**Chapter 1**

**PREAMBLE**

**1.1 Introduction**

A Network Monitoring System monitors the network for problems caused by overloaded and/or crashed servers, network connections or other devices. Simple Network Management Protocol (SNMP) is an application–layer protocol defined by the Internet Architecture Board (IAB) in RFC1157 for exchanging management information between network devices. It is a part of Transmission Control Protocol⁄ Internet Protocol (TCP⁄IP) protocol suite.

SNMP is one of the widely accepted protocols to manage and monitor network elements. Most of the professional–grade network elements come with bundled SNMP agent. These agents have to be enabled and configured to communicate with the Network Management System (NMS). It is a flexible technology that can be used across operating systems and device types. For example, the SNMP agent on a router can provide information about the device's network configuration and operations, such as the device's network interfaces, routing tables and network traffic statistics. The Management Information Base (MIB) is like a two-way interpreter. The MIB lets an SNMP agent know what Object Identifier (OID) or SNMP objects the remote device understands and provides a structure for the SNMP agent to communicate with the device. The OID is displayed as a series of numbers separated by dots that represent where on the MIB Tree the object is located. The target device, such as a router, already understands the commands that it supports and how it will respond to commands it receives. The structure, or rules, of the MIB is defined in SMI (Structure of Management Information) files also called MIB files.

**1.2 Existing System**

**Monitoring the existing network is very challenging**:

* It involves connecting to every node to monitor its status.
* It is very cumbersome and involves a lot of manpower.
* There is no dedicated Web interface where the entire set of nodes can be monitored at one shot.
* It involves an outdated model of network monitoring to fulfill its needs and is not a convenient and secure way of monitoring.

**1.2.1 Limitation of the Existing system**

* No centralized system to monitor a network.
* Troubleshooting is difficult (Finding the source or error in the network is difficult)
* Cannot see the status of all the system nodes at one shot.
* Each switch or router has its own configuration and interface.

**1.3 Problem Statement**

To develop a network monitoring system that is used to identify the different problems that arise in a network. These problems should be periodically reported to the administrator who would perform the required action remotely from any system connected to the network. We are reconfiguring the architecture by using an agent less monitoring system and providing existing system lacks a good GUI which must be rectified for ease of usage.

**1.4 Objective of the project**

* We intend to design and develop a network monitoring utility that is open source and customized for the use at an organization.
* The system to be designed supports agent-less access and also a different web interface through which central access to details of all switches can be provided.
* Each software has its own interface, but we are designing a system which helps to access data from all switches and routers irrespective of the interface.

**1.5 Proposed System**

* Provides a centralized system to monitor the whole network.
* Status of all the system nodes available at one shot.
* Easy troubleshooting.
* The system is accessed by the clients using a Web Interface (Web Browsers).
* Developed using SNMP programming in java.
* Network monitoring server: - It contains software installed.

**1.5.1 Advantages of Proposed System**

* Provides a centralized system to monitor the whole network.
* Status of all the system nodes available at one shot.
* Easy troubleshooting.
* The system is accessed by the clients using a Web Interface (Web Browsers).
* Developed using SNMP programming in java.
* Network monitoring server: - It contains software installed.

**1.6 Phase Description**

|  |  |  |
| --- | --- | --- |
| **Phase** | **Task** | **Description** |
| Phase 1 | Analysis | Analyzing the core of the IEEE paper and provide Literature review based on analysis. |
| Phase 2 | Literature survey | Collect raw data and elaborate on literature surveys. |
| Phase 3 | System analysis | Analyses the requirements of the project and lists the specific requirements needed. |
| Phase 4 | Design | Object designing and Functional description |
| Phase 5 | Implementation | Implement the code based on the object specification |
| Phase 6 | Testing | Test the project according to Test Specification |
| Phase 7 | Documentation | Prepare the document for this project with conclusion and future enhancement. |

**Table 1.6:** Phase Description

**1.7 Organization of the Project Report**

The project report is organized as follows:

**Chapter 2: Literature Review** - Gives a brief overview of the survey papers and the research sources that have been studied to establish a thorough understanding of the project under consideration.

**Chapter 3: Theoretical Background** - Establishes groundwork for the proposed project by giving a detailed analysis of the project topic, existing research relevant to the project, arguments in favor and against the existing solutions and finally explores the motivation behind the proposed solution.

**Chapter 4: System Requirement Specification** - Discusses in details about the different kinds of requirements needed to successfully complete the project.

**Chapter 5: System Analysis** - gives details about several analysis that are performed to facilitate taking decision of whether the project is feasible enough or not.

**Chapter 6:** **System** **Design -** Gives the design description of the project, conceptual and detailed design well supported with design diagrams.

**Chapter 7: Implementation** - Discusses the implementation details of the project and reasons the use of the programming language and development environment.

**Chapter 8: Testing -** Briefs the testing methods used for testing the different modules in the project.

**Chapter 9: Results and Performance Analysis -** Gives the snapshots and graphs of the proposed protocols.

**Chapter 10: Conclusion and Future Scope -** Gives the concluding remarks of the project, throwing light on its future aspects.

**References** Lists the websites and references referred during the project work.

**Chapter 2**

**LITERATURE SURVEY**

In order to get required knowledge about various concepts related to the present analysis existing literature were studied. Some of the important conclusions were made through those are listed below.

1. **“*An Efficient Network Monitoring and Management System*” - Rafiullah Khan, Sarmad Ullah Khan, Rifaqat Zaheer, and Muhammad Inayatullah Babar, International Journal of Information and Electronics Engineering, Vol. 3, No. 1, January 2013–** In this paper we learnt how large organizations always require fast and efficient network monitoring system which reports to the network administrator as soon as a network problem arises. This paper presents an effective and automatic network monitoring system that continuously monitor all the network switches and inform the administrator when any of the network switch goes down.
2. **“*A Network Monitoring System Design under LAN*” -Liang Fan Shaanxi Regional Electric. Power Group, Xi''an, China Li Fei, Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on (Volume:4 ) –** In this paper designed the network monitoring system in Ethernet environment, described the physical topology structure of the system running, and analyzed the system functional structure design in detail based on the overall software structure of the system. At the same time accounted for the detailed design of the database table.
3. **“*Agent-based Knowledge Acquisition in Network Management Domain* - Sameera Abar, Hideaki Hatori, Toru Abe, Tetsuo Kinoshita, Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA’05) -** In this paper we learn thatto cope with the complexity and broad scope of areas and functions that are involved with a network management system (NMS), a knowledge model should be provided for the system. The aim of this paper is to propose the design of a multi-agent based network management support system with well-organized representation of the formal semantics of networks’ experiential knowledge related to NMS fault diagnosis functional area.

**Chapter 3**

# THEORITICAL BACKGROUND

Theoretical background highlighting some topics related to project work. The description contains several topics which are worth to discuss and also highlight some of their limitation that encourage going on finding solution as well as highlights some of their advantages for which reason these topics and their features are used in this project.

**3.1 Overview on Network Monitoring System**

A computer network consists of network elements like nodes, links, routers etc. The network administrator (manager) is the one who manages the network. Administrator faces problems in managing the network. Adding a new network element requires configuration maintenance. When the network is functioning some of the network elements like nodes, links, routers may go down. It creates obstacles in network functions. Also, various nodes use the network for variable times; hence for accounting the information about the use of network by every node must be maintained. The administrator or some special user must be able to find out performance of a node. If data in the network is confidential then it must be protected from outside world. Also there are different categories defined for the user and the data should be accessible to them as per their privilege level.

**3.2 SNMP Basics**

**SNMP** stands for Simple Network Management Protocol and consists of three key components: Managed Devices, Agents, and Network-Management Systems (NMSs). A managed device is a node that has an SNMP agent and resides on a managed network. These devices can be routers and access servers, switches and bridges, hubs, computer hosts, or printers. An agent is a software module residing within a device. This agent translates information into a compatible format with SNMP. An NMS runs monitoring applications. They provide the bulk of processing and memory resources required for network management.

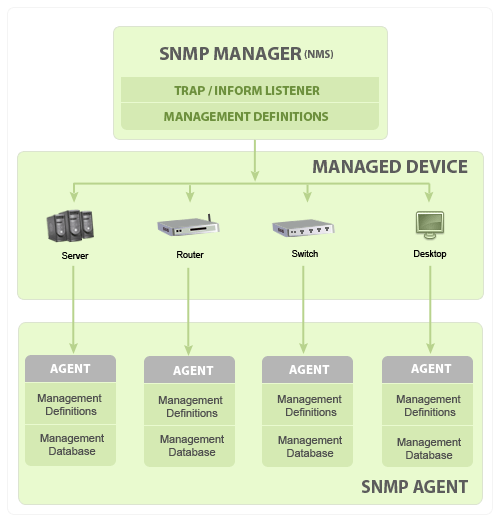


Figure 3.2: Working of SNMP

**3.3 What is an Object Identifier (OID)?**

An Object Identifier is a name used to identify an object. This object can be a country or an individual disk drive. The most common one, in the context of the IEEE-RAC, is the OUI (Organizationally Unique Identifier), and the organizationally derived, and assigned, assignments beyond the OUI. Most common subsequent identifiers include Ethernet address identifier, the Extended Unique Identifiers (EUI) or the World Wide Name (WWN) identifiers. Uniqueness, for conforming systems, is a prized property in both of these cases. Uniqueness is assumed by the building of a unique number starting with the OUI. This IEEE-RAC assigned OUI is an Object Identifier for the organization. This is a layer within a larger context of identifiers uniquely derived from the starting point of all OID’s, the International Telecommunication Union Telecommunication Standardization Sector (ITU-T) and described in the ASN.1 standard. The path traceable to the ITU-T starting point is called the “arc” of the OID. This arc extends to the OUI and other RAC assigned designators and through the assignments made by the organization to the end point object identification.

**MIB** stands for Management Information Base and is a collection of information organized hierarchically. These are accessed using a protocol such as SNMP. There are two types of MIBs: scalar and tabular. Scalar objects define a single object instance whereas tabular objects define multiple related object instances grouped in MIB tables.

**OIDs** or Object Identifiers uniquely identify managed objects in a MIB hierarchy. This can be depicted as a tree, the levels of which are assigned by different organizations. Top level MIB object IDs (OIDs) belong to different standard organizations. Vendors define private branches including managed objects for their own products.

SNMP version 1 was the initial development of the SNMP protocol. A description can be found in Request for Comments (RFC) 1157 and it functions within the specification of the Structure of Management Information (SMI). It operates over User Datagram Protocol (UDP), Internet Protocol (IP), OSI Connectionless Network Services (CLNS), AppleTalk Datagram Delivery Protocol (DDP), and Novell Internet Packet Exchange (IPX). SNMP v1 is considered the de facto network management protocol in the Internet community.

SNMP works on the basis that network management systems send out a request and the managed devices return a response. This is implemented using one of four operations: Get, Get Next, Set, and Trap. SNMP messages consist of a header and a PDU (protocol data units). The headers consist of the SNMP version number and the community name. The community name is used as a form of security in SNMP. The PDU depends on the type of message that is being sent. The Get, Get Next, and Set, as well as the response PDU, consist of PDU type, Request ID, Error status, Error index and Object/variable fields. The Trap consists of Enterprise, Agent, Agent address, Generic trap type, Specific trap code, Timestamp and Object/Value fields.

MIBs are a collection of definitions which define the properties of the managed object within the device to be managed (such as a router, switch, etc.) Each managed device keeps a database of values for each of the definitions written in the MIB. As such, it is not actually database but implementation dependent. Each vendor of SNMP equipment has an exclusive section of the MIB tree structure under their control.

In order for all of this to be properly organized, all of the manageable features of all products (from each vendor) are arranged in this tree. Each 'branch' of this tree has a number and a name, and the complete path from the top of the tree down to the point of interest forms the name of that point. This is the OID. Nodes near the top of the tree are extremely general I nature. For example, to get to the Internet, one has to reach to the fourth tier. As one moves further down, the names get more and more specific, until one gets to the bottom, where each node represents a particular feature on a specific device (or agent).

Samples

Here is a sample structure of an OID:

Iso(1).org(3).dod(6).internet(1).private(4).transition(868).products(2).chassis(4).card(1).slotCps(2)­.-cpsSlotSummary(1).cpsModuleTable(1).cpsModuleEntry(1).cpsModuleModel(3).3562.3

Or

1.3.6.1.4.868.2.4.1.2.1.1.1.3.3562.3

These numbers are the ones used in PRTG when setting up custom sensors, in order to access the appropriate elements of the device desired to be monitored. OIDs are generally provided by the hardware manufacturers or can be found in so-called OID repositories, where collections of MIB trees and the respective OIDs can be accessed. PRTG reads these OIDs and appoints them to the pertinent device, respectively monitoring the selected device and its OID specific.

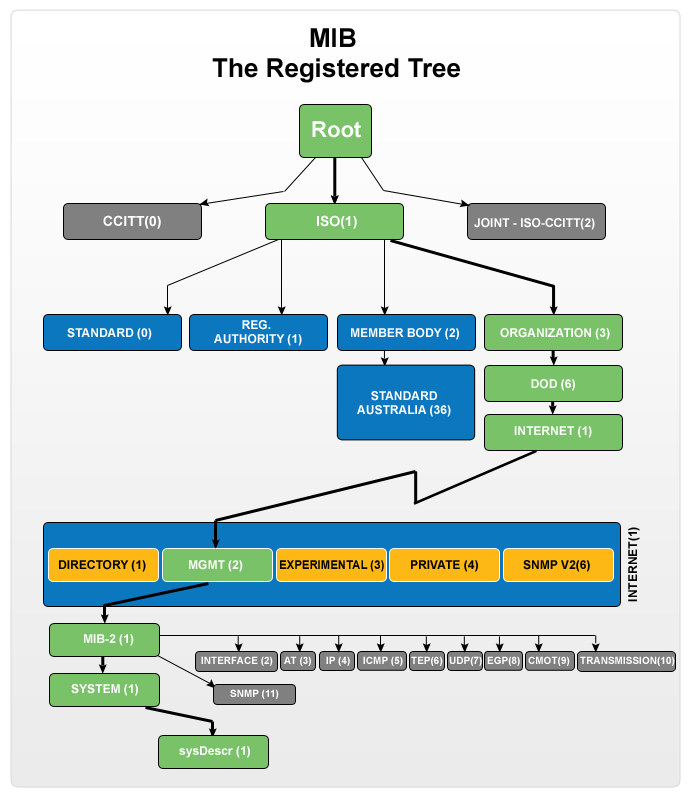


Figure 3.3: MIB Registered Tree

**3.4 Hubs, Switches and Routers Defined**

The functions of the three devices — the hub, switch and router

**3.4.1 Hubs**

These form the common connection point for devices in a network. Hubs are commonly used to connect segments of a LAN. A hub contains multiple ports. When a packet arrives at one port, it is copied to the other ports so that all segments of the LAN can see all packets.

**3.4.2 Switch**

In networks, a device that filters and forwards packets between LAN segments. Switches operate at the data link layer (layer 2) and sometimes the network layer (layer 3) of the OSI Reference Model and therefore support any packet protocol. LANs that use switches to join

Segments are called switched LANs or, in the case of Ethernet networks, switched Ethernet LANs.

**3.4.2 Routers are Completely Different Devices**

Routers are completely different devices. Where a hub or switch is concerned with transmitting frames, a router's job, as its name implies, is to route packets to other networks until that packet ultimately reaches its destination. One of the key features of a packet is that it not only contains data, but the destination address of where it's going. A router is typically connected to at least two networks, commonly two Local Area Networks (LANs) or Wide Area Networks (WAN) or a LAN and its ISP's network. For example, your PC or workgroup and EarthLink. Routers are located at gateways, the places where two or more networks connect. Using headers and forwarding tables, routers determine the best path for forwarding the packets. Router use protocols such as ICMP to communicate with each other and configure the best route between any two hosts.

**Chapter 4**

# SYSTEM REQUIREMENT SPECIFICATION

Software requirement Specification is a fundamental document, which forms the foundation of the software development process. It not only lists the requirements of a system but also has a description of its major feature. An SRS is basically an organization's understanding (in writing) of a customer or potential client's system requirements and dependencies at a particular point in time (usually) prior to any actual design or development work. It's a two-way insurance policy that assures that both the client and the organization understand the other's requirements from that perspective at a given point in time.

The SRS also functions as a blueprint for completing a project with as little cost growth as possible. The SRS is often referred to as the "parent" document because all subsequent project management documents, such as design specifications, statements of work, software architecture specifications, testing and validation plans, and documentation plans, are related to it. It is important to note that an SRS contains functional and nonfunctional requirements only; it doesn't offer design suggestions, possible solutions to technology or business issues, or any other information other than what the development team understands the customer's system requirements to be.

**4.1 Functional Requirement**

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. In this system following are the functional requirements:-

* Store the details of specific nodes such as switches, routers in a database.
* Display details of all switches used in each building involved in the network on a dashboard.
* Display the status of each port upon selecting the respective switch.
* Store connection details in a database which helps to pinpoint the exact reason for non- connectivity.
* Help the network administrator in implementing corrective measures to ensure good operation.

**4.2 Non Functional Requirement**

Non functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviors. They may relate to emergent system properties such as reliability, response time and store occupancy. Non functional requirements arise through the user needs, because of budget constraints, organizational policies, the need for interoperability with other software and hardware systems or because of external factors such as:-

* Product Requirements
* Organizational Requirements
* User Requirements
* Basic Operational Requirements
  + 1. **Product Requirements**

**4.2.1.1 Reliability** – The system is expected to be consistent. Alerts shall be provided to the administrators about the events happening in the system.

* + - 1. **Availability** - The software should be available all the time.

**4.2.1.3 Security** - Only authenticated and authorized users can access the monitoring system. Each of the functions specified should be accessed by only the users who have the right to access it. For instance, none other than the administrator shall get access to the NMS with an authentication.

* + - 1. **Maintainability**

Identify the flaws in system and take corrective action by providing regular updates

* + - 1. **Portability**

It should be able to deploy in a server with any type of operating system. The following things are recommended:

* The system should be developed as a web application
* Java platform can be used to develop the application, since it is portable among variety of operating systems.

Non functional requirements are also called the qualities of a system. These qualities can be divided into execution quality & evolution quality. Execution qualities are security & usability of the system which are observed during run time, whereas evolution quality involves testability, maintainability, extensibility or scalability.

**4.2.2 Organizational Requirements**

**Process Standards:** IEEE standards are used to develop the application which is the standard used by the most of the standard software developers all over the world.

**Design Methods:** Design is one of the important stages in the software engineering process. This stage is the first step in moving from problem to the solution domain. In other words, starting with what is needed design takes us to work how to satisfy the needs.

**4.2.3 User Requirements**

* The system requires a unique username and password for login
* Upon authentication, it provides all the details to the user
* The system also provides a dashboard to display the status of nodes in the network
* It consists of a database of all nodes present in the network
* The network manager would take the individual details of the node from the database and check for good operation
* The network monitor would report to the network manager which would in turn report the status to the client
* Administrator has the authority to provide or reject connectivity to the clients
* A normal user would be given only read only access to the system

**4.2.4 Basic Operational Requirements**

The customers are those that perform the eight primary functions of systems engineering, with special emphasis on the operator as the key customer. Operational requirements will define the basic need and, at a minimum, will be related to these following points:-

**Mission profile or scenario:** It describes about the procedures used to accomplish mission objective. It also finds out the effectiveness or efficiency of the system.

**Performance and related parameters:** It points out the critical system parameters to accomplish the mission

**Utilization environments:** It gives a brief outline of system usage. Finds out appropriate environments for effective system operation.

**Operational life cycle:** It defines the system lifetime

**4.3 Hardware Requirements**

* Processors **:** Pentium IV
* Processor Speed **:** 3.00 GHZ
* RAM **:** 2 GB
* Storage **:** 20 GB
* Monitor **:**  15 inches
* Keyboard **:**  Standard 102 keys
* Mouse  **:** Standard 3 buttons

**4.4 Software Requirements**

* Operating system **:** Windows 7
* Coding Language **:** Java/ SNMP Programming
* Visual Interface **:** Standard Web Browser

**Summary**

This chapter gives details of the functional requirements, non-functional requirements, resource requirements, hardware requirements, software requirements etc. Again the non-functional requirements in turn contain product requirements, organizational requirements, user requirements, basic operational requirements etc.

**Chapter 5**

**SYSTEM ANALYSIS**

**Overview**

Analysis is the process of finding the best solution to the problem. System analysis is the process by which we learn about the existing problems, define objects and requirements and evaluates the solutions. It is the way of thinking about the organization and the problem it involves, a set of technologies that helps in solving these problems. Feasibility study plays an important role in system analysis which gives the target for design and development.

**5.1 Feasibility Study**

All systems are feasible when provided with unlimited resource and infinite time. But unfortunately this condition does not prevail in practical world. So it is both necessary and prudent to evaluate the feasibility of the system at the earliest possible time. Months or years of effort, thousands of rupees and untold professional embarrassment can be averted if an ill-conceived system is recognized early in the definition phase. Feasibility & risk analysis are related in many ways. If project risk is great, the feasibility of producing quality software is reduced. In this case there are three primary areas of interest:-

**5.1.1 Performance Analysis**

For the complete functionality of the project work, the project should be deployed in a healthy network. Normally, the OS is windows 7. The main theme of this project is to design a network monitoring system that checks the status of the network from time to time. It gives an overview of the health of the network to the admin. Performance analysis is done to find out whether the proposed system is efficient and accurate when compared to previously existing system. It is essential that the process of performance analysis and definition must be conducted in parallel.

**5.1.2 Technical Analysis**

System is only beneficial only if it can be turned into information systems that will meet the organization’s technical requirement. Simply stated this test of feasibility asks whether the system will work or not when developed & installed, whether there are any major barriers to implementation. Regarding all these issues in technical analysis there are several points to focus on:-

**Changes to bring in the system:** All changes should be in positive direction, there will be increased level of efficiency and better customer service.

**Required skills:** Platforms & tools used in this project are widely used. So the skilled manpower is readily available in the industry.

**Acceptability:** The structure of the system is kept feasible enough so that there should not be any problem from the user’s point of view.

**5.1.3 Economical Analysis**

Economical analysis is performed to evaluate the development cost weighed against the ultimate income or benefits derived from the developed system. This system runs even on the basic computers. However it does require physical Switches and Routers to be connected to monitor them which is a costly affair. However once this is used it would reduce manpower and thereby the cost of maintaining the system.

**Summary**

The main aim of this chapter is to find out whether the system is feasible enough or not. For these reasons different kinds of analysis, such as performance analysis, technical analysis, economical analysis etc is performed.

**Chapter 6**

**SYSTEM DESIGN**

**Overview**

Design is a meaningful engineering representation of something that is to be built. It is the most crucial phase in the developments of a system. Software design is a process through which the requirements are translated into a representation of software. Design is a place where design is fostered in software Engineering. Based on the user requirements and the detailed analysis of the existing system, the new system must be designed. This is the phase of system designing. Design is the perfect way to accurately translate a customer’s requirement in the finished software product. Design creates a representation or model, provides details about software data structure, architecture, interfaces and components that are necessary to implement a system. The logical system design arrived at as a result of systems analysis is converted into physical system design.

**6.1 System development methodology**

System development method is a process through which a product will get completed or a product gets rid from any problem. Software development process is described as a number of phases, procedures and steps that gives the complete software. It follows series of steps which is used for product progress. The development method followed in this project is waterfall model.

**6.1.1 Model phases**

The waterfall model is a [sequential](http://en.wikipedia.org/wiki/Sequence) software development process, in which progress is seen as flowing steadily downwards (like a [waterfall](http://en.wikipedia.org/wiki/Waterfall)) through the phases of Requirement initiation, [Analysis](http://en.wikipedia.org/wiki/Analysis), [Design](http://en.wikipedia.org/wiki/Design), Implementation, [Testing](http://en.wikipedia.org/wiki/Software_testing) and [maintenance](http://en.wikipedia.org/wiki/Software_maintenance).

**Requirement Analysis:** This phase is concerned about collection of requirement of the system. This process involves generating document and requirement review.

**System Design:** Keeping the requirements in mind the system specifications are translated in to a software representation. In this phase the designer emphasizes on:-algorithm**,** data structure**,** software architecture etc.

**Coding:** In this phase programmer starts his coding in order to give a full sketch of product. In other words system specifications are only converted in to machine readable compute code.

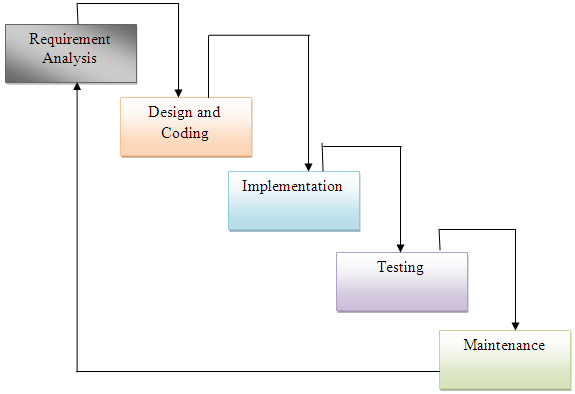
**Implementation:** The implementation phase involves the actual coding or programming of the software. The output of this phase is typically the library, executables, user manuals and additional software documentation.

**Testing:** In this phase all programs (models) are integrated and tested to ensure that the complete system meets the software requirements. The testing is concerned with verification and validation.

**Maintenance:** The maintenance phase is the longest phase in which the software is updated to fulfill the changing customer need, adapt to accommodate change in the external environment, correct errors and oversights previously undetected in the testing phase, enhance the efficiency of the software.

**6.1.2 Reason for choosing Waterfall Model as development method**

* Clear project objectives.
* Stable project requirements.
* Progress of system is measurable.
* Strict sign-off requirements.
* Helps you to be perfect.
* Logic of software development is clearly understood.
* Production of a formal specification
* Better resource allocation.
* Improves quality. The emphasis on requirements and design before writing a single line of code ensures minimal wastage of time and effort and reduces the risk of schedule slippage.
* Less human resources required as once one phase is finished those people can start working on to the next phase.

****

**Figure 6.1.2:** Waterfall model

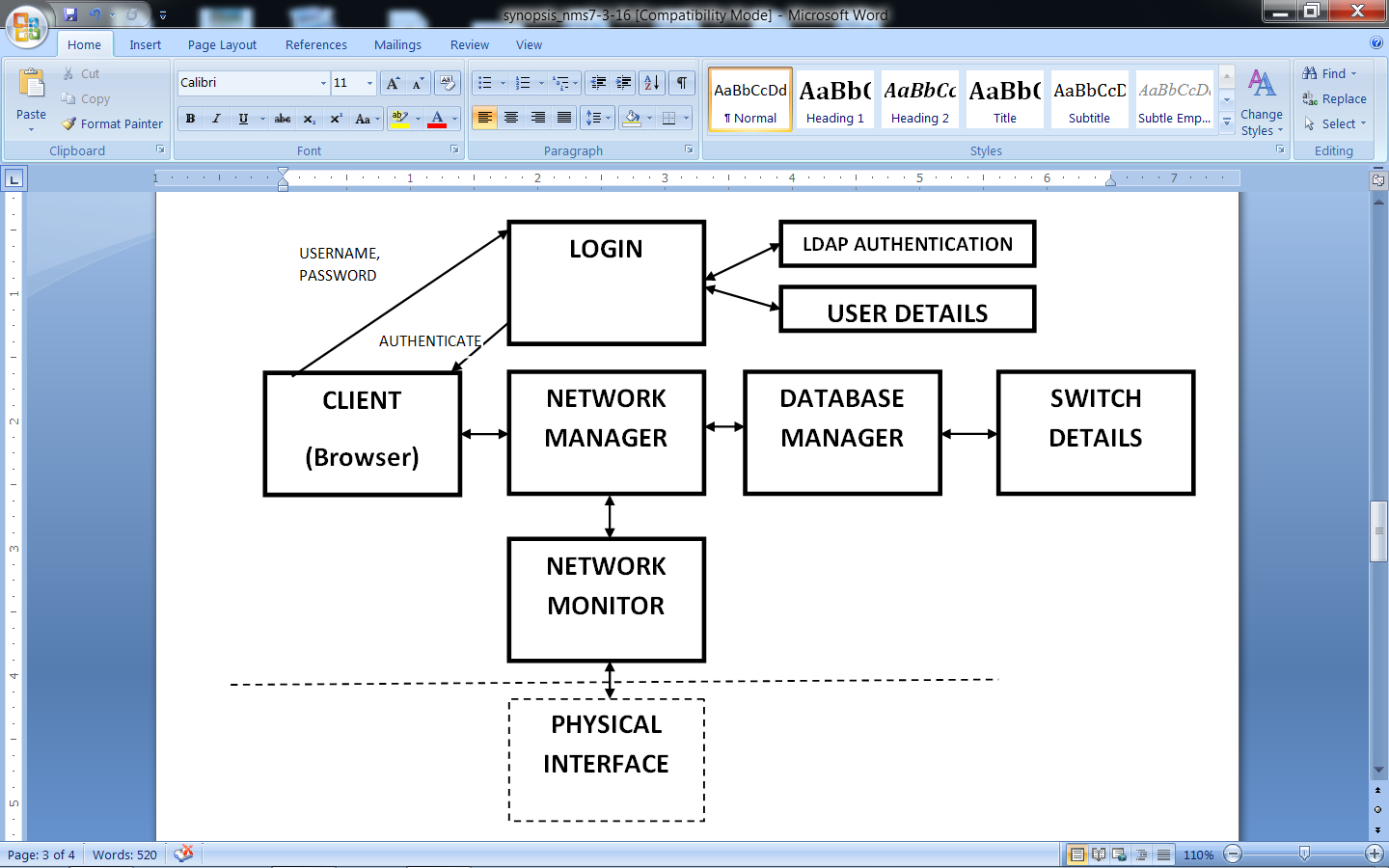
**6.2 Design Using UML**

Designing UML diagram specifies, how the process within the system communicates along with how the objects with in the process collaborate using both static as well as dynamic UML diagrams since in this ever-changing world of Object Oriented application development, it has been getting harder and harder to develop and manage high quality applications in reasonable amount of time. As a result of this challenge and the need for a universal object modeling language every one could use, the Unified Modeling Language (UML) is the Information industries version of blue print. It is a method for describing the systems architecture in detail. Easier to build or maintains system, and to ensure that the system will hold up to the requirement changes.

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl. No** | **Symbol Name** | **Symbol** | **Description** |
| 1 | Class |  | Classes represent a collection of similar entities grouped together. |
| 2 | Association |  | Association represents a static relation between classes. |
| 3 | Aggregation |  | Aggregation is a form of association. It aggregates several classes into a single class. |
| 4 | Composition |  | Composition is a special type of aggregation that denotes a strong ownership between classes. |
| 5 | Actor |  | Actor is the user of the system and other external entity hat react with the system. |
| 6 | Use Case |  | A use case is an interaction between system and the external environment. |
| 7 | Relation (Uses) |  | It is used for additional purpose communication. |
| 8 | Communication |  | It is the communication between use cases. |
| 9 | State |  | It represents the state of process. Each state goes through various flows. |
| 10 | Initial State |  | It represents initial state of object. |
| 11 | Final State |  | It represents final state of object. |
| 12 | Control Flow |  | It represents decision making process for object. |
| 13 | Decision Box |  | It represents the decision making process from a constraint. |
| 14 | Data Process/ State |  | A circle in a DFD represents a state or process which has been triggered due to some other event or action. |
| 15 | External Entity |  | It represents external entity such as Keyboard, sensors, etc which are used in the system. |
| 16 | Transition |  | It represents any communication that occurs between processes. |
| 17 | Object Lifeline |  | Object lifeline represents the vertical dimension that object communicates. |
| 18 | Message |  | It represents messages exchanged. |

**Table 6.2:** Symbols used in UML

**6.2.1 Architectural Design**

****

**Figure 6.2.1: Architecture of Monitoring Application**

As shown in the above figure 6.2.1, the client has to authenticate himself in order to gain access to the system. Once authenticated he will gain access to the dashboard which is under the control of the net manager. The net manager returns its response in the form of web pages. This is done for the purpose of compatibility. The net manager is an interface between net monitor and database manager thereby has to interact with them in order to update the dashboard.

The net monitor interacts with the physical interface such as switches and routers using OIDs who's response is recorded and forwarded to the net manager. The net manager has to collect switch details such as IP in order to poll that switch for connectivity. This IP is returned to net manager who forwards it to the net monitor for polling. This process repeats continuously checking the network from time to time.

**6.2.2 Data Flow Diagram**

A data flow diagram (DFD) is graphic representation of the "flow" of data through an information system. A data flow diagram can also be used for the visualization of data processing (structured design). It is common practice for a designer to draw a context-level DFD first which shows the interaction between the system and outside entities. DFD’s show the flow of data from external entities into the system, how the data moves from one process to another, as well as its logical storage. There are only four symbols:

**1.** Squares representing *external entities*, which are sources and destinations of information entering and leaving the system.

**2.** Rounded rectangles representing *processes*, in other methodologies, may be called 'Activities', 'Actions', 'Procedures', 'Subsystems' etc. which take data as input, do processing to it, and output it.

**3.** Arrows representing the *data flows*, which can either, be electronic data or physical items. It is impossible for data to flow from data store to data store except via a process, and external entities are not allowed to access data stores directly.

**4.** The flat three-sided rectangle is representing data stores should both receive information for storing and provide it for further processing.

**6.2.3 Module Description**

Three modules that are included in the project are given below

* **MODULE 1:** Database Manager
* **MODULE 2:** Net Monitor
* **MODULE 3:** Net Manager
* **MODULE 4:** Login

**6.2.3.1 MODULE 1: DATABASE MANAGER**

Database Manager is a module which acts as an interface between the NetManager and Switch Details. It collects data of switches such as IP Address, Status, Location, Name and Type of Device. These details are collected in a database that is queried by the database manager. The results are communicated to the NetManager which performs several operations.

The database manager is designed using Hibernate Utility.

**6.2.3.2 MODULE 2: NET MONITOR**

Netmonitor is a module which acts as an interface between the physical interface and the NetManager. The NetMonitor consists of the Management Information Base (MIB) which consists of the several Object ID (OID). The OIDs are responsible for querying the Switches and Routers and extracting their details.

**6.2.3.3 MODULE 3: NET MANAGER**

NetManager is the main module whose function is to provide an interface for the client to access the system. The NetManager integrates all modules in the system together to form a single effective NETWORK Monitoring Utility.

It integrates the entire project and provides a single interface for performing related activities and queries the Network. When the system is invoked the client is asked for authentication, once the client is authenticated he/she will be directed to the Monitoring options by the NetManager.

The NetManager is the first Program to be run and it continuously polls the Nodes in the network in order to check for connectivity ad returns the Status.

**6.2.3.4 MODULE 4: LOGIN**

Security is an important aspect in an organization. Hence we have developed a login module which will authenticate the user only if he holds valid credentials. The login module is connected to a database that has the username and password of the employees.

Once the session ends the user can logout from his account which redirects to the Login page.

**6.2.4 Class Diagram**

UML class diagram shows the static structure of the model. The class diagram is a collection of static modeling elements, such as classes and their relationships’, connected as a graph to each other and to their contents.

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects and or interactions in the application and the objects to be programmed.

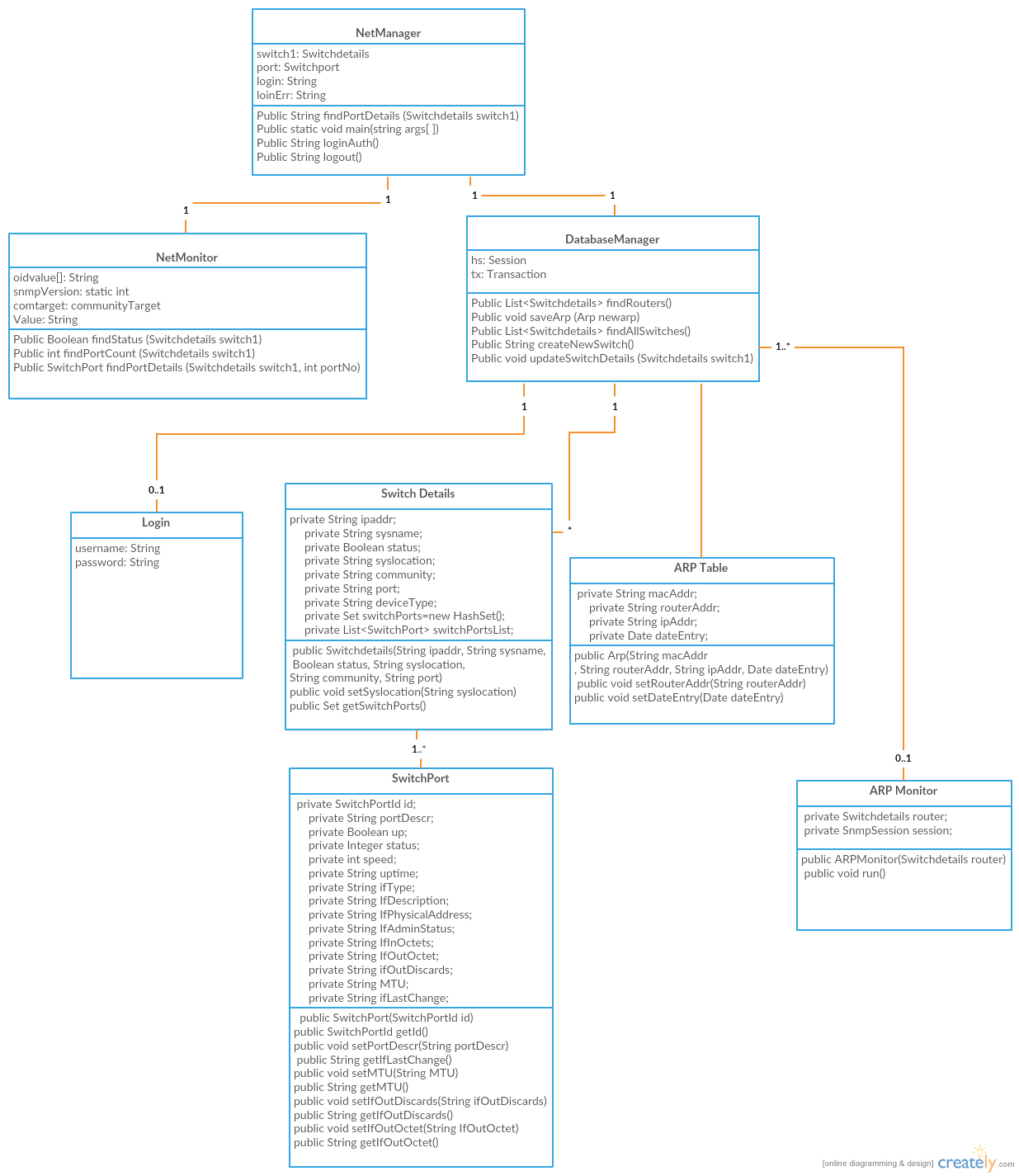
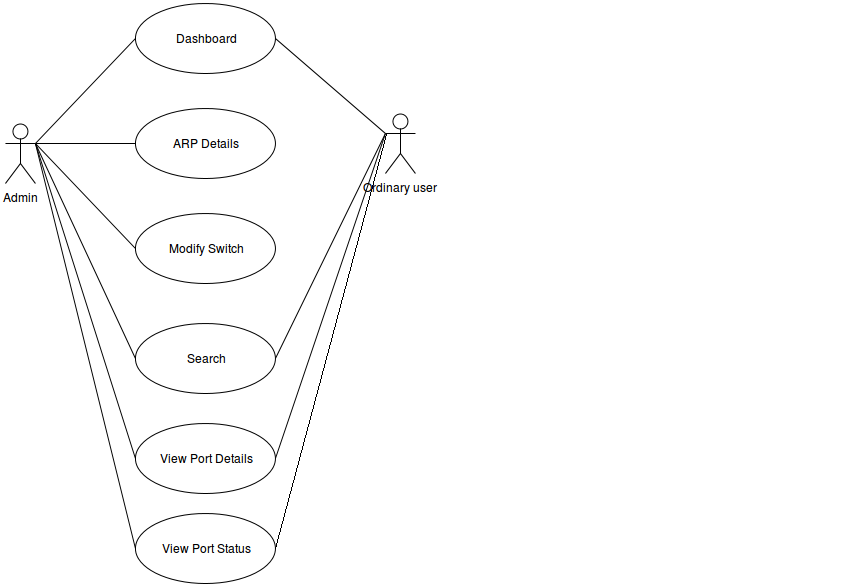
****

Figure 6.2.4 shows a Network composed of several Nodes and one or more Sinks.

**6.2.5 Use Case Diagram**

A use case defines a goal-oriented set of interactions between external entities and the system under consideration. The external entities which interact with the system are its actors. A set of use cases describe the complete functionality of the system at a particular level of detail and it can be graphically denoted by the use case diagram.



**Figure 6.2.5:** Use Case diagram

* + - 1. **Use Case Description**

**6.2.5.1.1 Modify Switch Details**

|  |  |
| --- | --- |
| Use case: | Modify switch details. |
| Summary: | Switch details are modified (add/remove) by the administrator. |
| Actors: | Administrator. |
| Pre condition: | Authenticate as network administrator. |
| Typical flow of events: | 1) Capture details of IP address, location, symbolic name, SNMP community name.  2) Capture port details.  3) Add or remove details using forms. |

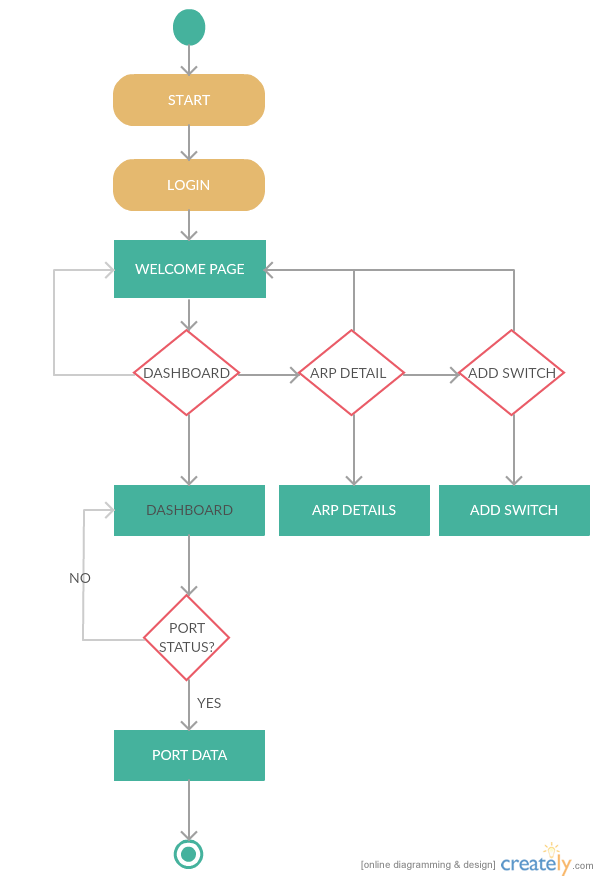
|  |  |
| --- | --- |
| Use case: | Modify switch details. |
| Summary: | Switch details are modified (add/remove) by the administrator. |
| Actors: | Administrator. |
| Pre condition: | Authenticate as network administrator. |
| Typical flow of events: | 1) Capture details of IP address, location, symbolic name, SNMP community name.  2) Capture port details.  3) Add or remove details using forms. |

**6.2.5.1.2 Display Dashboard**

|  |  |
| --- | --- |
| Use case: | Displaying dashboard. |
| Summary: | It is the reflection of connectivity of the network in real time. |
| Actors: | End user, Administrator (All users). |
| Pre condition: | Should be a valid user. |
| Typical flow of events: | 1) Upon login, it displays the location tabs.  2) Upon clicking these tabs, it redirects to the switches in that location and displays the switch’s status.(Up/Down)  3) Upon clicking on switches, it displays the port details. |

**6.2.6 Activity Diagram**

An activity diagram shows the sequence of steps that make up a complex process. An activity is shown as a round box containing the name of the operation. An outgoing solid arrow attached to the end of the activity symbol indicates a transition triggered by the completion.



**Figure 6.2.6:** Activity Diagram

Figure 6.2.6 shows the stepwise activities that take place when a user wants to test a protocol. Initially the user is expected to login with his/her credentials which would redirect them to the welcome page. The welcome page gives the user several decision making steps where he could choose to go to the Dashboard or ARP Details or Add Switch Module.

The Dashboard module would show the entire network in the form of nodes. Each node when selected would redirect to another page indicating the details of the node. In this case when a switch is selected it would redirect to the switch details (a page showing the various ports in the switch with the name and location of the switch). On selecting the ports of the switch/ router that are up we can further redirect to another page that shows the details of that port such as speed, uptime, administrator status, bandwidth utilization details.

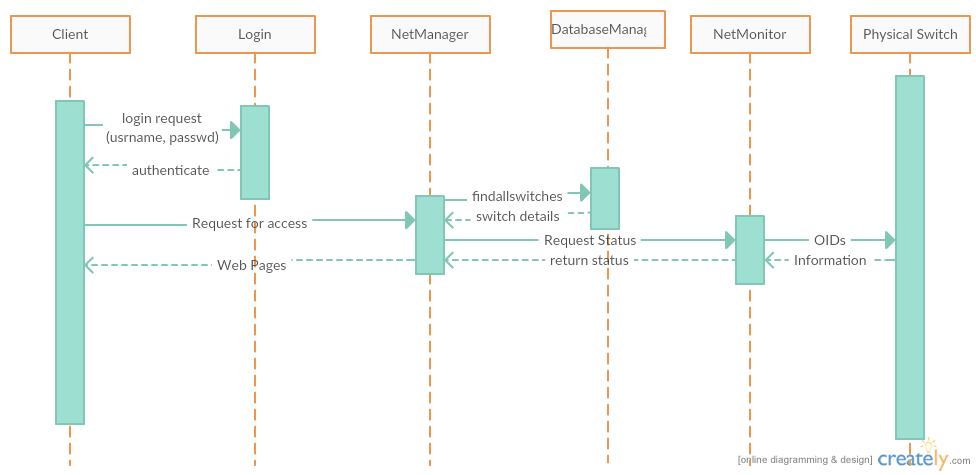
The ARP module displays the details of all the systems connected to the routers along with their MAC addresses. The Add Switch Module would add a new switch to the database which would be polled from time to time to check for connectivity. Hence whenever a new switch/router is added to the database it should be registered with the administrator.

**6.2.7 Sequence Diagram**

Sequence diagram are an easy and intuitive way of describing the behavior of a system by viewing the interaction between the system and the environment. A sequence diagram shows an interaction arranged in a time sequence. A sequence diagram has two dimensions: vertical dimension represents time, the horizontal dimension represents the objects existence during the interaction.

**Basic elements:**

* Vertical rectangle: represent the object is active (method is being performed).
* Vertical dashed line: represent the life of the object.
* ”X”: represent the life end of an object. (Being destroyed from memory)
* Horizontal line with arrows: messages from one object to another.



**Figure 6.2.7:** Sequence Diagram for communication between network modules

As shown in the above figure 6.2.7, the client has to authenticate himself in order to gain access to the system. Once authenticated he will gain access to the dashboard which is under the control of the net manager. The net manager returns its response in the form of web pages. This is done for the purpose of compatibility. The net manager is an interface between net monitor and database manager thereby has to interact with them in order to update the dashboard.

The net monitor interacts with the physical interface such as switches and routers using OIDs who's response is recorded and forwarded to the net manager. The net manager has to collect switch details such as IP in order to poll that switch for connectivity. This IP is returned to net manager who forwards it to the net monitor for polling. This process repeats continuously checking the network from time to time.

**Summary**

This chapter mainly concentrates on few fundamental design concepts such as system development methodology, system architecture, class diagram, flowchart, sequence diagram, use-case diagram, activity diagram, data flow diagram etc.

**Chapter 7**

**IMPLEMENTATION**

The implementation phase of the project is where the detailed design is actually transformed into working code. Aim of the phase is to translate the design into a best possible solution in a suitable programming language. This chapter covers the implementation aspects of the project, giving details of the programming language and development environment used. It also gives an overview of the core modules of the project with their step by step flow.

The implementation stage requires the following tasks.

* Careful planning.
* Investigation of system and constraints.
* Design of methods to achieve the changeover.
* Evaluation of the changeover method.
* Correct decisions regarding selection of the platform
* Appropriate selection of the language for application development

**7.1 Introduction to NetBeans**

NetBeans is a [software development](https://en.wikipedia.org/wiki/Software_development) platform written in [Java](https://en.wikipedia.org/wiki/Java_(programming_language)). The NetBeans [Platform](https://en.wikipedia.org/wiki/Platform_(computing)) allows applications to be developed from a set of modular [software components](https://en.wikipedia.org/wiki/Software_component) called modules. Applications based on the NetBeans Platform, including the NetBeans [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE), can be extended by [third party developers](https://en.wikipedia.org/wiki/Third_party_developer).

The NetBeans IDE is primarily intended for development in Java, but also supports other languages, in particular [PHP](https://en.wikipedia.org/wiki/PHP), [C](https://en.wikipedia.org/wiki/C_(programming_language))/[C++](https://en.wikipedia.org/wiki/C%2B%2B)and [HTML5](https://en.wikipedia.org/wiki/HTML5).

NetBeans is [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) and runs on [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows), [Mac OS X](https://en.wikipedia.org/wiki/Mac_OS_X), [Linux](https://en.wikipedia.org/wiki/Linux), [Solaris](https://en.wikipedia.org/wiki/Solaris_(operating_system)) and other platforms supporting a compatible [JVM](https://en.wikipedia.org/wiki/Java_Virtual_Machine).

**7.1.1 Modularity**

All the functions of the IDE are provided by modules. Each module provides a well-defined function, such as support for the [Java language](https://en.wikipedia.org/wiki/Java_(programming_language)), editing or support for the [CVS](https://en.wikipedia.org/wiki/Concurrent_Versions_System) versioning system, and SVN. NetBeans contains all the modules needed for Java development in a single download, allowing the user to start working immediately. Modules also allow NetBeans to be extended. New features, such as support for other programming languages, can be added by installing additional modules. For instance, [Sun Studio](https://en.wikipedia.org/wiki/Sun_Studio_Compiler_Suite), Sun Java Studio Enterprise, and [Sun Java Studio Creator](https://en.wikipedia.org/wiki/Sun_Java_Studio_Creator) from [Sun Microsystems](https://en.wikipedia.org/wiki/Sun_Microsystems) are all based on the NetBeans IDE.

**7.1.2 NetBeans IDE Bundle for Web and Java EE**

The NetBeans IDE Bundle for Web & Java EE provides complete tools for all the latest Java EE 6 standards, including the new Java EE 6 Web Profile, Enterprise Java Beans (EJBs), servlets, Java Persistence API, web services, and annotations. NetBeans also supports the JSF 2.0 (Facelets), Java Server Pages (JSP), Hibernate, spring, and Struts frameworks, and the Java EE 5 and J2EE 1.4 platforms. It includes [GlassFish](https://en.wikipedia.org/wiki/GlassFish) and [Apache Tomcat](https://en.wikipedia.org/wiki/Apache_Tomcat). Some of its features with javaEE include

* Improved support for CDI, REST services and Java Persistence
* New support for Bean Validation
* Support for JSF component libraries, including bundled Prime Faces library
* Improved editing for Expression Language in JSF, including code completion, refactoring and hints

**7.2 Modules in Implementation**

For the purpose of executing the code properly, the first step is to make sure that all the switches and nodes required to be monitored have well established physical connection. This so called physical connection can be checked by looking at the lights which are given alongside each Ethernet port in the switch. The code will run if and only if the system which contains the code is connected to at least one switch and if there is any break in the cable, it throws an error. Before executing the code, there are certain libraries which need to be installed, servers and database details which needs entered into the database, and so on.

**7.2.1 Network Monitoring System Implementation**

The main functions required for implementing Network Monitoring System are:

* Functions for Network Manager
* Functions for Network Monitor
* Functions for Database Manager
* Functions for ARP Details
* Functions for Switch Details

**1. Functions Used for Network Manager**

* public String findPortDetails (Switchdetails switch1)-

The function is used to find all the port details that are to be displayed. It acts as an interface to the findPortDetails function in Netmonitor.

* public String loginAuth()-

The function is used to authenticate the client based on string comparision. If the string entered is similar to the string in the database it would login.

* Public String logout()-

As the name suggests the function is responsible for logging out of the system. At the end of every session the client is recommended to logout.

* Public static void main(string args[ ])-

The network manager acts as an interface between the network monitor and the database manager. This function would receive data from switch details database and poll them continuously to check for connectivity.

**2. Functions Used for Network Monitor**

* Public Boolean findStatus (Switchdetails switch1)-

The function is used to find the status of the switch and return whether it is up or down.

* Public int findPortCount (Switchdetails switch1)-

The function is used to return the port count on the switch/router.

* Public SwitchPort findPortDetails (Switchdetails switch1, int portNo)-

The function is used to return the details of the switch by passing the OIDs to that switch/router.

**3. Functions Used for Database Manager**

* Public List<Switchdetails> findRouters()-

This enumeration creates a list of all the routers in the network. It displays the ARP details of only the routers.

* Public void saveArp (Arp newarp)-

The arp details have to be stored in the database for future reference on history of device.

* Public List<Switchdetails> findAllSwitches()-

This enumeration would return a list of all the switches/routers in the network. This function helps to poll those nodes from time to time.

* Public String createNewSwitch()-

This function helps to create a new switch so that it can be added to the database and polled later for connectivity.

* Public void updateSwitchDetails (Switchdetails switch1)-

The switch details should be updated whenever there is a change in the network.

**4. Functions Used for ARP Details**

* Public Boolean readNextEntry(String communityName)-

This function is used to read the next entry in the database and exhibit its ARP details.

* Public String findMACAddress (String IPAddr, String communityName)-

This function is used to find the MAC address of the device that connected to the ports of the router.

**5. Functions Used for Switch Details**

* Functions related to get and set data (i. e) add, delete, update or retrieve data from the database

**Summary**

The chapter discusses the implementation details of the different modules of the system and gives the step by step flow of each of them. Along with these, this chapter also highlights some of the important features of the platform and language used for implementation purpose.

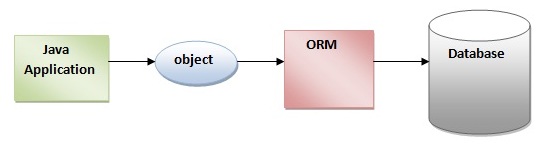
**7.3 Backbone of Design and Implementation**

Two main components that are used in designing and implementation are Hibernate and PrimeFaces.

**7.3.1 HIBERNATE**

Hibernate framework simplifies the development of java application to interact with the database. Hibernate is an open source, lightweight, [ORM (Object Relational Mapping)](http://en.wikipedia.org/wiki/Object-relational_mapping) tool.

An ORM tool simplifies the data creation, data manipulation and data access. It is a programming technique that maps the object to the data stored in the database.



**Advantages of Hibernate Framework:**

There are many advantages of Hibernate Framework. They are as follows:

**1) Open source and Lightweight:** Hibernate framework is opensource under the LGPL license and lightweight.

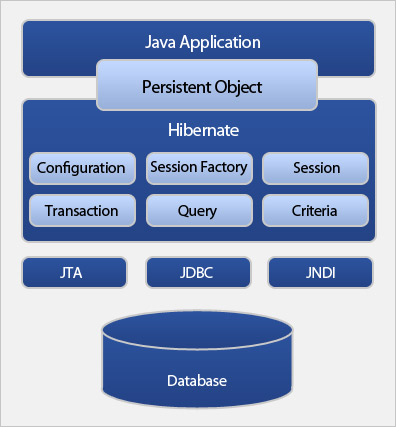
**2) Fast performance:** The performance of hibernate framework is fast because cache is internally used in hibernate framework. There are two types of cache in hibernate framework first level cache and second level cache. First level cache is enabled by default.

**3) Database Independent query:** HQL (Hibernate Query Language) is the object-oriented version of SQL. It generates the database independent queries. So you don't need to write database specific queries. Before Hibernate, If database is changed for the project, we need to change the SQL query as well that leads to the maintenance problem.

**4) Automatic table creation:** Hibernate framework provides the facility to create the tables of the database automatically. So there is no need to create tables in the database manually.

**5) Simplifies complex join:** To fetch data form multiple tables is easy in hibernate framework.

**6) Provides query statistics and database status:** Hibernate supports Query cache and provide statistics about query and database status.

****

**7.3.2 PRIMEFACES**

PrimeFaces is an open source JSF component suite with various extensions.

• Rich set of components (HtmlEditor, Dialog, AutoComplete, Charts and many more).

• Built-in Ajax based on standard JSF Ajax APIs.

• Lightweight, one jar, zero-configuration and no required dependencies.

• Push support via Atmosphere Framework.

• Mobile UI kit to create mobile web applications.

• Skinning Framework with 35+ built-in themes and support for visual theme designer tool. • Extensive documentation.

• Large, vibrant and active user community.

• Developed with "passion" from application developers to application developers.

**7.3.3 Hibernate Reverse Engineering**

Hibernate provides "click-and-generate" reverse engineering and code generation facilities. This allows you to generate a range of artifacts based on database or an existing Hibernate configuration, be that mapping files or annotated classes. Some of these are POJO Java source files, Hibernate .hbm.xml files, hibernate.cfg.xml generation and schema documentation.

**Chapter 8**

**TESTING**

System testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. Although each test has a different purpose, all work to verify that all the system elements have been properly integrated and perform allocated functions. The testing process is actually carried out to make sure that the product exactly does the same thing what is supposed to do. In the testing stage following goals are tried to achieve:-

* To affirm the quality of the project.
* To find and eliminate any residual errors from previous stages.
* To validate the software as a solution to the original problem.
* To provide operational reliability of the system.

**8.1 Testing Methodologies**

There are many different types of testing methods or techniques used as part of the software testing methodology. Some of the important testing methodologies are:

* + 1. **White box testing**

White box testing (clear box testing, glass box testing, and transparent box testing or structural testing) uses an internal perspective of the system to design test cases based on internal structure. It requires programming skills to identify all paths through the software. The tester chooses test case inputs to exercise paths through the code and determines the appropriate outputs. While white box testing is applicable at the unit, integration and system levels of the software testing process, it is typically applied to the unit. While it normally tests paths within a unit, it can also test paths between units during integration, and between subsystems during a system level test.

Though this method of test design can uncover an overwhelming number of test cases, it might not detect unimplemented parts of the specification or missing requirements, but one can be sure that all paths through the test object are executed. Using white box testing we can derive test cases that:

* Guarantee that all independent paths within a module have been exercised at least once.
* Exercise all logical decisions on their true and false sides.
* Execute all loops at their boundaries and within their operational bounds.
* Execute internal data structure to assure their validity
  + 1. **Black box testing**

Black box testing focuses on the functional requirements of the software. It is also known as functional testing. It is a [software](http://www.webopedia.com/TERM/B/software.html) testing technique whereby the internal workings of the item being tested are not known by the tester. For example, in a black box test on software design the tester only knows the inputs and what the expected outcomes should be and not how the program arrives at those outputs.

The tester does not ever examine the programming [code](http://www.webopedia.com/TERM/B/code.html) and does not need any further knowledge of the program other than its specifications. It enables us to derive sets of input conditions that will fully exercise all functional requirements for a program. Black box testing is an alternative to white box technique. Rather it is a complementary approach that is likely to uncover a different class of errors in the following categories:-

* Incorrect or missing function.
* Interface errors.
* Performance errors.
* Initialization and termination errors.
* Errors in objects.

**Advantages**

* The test is unbiased as the designer and the tester are independent of each other.
* The tester does not need knowledge of any specific programming languages.
* The test is done from the point of view of the user, not the designer.
* Test cases can be designed as soon as the specifications are complete.

**8.2 Unit Testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**8.2.1 Functions for NetManager**

|  |  |  |
| --- | --- | --- |
| **Function** | **Tests done** | **Remarks** |
| Public String findPortDetails (Switchdetails switch1) | Tested to check the port status on every switch/router | Success |
| Public static void main(string args[ ]) | This function is tested to check if the system is polling and returning the IP’s and their status. | Success |
| Public String loginAuth() | Tested to check if it compares the username password strings and authenticates the user. | Success |
| Public String logout() | Tested to check if the system logs out of the session. | Success |

**Table 8.2.1:** Functions for NetManager

**8.2.2** **Functions for NetMonitor**

|  |  |  |
| --- | --- | --- |
| **Function** | **Tests done** | **Remarks** |
| Public Boolean findStatus (Switchdetails switch1) | Tested to check if the OID returns a value about the status of the switch/Router | Success |
| Public int findPortCount (Switchdetails switch1) | Tested to check the number of ports in the switch/Router. | Success |
| Public SwitchPort findPortDetails (Switchdetails switch1, int portNo) | Tested to check if the port is returning the respective port details based on the OID. | Success |

**Table 8.2.2:** Functions for NetMonitor

**8.2.3 Functions for Database Manager**

|  |  |  |
| --- | --- | --- |
| **Function** | **Tests done** | **Remarks** |
| Public List<Switchdetails> findRouters() | Tested to check for routers in the network | Success |
| Public void saveArp (Arp newarp) | Tested to check if it saves arp details for future reference by admin | Success |
| Public List<Switchdetails> findAllSwitches() | Tested to check if it returns all the Switches/Routers in the network | Success |
| Public String createNewSwitch() | Tested to check if creates a new switch | Success |
| Public void updateSwitchDetails (Switchdetails switch1) | Tested to check if it updates switch details whenever there is a change in network | Success |

**Table 8.2.3:** Functions for Database Manager

**8.2.4 Functions for IPMacTable**

|  |  |  |
| --- | --- | --- |
| **Function** | **Tests done** | **Remarks** |
| Public Boolean readNextEntry(String communityName) | This function is tested to check whether the OIDs return the ARP Table of the ports that they are connected | Success |
| Public String findMACAddress (String IPAddr, String communityName) | Tested to check for finding MAC Address | Success |

**Table 8.2.4:** Functions for IPMacTable

**8.3 Integration Testing**

Upon completion of unit testing, integration testing begins. Individual modules are combined and tested as a group. Integration testing is black box testing. The purpose of integration testing is to ensure distinct components of the application still work in accordance to user requirements. Integration testing is considered complete, when actual results and expected results are either in line or differences are explainable based on client input. It concentrates on data transfer between modules. Integration testing is a logical extension of unit testing. Two units that have already been tested are combined into a component and the interface between them is tested. Integration testing identifies problems that occur when units are combined .The errors that arise can be attributed to those occurring due to the combination of modules, resulting from errors across interface.

The Integration Testing Table 8.3 shows the functions that are combined into different classes and the module as a whole tested for its functionality. Finally all the modules are integrated and tested. This is important to check for error-free interaction between various classes and its modules. The integration testing table shows the important modules integrated.

|  |  |  |
| --- | --- | --- |
| **Modules** | **Functions integrated** | **Tests done** |
| NetManager | * Functions used to show Port Information * Functions used to find Port Details * Functions for creating a list of switches and returning their IP and Status | Tested the function of NetManager. |
| NetMonitor | * Functions to send OIDs to switches/routers and collect the response data * Functions to constantly poll the network * Functions that count the number of ports in a switch/ router * Functions that find the details of the ports | Tested the function of NetMonitor. |
| Database Manager | * Functions that create a new switch * Functions that are used to find all switches from switch details database * Functions to update switch details | Tested the function of Database manager |
| Switch Details | * Functions that store all the values into the database | Tested the function of Switch Details |

**Table 8.3:** Integration Testing

**8.4 System Testing**

System testing checks complete end-end scenarios, as a user would exercise the system. The system has to be tested for correctness of the functionality by setting it up in a controlled environment. System testing includes testing of functional and non functional requirements. It helps to verify and validate the system. All components of system should have been successfully unit tested and then checked for any errors after integration.

|  |  |
| --- | --- |
| **Functionality to be tested** | **Input** |
| Working of Front-End | User interaction with through Web interface |
| Working of the main Netmonitor, NetManager and Database manager modules | User defines the network simulation parameters such number of nodes, topology size, etc |

**Table 8.4:** System Testing

**8.5 Quality Assurance**

Quality assurance consists of the auditing and reporting functions of management. The goal of quality assurance is to provide management with the data necessary to be informed about product quality, thereby gaining insight and confident that the product quality is meeting its goals. This is an “umbrella activity” that is applied throughout the engineering process***.*** Software quality assurance encompasses:-

* Analysis, design, coding and testing methods and tools
* Formal technical reviews that are applied during each software engineering
* Multi tiered testing strategy
* Control of software documentation and the change made to it.
* A procedure to ensure compliance with software development standards.
* Measurement and reporting mechanisms.

**8.5.1 Quality Factors**

An important objective of quality assurance is to track the software quality and assess the impact of methodological and procedural changes on improved software quality. The factors that affect the quality can be categorized into two broad groups:

* Factors that can be directly measured.
* Factors that can be indirectly measured

These factors focus on three important aspects of a software product

* Its operational characteristics
* Its ability to undergo changes
* Its adaptability to a new environment.
* Effectiveness or efficiency in performing its mission
* Duration of its use by its customer.

**Summary**

The chapter discusses the tests that are done on the system to check its functionality. Testing is carried out at three different levels from the module level to the system level checking for errors at each stage. The remarks have also been documented.

**Chapter 9**

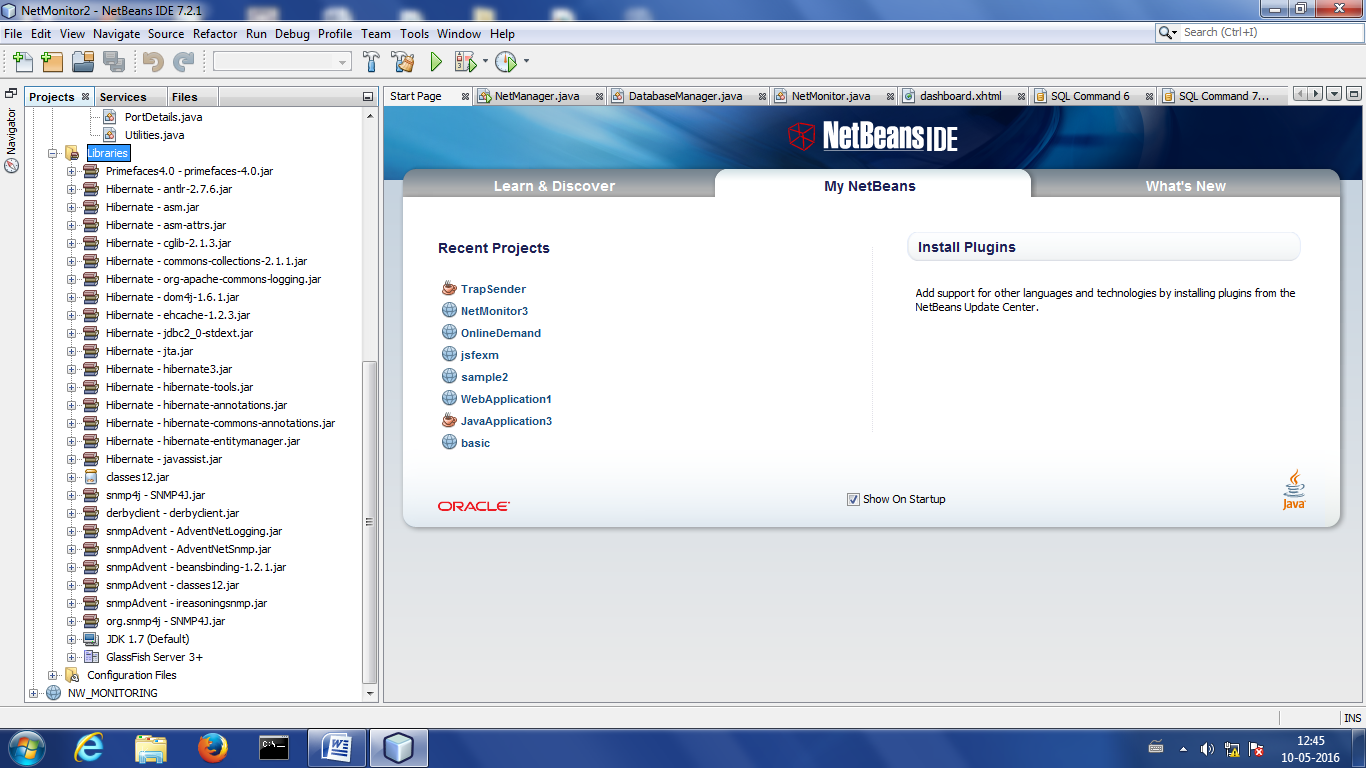
**RESULTS AND PERFORMANCE EVALUATION**

The following snapshots and graphs define the results or outputs that we will get after step by step execution of each proposed protocol for different values of time and speed.

**9.1 Snapshots**

**9.1.1 Libraries:**

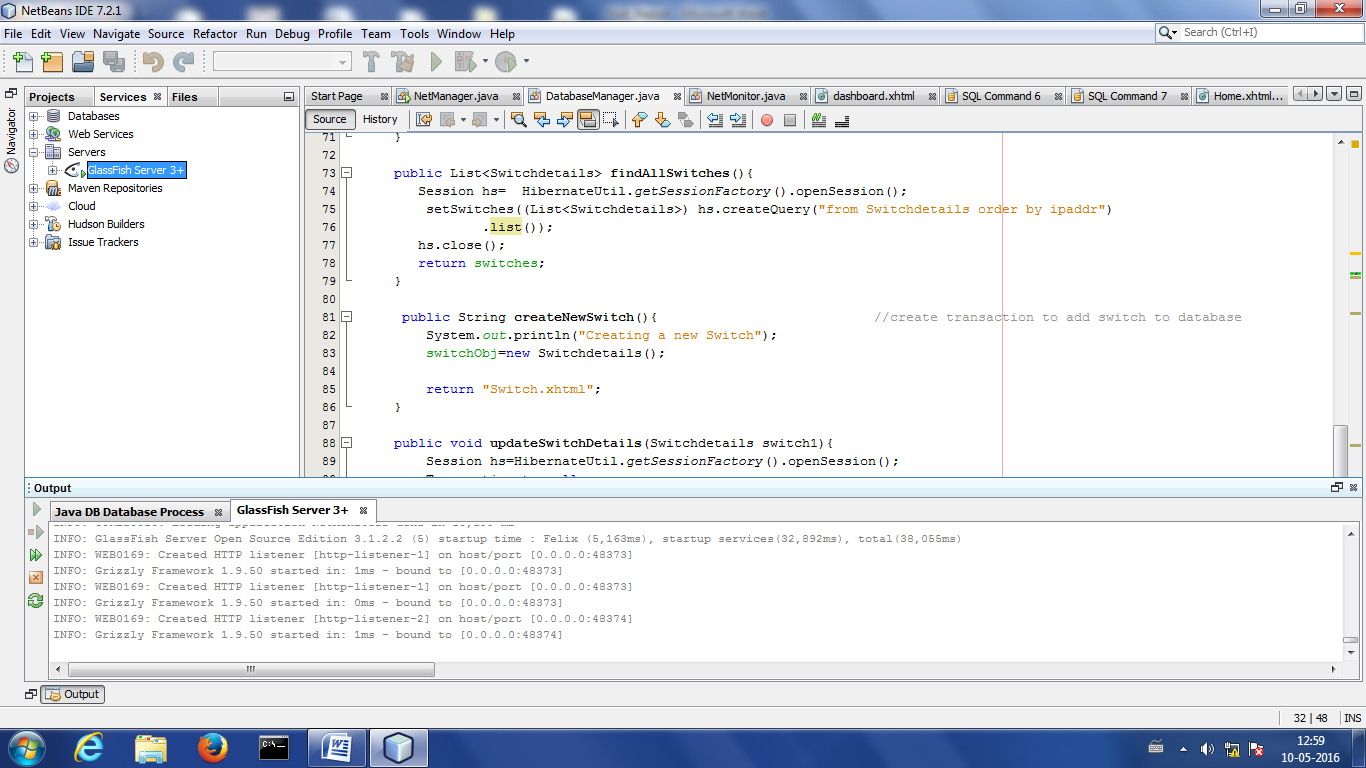
We have to make sure that the following libraries to be used are installed. These libraries are a set of “jar” files that need to be present in the NetBeans platform in order to run the code. Some of the libraries used are Hibernate, Primefaces, SNMP 4J, SNMP Advent, etc. as shown below:



**9.1.2: Servers:**

For the purpose of executing the code, we make use of an open source application server known as Oracle Glassfish Server. It is the [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation) of Java EE and also supports [Enterprise JavaBeans](https://en.wikipedia.org/wiki/Enterprise_JavaBeans), [JPA](https://en.wikipedia.org/wiki/Java_Persistence_API), [JavaServer Faces](https://en.wikipedia.org/wiki/JavaServer_Faces), [JMS](https://en.wikipedia.org/wiki/Java_Message_Service), [RMI](https://en.wikipedia.org/wiki/Remote_Method_Invocation), [JavaServer Pages](https://en.wikipedia.org/wiki/JavaServer_Pages), [servlets](https://en.wikipedia.org/wiki/Java_Servlet), etc. and optional components can also be installed for additional services.

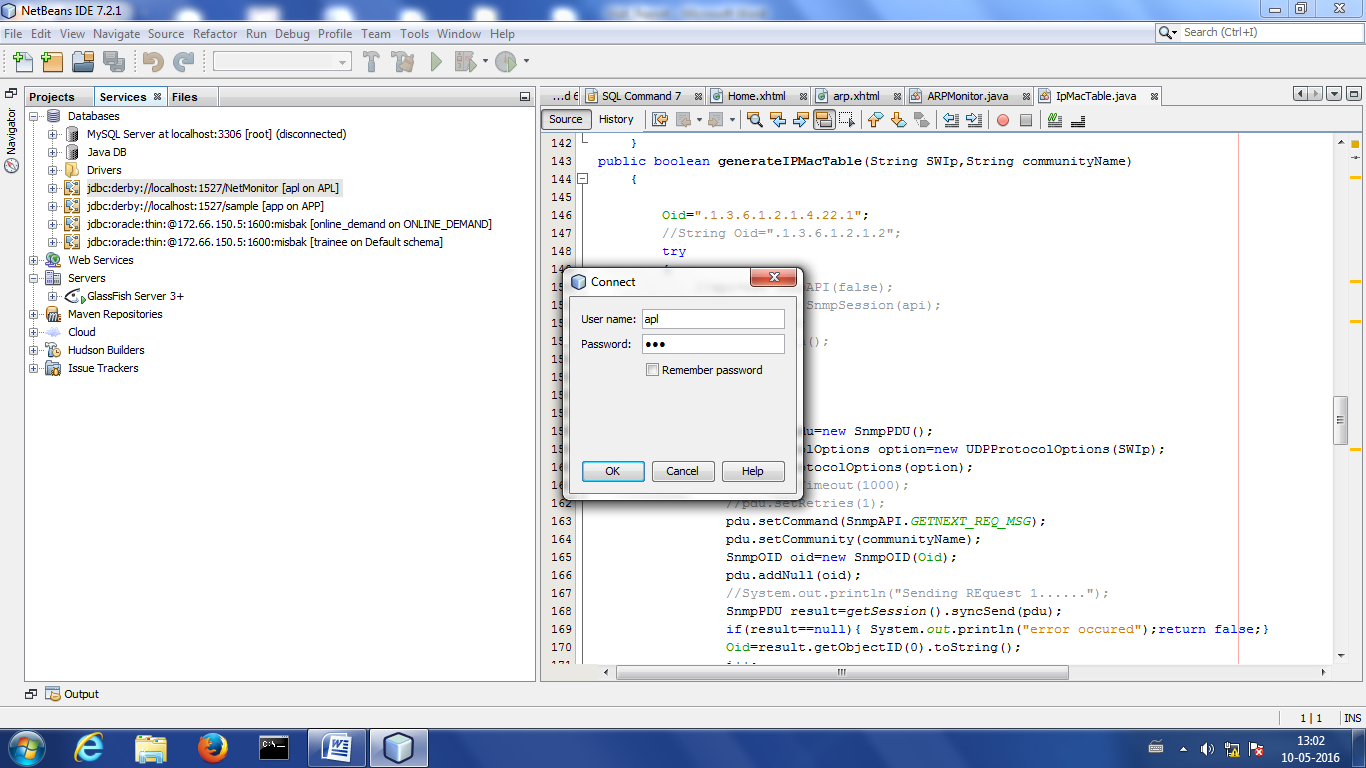
We have to make sure that this server is up and running in the background at all times.

**9.1.3: Database Details:**

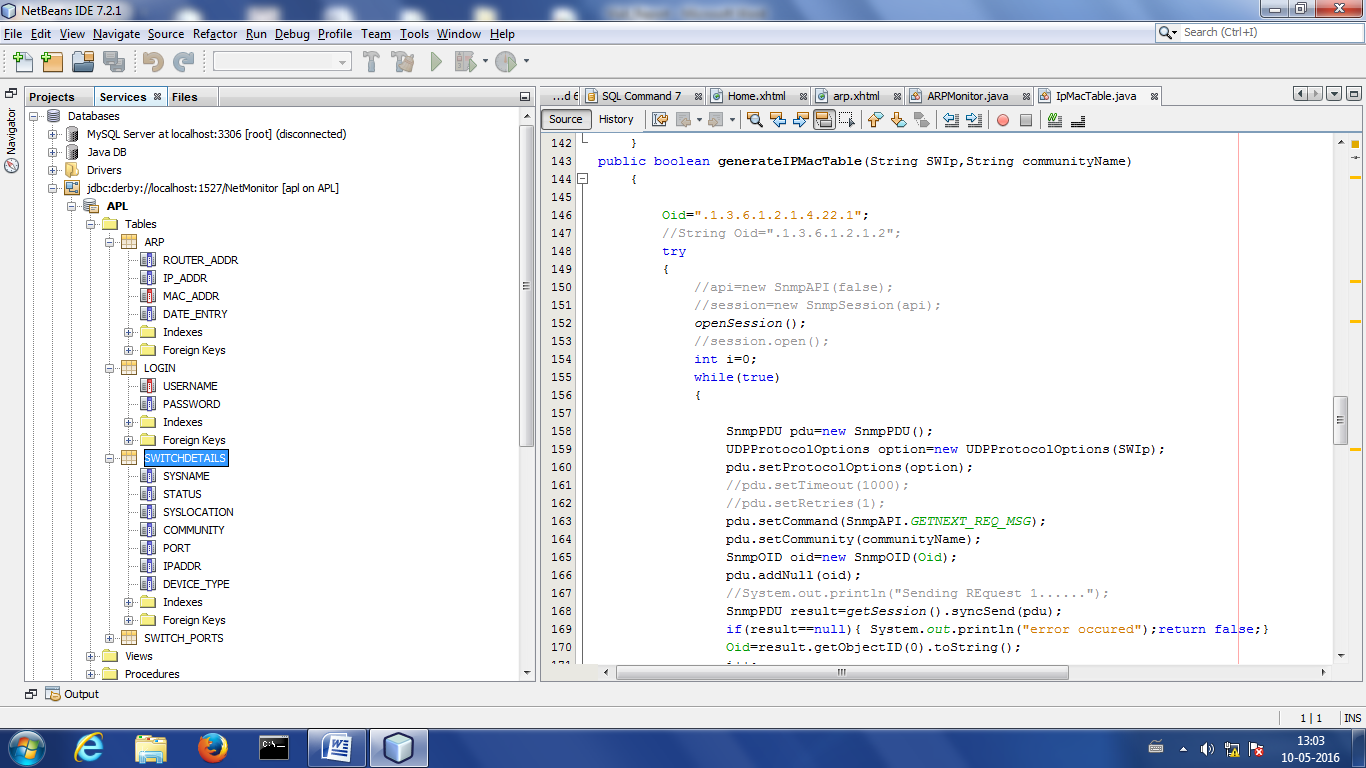
Apache Derby is a [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS) that can be embedded in [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) programs and used for [online transaction processing](https://en.wikipedia.org/wiki/Online_transaction_processing) and is open source.

Derby has two JDBC drivers. The Embedded Driver called "org.apache.derby.jdbc.EmbeddedDriver" and the Client-side driver called "org.apache.derby.jdbc.ClientDriver". The first Driver embeds the Derby Database Driver into Java Code. The second Driver connects the Derby Database via network.

It is to be ensured that the Apache Derby system is connected to the database so that the java programs can access and manipulate the database with ease. This is done by right-clicking on the local host derby tab and clicking on “connect”. The connect dialog box appears in which you have to enter the user name and password as shown in the screenshot below:



Once connected, you need to create the tables which are necessary for implementation (the schema of the tables is as shown in the below screenshot). We also need to enter the IP addresses and other details of the switches to be monitored into the “SWITCHDETAILS” table. The system will monitor the network if and only if the switches to be monitored are entered into this table since a polling of only these switches is done by the code and the other switches will not be considered.

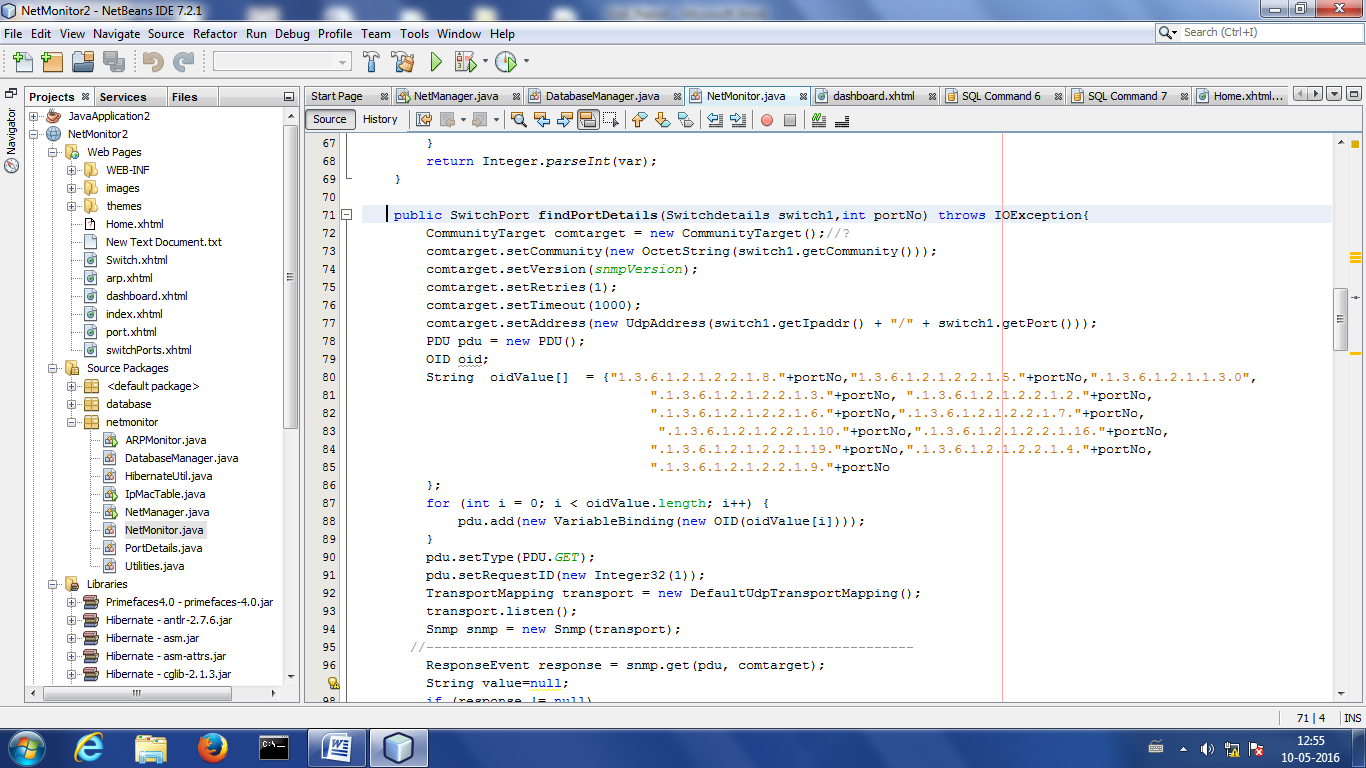


**9.1.4 Modules:**

The system is modularised into various categories and the screenshots of some of these modules are given below. Each module does it own job and has its own set of functionalities. These modules need to be running in the background for the process of monitoring the switches and other nodes in the network. Some of these modules are called from inside other modules.

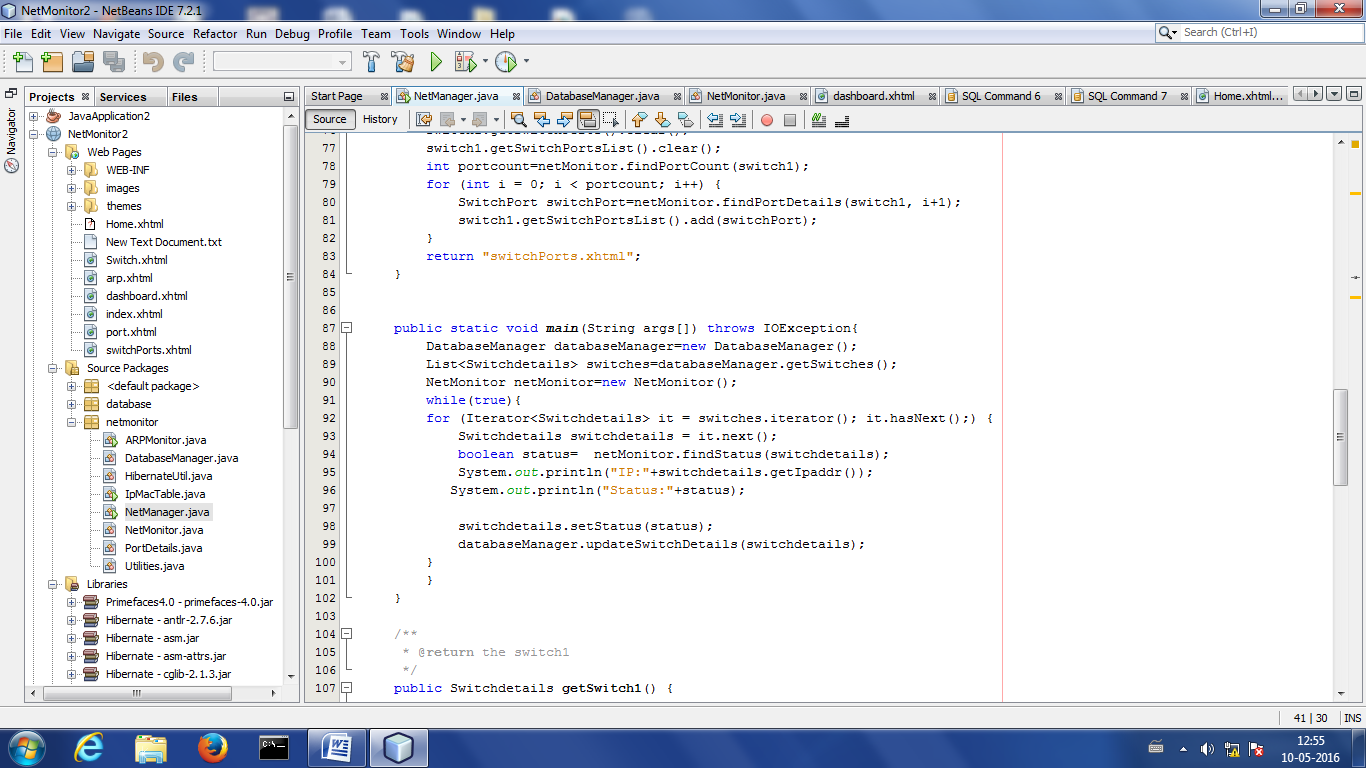
**Network Monitor**

The name of this module is NetMonitor and it is called by the netManager module. This module is used to set the various parameters and is also used to retrieve the port details of the switch. This is done by passing various object identifiers (or OIDs) to the switch and the information corresponding to that OID is returned by the switch back to this module.



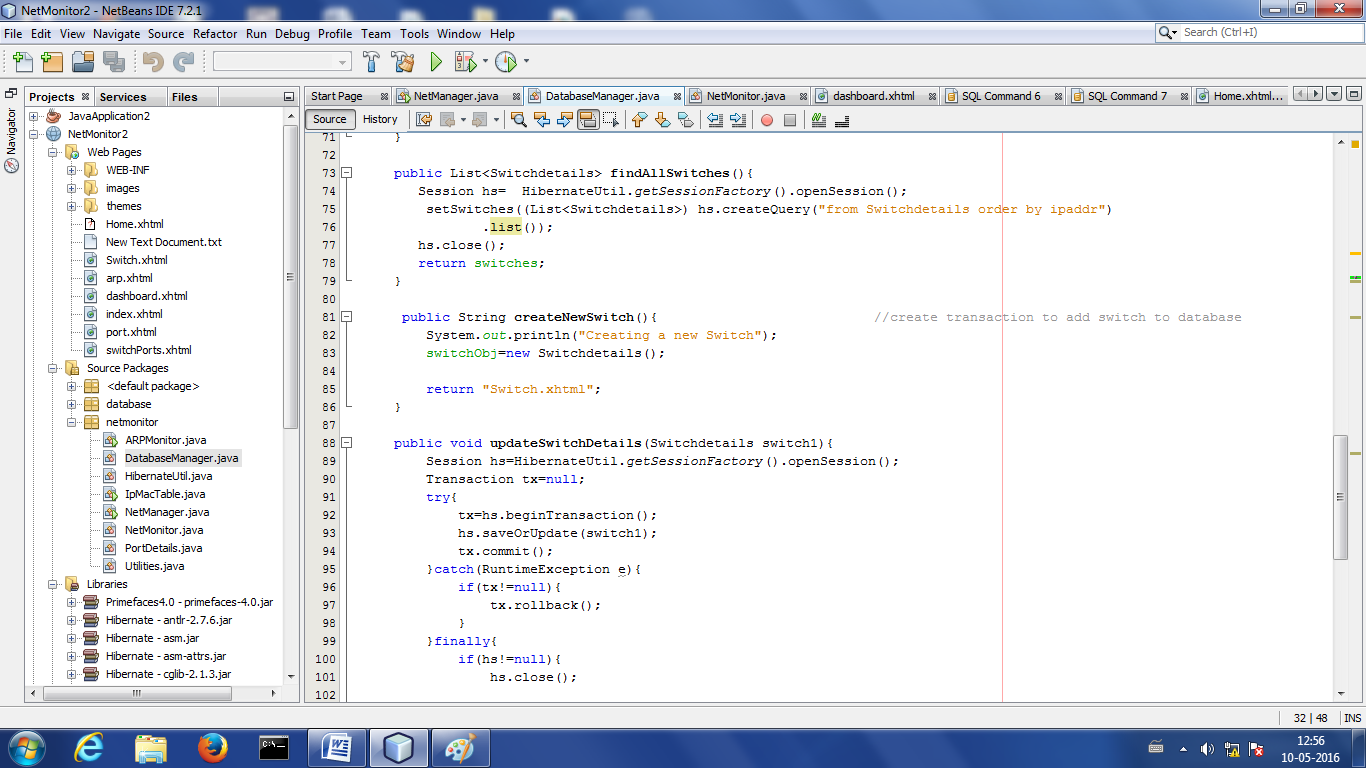
**Network Manager**

This module is the start of execution of the program since it has the main function. It calls the NetMonitor module and also checks if the switch is up and running. It also collects the various information of the switch by passing OIDs to the switch. It acts as the central system which calls other modules and it needs to be running continuously in the background.



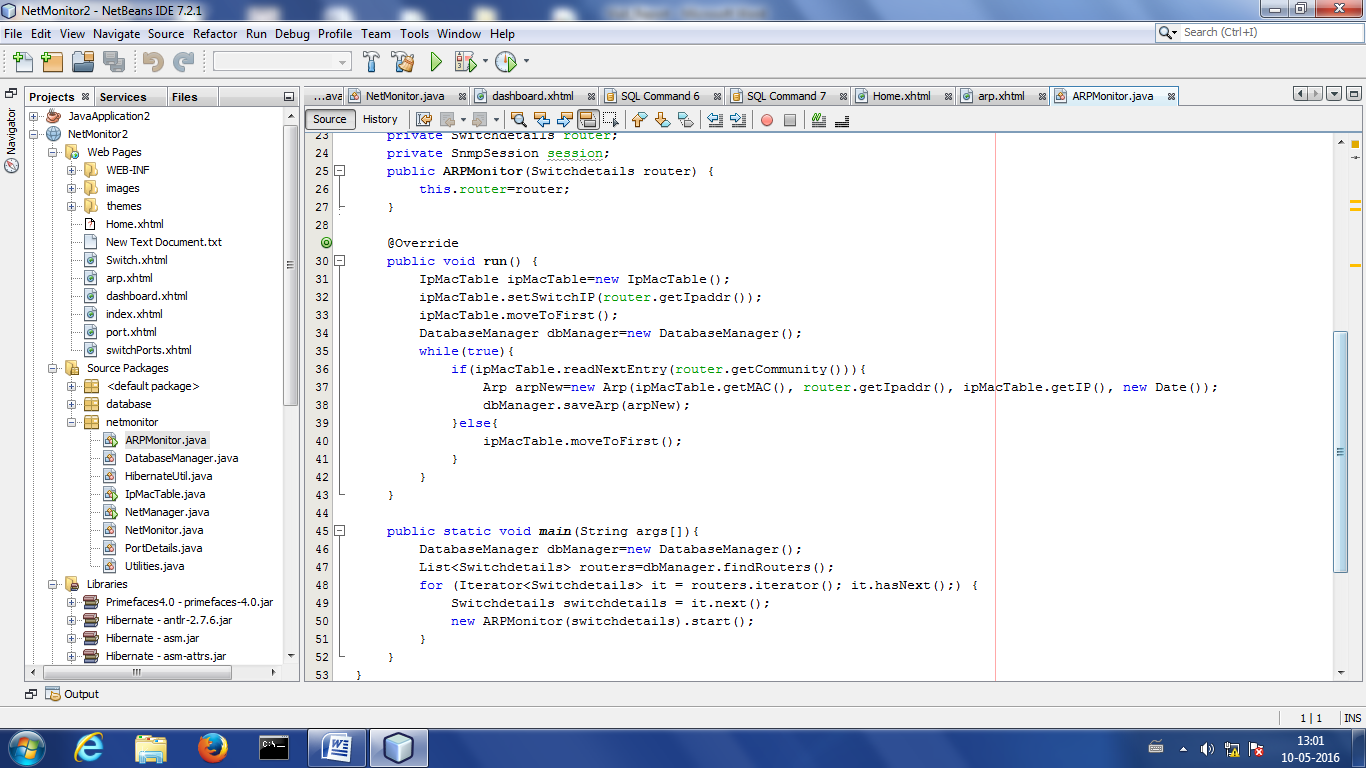
**Database Manager**

This module is used to update or modify the details in the tables which are present in the database. It performs the job by creating sessions to perform each transaction. At the end of each session, it commits the transaction or does a roll back if necessary.

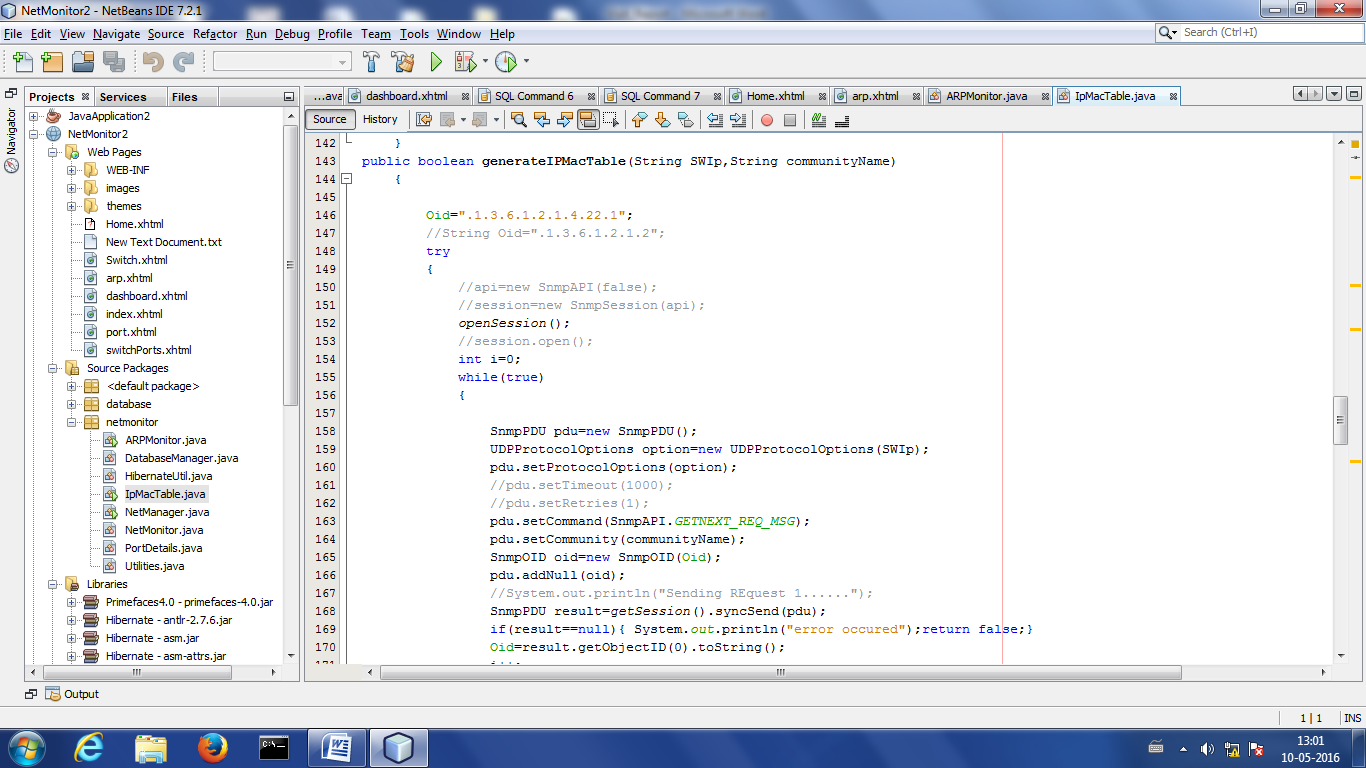


**ARP Monitor**

This module is used to monitor the ARP details of the switch and store it in the ARP table. It contains a main module and needs to be running by itself in the background. The ARP table has four fields (router address, IP address, MAC address, and date of entry) which are updated when this module is called. This module can only retrieve the ARP details of a router, but not a switch, since the switch does not contain an ARP table.



**IP Mac Table**

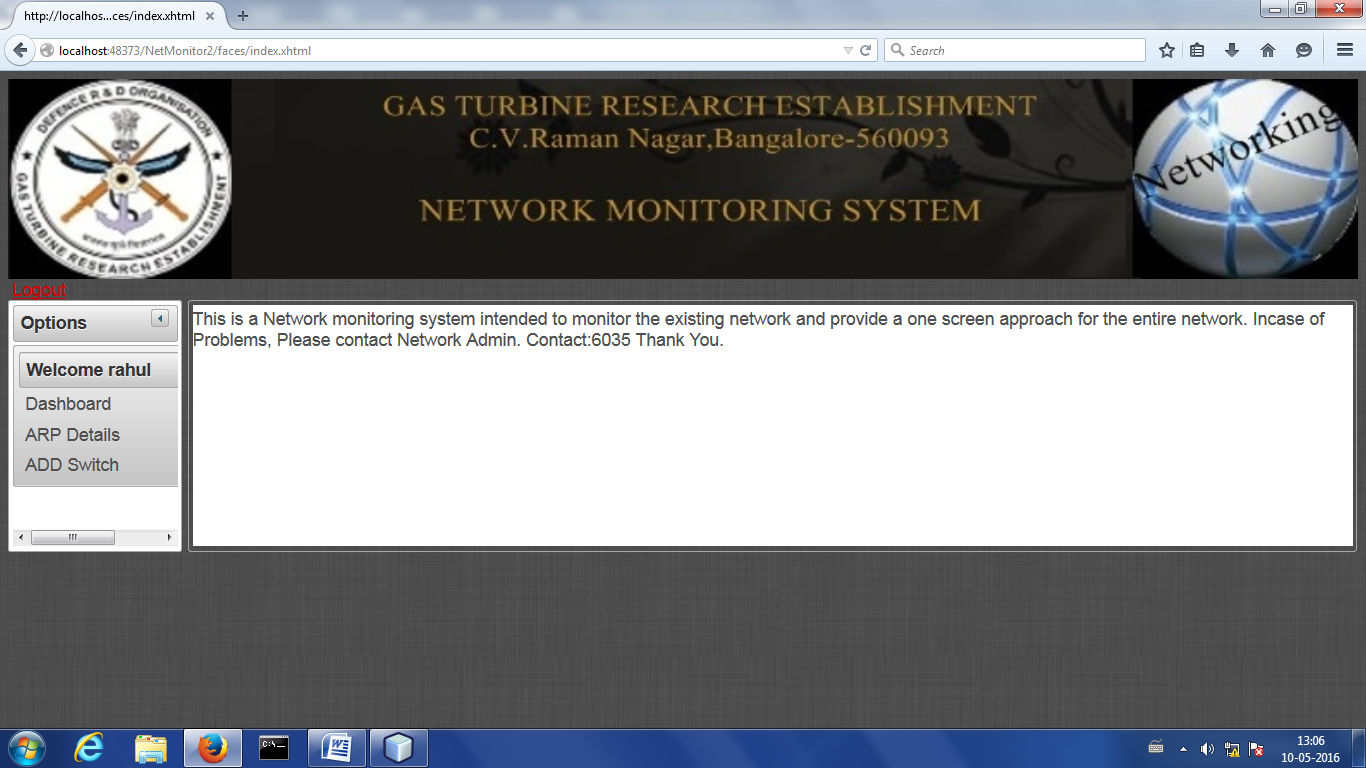


These are the screenshots (taken in order) which are opened in any web browser and form the Graphical User Interface:

**1) Login**

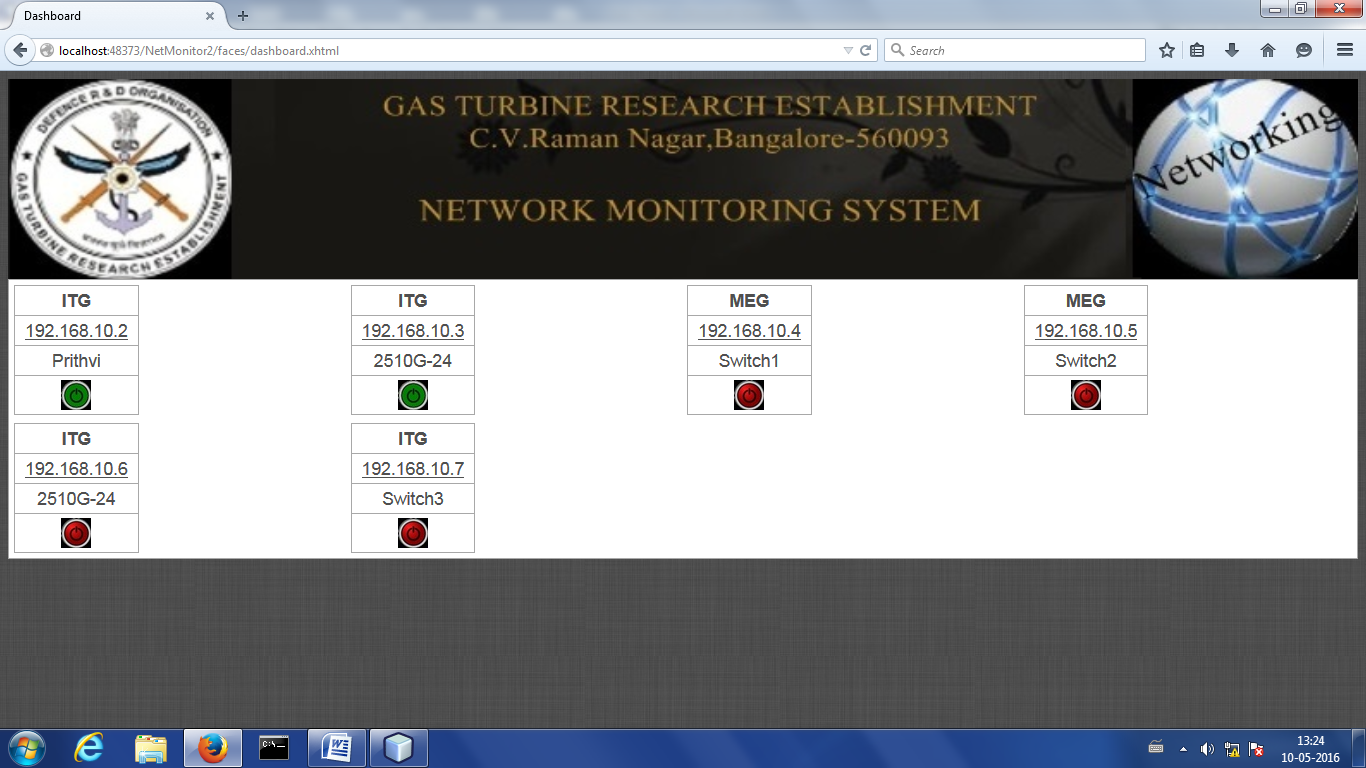


**2) Welcome Page**

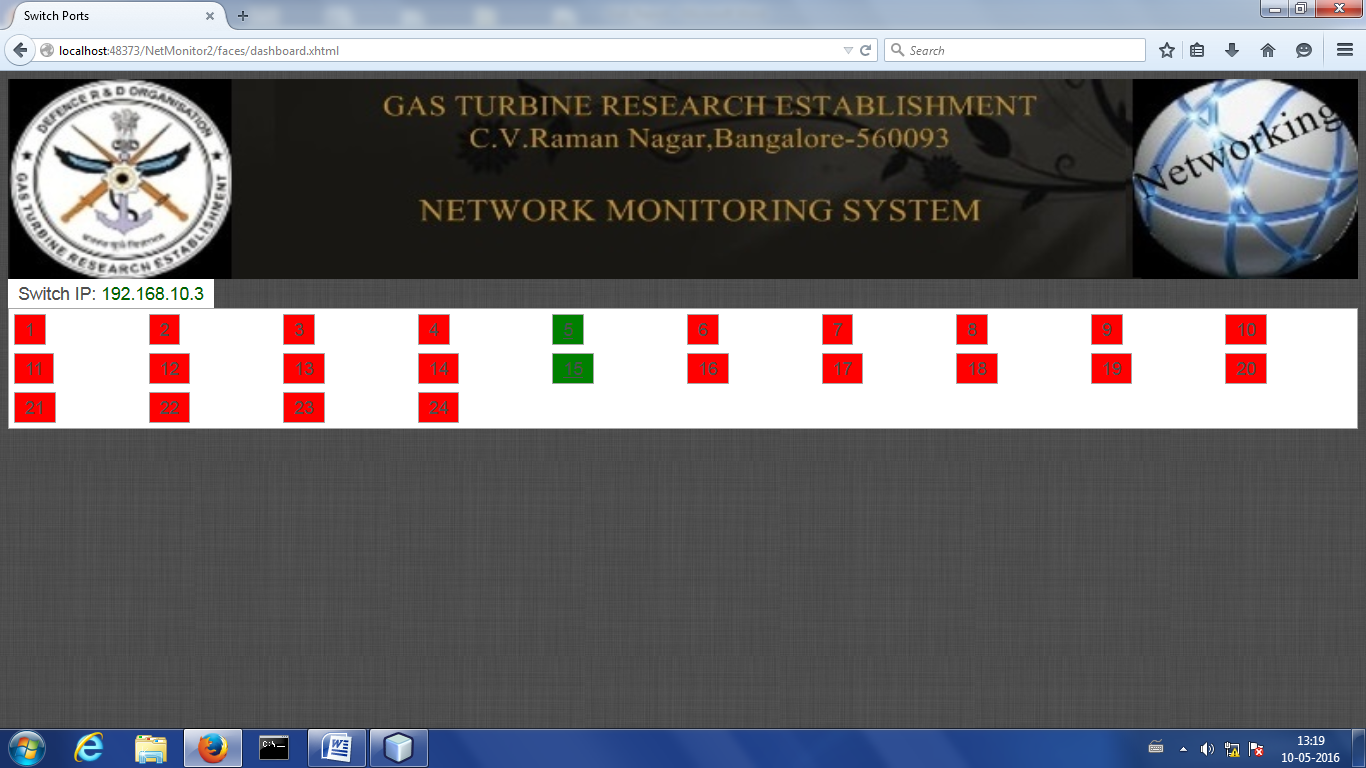


**3) Dashboard**

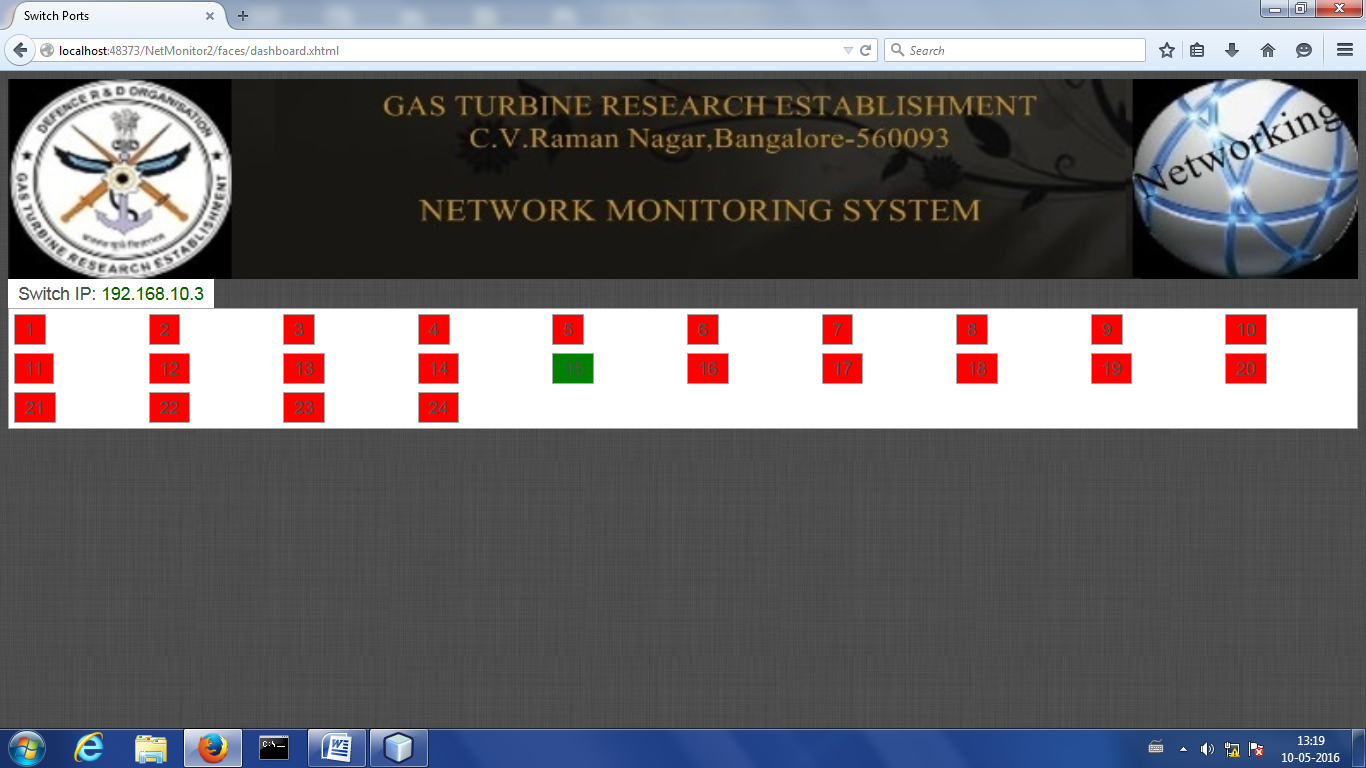
**3.1) Switch UP**



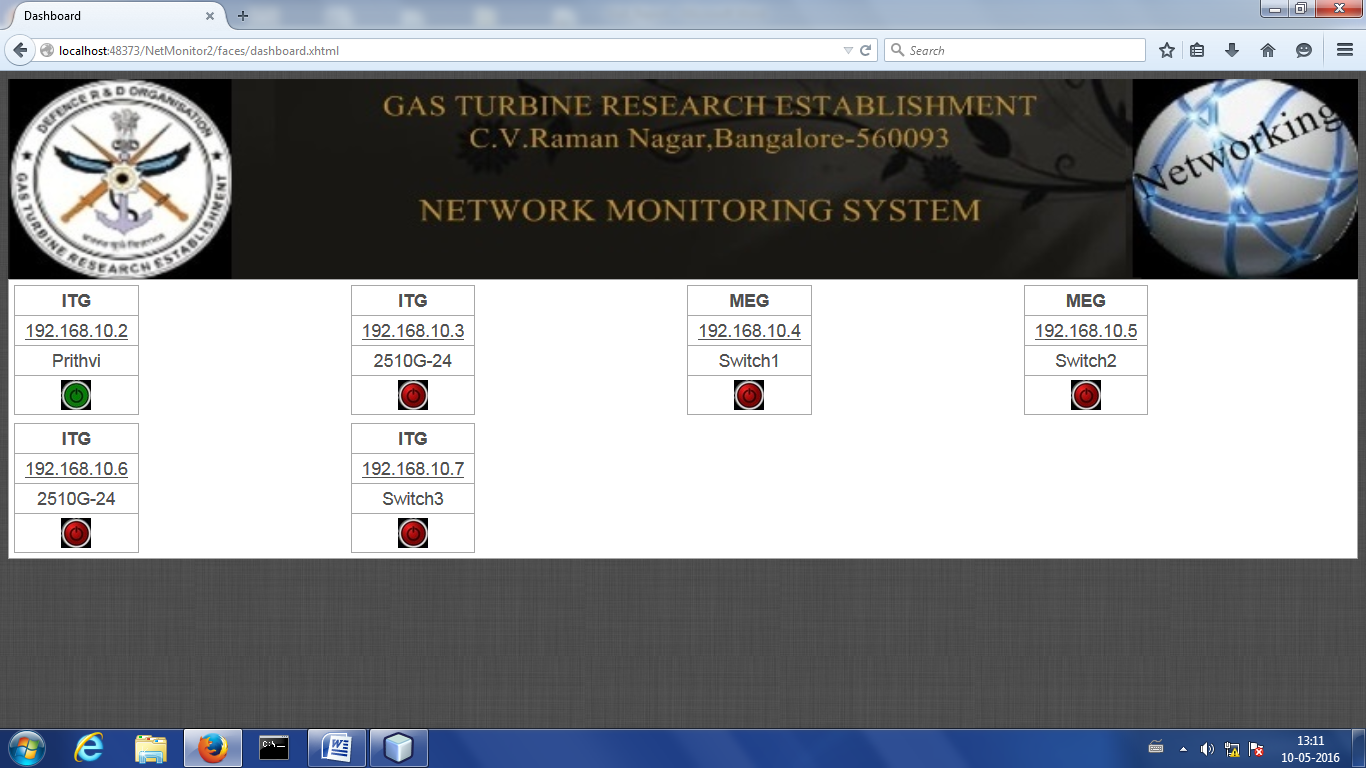
**3.1.1) Port UP**



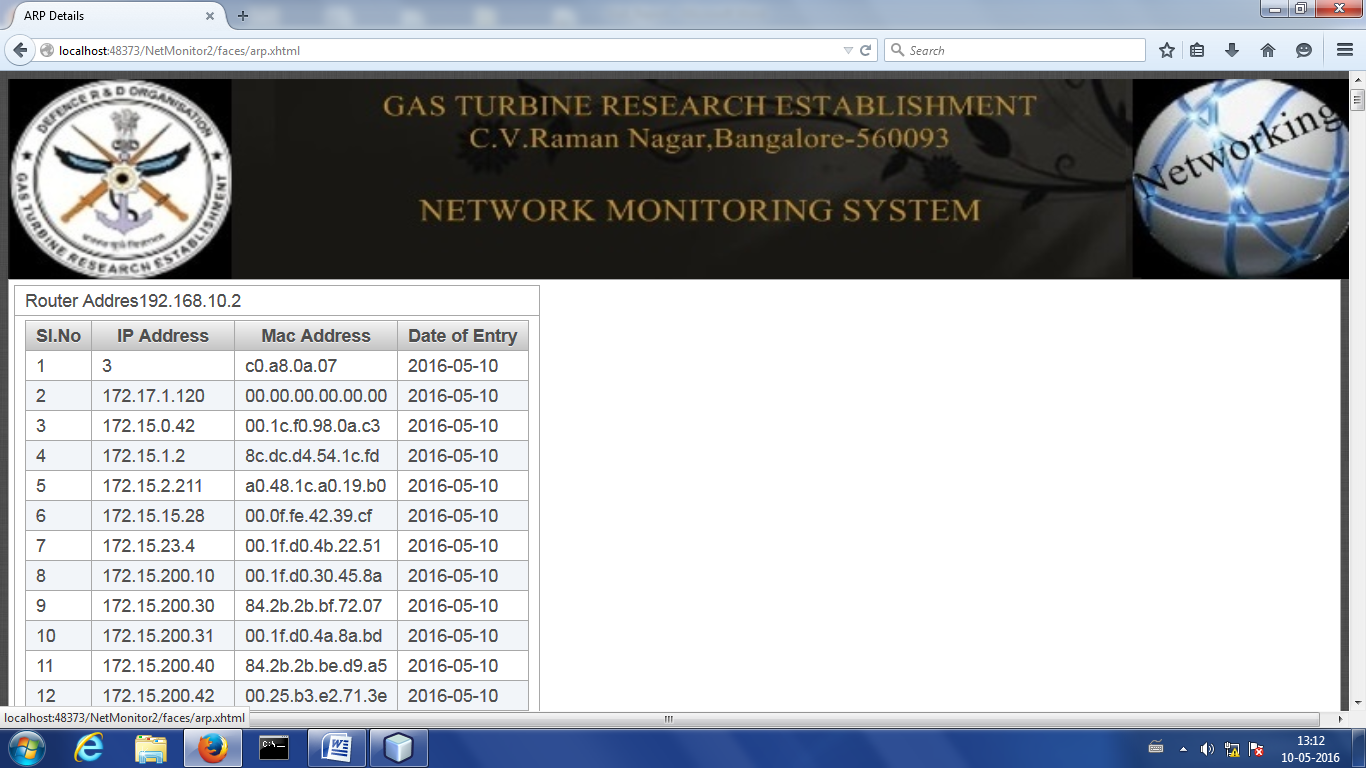
**3.1.2) Port DOWN**



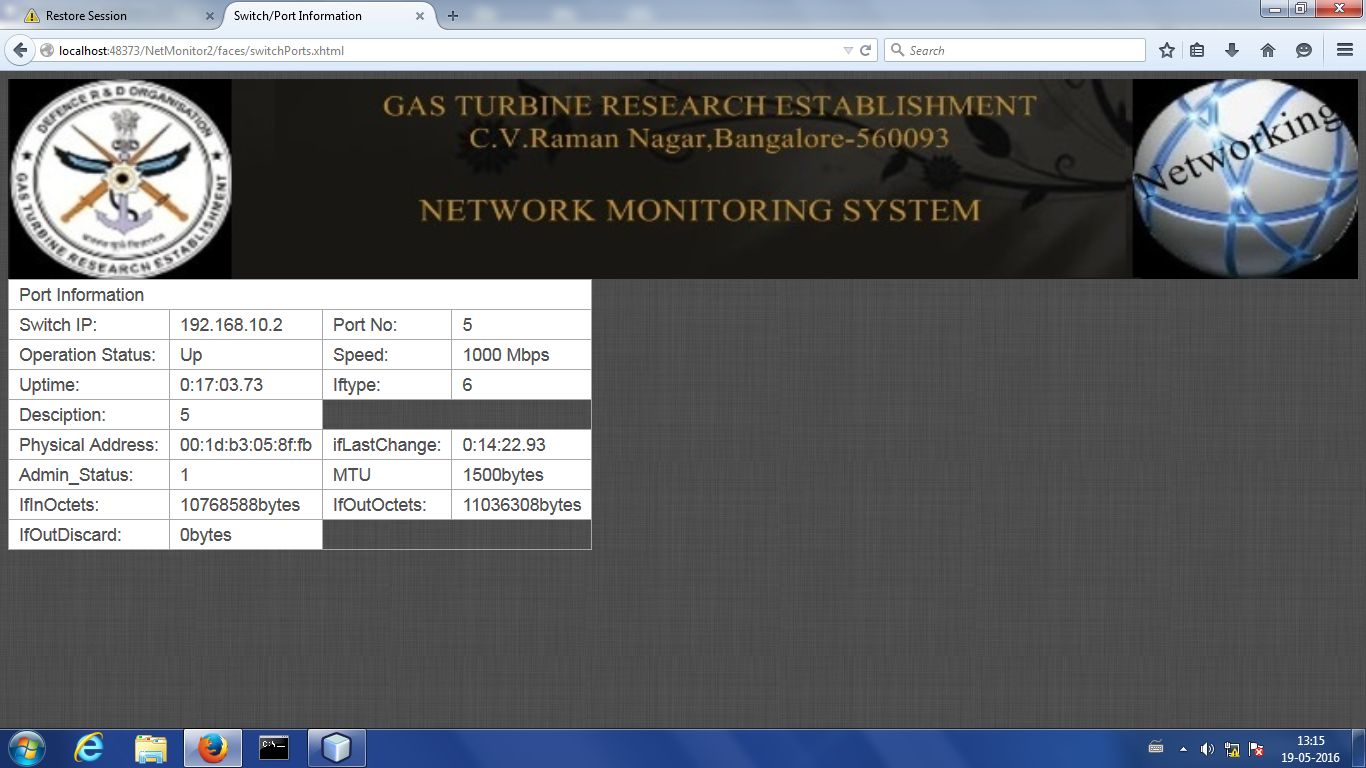
**3.2) Switch DOWN**



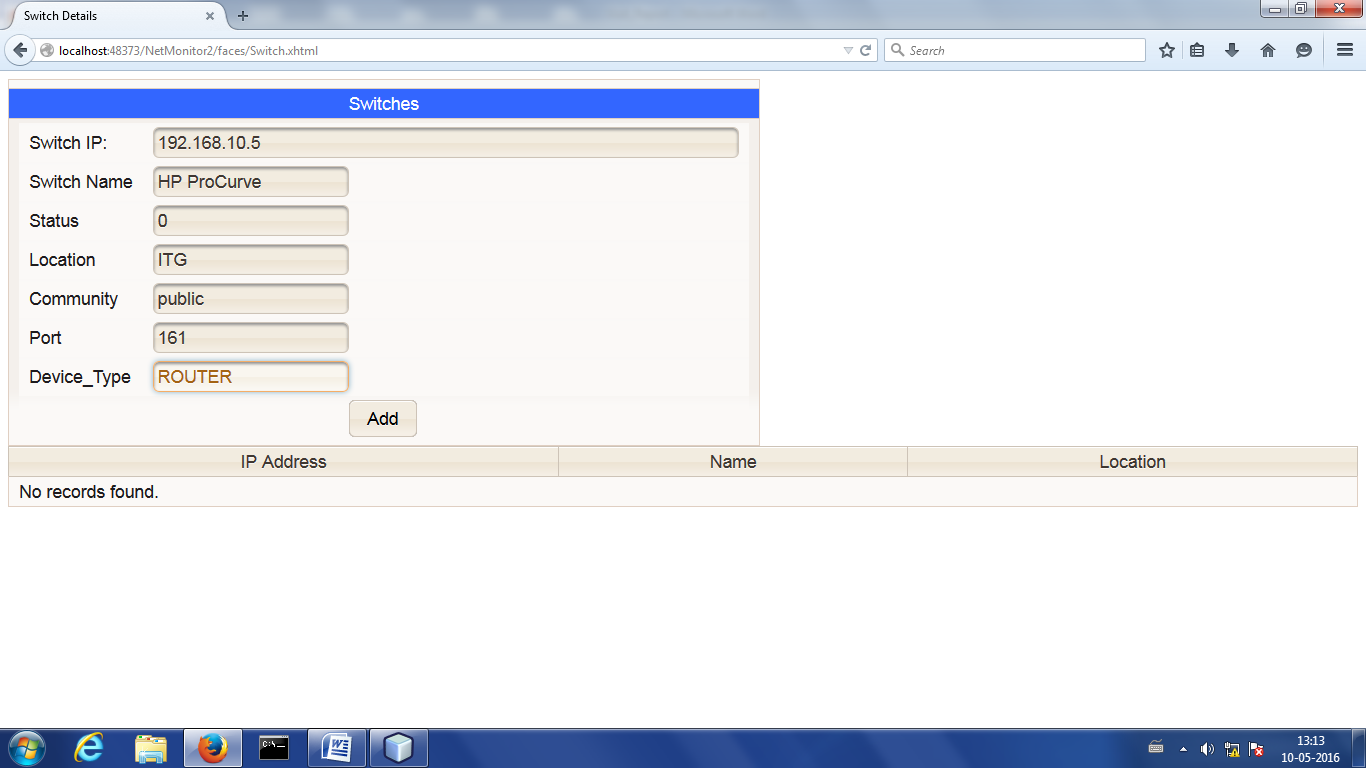
**4) ARP Details**



**5) Port Details**

****

**6) ADD Switch**



**Summary**

This chapter gives a graphic view of the execution of the system. The output screens show each of the execution stages for each protocol. Based on the analysis the proposed protocol performs well in terms of energy consumption when compared to the existing AODV protocol.

**Chapter 10**

**CONCLUSION & FUTURE SCOPE**

**10.1 Conclusion**

An efficient network monitoring system is essential for the successful functioning of any organization. However it must be designed as per the requirements of the organization and tailored for its everyday use. The network monitoring system developed is complying with the requirements and fits perfectly by providing security and several agent less features. It acts as a revamp of the existing system where nodes have to be pinged for checking their availability.

This system acts as an interface where the administrator/client can access the entire network on a single screen- Dashboard. Furthermore, it is possible to detect the different systems connected to that node- specifically a router- so that the administrator has data to make changes in case of outage.

**10.2 Future Scope**

In future, the system could have an SMS or Email alert system to alert the administrator in case of outage. It would also be easier to track performance if it is represented in the form of graphs. Finally the system could have ticket management system and a better interface.

**REFERENCES**

**[1]. Rafiullah Khan, Sarmad Ullah Khan, Rifaqat Zaheer and Muhammad Inayatullah Babar, “*An Efficient Network Monitoring and Management System*”,** International Journal of Information and Electronics Engineering, Vol. 3,No. 1, January 2013

**[2].** **M. Schubert, A. Hay, D. Bennett et al., “*Nagios3 Enterprise Network Monitoring*”** Designing Configurations for Large Organizations, Chap:2, pp.25-84, 2008.

**[3].** **Mohd Izuan Mohd Saad, Zuriati Ahmad Zukarnain “*Performance Analysis of Random-Based Mobility Models in Ad Hoc Routing Protocol”***- International Conference on WirelessCommunication and Sensor Computing, ICWCSC IEEE FEB 2010.

**[4].** **A. Gomez, C. Dafonte, and B. Arcay, “*3D Visualization for system and networks monitoring support*”,** presented at 3rd IEEE Conference on Human System Interactions (HSI-10), A Coruna, Spain, July 2010.

**[5]. Junguo Zhang, Wenbin LI, Dongxu CUI, Xueliang ZHAO, Zhongxing YIN** “***The NS2-based Simulation and Research on Wireless Sensor Network Route Protocol***” -, Beijing Forestry University, Beijing, P.R.China, 978-1-4244-3693-4/09/-2009 IEEE

**[6]. D. Oliveira, T. Vasques, F. Vieira, G. de Deus et al ,“*A management system for PLC networks using SNMP Protocol*”,** presented at IEEE International Symposium on Power Line Communications and Its Applications (ISPLC), 10, Goias, Brazil, June 2010

**[7]. Liang Fan Shaanxi Regional Electric. Power Group, Xi''an, China Li Fei** ***“A Network Monitoring System Design under LAN” -***, Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on (Volume:4 )

**[8]. IJARCSSE Notes on Network Monitoring,** International Journal of Advanced Research in Computer Science and Software Engineering

**[9].**  “***Data communication and Networking*”,** 4th Edition, Beherouz A Forouzan

**Websites**

**[10].** <http://www.en.wikipedia.org>

**[11].** <http://ieeexplore.ieee.org/xpl>

**[12].** https://www.novell.com/coolsolutions/feature/16723.html

**[13].** http://www8.hp.com/h20195/V2/GetPDF.aspx/ 4AA5-1319ENW.pdf

**[14].** http://www.javatpoint.com/hibernate-tutorial

**APPENDIX**

**Acronyms & abbreviation**

**Acronyms : Abbreviation**

GNU : GNU's Not Unix

HTTP : Hyper Text Transfer Protocol

HTML : Hyper Text Markup Language

IAB : Internet Architecture Board

IIS : Internet Information Server

NetBIOS : Network Basic Input / Output System

RT : Real Time

SQL : Structured Query Language

SSH : Secure Shell

TELNET : TELecommunication NETwork

UI : User Interface

URL : Universal Resource Locator

UML : Unified Modelling Language

**Paper Presented During the Course of Project Work**

The following paper related to dissertation work was presented in an International conference during the course of project work.

1. **Title of paper**: “Advanced Network Monitoring System Using A Hybrid Of Agent And Agentless Concepts”

**Author**: Rahul S Sreedhar, Prithvi Vihari R, Dr. N Guruprasad

**Conference**: International Conference on Innovations in Computing and Networking- ICICN 2016

**Organizer of Conference**: Rajarajeshwari College of Engineering in association with Computer Society of India (CSI)

**Date on which paper presented**: 12th May, 2016.