use of the network, since it is used by humans with certain behaviors. The study developed by this project, may have several applications, like helping to determine the critical network areas, and usage as function of time. This application, when available to students with database querying, provides information about resource usage and it enables, for example, students to learn if its worth to move from the room to the library.

The project FLUX* was tested using the IST wireless network. The goal of this project has some similarities with the iSPOTS project, since it also makes the study of users mobility patterns, but with some peculiarity, because its used the Google Earth graphical interface to view the roaming between the AP's in real-time. The IST network, was represented in Google Earth with such detail that buildings were designed in transparent 3D models, including precise AP's location.
Like iSpots, there's a large amount of data recorded, and Google Earth allows a simple way to view specific information, by creating visualization layers. These layers are a vital tool for this work, helping to analyze the data at every instant of time. Some layers allow to analyze roaming during a certain period of time. On this project, it was possible to see the number of student who roamed between the two IST polo, since both Taguspark and Alameda were represented.
With this study, it is possible to understand roaming patterns, but it is also possible to watch in real-time other relevant network information, useful to determinate is current status, like the APs usage. To give an overview from AP usage, it was used 2D circles around the AP's, with the radius depending on the number of users. When clicking into a desired AP, statistical usage graphics are displayed, showing for example, upstream and downstream traffic. The figures 2.7 and 2.8 illustrate the environment of this project.

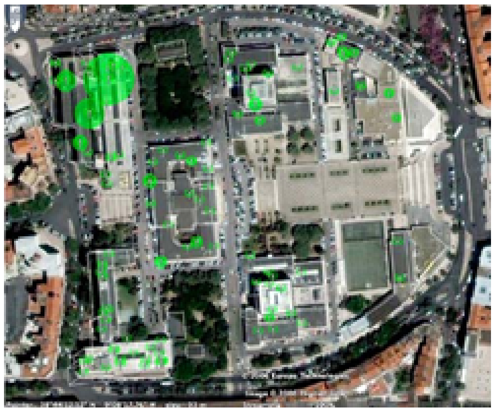


Figure 2.7: Network Usage

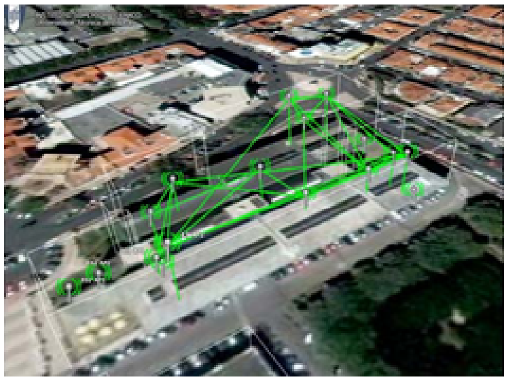


Figure 2.8: Roaming between APs

2.2 Monitoring P2P networks

Peer-to-peer (P2P) are distributed systems that consist on the interconnection of nodes able to self organize into structured or unstructured networks in order to share resources such as content (files), CPU cycles, storage space or bandwidth. The monitoring of such networks is complex, since the network can be extremely large and subject to a huge flow of input/output nodes, and node failure too. The work done in this area has more focus on the load management of files/keys storage, as in monitoring of resources like CPU, memory or network interfaces.

Therefore, there is little work under monitoring P2P networks, which deal basically with monitoring the state of each neighbours node, to check if any of these have failed, implying a very small view of the P2P network state. Below are some projects addressed in this context, i.e., the collection of node information of the P2P network.

2.2.1 YouMonitor.Us

The YMU (YouMonitor.Us) [5] is a distributed peer-to-peer monitoring service enabling monitoring other sites for downtime, while other sites keep an eye on it, resulting in a mutual monitoring (figure 2.9). Each site volunteer some CPU cycles and bandwidth from is server to monitor other sites, and YMU datacenter provides detailed downtime reports and instant notification of outages via SMS or email.

To join this network, and take advantage of its monitoring service, each node must install a script on its web server. The script receives monitoring tasks from the YMU datacenter, and ping other web sites about 10 times per minute. Other web sites in the network will ping the source site about 5 times per minute. If anything goes wrong with the site, a text message or email will be dispatched. The users of this service will receive regular reports about the site activity over time.

In this solution, the resource consumption by the script is very low, approximately equal to 10 page views per minute. Running the script 100 times per minute would result in a modest CPU usage of 3%, memory usage of 8.5-9.5 MB, and bandwidth of 200 B/s for HTTP monitoring and 2 KB/s for HTTP transaction [6]. These service could became very cost effective and useful for webmasters.

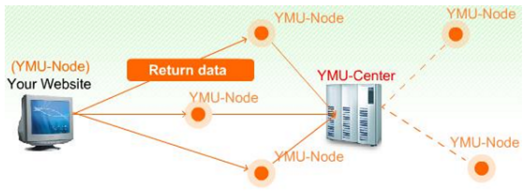


Figure 2.9: Data collect from P2P network

In this project we must highlight the fact that its possible to collect the status of each of the nodes, and then store and process that information in the YMU datacenter.

2.2.2 Monitoring Chord-based P2P System

This algorithm allows the monitoring of structured P2P networks of type Chord [22]. This algorithm consists of two phases. In the first phase the overlay is divided recursively into sub parts of a default size. For the second and last phase, its carried out the collection of statistical information for each of these sub parts.

This collection, its based on a token mechanism. The token is created by the first node (Rs), that send to his successor, and it successor do the same thing, all recursively. This token passage, ends when it reaches a node with an id superior to the last node (Re). The node Re is responsible to retrieve the CP (Collection Point) IP from the token, and also the information collected along the token trip, and send it to the CP.

These nodes must add to the token the values obtained locally, in order to make possible to the CP to get the required statistics, based on the number of nodes visited by the token. The value Smin can be adjusted, determining the period of checkpoints in which the nodes must notify the CP, instead of send just in the final node (Re). This mechanism is illustrated on figure 2.10.

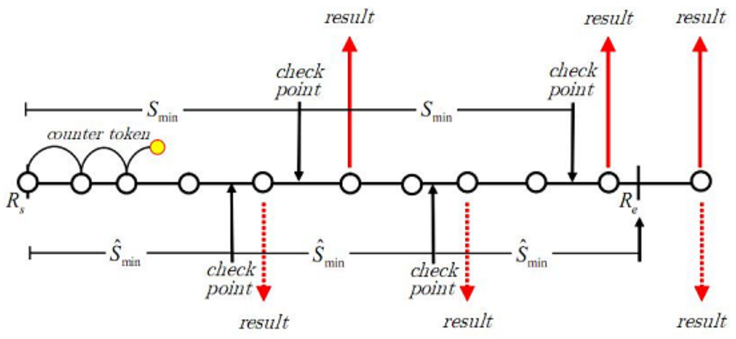


Figure 2.10: Data send at each checkpoint

The statistics can be obtained in the form of averages or histograms. For averages, its necessary two variables, one responsible to aggregate data and the other to count the number of nodes where the token has passed. To create histograms, its required a set of variable counters for given interval values. A given variable counter is incremented, when the node obtains a local value and that value belongs to its interval.

With this approach, some statistics can be collected, such the following averages: number of files; number of served queries; downstream and upstream bandwidth; disk space; used memory; uptime.

2.3 Location Systems

Location systems share the same main goal: to determine objects locations with great accuracy. These systems have an important contribute to create location-aware applications, that gain more quality as more efficient and reliable the location systems can be.
Several systems have been proposed for both indoor and outdoor environments, so as hybrid solutions, but only the indoor/outdoor study will be performed. The following systems represent some of the most relevant works for each technology (radio frequency, ultrasound and infrared).

2.3.1 Indoor

2.3.1.1 Radio frequency (WLAN, Bluetooth)

Radar [12] is a location-aware tracking system based on 802.11 wireless networking technology. It allows mobile terminals to compute their location based on the signal strength of known infrastructure APs. The system calculates the position coordinates of a terminal either by empirical methods based on comparison (fingerprinting), with previously measured locations mapped on a radio map (comparing the current measurement of the terminal with a database of previous measurements to see if it matches), or by using a mathematical model of indoor radio signal propagation. Radar overcomes the noisy environment by creating a radio map in the offline phase by collecting signal strength samples for each user location and orientation. In the real time phase, each AP measures signal strength of the mobile terminal and searches through the radio map database to determinate the location of the mobile terminal. The radio map-based method is an empirical method. A mathematical method using a mathematical model of indoor RF propagation and floor layout information instead of radio map has also proposed of radar. The system is able to estimate a user's location to within 2 to 3m of this or her actual position (with 50% probability). This is achieved using radar's scene analysis.

Some radar's limitations are the following:

1. Object tracking. It must support a WLAN NIC, which may be impratical on small or power-constrained devices.

2. Effect of multifloored buildings. Signal aliasing between points on adjacent floors could cause the system to place the user on the wrong floor.

3. Interference from other radio sources. The radio map database assumes that the signal from the Wi-Fi base stations does not change. However, either interference from other transmitters, or moving objects in rooms can change the signal.

The Ekahau Positioning Engine (EPE) uses its predictive capabilities to determine location based on the process of site calibration that builds a signal strength model of the environment. Site calibration involves taking signal samples at different known locations to construct a signal strength model based on signal strengths from various radio channels on the floor. The model is stored in the Positioning Engine. Ekahau clients will retrieve received signal strength values and return the values to the engine for location calculations. The dynamic indoor environment ensures that the radio signal propagation is susceptible to multipath effect thus making the generated signal strength model inconsistent with the actual signal propagation in the environment. Site calibration creates the empirical model of the signal strength in the environment. Therefore, modifying the existing wireless LAN layout after calibration will result in the positioning model to be inconsistent with the new environment. Signal strength at different areas will fluctuate in time due to movement of objects or new installations. This will require recalibration of the EPE to re-produce a new positioning model. The EPE exposes location information in the form of (x, y, floor) and logical area. The accuracy of the system is about one meter if there is a minimum of seven access points in sight but in a long term the accuracy decreases due to fluctuations in signal strength. The error in EPE can range from 1.5 meters when no one is in the room to 3.0 meters when the room is half filled with people. This is one of the problems noted in RADAR. Applications that obtain location information from the EPE do not know whether the location information is accurate as there is no form of location verification. EPE only knows the signal strength model of the environment but unaware of walls and obstacles. Therefore if a device is near a wall, the EPE may see the terminal on the other side of the wall. The location given by EPE has greater inaccuracy when a device is near a wall then when it is in the middle of the room. This could be due to radio signal distortion and multi-path effects caused by the wall.

The BlipNet system consists on a managed Bluetooth network offering access to LAN/WAN via Bluetooth. This system consists of four kinds of elements: bluetooth devices (e.g., mobile phones), the blipnodes, a blipserver and server application. The Blipnodes are Bluetooth AP to which terminals such as mobile phones and PDAs may be connected for receiving information. The BlipNode provides several protocols like OBEX File Transfer, PAN Networks and device detection. The BlipServer is the core component of the BlipNet architecture, responsible to configure, monitor the blipnodes in the BlipNet. It provides a rich open Java API (BlipNet API), which makes it possible for third-party developpers to interface with the BlipNet and create custom applications. The system inclues a positioning module to locate the devices, using the information provided by the blipnodes. The BlipNodes continuously send inquiries messages, to discover devices in range and then filter the Bluetooh device found, for example, by searching for those supporting certain bluetooth profile. After discovered, the device is logged into the blipserver database, and using the received RSS values on each blipnode, the positioning module can use this information to determinate it's position. BlipNet uses multilateration to locate the devices, that must be within 100m of a Blipnode to be located. To achieve a more accurate location, it's necessary several

BlipNodes and deployed in specific areas. Some problems arise with these bluetooth systems, for example, installation and deployment require calibration, special hardware, previously decided AP's locations to serve as anchors, maintenance, scalability and recalibration of the system.

2.3.1.2 Ultrasound

The Active Bat [15] System is the pioneer work in the development of a broadband ultrasonic positioning system. It consists of roaming Active Bat tags attached to the object to be located, which transmit an ultrasonic pulse, and are received by ultrasonic receivers mounted on the ceiling. The Active Bat system measures the distance between a tag and a receiver based on the time of flight of the ultrasonic pulse, and computes each tag's position by performing multilateration. The Active Bat system also provides direction information, which is useful for implementing many ubiquitous computing applications. It was shown to have 3cm accuracy. The 3D accuracy of a synchronous receiver is better than 5cm in 95% of the cases. However, Active Bat employs centralized system architecture and requires a large number of fixed sensor infrastructure throughout the ceiling and is rather sensitive to the precise placement of these sensors. Thus, scalability, ease of deployment, and cost are disadvantages of this approach.

Complementing the Active Bat system, the Cricket [19] Location Support System uses ultrasound emitters to create the infrastructure and embeds receivers in the object being located. This approach forces the mobile objects to perform all their own triangulation computations. Cricket uses the radio frequency signal not only for synchronization of the time measurement, but also to delineate the time region during which the receiver should consider the sounds it receives. The system can identify any ultrasound it hears after the end of the radio frequency packet as a refection and ignore it. A randomized algorithm allows multiple uncoordinated beacons to coexist in the same space. Each beacon also transmits a string of data that describes the semantics of the areas it delineates using the short-range radio. Like the Active Bat system, Cricket uses ultrasonic time of flight data and a radio frequency control signal, but this system does not require a grid of ceiling sensors with fixed locations because its mobile receivers perform the timing and computation functions. Cricket, in its currently implemented form, is much less precise than Active Bat in that it can accurately delineate 4x4 square-foot regions within a room, while Active Bat is accurate to 9cm. However, the fundamental limit of range-estimation accuracy used in Cricket should be no different than Active Bat, and future implementations may compete with each other on accuracy. Cricket implements both the lateration and proximity techniques. Receiving multiple beacons lets receivers triangulate their position. Receiving only one beacon still provides useful proximity information when combined with the semantic string the beacon transmits on the radio. Cricket's advantages include privacy and decentralized scalability, while its disadvantages include a lack of centralized management or monitoring and the computational burden, and consequently power burde that timing and processing both the ultrasound pulses and RF data place on the mobile receivers.

2.3.1.3 Infrared

Active badge [23] was originally developed from 1989-1992 at AT&T labs. Active badge makes it possible to locate any person or object at any given time with an acceptable accuracy. This device uses infrared technology (IR) to transmit data. The advantage of IR solid state emitters is the ability to produce them very small and very cheap. Every badge has an IR emitter which sends a unique pulse signal every specified time interval. In every room of the building there are sensors that can detect the signals coming from these badges. Since IR reflects off the walls therefore the signals are confined in only one room making it possible to locate badges. Since infrared also bounces off other objects in a room, it is possible for the sensors in the room to receive the signal without having to be exactly aligned with the infrared emitters. As with any diffuse infrared system, Active Badges have difficulty in locations with fluorescent lighting or direct sunlight. Diffuse infrared has an effective range of several meters, which limits cell sizes to small or medium sized rooms.

2.3.2 Outdoor

2.3.2.1 GPS

The Global Positioning System (GPS) [17] is a satellite-based navigation system made up of a network of 24 satellites placed into orbit by the U.S. Department of Defense. GPS was originally intended for military applications, but in the 1980s, the government made the system available for civilian use. GPS works in any weather conditions, anywhere in the world, 24 hours a day. There are no subscription fees or setup charges to use GPS. GPS satellites circle the earth twice a day in a very precise orbit and transmit signal information to earth. GPS receivers take this information and use triangulation to calculate the user's exact location. Essentially, the GPS receiver compares the time a signal was transmitted by a satellite with the time it was received. The time difference tells the GPS receiver how far away the satellite is. Now, with distance measurements from a few more satellites, the receiver can determine the user's position and display it on the unit's electronic map. A GPS receiver must be locked on to the signal of at least three satellites to calculate a 2D position (latitude and longitude) and track movement. With four or more satellites in view, the receiver can determine the user's 3D position (latitude, longitude and altitude). Once the user's position has been determined, the GPS unit can calculate other information, such as speed, bearing, track, trip distance, distance to destination, sunrise and sunset time and more.

The 24 satellites that make up the GPS space segment are orbiting the earth about 12.000 miles above it. They are constantly moving, making two complete orbits in less than 24 hours. These satellites are travelling at speeds of roughly 7.000 miles an hour. GPS satellites are powered by solar energy. They have backup batteries onboard to keep them running in the event of a solar eclipse, when there's no solar power.

2.3.2.2 A-GPS

With AGPS, a wireless network sends information directly to the GPS receiver, which allows the receiver to quickly locate the four satellites and process the data contained in their signals. The AGPS information includes identification of the visible satellites. Because the receiver is now searching only for specific signals, the amount of time it takes for a GPS receiver to obtain its first location or time-to-first-fix (TTFF) is reduced from minutes to seconds. Assistance is also provided to the GPS receiver by sending the ephemeris data for each satellite so that this data does not have to be decoded from the GPS signals. The receiver must still obtain signals from at least four satellites to determine the time it took each signal to arrive at the receiver, however it does not have to decode the entire signal. Assisted GPS effectively increases the sensitivity of the receiver so that it is able to obtain and demodulate the satellite signals in areas where unassisted GPS could not. Further, since the ephemeris data is already provided to the receiver, it can determine position more quickly than if unassisted, even in clear view of the sky. It is important to note that these advantages will be seen primarily under circumstances present when the device is in an unfriendly RF environment. The most obvious situation is when the device is first powered. When first powered, there is no valid ephemeris data on the GPS receiver, so the positions of the satellites in the sky are unknown. In this circumstance, the assistance information enables the receiver to obtain a more fix quickly than an unassisted device and in some cases, obtain a position fix where an unassisted device could not obtain one at all. If a GPS receiver has been functioning and has been demodulating the satellite signals prior to entering an unfriendly RF environment, the assistance data is initially unnecessary and offers no advantage. However, if the receiver remains in this unfriendly RF environment for a period of time, the satellites viewable over its position will change. In addition, the ephemeris data of each satellite will also change, as corrections are made to its orbit on a regular basis. For these reasons, ephemeris data becomes stale and needs to be updated on the GPS receiver. Regular updates of ephemeris data to the receiver enable the device to continue to operate in conditions where an unassisted device would cease to operate.

2.3.2.3 Cell ID

The Cell ID [13] based systems, uses network information to identify which cell site or sector a mobile user is in. The cell site is derived from the base station identity in the network, and the sector can be determined by checking which antenna is receiving the signal. As a result, location accuracy is dependent on cell size (antennas coverage). This method works for networks such as GSM. The cell ID-based systems can work in two ways: the terminal derives its position by receiving cell information from the network and calculating the position using a database of cell positions, or the terminal identity is received by the network and the network identifies which cell the terminal is in. In case the network identifies the cell, it can also determine which sector of the cell the terminal is in, if the antennas of the base station can be identified. The cell sector can also be determined using trilateration. Future systems may be deployed with smaller cells than in today's GSM networks, and providing more accurate systems.

2.4 SNMP

The Simple Network Management Protocol is a standard application layer protocol (defined by RFC 1157) 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 servers 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. 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:

- The get command collects statistics on SNMP devices.
- The set command changes the values of variables stored within the device.
- The trap command reports on unusual events that occur on the SNMP device.

The SNMP management console reviews and analyzes the different variables maintained by that device to report on device uptime, bandwidth utilization, and other network details.

2.5 Discussion

From the study presented in this chapter was found that most georeferenced systems are proprietary solutions with software typically developed from scratch, that in many cases include some integration with other management applications, as in the Belview NMS that supports integration with systems HP Open-View and IBM NewView. Without any related open source projects, this fact motivates the integration of an open source NMS solution with Google Earth, since it is an innovation.

In terms of graphical interface, it highlights a major usage of Google Earth by the georeferenced systems. This choice is due to the graphical environment of Google Earth, allowing comprehensive coverage of the planet, with enough quality and precision to place network information with all its 2D/3D features. For example, its extremely simple view the state of equipments (up/down), networks link, and some services (e.g. web, ftp, mail, dns or voip). Some problems may arise from this application, resulting from the lack of image quality in some areas of the planet, and in particular the difficulty of representing indoor networks, that may include a great number of equipment in many rooms and several floors, and for the current tools, Google Earth cannot represent this networks properly. In this type of problems, the sub-map approach is more efficient, since it can represent the indoor network, for example, one sub-map for each floor. To minimize this lake, the visualizations layers can be explored, introducing new layers and rules to turn more flexible to display the data of interest, filtering the rest.

Another important issue, is related to the Google Earth plugin, that haven't been used by any solution. This plugin enable loading a Google Earth instance into a web browser, which is a very powerful feature to integrate with other web-based NMS's, only implying that the client browser has support to Google Earth plugin. By using this plugin, this project becomes innovative, since this plugin hasn't been applied on other projects, due to its recent availability to developers.

Relatively to P2P systems, their monitoring approaches change according to this level of centralization and structure. For example, to structured systems like Chord, can be used a token based mechanism, while to pure unstructured and decentralized systems as Gnutella [20] there must be used another method to monitor the maximum nodes as possible, since the nodes communication works with floadings. In the absence of research maded in this area, it wasn't possible to find a great number of monitoring approaches to P2P network, and gather some relevant ideas. Some topics like mobile agents or distributed queries mechanisms, could have been explored.

From the analysed location systems, its notorious the number of research made for the indoor environments, in the most varied technologies (RF, ultrasound and infrared). This systems can achieve good accuracies, depending on the environment characteristics (small/big office, cubicolis, great medals objects) and the applied technology. When using infrared systems is necessary to use them in the short distances with LOS and apply specialized hardware. For RF systems there is no LOS problems, or need specialized hardware, but other issues arise from very common problems such as multi-path propagation, signal reflection and absorption (for example people's bodies). In the outdoor location systems, the GPS is the more important system, since it's adopted worldwide in an endless list of applications, like air force, navy, urban transports, object tracking or geocaching. Since the Google Earth uses GPS coordinates, the only suitable tecnologie is the GPS. The GPS have some well known limitations for indoor environments, but this problems can be overcomed, since the user can specify the network element localization manually. An automatic location update can be performed too, but more efficiently in outdoor environment, without using other technology.

Chapter 3

Architecture Solution

This chapter describes the main requirements and the proposed architecture to implement the LBNMS, which explore some ideas as presented in the discussion of the state of the art.

3.1 Requirements

At this point the essential requirements that must be included in the solution, are the following:

- Embedding Google Earth on the graphical environment of the NMS
- Ability to add/edit/remove equipment and network links in Google Earth
- Creating visualization layers to filter information
- Add graphics generated by the NMS into Google Earth
- View the status of equipment and links in Google Earth, according to the color of its icon
- Persistent storage of geographic information
- Concurrency control between users and ensure the data consistency
- SNMP agent for monitoring the P2P network over MANET

3.2 Architecture

The overall architecture is presented in figure 3.1.

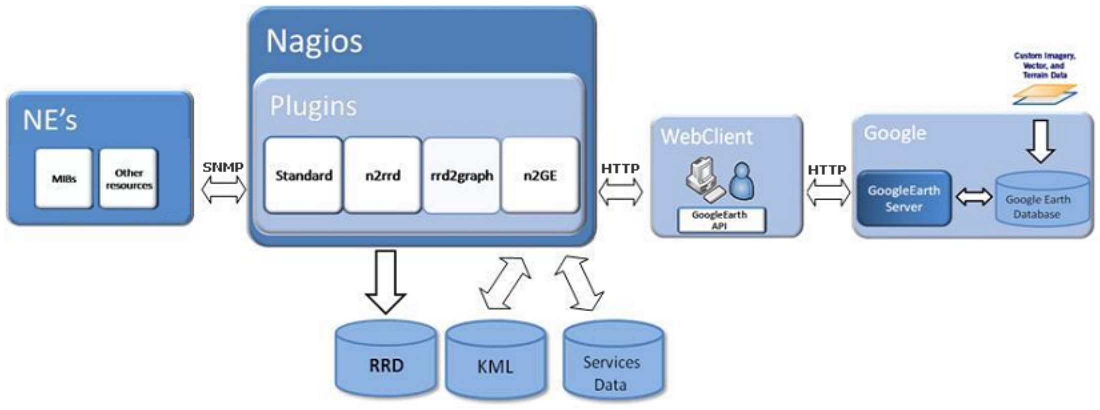


Figure 3.1: Architecture

3.3 Components

The architecture consists in 4 main components: network elements; NMS Nagios; Web Clients; Google Earth service.

3.3.1 Network Elements

The monitored network elements are desktops, laptops (Asus EeePC), switchs, routers and APs. The managment protocol used was the SNMP, usually the version 2c, to collect status informations from MIB tables. Other protocols were used too, like ICMP (Internet Control Message Protocol).

3.3.2 Nagios

The Nagios was the chosen NMS to build the georeferenced system. This open source application includes a large number of plugins and add-ons, with many potential to monitor the resources on the network. The work presented in this thesis corresponds to an additional Nagios plugin, providing location aware behavior.
Nagios is a very complete tool in the alarmistic field, but doesn't provide native mechanisms for graph generation, unlike other monitoring tools, namely Cacti [7] and Zabbix [9]. To overcome this limitation, the plugins n2rrd and rrd2graph are used to reproduce the graphics, based on the collected data from network elements when Nagios does its periodic pollings.
The plugin n2rrd is responsible to store the collected data from the network elements into a RRD database, while the plugin rrd2graph is responsible to generate graphs from it.
The add-on N2GE, corresponds to a web service which provide the integration between the world of Nagios and Google Earth, by processing data from Nagios, typically images of statistical graphics and other information to monitor, and convert this into the KML format. When these KML file is loaded by a network administrator on a web browser featuring the Google Earth plugin API, all this information will be interpreted and displayed.

3.3.3 Web Client

This component corresponds to the network administrators, carrying out the access to NMS by a web browser. To achieve this access, it is required to have the Google Earth API plugin installed in order to view the Google Earth in the web browser. Currently, this plugin is only available for MacOSX and Windows platforms, which have support for most browsers (IE, Firefox, Safari and Chrome), and will be available for Linux in a near future.
This plugin will allow embbed Google Earth inside the Nagios administrative webpages, becoming unnecessary the usage of Google Earth as a standalone application.

3.3.4 Google Earth

This component is the service provided by Google, which offers a set of technologies for creation of GIS and LBS. Running the Google Earth plugin on the client side, the administrators can interact with this geographical platform, an export the information of the network state, both software and hardware to this environment and perform their monitoring.

There are many similar tools such as NASA World Wind, Microsoft Live Earth, but Google it's more mature, providing practically all the rivals features. Another advantage appears from the many operation system support, such as Linux, Mac and Windows.

Chapter 4

Implementation

4.1 Tecnologies Used

4.1.1 Data Representation

The technology XML (eXtensible Markup Language) is implicit on the KML (Keyhole Markup Language) files. KML is a file format used to display geographic data in an earth browser, such as Google Earth, Google Maps, and Google Maps for mobile. A KML file is processed in much the same way that HTML (and XML) files are processed by web browsers. Like HTML, KML has a tag-based structure with names and attributes used for specific display purposes. The KML can be compressed using the ZIP format into KMZ archives, occupying less space on disk/bandwidth and still the same to be interpretable by both Google Earth and Maps. On the KML files, where the objects are defined, it is also included the geographical information with longitude and longitude values.

The KML can be used to:

- Specify icons and labels to identify locations on the planet surface;
- Create different camera positions to define unique views for each of your features;
- Use image overlays attached to the ground or screen;
- Define styles to specify feature appearance;
- Write HTML descriptions of features, including hyperlinks and embedded images;
- Use folders for hierarchical grouping of features;
- Dynamically fetch and update KML files from remote or local network locations;
- Fetch KML data based on changes in the 3D viewer;
- Display COLLADA textured 3D objects.

One of the most powerful features within KML is the so called "network link". A network link is a pointer to another KML file. The specified file can be either a local file or a file on a remote server. The information can basically be anything the Google Earth files (KML files) can support. For example: Placemarks, lines, polygons, image overlays, and even 3D models. In their simplest form, network links are a useful way to split one large KML file into smaller, more manageable files on the same computer. Even more important, when KML files have to be modified on the server, users must be aware of such update. For this reasons, the network links have a timer to perform automatic KML refresh, detecting if the pointed KML file have been updated. Depending on the nature of the KML file, we can indicate how fast or slow the KML file will be checked. In this thesis, the KML file is checked every 5 minutes, a time span small enough to see events in real-time. The target KML file is only download in case of changes, saving time and bandwidth.

4.1.2 Web Services

To develop the webservice, it was selected the open source project GWT (Google Web Toolkit). The GWT provides a set of tools that allow web developers to write their AJAX applications in Java language. The GWT framework cross-compiles the source code into optimized Javascript that automatically works across all major browsers. The GWT compiler safely eliminates dead code, aggressively pruning unused classes, methods, fields, and even method parameters to ensure that the compiled script is the smallest it can possibly be.
During the development stage, this webkit was design for rapid development of web applications, facilitating the cycle "edit-refresh-view" very common in JavaScript, because it allows debug through Java code line by line. It uses a simplified web browser that corresponds to the GWT's hosted mode browser, where the developper can refresh and see immediate the changes made in Java code, avoiding compiling to JavaScript. This toolkit uses a customized Tomcat to provide a servlet container to run the servlet created in this project. Tomcat is already configured, decreasing the initial installation and configuration time.
It is the goal of this project to embedd Google Earth in the graphical environment of Nagios. In order to accomplish this task its used the technology provided by the Google Earth Plugin. Fortunately GWT projects that implement the JavaScript APIs already exists, with the most important features implemented, facilitating the development of this thesis.

4.1.3 Virtual Map

To make a physical representation of the network in a virtual map, Google Earth was the selected as the reference application. This program offers complete coverage of the globe, with a great resolution from images obtained by satellite and aircraft photography, and it is always in constant improvement. In this virtual map, there is support for a GPS location system, enabling equipment with GPS support to be mapped in this virtual representation.
With this system there is the possibility of determining the distances between objects in the map, providing a useful metric for analysis, e.g., for planning and design of networks. The easy navigation through a three-dimensional world is also a point taken into account, supporting many functions

that deal with the camera viewpoint, manipulating is translation and rotation values.
There are several objects that can be created on the map, such as placemarks, lines, and 3D models. The placemarks are the icons used to represent the network equipment, since it is possible to customize the icon's image, establishing a correspondence to the type of equipment (e.g., switch, router or AP). These placemarks have an associated HTML page that is displayed when some click is performed on the placemark icon, allowing to display performance graphics and other important information about the status of the selected equipment or link. The lines were used to create network connections between the equipments and describe the state of the connection based on its color, which may be updated depending on its use. The 3D models are used only to represent building structures, allowing to discriminate different interior spaces, such as rooms and hallways. Another feature of great importance are the visualization layers, that allow to select only the data of interest. Google Earth already contains layers for roads, land, buildings, views of streets and places of interest (pubs, accommodations, pharmacies, sports, etc.). However, others can be created by direct manipulation of KML data, filtering only relevant information that are intended to analyze in the network. As an example, it is possible to see all equipments by: subnet, type of equipment or status (normal or critical). Given the ability to combine these layers, it is offered more flexibility for administrators to focus their attention on a given class or type of network information.

4.1.4 DataBase

The MySQL database was chosen for this project, given its great popularity and since it provides all the necessary features to implement the project. The required database resources are basic tables and relations, without major relational complexity. In order to implement the interaction between the servlets and the database, it was adopted the "Hibernate framework" [4]. The Hibernate is an object-relational mapping (ORM) library for the Java language, providing a framework for mapping an object-oriented domain model to a traditional relational database. This framework can map Java classes to database tables (and from Java data types to SQL data types). Hibernate also provides data query facilities. Hibernate generates the SQL calls and relieves the developer from manual result set handling and object conversion, keeping the application portable to all SQL databases, with database portability delivered at very little performance overhead. All these facilities help in development and code quality, since it creates abstractions from the SQL instructions, providing an easy way to program database transactions.

Another database technology used was the RRD (Round Robin Database). In this type of database there are a fixed number of records and once the last record has been written, the next update goes to the first record, and around and around as it goes. In this way, the databases will never grow out of control. The only downside of this approach is that we have to got to know the maximum time window frame and time resolution of each window view such that when the database is generated there is enough space reserved. The storage space can be dimensioned to days or even months of information, depending on the administrator choice. When creating a RRD database it will be required to specify a several parameters, namely one or more Data Sources and one or more Round Robin Archives. The data source (DS) defines what type of data is accepted (e.g., Integer, Double or String) and some boundaries on what represents "good" data. The round robin archives (RRA) can almost be thought of as data views, defining the different ways we can store and retrieve data. For this thesis, Nagios collects network information by performing pollings, and then generates statistical graphics, such as temperature, CPU usage, memory free/used or network interfaces downstream/upstream traffic. To perform these operations, its needed the open source tool RRDTool [3], which is used by the Nagios plugins n2rrd and rrd2graph, described in the architecture of the solution.

Finally, in order to access to data, MIB's (management information base) were used. MIB's correspond to a database containing Object Identifiers (OID) information. Depicted as a hierarchical structure, the MIB is the "tree" and each individual object is a l̈eafidentified by an OID. Levels within the MIB are assigned to different organizations. The top-level MIB OIDs belong to several standards organizations, while lower-level OIDs belong to institutions and organizations, such as network equipment manufacturers, who assign OIDs that extend the MIB with proprietary values. For this thesis, a MIB was created so that the P2P middleware can export a set of statistics, which can be made available by SNMP. This MIB can be found rooted under the registered Instituto Superior Técnico OID (1.3.6.1.4.1.21512), with the child number 99.

4.1.5 3D Modeling

The Google Skectchup [2] was the chosen 3D modeling program to create 3D building. With this 3D drawing tools we can practically make all types of building, applying transparency and to see only the building infrastruture, with all its internal division. One great motivation, is related with the module that enables the export of 3D models to Google Earth, by converting the 3D data to KML format. In this way, the development and populating of a 3D model becomes a quick and easy process.

4.1.6 Scripting

The Python [8] language was chosed to create a script that modifies the GPS coordinates of some pre-defined terminals. This was useful to accomplish the mobility test for the Ad-hoc network, simulating node movement through the campus.

4.2 N2GE implementation

4.2.1 Description

The N2GE is composed by a single webservice, which is responsible to implement the integration of Nagios and Google Earth. This webservice is accessed by a single url placed on the Nagios administrative webpages.

The N2GE was developed using the GWT framework on the Eclipse IDE. The GWT provided the tools to develop the client and server side code. For the client side, it generates Javascript and for the server side, a java servlet. The client side code, its loaded by web clients, and it includes the Google Earth interface and a set of HTML forms to export Nagios data to it. The server side

Icon	Equipment type
●	router
●	switch
●	acess point
●	server
●	laptop

Table 4.1: Available icons

Icon	Status
🟢	Ok
🟡	Warning
🔴	Critical
🟠	Unknown

Table 4.2: Equipment Status

code runs typically on the Nagios machine and it implements a set of methods to receive the RPC's invoked by web clients, such that they may perform some reads and write operations, resulting in accesses to database and KML files.

4.2.2 GUI Objects

To represent the network in the Google Earth graphical environment, four types of objects were defined: the network equipments, network links, cameras and building structures. These objects are sufficient, since we can design the network topology with the network equipments and their links, and additionally, using 3D buildings to better understand its location in indoor locations. The camera object allows to move the user viewpoint to places of interest in the map.

4.2.2.1 Network equipments

The network equipments were represented in the geographical map using Google Earth's placemarks. Each placemark represents a position on the earth's surface, with some icon and description in HTML. These features, enable to construct a rich object with many vital information about the current status of the equipment.

The icon image was used to represent the equipment's type. The supported icons in this work are listed on the table 4.1. Using different icons, it is possible to better differentiate the equipment functions, such as access or core nodes.
The icon's color correspond to the equipment's status, which can be four (ok, warning, critical and unknown), as shown in table 4.2. This color is updated in real-time. For example, if a equipment or service becames inactive, the icon becomes red, providing an intuitive way to view the network state.

These icons, when selected, can display an HTML page. This page, displays informations about the equipment, like its geographical information, monitoring graphics, IP address and the equipment status (up or down).
The included geographical information are referenced to equipments floor, and must be provided by the administrators.
For the monitoring graphics, each graphic has associated status, that are obtained by Nagios plugins, when collecting network information. The Nagios plugins are configured with some thresholds,

that allows determine the state of each collected sample.

4.2.2.2 Network Link

The network link between the equipments were represented using 2D lines. A line is created with two GPS coordinates of both network elements. This line only shows the link status, with a different color for each network status.
To determine a status line, it is required to known the NICs (Network Interface Card) status, which can be obtained by Nagios.

4.2.2.3 Camera

The camera provides a way to define the observer's viewpoint in terms of position and viewing direction, only requiring 2D coordinates, longitude and latitude. The third coordinate (altitude), is not available in the Google Earth API, so it is reset to the default value.

This object is important to save all relevant locations in the globe, becoming easy and fast to look at any place, avoiding manual navigation to the desired location.

4.2.2.4 3D Buildings

The creation of 3D building is performed using the Google SketchUp. After created, they can be uploaded to the N2GE server, and placed in a 3D models folder, to further be loaded on demand by users.

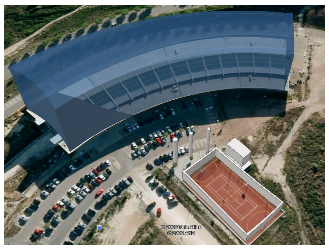


Figure 4.1: 3D buildings example

4.2.3 Database E-R model

The database was designed with six tables, to ensure the persistence storage of the network elements and link information, viewpoint cameras and log all transactions made. The figure 4.2 shows the table fields and its relations.

The network element table provides all the required information to draw a network element object, placed into the geographical coordinates indicated on the longitude and latitude fields. To gather more geographical information, the floor field was included, providing a richer layer filtering operations. This table is related with the MonitoringService table, to associate all the services that are being monitored to their respective network element, e.g., a network element named server02 may have associated several services being monitored, like HTTP response time, free space on a disk partition or the CPU temperature. All these relations between network elements and the

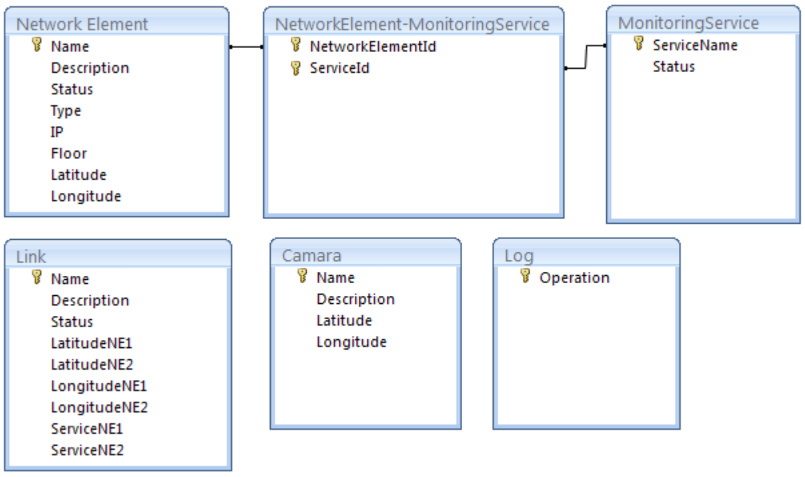


Figure 4.2: Implemented Database

monitored services are defined in the Nagios status.dat file. The N2GE performs constant parse of this information to load it into database.

The link table includes the two GPS coordinates of each network element, in order to create a 2D line between them. The line status is obtained from the services fields, which must be associated with the NIC status of both network elements. The link name does not have any relation to the network elements table, because its name already includes the two participating network elements, and concatenating each name results into a unique Id for the link table. When some of these network elements are removed from the Google Earth interface, the link loses its importance, and by this reason it must be equally removed.

The camera table stores the geographical information on the latitude and longitude fields, attached with some location description, making it possible to load a previously saved place of interest. In this object, is not possible to get the altitude value, since the implementation of the javascrit API does not provide this information, so it is assumed a default value of about 300 meters.

The log table records all the transactions made by the system, and for this purpose its used the operation field. These logs are very important to determine the system current view and additionaly to implement an automatic refresh mode on the client side.

4.2.4 Implemented Features

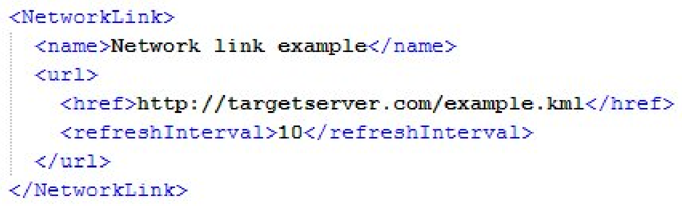
4.2.4.1 Create/Edit/Remove N2GE Objects

This feature is absolutely a must-have on every LBNMS, enabling the manipulation of every GUI Object on Google Earth. With this feature, administrators can start building the virtual environment with all information present in Nagios.

4.2.4.2 Real-time data

Nagios monitors the network in real-time, keeping all the adminstrators informed about the network health. On the Google Earth we want to accomplish the same goal, displaying the network information updated in real time. The adopted solution focus on simply updating the KML files, which are associated with every N2GE Object. When some information carried by the object changes, the respective KML is updated. This can happen, for example, when some service reaches its threshold and switchs to a warning state. Another very common case is when the graphics associated to some equipment is updated. This change must be reflected on the equipment KML. The KML files are on serverside, and loaded initially on the clienside, which periodically checks for new updates.

This periodic checks are performed by KML's Network Links. This type of KML file, have a simple syntax, as illustrated in the next figure. The href tag points to the kml file location, and the refresh rate is indicated on the refreshInterval tab.



```
<NetworkLink>
  <name>Network link example</name>
  <url>
    <href>http://targetserver.com/example.kml</href>
    <refreshInterval>10</refreshInterval>
  </url>
</NetworkLink>
```

Figure 4.3: Network Link KML example

4.2.4.3 Web SSH

This feature takes advantage of the web-based interaction between the client and the N2GE, integrating in the browser an SSH shell-client to perform remote access. The displayed network elements may be targeted of some problems or maintenance tasks with some updates, and having a tool to perform quick remote access is very useful. The software used was the applet Mindterm, which supports the first and second SSH versions and the ciphers RSA, DES, 3DES, Blowfish, IDEA and RC4. This software is also lightweight, taking small time to download it.

4.2.4.4 Location Search

In this feature the Google location search service is used to look out for a particular address. This service will return the GPS coordinates of the specified address and translates the camera viewpoint to that address. Using this service, it is avoided to waste time making a manual search for each place on the earth globe.

4.2.4.5 Visualization Layers

There is often irrelevant information being displayed that can be turned off, helping the administrator to focus this attention on certain information of the network. The visualization layers, are the concept used to provide this feature. In this thesis, we try to explore this feature, creating different layers types, and providing options to make layer combinations. The five layers that were implemented are the following:

- Equipment type (router, switch, AP, server, or laptop)
- Object type (network element or link)
- Status type (up, down, warning, unknown)
- Floor
- IP address

For the floor layer, it was possible to add several rules, such equal, above or below a certain floor value. For example, we can create two rules to filter all the network elements above the third floor, and those in the six floor. This feature is important to represent each floor separately, because Google Earth does not support the creation of placemark above the ground, i.e., it can not create network elements for different altitudes, other than zero.

Other useful layer is the selection of network elements based on the IP address. This layer also supports rules, including a filter by a specific IP address or subnet, using the CIDR notation for the last one. This is a powerful layer, because it is possible to show only individual equipments and specific parts of the network.

The other layers provide filtering according to the selected type.

4.2.4.6 Client Concurrency

In order to deal with client concurrent accesses, a single exclusive lock for destructive operations was implemented, as edit and delete. When multiple users compete for these updates, the first winning access to the critical zone will gain the lock and perform their operation successfully.. Others users attempting the same operations will fail to accomplish it, and they will be notified about the changes occurred in the server, by loading these changes.

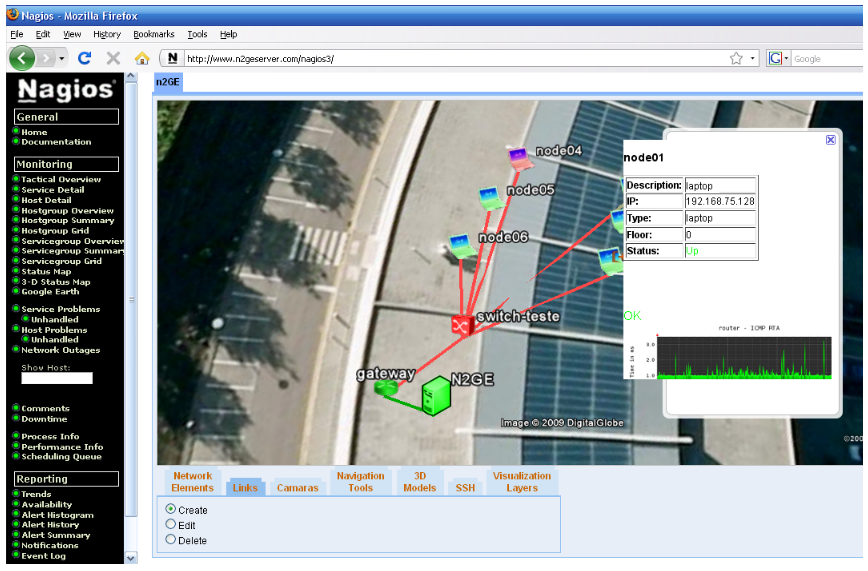
4.2.4.7 Client Update System

Nagios may have simultaneous user access, and the information will be changed over time, is extremely important to implement a feature to update the client current system view. By logging all the operations, and knowing the actual system view and client view, it is possible to determine the steps required to make the client updated.

4.2.4.8 P2P SNMP Agent

To monitor the P2P network, another SNMP agent was defined using the TwistedSNMP Project [10], providing access to all v1 and v2 SNMP functionalities. The SNMP agent runs in port 20000, for each p2p node. The created MIB tables (see Appendix A), are grouped in four main tables: routing service, storage services, peer selection service and service registration service. All the variables data type are defined as integer, easing the create of graphical templates for each variable, because they share the same base template.

4.2.5 Environment



node01

Description:	laptop
IP:	192.168.75.128
Type:	laptop
Floor:	0
Status:	Up

(●) Create
( ) Edit
( ) Delete

Figure 4.4: N2GE environment

4.3 Project Workflows

4.3.1 Client Load

The client starts by making an http request to the N2GE wepage, and the webserver sends the webpage with the javascript code and the ssh applet. After the webpage is received, the client loads it on it browser. On this loading, one startup javascript event is executed, invoking an RPC (Remote procedure Call) with the N2GE endpoint. The N2GE receives the RPC call and fulfills the request.

To serve this request, the webservice performs a database query, taking all the system information, network elements, link, cameras and 3D models. After receiving this data, they are structured into a list of N2GE objects, serialized and sent back to the client.

In the end, the client extracts all the objects and loads them into the graphical environment.

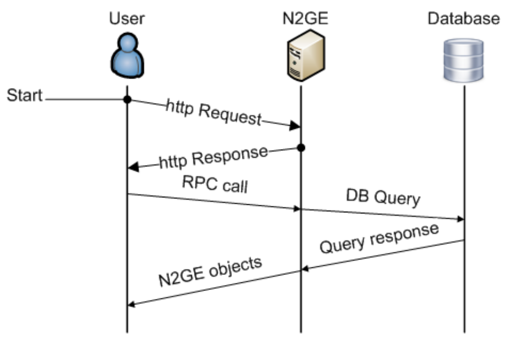


Figure 4.5: Client Load

4.3.2 Server Load

The server load follows the steps illustrated on figure 4.6. The server starts to check the database existence and then creates a thread responsible to analyse database consistency. This thread is important, since the Nagios data changes upon time. If the database is not updated, it becomes obsolete. Therefore, the lauched thread is responsible to analyse the Nagios data and the database, and determine if this last one is update. The Nagios data is obtained from the file status.dat, which contains the status of every monitored service. To avoid unnecessary readings on the status.dat file when no changes happen, an MD5 hash is applied to the file content and recorded. This hash is compared to the next generated hash, and if they match, the read operation will be cancelled.

Other files are used too, like the equipments configuration files, that contain the IP address, the only retrieved information. This project was implemented using the same configuration files used by CIIST, becoming indispensable to respect the adopted file organization.
Using this information, when confronting the database validity, if some register appears to be invalid, the database and the affected KML file will be updated.

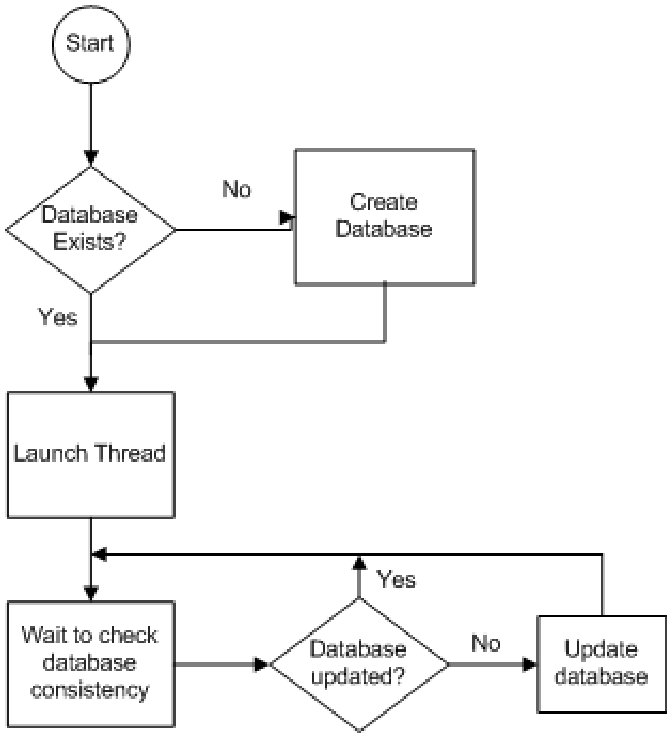


Figure 4.6: Server Load

4.3.3 Client Requests

The client can request three types of operations, create, edit and remove. On every operation some html forms are filled, gathering the required information to execute the respective RPC operation.
On the create operations, the records are created in the database and also in the KML file. In edit operations, the database and KML files are update. The remove operation consists in clearing the element from the database and the KML files.
In normal operations, the server always responds with operation results. The received results can be success or fail. If they are well succeeded, the client reflects the operation data into Google Earth.

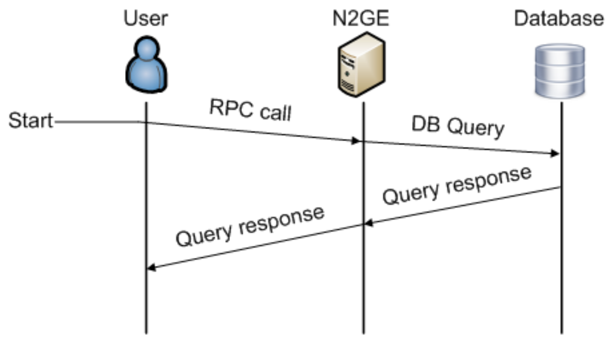


Figure 4.7: Client Request

4.3.4 Client Updates

The clients may perform several operations through time, and the system will be affected by these actions. All clients must be aware of the currents changes. To stay update, the clients have a timer that when its triggered invokes a RPC. The invoked RPC has the current client system view as argument. Is is just an integer value, that correspond to the last operation Id observed by the client.
When the webservice receives this RPC and the handler method receives the client system view, it compares it to the current server system view. Based on this information, the server knows the last operation unknown to the client, and creates a list with all missing information, and other list with the information that must be deleted. When these compiled results reach the client, it will be updated.

This approach is more efficient, since the client does not have to download all the information from the database, only the missing information.

4.3.5 P2P MIB Monitoring

The P2P network used as a testbed of the LBNMS system is a project where a middleware was implemented, in order to improve the performance of P2P networks over MANETs. This middleware, the P2P-HWMP, uses a cross-layer approach to collect information from the Mac layer (e.g., mesh path, peer nodes), and acts into the routing decisions using the application layer information (e.g., database with the files and peers storage them), improving the p2p applications performance. This P2P network works in an unstructured overlay, and uses the HWMP routing protocol.

To perform the P2P network monitoring, the adopted solution consists in to install the monitoring system in a P2P node charged to monitor the other overlay peers. This particular node runs the Nagios with the N2GE instance. As figure 4.8 suggests, it is a very simple way of collecting information, since the node can monitor every other node in the overlay, if there are some anchors to every node. This solution works very well for closer nodes or mesh topologies without any partition, where every node can be reached by using mesh forwarding. Once the SNMP messages reaches all the nodes, all the peer information can be collected from the P2P MIB's, offering detailed P2P information.

The MIB's information are filled by the P2P-HWMP middleware, for certain type of events, like the bytes sent/received, peers entering/exiting the mesh, search queries received or dropped.

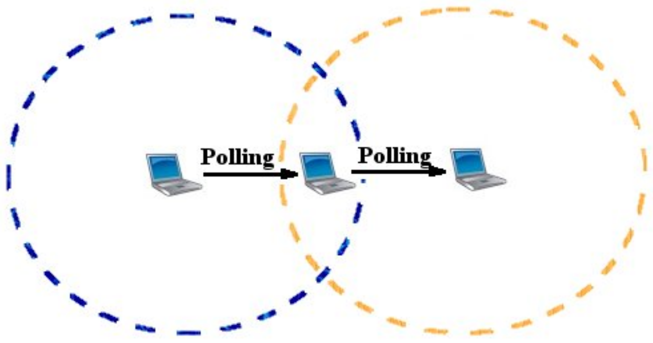


Figure 4.8: Example of polling messages forwarded in the mesh network

Chapter 5

Tests

This chapter presents the tests performed, in order to ensure proper implementation of the developed NMS. The performed tests were applied in two different scenarios, in order to test both static and mobile networks. The IST network was used to test a network with fixed equipments, while the ad-hoc over Manet was used to test the mobility component, made by mobile terminals.

5.1 IST Network

5.1.1 Scenario

To test the implemented NMS in fixed networks, a special test access was made available by CIIST to both the Taguspark and Alameda networks. To perform monitoring of the network elements of the two campus, the NMS server had access to the management VLAN.

This network provides an important test bed, because it is a complex network, composed by a large number of heterogeneous equipments, such as AP's, switchs, servers and routers from many vendors and different series.

5.1.2 Test Specification and results

5.1.2.1 Network Mapping

This test aims to represent some N2GE objects in both IST campi, and observe the correct operation of them. The first objects created were the 3D models, representing the buildings structure. After this step all network elements and links on both campus were created. To finish this test, two cameras were created, one for each campi.

The Google Earth has some problems concerning indoor representation. For this reason, only some network elements for each zones of the map were represented, avoiding a confusing map with too many equipments together.

After lauching the N2GE server and execute all test phases, the outcome results are illustrated in figures 5.1 to 5.4.

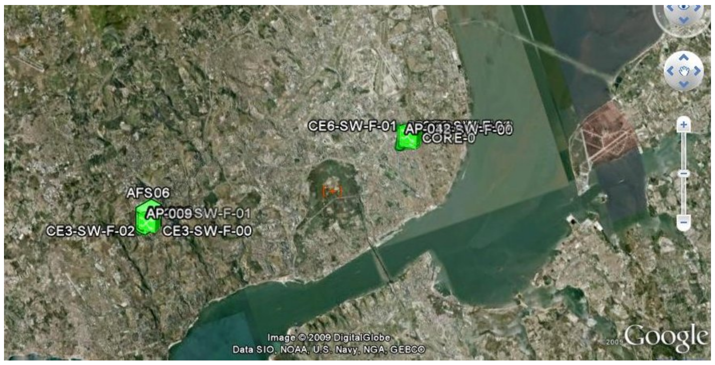
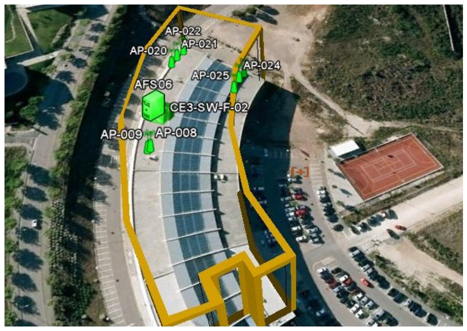


Figure 5.1: Complete IST-Network View



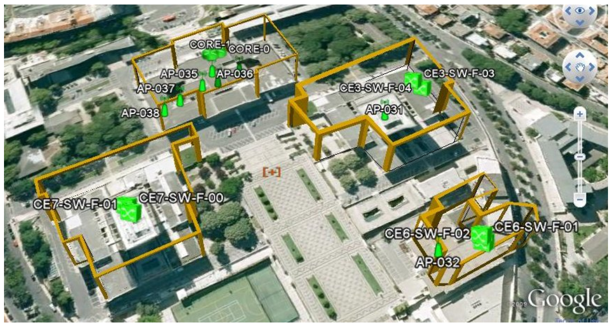
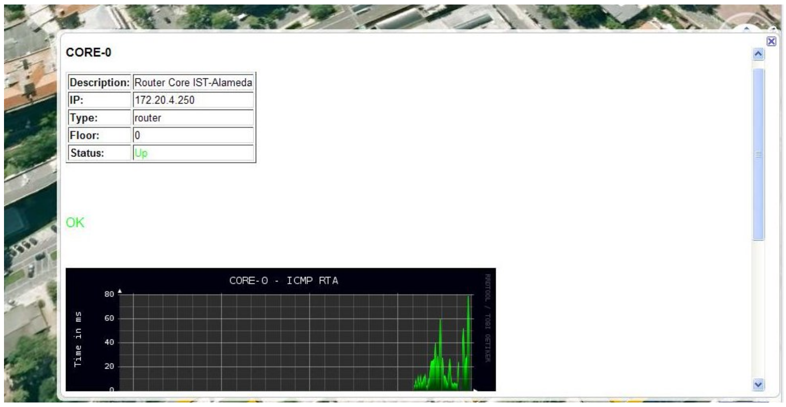


Figure 5.2: IST-Taguspark network mapped to Google Earth

Figure 5.3: IST-Alameda network mapped to Google Earth

During this mapping did not occur any problem, and all the N2GE objects were placed correctly. Additionaly, the HTML pages for each node were analyzed, looking at their values and graphs. The figure 5.4 shows the HTML page for the Core-0 router, which is like all others equipments, and was correctly represented.



CORE-0

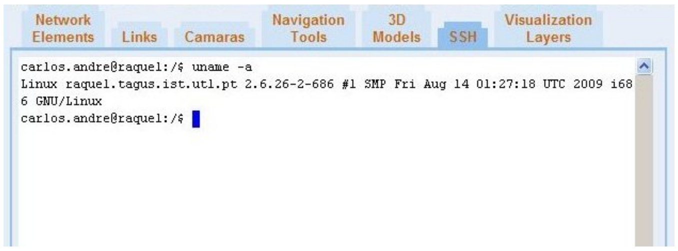
Description:	Router Core IST-Alameda
IP:	172.20.4.250
Type:	router
Floor:	0
Status:	Up

OK

Figure 5.4: Core-0 HTML page

5.1.2.2 WebSSH

To test the ssh applet, a simple remote access to the server raquel was established. Figure 5.5 shows the ssh applet working in the N2GE environment.

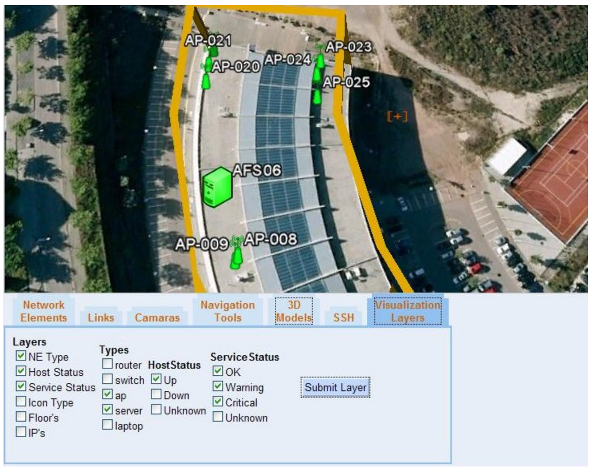


```
carlos.andre@raquel:/$ uname -a
Linux raquel.tagus.ist.utl.pt 2.6.26-2-686 #1 SMP Fri Aug 14 01:27:18 UTC 2009 i68
6 GNU/Linux
carlos.andre@raquel:/$
```

Figure 5.5: Remote access using mindterm ssh applet

5.1.2.3 Visualization Layers

The N2GE has several visualization layers. To cover all of them, three tests were performed. On the first test, a combination of three layers were used, related to the type of equipment and the state of each one. This layer will filter other equipments that are not server or AP type, and different from Up state. After the test was performed, since all equipments are Up, only the switchs were filtered, as expected. This result is shown on figure 5.6.



Layers
- [✓] NE Type
- [✓] Host Status
- [✓] Service Status
- [ ] Icon Type
- [ ] Floor's
- [ ] IP's

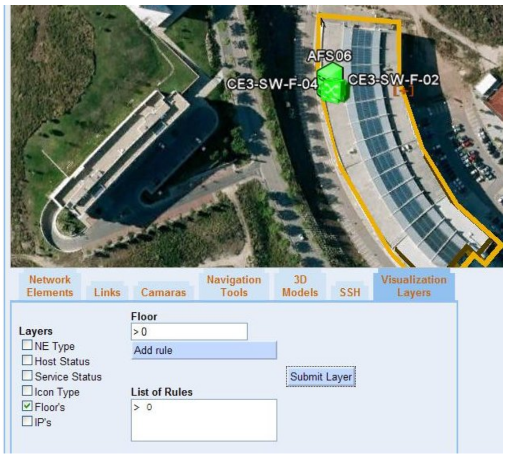
Types
- [ ] router
- [ ] switch
- [✓] ap
- [✓] server
- [ ] laptop

HostStatus
- [✓] Up
- [ ] Down
- [ ] Unknown

ServiceStatus
- [✓] OK
- [✓] Warning
- [✓] Critical
- [ ] Unknown

Figure 5.6: Layer applied into IST-Taguspark network

On the second test, the floor feature was tested, trying to filter all the equipments above the ground floor. All the AP's were positioned at the ground floor on every class room, while the mapped switchs and the server are on the first floor, at CIIST premises. Applying the floor rule, i.e., higher than zero, all the AP's should be filtered, and only the equipments in higher floors should remain. After the test was performed, the obtained results are shown in figure 5.7, and prove that the layer works correctly.



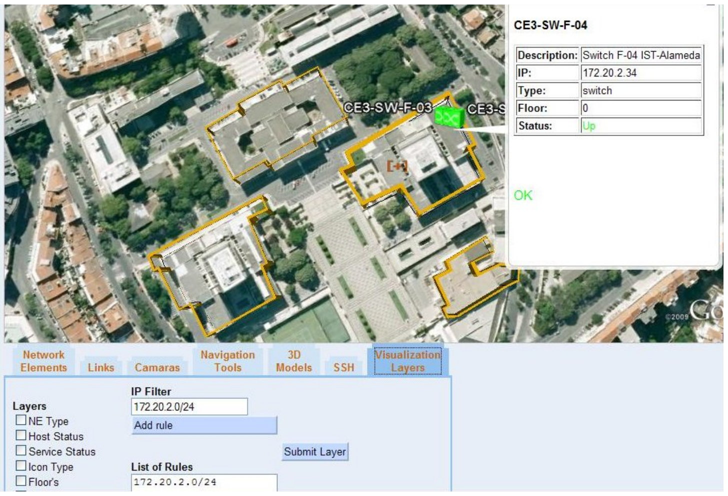
Layers
☐ NE Type
☐ Host Status
☐ Service Status
☐ Icon Type
☑ Floor's
☐ IP's

Floor: > 0

List of Rules: > 0

Figure 5.7: Floor layer applied into IST-Taguspark network

For the third and last test, the IP feature was tested, its aim was to test the subnet filter. For the Alameda network, the mapped switchs are the only equipments in the 172.128.2.0/24 subnet. Performing the filter based on this information, we can demonstrate the correct operation of this layer. As it is shown in figure 5.8 shows the layer is applied, the remaining equipments were the switchs, which is correct.



CE3-SW-F-04

Description:	Switch F-04 IST-Alameda
IP:	172.20.2.34
Type:	switch
Floor:	0
Status:	Up

Layers
☐ NE Type
☐ Host Status
☐ Service Status
☐ Icon Type
☐ Floor's

IP Filter: 172.20.2.0/24

List of Rules: 172.20.2.0/24

Figure 5.8: IP layer applied into IST-Alameda network

5.1.2.4 Concurrency

This test pretends to launch simultaneous destructive operations (write/delete) by multiple users, and prove that only one user can be successful. The selected number for concurrent access were three users, with the current system view, because a client cannot perform destructive operation, unless he is updated, seeing all the changes made on the NMS.

Since it is impossible to launch the three requests ate the same time, the adopted solution consists in adding some line codes into the N2GE servlet. These line codes locks every thread created on the server, during a specified time. This approach provides the required time to fill HTML forms on every client, and submit a destructive operation. On the server side, all the client threads are locked and released at the same time. Just one client should gain access, and the others must fail to make their operation.

After the test, all went well because two of them received an popup message saying that the operation failed, and only one submitted the operation sucessfuly.

5.1.2.5 Consistency

The consistency tests are important to avoid an inconsistent NMS. These inconsistencies could appear when some user tries to make an update, and he/she is not aware of the last submited operation. Another and very frequent event for this kind of application, results from changes on the Nagios configuration files, in order to add another equipment/service to monitor, or even to delete.

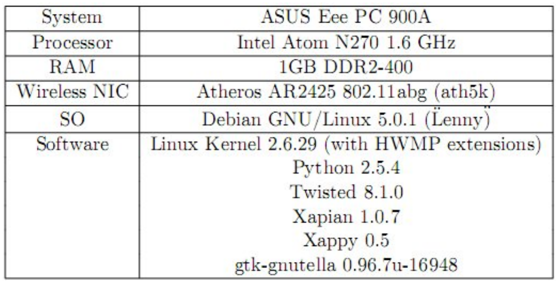
As described on the implementation chapter, all user operations include the system view. When it is received by the server side, it is compared to the current system view, and from this action we can check if the user did not lost any destructive operation (update, insert or delete). To test this issue, two users were used. User A starts to make some operations, as the creation of two network elements, and after that, the user B logs into Nagios and receives the current view. Immediately afterwards, the user A creates another network element and a network link. If the user B tries some destructive operation, he/she should not succeed. This test run as expected.

For the other test, the worst case happens when some administrator deletes some information, and the N2GE must be aware of such changes. For this test, typical configuration changes were performed, namely removing completly one equipment, updating the IP address or deleting one service. It was expected that the thread responsible by analysing the Nagios configuration files would be able to deal with these situations. For each situation, the thread was able to notice this incoherence, and performed the expected updates on the database and the afected KML files.

5.2 P2P Network

5.2.1 Scenario

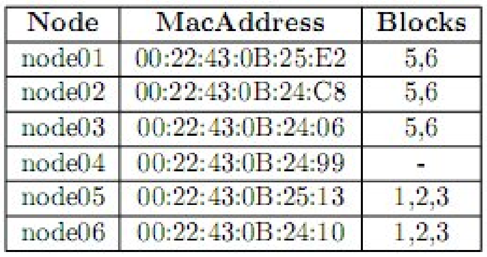
To accomplish this test, a P2P network running over a MANET implemented by 6 netbooks (Asus EeePC) was used, with identical software and hardware configuration, except for the special node, responsible to run the Nagios with the N2GE instance and monitor the entire P2P network. A detailed information about these nodes configuration is listed below.



System	ASUS Eee PC 900A
Processor	Intel Atom N270 1.6 GHz
RAM	1GB DDR2-400
Wireless NIC	Atheros AR2425 802.11abg (ath5k)
SO	Debian GNU/Linux 5.0.1 (L̈enny")
Software	Linux Kernel 2.6.29 (with HWMP extensions) Python 2.5.4 Twisted 8.1.0 Xapian 1.0.7 Xappy 0.5 gtk-gnutella 0.96.7u-16948

Figure 5.9: Node Configuration

In this network, there are two diferent networks operating simultaneously, one cabled and another one mobile. The cabled network is used to receive a WAN connection from the gateway (Linksys wireless-G), to access Google Earth service and facilitate the configuration of the P2P nodes using SSH. The attributed IP ranges distributed to cabled and mobile networks were 192.168.2.0/24 and 192.168.1.0/24, respectively. The middleware nodes includes an option to block the desired peers neighbours, dropping their packets, this way refusing peer links with them (at link layer). Using this option, we can create an overlay without to move the mobiles positions, remaining on the same place (testbed table), and forcing the desired overlay. The selected overlay enables to test multi-hop search throughout the network, with multiple possible paths. Figure 5.10 and 5.11 ilustrate the selected network topology, and their respective blocks table.



Node	MacAddress	Blocks
node01	00:22:43:0B:25:E2	5,6
node02	00:22:43:0B:24:C8	5,6
node03	00:22:43:0B:24:06	5,6
node04	00:22:43:0B:24:99	-
node05	00:22:43:0B:25:13	1,2,3
node06	00:22:43:0B:24:10	1,2,3

Figure 5.10: Bloked nodes

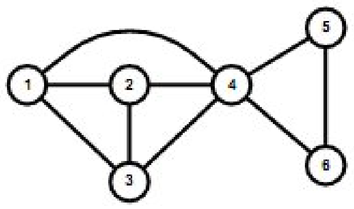


Figure 5.11: Network Topology

5.2.2 Test Specification and results

5.2.2.1 Network Mapping

This test validates the correct operation of the N2GE basic operation when applied to P2P over MANET networks. The procedure consists in the creation of all peer nodes and the gateway. Furthermore, a network link was added between the monitor node and the gateway. The obtained result, is illustrated in figure 5.12.

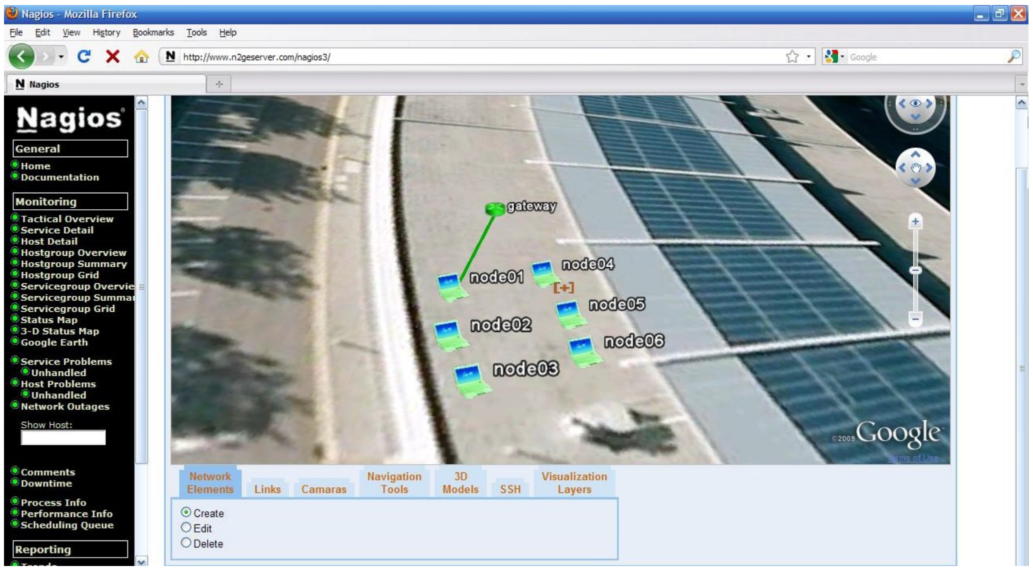


Figure 5.12: Ad-Hoc network mapped to Nagios

The green icon color of each equipment suggests the correct operation of each equipment. In order to perform a more detailed analysis of each object, all the extra information displayed in the HTML ballons were analysed. All the IP's were correct, and the graphics were correctly extracted from Nagios.

After the network mapping was created, it was tested an external access, by a laptop connected to the cabled network. Everything run as expected, since the laptop logged into Nagios and loaded the N2GE from the monitor node, and it observed the current mapping network. This access indicates an important feature, that the network can be monitored remotely.

5.2.2.2 P2P operations monitoring

This test tries to reproduce some typical operations in an P2P network, such as a search for a given document and other resources. This operations will affect the MIB tables, and the goal of this test is to use the N2GE to observe this events, by analysing some graphics generated for some MIB variables.

Before starting the test, it was used the P2P resource search service to find which P2P SNMP agents are running, and from the nodes that reply to this query, it was checked that all of them were running the SNMP agent.

With the guarantee about the running agents, several requests were generated from the monitor node (other node could be used). These requests have modified the nodes MIB tables. The used syntax for the search command was the following, searching for debian packages.

./search.py -a file ”extension:deb”

After several queries were generated, some traffic was generated. By the graphics shown in figures 5.13-5.18, we can conclude that all nodes received approximately the same number of queries, which result in this traffic. Other graphics were observed, but remained static, e.g., the number of documents and database shared by every node. Other interesting test was related with the monitoring of the current number of peers neighbours. However, this variable had some operational problems and it was not possible to populate it, remaining empty. If all run as expected sending down node4, node01, node02 and node03 would have two neighbours, while node05 and node06 only one.

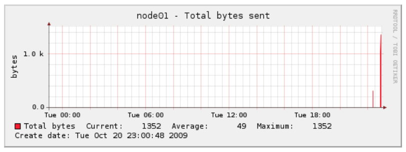


Figure 5.13: Bytes Sent from node01

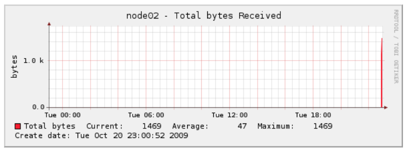


Figure 5.14: Bytes Received from node02

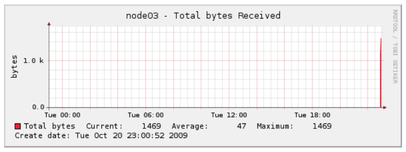


Figure 5.15: Bytes Sent from node03

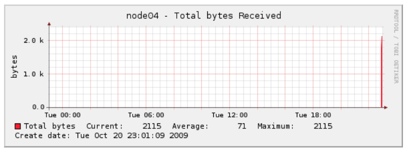


Figure 5.16: Bytes Received from node04

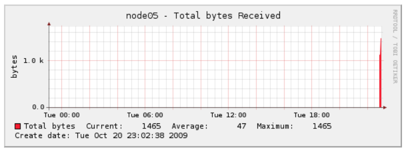


Figure 5.17: Bytes Sent from node05

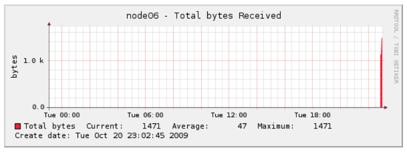


Figure 5.18: Bytes Received from node06

5.2.2.3 Mobility Test

For this test, the desired goal was to test a mobility situation that can ocur in a mobile network. The test scenario consists on the separation of two nodes relative to the overlay, creating two islands when they become unreachable from the monitoring node.

The developped NMS does not include any mechanism to perform an automatic mapping of network elements to Google Earth, only manual, implying intervention of the network administrators to specify the precise location. In contrast, some of the applications seen on the state of the art have features to perform automatic network mapping, without the need to network administrators to manually specify the GPS coordinates. To resolve this issue, it could be used a new variable on the MIB tables indicating the current equipment's GPS coordinates (longitude and latitude), collected from a GPS receptor attached to the laptop. But since no GPS equipment was available, and the tests were performed indoor, one python script was used to simulate mobility. This script inject GPS positions into the database, for nodes 5 and 6. These positions were collected in multiple campus positions, from a certain path that increase its distance from the overlay. At certain point the distance between nodes 5 and 6 became unreachable, and this will result in another island.

Therefore, the test objective was to separate the nodes and see their positions being updated in Google Earth, and after the monitor node loses their radio signal, to see them become red. The script has an ending condition that stops to inject GPS coordinates when the respective node becomes into critical status. When run for each of the nodes, it inject new GPS coordinates every 2 minutes, giving time to move the network element. The following figure shows the location of the peers when they have totaly separated from the overlay. It was possible to see their mobility by the campus, and their connections falling down.

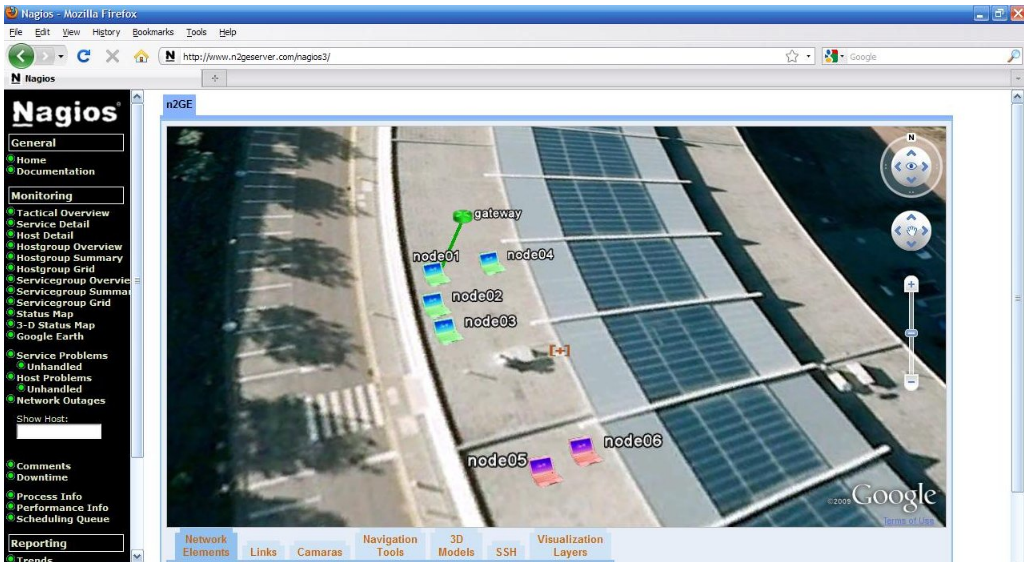
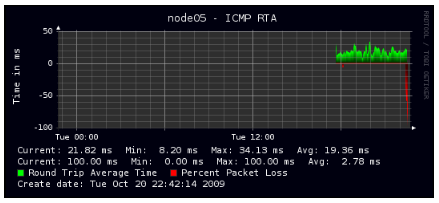


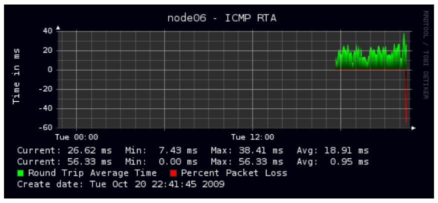
Figure 5.19: Network status after node 5 and 6 signal faded away

After the test, the generated graphics on the monitor node were analysed, it was observed that the connections failed slowly, with the RTT increasing with time, until complete loss of the source signal.



```
Current: 21.82 ms   Min:   8.20 ms   Max: 34.13 ms   Avg: 19.36 ms
Current: 100.00 ms   Min:   0.00 ms   Max: 100.00 ms   Avg:  2.78 ms
Create date: Tue Oct 20 22:42:14 2009
```

Figure 5.20: Ping for node05



```
Current: 26.62 ms   Min:   7.43 ms   Max: 38.41 ms   Avg: 18.91 ms
Current: 56.33 ms   Min:   0.00 ms   Max: 56.33 ms   Avg:  0.95 ms
Create date: Tue Oct 20 22:41:45 2009
```

Figure 5.21: Ping for node06

Chapter 6

Conclusion

This thesis proposes the development of a georeferenced NMS, based on the integration of the Nagios and the Google Earth systems. This solution explores the Google Earth technological potential to implement a rich NMS, displaying in real-time all the network information in a very interactive way.

One desired feature of this project is associated with the support of different scenarios. For this purpose, the developped NMS was tested using the IST network, and also a P2P network over a MANET. With these tests, it was possible to prove how important these systems can be, providing extra information and tools to monitor the network, helping even more network administrators.

The test results allowed to validate all the NMS features, namely the network mapping to Google Earth, WebSSH and Visualization Layers. The mobility test gave some positive notes about the NMS potential to create an automatic update for each node location.

We expect that this work represents a valuable contribution to the open source community, providing a suitable alternative to existing proprietary software.

6.1 Future Work

For future work, this application can include support for GPS receivers, and perform an automatic location update, like the project BelView. To implement this feature, it is only required to update the N2GE servlet to receive data from the GPS receiver and inject the coordinates into the database or directly into the KML files.

Another interesting feature is the possibility of improving network planning and design, which is a specially important feature for wireless networks. Since Google Earth can export 3D models to the graphical environment, and to populate some kml files with the levels of fog/humidity/temperature/rain for each point in the map, some works can benefit from this data and generate more accurate signal coverage into the map.

Regarding to P2P networks, the used solution for this work was very simple, but it contributes with some useful results to more sophisticated future developments. To improve the adopted solution, the P2P resource discovery service could be used to find new peers and to know their new IP address and available resources to be monitored. This approach does not need manual configuration in order to add another node to be monitored. In contrast, the implemented solution requires pre-defined IP nodes and monitoring services.

Other important feature is the representation of huge indoor networks. To solve this problem, the support of Flash [1] in the HTML ballons could help. With Flash programs, it is possible to create all possible building scales, and watch every place on it. The idea would be to use regular Google Earth for outdoor environment, and to develop a Flash-based 3D model for indoor environment when required.

Appendix A

Appendix

A.1 P2P Middleware - SNMP Specification

The P2P middleware exports a set of statistics using an embedded agent which can be queried using SNMP. The MIB can be found rooted under the Instituto Superior Técnico arc(.1.3.6.1.4.1.21512) with a child number of 99. The tree is further subdivided according to the internal middleware services, namely query routing, storage, peer discovery and service registration. A brief description of each available OID follows.

- .1.3.6.1.4.1.21512.99.1 - Routing service
  - .1.0 - Search requests sent(INTEGER): Number of search requests sent by the peer.
  - .2.0 - Search requests received(INTEGER): Number of unique search requests received from other peers in the network.
  - .3.0 - Search requests replied(INTEGER): Number of unique search requests the peer has replied to.
  - .4.0 - Pending search requests(INTEGER): Number of search requests sent by the peer which a repending,i.e., search requests for which no reply has been received and which haven't yet timed out.
  - .5.0 - Search replies received(INTEGER): Number of search replies received in response to search requests sent by the peer.
  - .6.0 - Bytes sent(INTEGER): Aggregate size in bytes of all the search request and search reply messages sent by the peer.
  - .7.0 - Bytes received(INTEGER): Aggregate size in bytes of all the unique search request and search reply messages received by the peer.
  - .8.0 - Mean search reply delay(INTEGER): Mean time in milliseconds between sending a search request and receiving a corresponding search reply.
  - .9.0 - Mean query result delay(INTEGER): Mean time in milliseconds between accepting a new query and returning its results.

  - .10.0 - Resolved queries(INTEGER): Number of queries sent by the peer for which at least one result was received.
  - .11.0 - Search replies with embedded results(INTEGER): Number of search replies sent by the peer containing the entire set of results.

- .1.3.6.1.4.1.21512.99.2 - Storage service
  - .1.0 - Number of databases(INTEGER): Number of document databases managed by the storage service.
  - .2.0 - Numberofdocuments(INTEGER): Total number of documents stored in the document databases.
  - .3.0 - Numberofshareddatabases(INTEGER): Number of document databases managed by the storage service which are searchable by remote peers.
  - .4.0 - Number of shared databases(INTEGER): Total number of documents stored in shared document databases.

- .1.3.6.1.4.1.21512.99.3 - Peer selection service
  - .1.0 - Number of neighbours(INTEGER): Number of mesh neighbours of the peer(includes legacy and P2P nodes).
  - .2.0 - Number of mesh paths(INTEGER): Number of active mesh paths in the peer's HWMP forwarding table.
  - .3.0 - Number of known peers(INTEGER): Number of peers known by the peer.

- .1.3.6.1.4.1.21512.99.4 Service registration service
  - .1.0 Number of registered services(INTEGER): Number of services registered with the service registration service and which are discoverable by other peers in the network.

Bibliography

[1] http://pt.wikipedia.org/wiki/AdobeFlash Accessed in 17 October 2009. 48

[2] http://sketchup.google.com Accessed in 17 October 2009. 26

[3] https://oss.oetiker.ch/rrdtool Accessed in 17 October 2009. 26

[4] https://www.hibernate.org Accessed in 17 October 2009. 25

[5] http://youmonitor.us Accessed in 17 October 2009. 10

[6] http://youmonitor.us/services.shtml Accessed in 17 October 2009. 10

[7] www.cacti.net Accessed in 17 October 2009. 20

[8] www.python.org Accessed in 17 October 2009. 26

[9] www.zabbix.com Accessed in 17 October 2009. 20

[10] http://twistedsnmp.sourceforge.net/ Accessed in 28 September 2009. 31

[11] http://www.nagios.org/ Accessed in 28 September 2009. 3

[12] Paramvir Bahl, , Paramvir Bahl, and Venkata N. Padmanabhan, *Radar: An in-building rf-based user location and tracking system*, 2000, pp. 775–784. 12

[13] M. N. Borenovic, M. I. Simic, A. M. Neskovic, and M. M. Petrovic, *Enhanced cell-id + ta gsm positioning technique*, Computer as a Tool, 2005. EUROCON 2005.The International Conference on, vol. 2, 2005, pp. 1176–1179. 16

[14] J. Case, M. Fedor, M. Schoffstall, and J. Davin, *A simple network management protocol (snmp)*. 1

[15] Andy Harter, Andy Hopper, Pete Steggles, Andy Ward, and Paul Webster, *The anatomy of a context-aware application*, MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, ACM Press, 1999, pp. 59–68. 14

[16] Teresa Heitor, Ana Tomé, Paulo Dimas, and João Pedro Silva, *Measurability, representation and interpretation of spatial usage in knowledge-sharing environments - a descriptive model based on wifi technologies*, December 2007. 8

[17] Werner Kumm, *Gps global positioning system*, 2000. 15

[18] António Marques, *P2P over mobile ad-hoc network*, September 2009. 3

[19] Nissanka Bodhi Priyantha, Anit Chakraborty, and Hari Balakrishnan, *The Cricket Location-Support System*, 6th ACM MOBICOM (Boston, MA), August 2000. 14

[20] M. Ripeanu, *Peer-to-peer architecture case study: Gnutella network*, August 2001, pp. 99–100. 18

[21] A. Sevtsuk and C. Ratti, *ispots: How wireless technology is changing life on the mit campus*, 2005. 8

[22] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan, *Chord: A scalable peer-to-peer lookup service for internet applications*, 2001. 11

[23] Roy Want, Andy Hopper, Veronica Falcão, and Jonathan Gibbons, *The active badge location system*, ACM Trans. Inf. Syst. **10** (1992), no. 1, 91–102. 15

[24] U. Warrier, L. Besaw, L. LaBarre, and B. Handspicker, *Common Management Information Services and Protocols for the Internet (CMOT and CMIP)*, October 1990. 1