CONSTELLA PLATINUM

AUTONOMOUS SEMANTIC BOUNDARY AWARE NETWORK TOPOLOGY DISCOVERY OF LARGE, MULTI-SUBNET NETWORKS

By

Hamid Mukhtar

(2002-NUST-BIT-118)

A project report submitted in partial fulfillment of

the requirement for the degree of

Bachelors in Information Technology

In

NUST Institute of Information Technology

National University of Sciences and Technology

Rawalpindi, Pakistan

(2006)

In the name of Almighty Allah

The Most Beneficent, the Most Merciful

CERTIFICATE

Certified that the contents and form of thesis entitled "**Constella Platinum: Autonomous Semantic Boundary Aware Network Topology Discovery of Large, Multi-subnet Networks**" submitted by Hamid Mukhtar have been found satisfactory for the requirement of the degree.

Advisor: _____

Wg Cdre Ramzan

Co Advisor: _____

Dr. Hafiz Farooq Ahmad

Committee Member:_____

Mr Aamir Jillani

Committee Member:_____

Mr Ali Sajjad

DEDICATION

In the name of Allah, the Most Gracious, the Most Merciful

To my beloved Parents and my family

ACKNOWLEDGEMENTS

For this project I am highly thankful to my supervisors Dr Farooq and Dr Arshad for their continuous dedicated support and guidance without which the project would not have been possible. Their presence and motivation was just like the basic stimulus for my work and I am extremely thankful for the time, help, motivation and facilities they provided to me during my project. The exposure that I got under their supervision will be there with me for life time. I really enjoyed working with them and they were the real mentors for my research work.

I would like to express my gratitude to Wg Cdre Ramzan for his valuable implication and comments to improve the dissertation and the encouragement during my project work. I am highly grateful to my external advisor Dr. Hiroki Suguri (Comtec Japan) for his worthy supervision and support to perform such a valuable research work and studies in the field of networking.

I am also deeply beholden to Mr Aamir Jillani and Mr Ali Sajjad for their wholehearted motivation and help whenever I needed it and for their valued suggestions during the project work.

I would especially like to thank the all my faculty members because their guidance was always there throughout my degree especially Muhammad Atif, Nadeem Ahmed, Wg Cdr Maqsood ul Hassan, Wg Cdre Nasir Mehmood, Kamran Munir, Arshad Farooq, Dr Inam, Ms Sarah and Ms Abira Masood.

I would really like to appreciate the efforts of my beloved parents for all their support throughout my life starting from the childhood till my graduation and nothing could be possible without their support and efforts.

In the end I would like to thank the whole Constella Team which was always there with me and I enjoyed a lot working with them throughout this project. I would like to especially thank the efforts of Mr Tallat Hussain Tarrar who was always there for me to motivate me whenever I was down and Mohsan Jameel my project fellow who was always there with me and we really enjoyed working together for the whole life of the project and also Faran Javed who was there when we needed him.

I am extremely grateful to my friends Shehryar Mohammad Khan, Qasim Bilal Lone, Rizwan Ur Rasheed, Fatima Ather, Maryam Zafar, Mohsan Jameel, Khawar Hasham, Ammar Hassan, Muhammad Amin, Hafiz Muhammad Fahad, Aftab Ahmed and Noman Latif for supporting helping encouraging me when ever I needed them and I really owe them for that.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

[A]

ARP Address Resolution Protocol

AFT Address Forwarding Table

API Application Programming Interface

[C]

COMTEC Communication Technologies Sendai Japan.

[G]

GUI Graphical User Interface

[I]

ICMP Internet Control Message Protocol

IP Internet Protocol

[L]

LAN Local Area Network

[M]

MAN Metropolitan Area Network

MIB Management Information Base

[N]

NIC Network Interface Card

NMS Network Management station

[O]

OID Object Identifier

	OWL	Web Ontology Language
[P]		
	Ping	Packet Internet Groper
[R]		
	RDF	Resource Description Framework
	RTT	Round Trip Time
	RMON	Remote Network Monitoring
	RARP	Reverse Address Resolution Protocol
	RT	Routing Tables
[S]		
	SNMP	Simple Network Management Protocol
	SMI	Structure Management Information
[T]		
	TCP	Transmission Control Protocol
	TPING	TCP Based Probing module
[U]		
	UDP	User Datagram Protocol
[W]		
	WAN	Wide Area Network
[X]		
	XML	Extensible Markup Language

ABSTRACT

Active Network Topology discovery is a mechanism to keep track of the status of network resources and their interconnections. The knowledge of the real-time topology of a network is crucial for management tasks such as resource management, resource discovery, congestion avoidance, failure detection and correction as well as network state visualization and analysis. In today's networks ensuring scalability by manual entry of topology information is very difficult due to their size and dynamic nature. Therefore we need to define mechanisms to automatically discover the network topology. Also sending request to all possible IP addresses in a network is not feasible because of the large number of possible IP addresses e.g. there can be more than sixteen million possible addresses for a Class A network. Therefore in this paper we propose an efficient algorithm for the automatic discovery of the network resources where we use ARP cache based guessing to check the subnets with high probability of being alive. For status checking of network nodes current techniques use ICMP based probing, which are be blocked or filtered by many routers. Therefore we use an open source TCP based ping mechanism (TPing) [1] for discovery of network elements. We make use of concurrent TPing to discover the topology in multi-subnet environments and our algorithm performs discovery of networks employing CIDR (Classless Inter-Domain Routing) based IP addressing scheme efficiently and accurately. TPing is only available for Linux and for platform independence I developed my own windows version of that. Our algorithm also discovers the MAC addresses with multiple hop distance which are transparent from subnet to subnet. The algorithm only requires

SNMP (Simple Network management protocol) on routers and switches and do not need SNMP on end hosts for the discovery process.

The data is exported in a semantically enriched form. By defining shared and common vocabularies, ontologies help both people and machines to communicate concisely, supporting the exchange of semantics instead of syntax. We are using Resource Description Framework (RDF) to export the data from the discovery algorithm. This approach will reduce the human intervention for the information retrieval about a particular network so that it can be further used by FIPA compliant Multi-agent systems to provide an agent understandable format where agents will be acting on behalf of the user for automatically discovering and querying the topology ontologies and retrieving the required data.

Chapter 1

INTRODUCTION

Information and communication are two of the most important challenges for the success of every enterprise. While today nearly every organization uses a substantial number of computers and communication tools (telephones, fax, and personal handheld devices), they are often still isolated. While managers today are able to use the newest applications, many departments still do not communicate and much needed information cannot be readily accessed.

To overcome these obstacles in an effective usage of information technology, computer networks are necessary. They are a new kind (one might call it paradigm) of organization of computer systems produced by the need to merge computers and communications. At the same time they are the means to converge the two areas; the unnecessary distinction between tools to process and store information and tools to collect and transport information can disappear. Computer networks can manage to put down the barriers between information held on several (not only computer) systems. Only with the help of computer networks can a borderless communication and information environment be built.

Computer networks allow the user to access remote programs and remote databases either of the same organization or from other enterprises or public sources. Computer networks provide communication possibilities faster than other facilities.

Because of these optimal information and communication possibilities, computer networks may increase the organizational learning rate, which many authors

declare as the only fundamental advantage in competition. As having networks is very important so is there management. Network Management is a critical job that needs a full understanding of the technology of computer networks. Whenever we talk about monitoring and managing large network the most important issue is to have a single snapshot of the whole network, which actually explains that, how networks are laid out. This creates a need for physical topology map of the network.

Network topology is a representation of the interconnection between directly connected peers in a network. In a physical network topology, peers are ports on devices connected by a physical transmission link. A physical topology corresponds to many logical topologies, each at a different level of abstraction. At the IP level, peers are hosts or routers one IP hop from each other, and at the workgroup level, the peers are workgroups connected by a logical link. Network topology constantly changes as nodes and links join a network, personnel move offices, and network capacity is increased to deal with added traffic. Keeping track of network topology manually is a frustrating and often impossible job. In large and constantly evolving networks, it is difficult to determine how the network is actually laid out. It is sometimes impossible to identify the existence of multiple paths between hosts, switches, routers and printers. Network topology knowledge including the path between endpoints, can play an important role in analyzing, engineering, and visualizing networks. Network topology can be used in order to simulate real networks. Network topology information is useful in deciding whether to add new hosts, switches, routers, printers and to figure out whether current hardware is configured correctly. It also allows network managers to find bottlenecks and failures in the network.

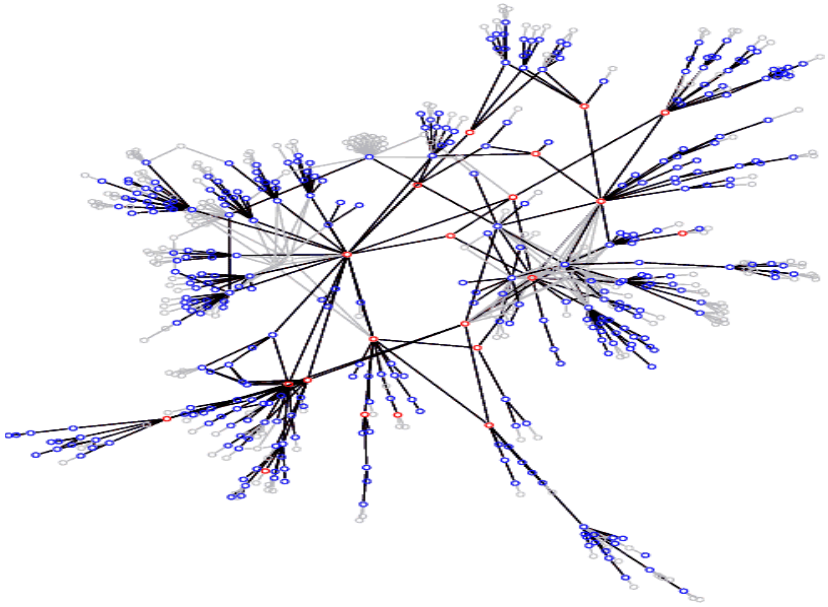

Figure 1: The internet and the networks today

Discovering the physical layout and interconnections of network elements is a prerequisite to many critical network management tasks, including reactive and proactive resource management, server sitting, event correlation, and root-cause analysis. This situation clearly mandates the development of effective, general-purpose algorithmic solutions for automatically discovering up-to-date physical topology of an IP network. One of the main challenges in the design of such algorithms is dealing with the lack of established, industry-wide standards on the topology information maintained locally by each network element, and the diversity of elements and protocols present in today's multi-vendor IP networks. Traditional topology discovery algorithms are based on SNMP, which is not universally deployed. Both network management and performance analysis benefit from network topology knowledge. Network managers, with access to an automatically derived topology of their network, are better able to react to and prevent problems. They can not only observe the exact

location of a problem and trace it back to the source, but also analyze the use of the network under normal operations. This knowledge allows them to anticipate problems and plan for them before services are impacted. For planning parallel applications, topology knowledge allows the effects of link-sharing on an application's performance to be predicted in advance and nodes to be selected with appropriate network connections to match the application's needs. While the bottlenecks on LANs are generally not as severe as those on WANs, reliance on commodity components and the emergence of grid-based computing has increased the need for topology discovery. The goals of our work are to automatically discover network topology and visualization of the topology discovered.

Developing an efficient algorithm for automatically discovering up-to-date physical topology of a large multi-subnet network poses several difficult challenges.

1.1 Different views of one network

1.1.1 View of a business manager

Managers are usually more interested in the inventory that is present in the network and how they are being utilized. They need information about how the devices are operating and what will be the future need so that they can plan the budgets for the upgradation of the network.

In this case the information of the topology is needed check how much inventory is being utilized in the proper manner and to plan for the maximum utilization of resources.

1.1.2 View of a software engineer

For a developer of distributed applications it is really important to have an insight of the network topology of the network he is designing his application for. With this knowledge, possible bottleneck or faulty operations can easily be avoided.

1.1.3 View of a network manager

The System administration staff is most interested in network topologies. It is the part of their work to have knowledge of the network topology. They have to plan the changes in the network every time a link goes down and they want to be constantly updated about the real picture of their network so that network efficiency reliability and fault tolerance can be achieved. As in dynamic networks the network requirements are constantly changing so network managers are the people who are best benefited from the information.

1.2 Computer Networks and Network Monitoring

Computer data networks enable communication between autonomous computers (devices) connected to the network. The devices are called nodes, elements, hosts or stations. Devices that use a network are also called end systems. Those devices that participate in the operation of the network are called intermediate systems.

Examples are bridges, and routers. Additionally, physical infrastructure use devices such as repeaters.

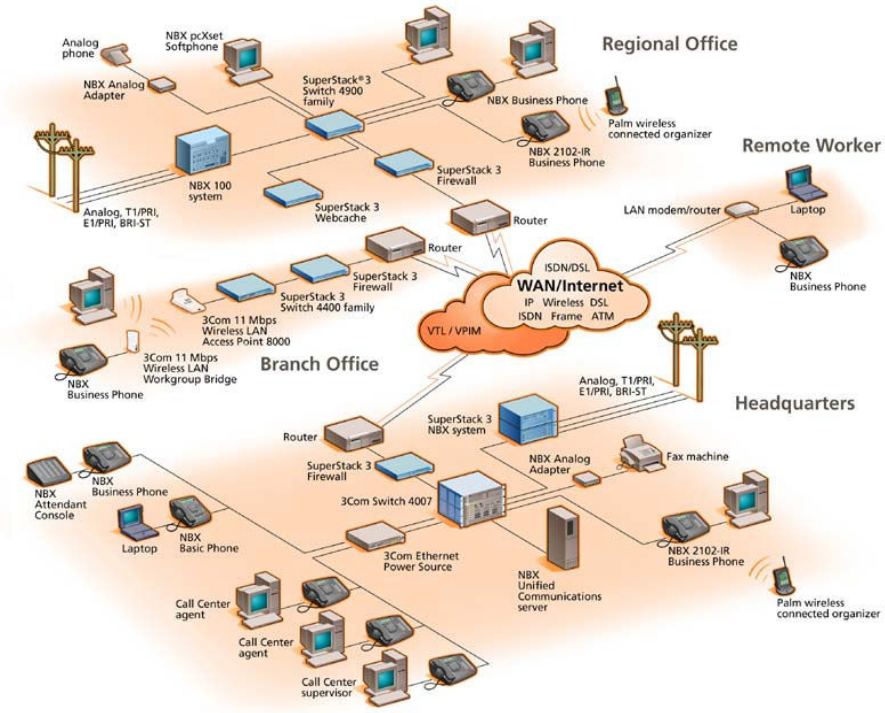

Figure 2: Today's Networks

Generally, a network is classified as Local Area Network (LAN) or a Wide Area Network (WAN). A LAN is based on technology that typically provides high transmission rates, has shared access, is limited in geographical distance, and has low cost. A WAN is based on technology that typically allows large geographical distance, had dedicated access, has lower transmission rates, and has a high cost. Examples of LAN technology are Ethernet, Token Ring, and FDDI networks. Typical examples of WAN technology are slow speed serial interfaces, ISDN interfaces, and high-speed serial interfaces. The WAN protocols include PPP, Frame relay, and proprietary point-to-point.

Network administrators are responsible to monitor there networks. Network monitoring actually means that how networks are performing. Network monitoring software can be very helpful when trying to troubleshoot network related issues, watching your network equipment and providing performance analysis. Network monitoring software is a must have if you are a network administrator. Network monitoring software can allow you to monitor a single network device or all your network devices. If you have several network devices then a network monitoring software may be what you need.

1.2.1 Types of Network Monitoring

1.2.1.1 Proactive Monitoring

Proactive network management also is just what it sounds like: Network managers use their network management infrastructure to proactively attempt to prevent problematic events and conditions from arising. This maximizes perceived service value by minimizing service disruptions, which translates directly into greater service reliability and availability, while also minimizing consumption of support resources.

Proactive network management is what I consider to be "advanced network management." And it’s not just the network that needs management—it’s the entire service solution infrastructure. It’s also not just the digital MIB information you need to use. Other information such as time-of-day and experience-based device behavioral characteristics also need to factor into correlated intelligence decisions.

1.2.1.2 Reactive Monitoring

Reactive network management is just what it sounds like: Network managers react to problematic events and conditions discovered in the network by their network management system by adjusting various system parameters to eliminate, accommodate and/or compensate for those conditions.

This consists of fallible humans attempting to respond to a growing number of potential traps and alarms from various solution components. As the number of services offered and related complexity grows, so will the traps and alarms. This approach is madness.

Figure 3 : Managing Large Networks and Multiple Types of Network Devices

1.2.2 MANAGING NETWORKS

Managing the networks means configuring the networks and then monitoring and controlling it so that it provides the contracted services at or above agreed levels within a budget. Unfortunately for network managers, most networks are dynamic organisms whose configurations and services are constantly changing. The expectation for network managers is to do more with the same or fewer resources. In some environments there is a clear distinction between "the network" and the "end systems" connected to the network. Managing the end systems is system management. Managing the network is network management. In this environment, different organizations are responsible for network and system management. In other environment, the same organization managed both system and the network, so a little distinction is made. A user just want to get a job done and typically does not know if a problem is due to a network or system failure. With client-server computing and the need to access information with a Web browser, a working network is a requirement. So there is a need to use an efficient and effective technology that helps in monitoring a network and that can be used to assist in determining configuration changes and control procedure.

1.2.2.1 SIMPLE NETWORK MANAGEMENT PROTOCOL (SNMP)

The Simple Network Management Protocol (SNMP) is an application layer protocol that facilitates the exchange of management information between network devices. It is part of the Transmission Control Protocol/Internet Protocol (TCP/IP) protocol suite. SNMP enables network administrators to manage network performance,

find and solve network problems, and plan for network growth. An SNMP-managed network consists of three key components: managed devices, agents, and network-management systems (NMSs). A managed device is a network node that contains an SNMP agent and that resides on a managed network. Managed devices collect and store management information and make this information available to NMSs using SNMP.

Managed devices, sometimes called network elements, can be routers and access servers, switches and bridges, hubs, computer hosts, or printers. An agent is a network-management software module that resides in a managed device. An agent has local knowledge of management information and translates that information into a form compatible with SNMP. An NMS executes applications that monitor and control managed devices. NMSs provide the bulk of the processing and memory resources required for network management. One or more NMSs must exist on any managed network. More Detail about SNMP can be found in the appendices.

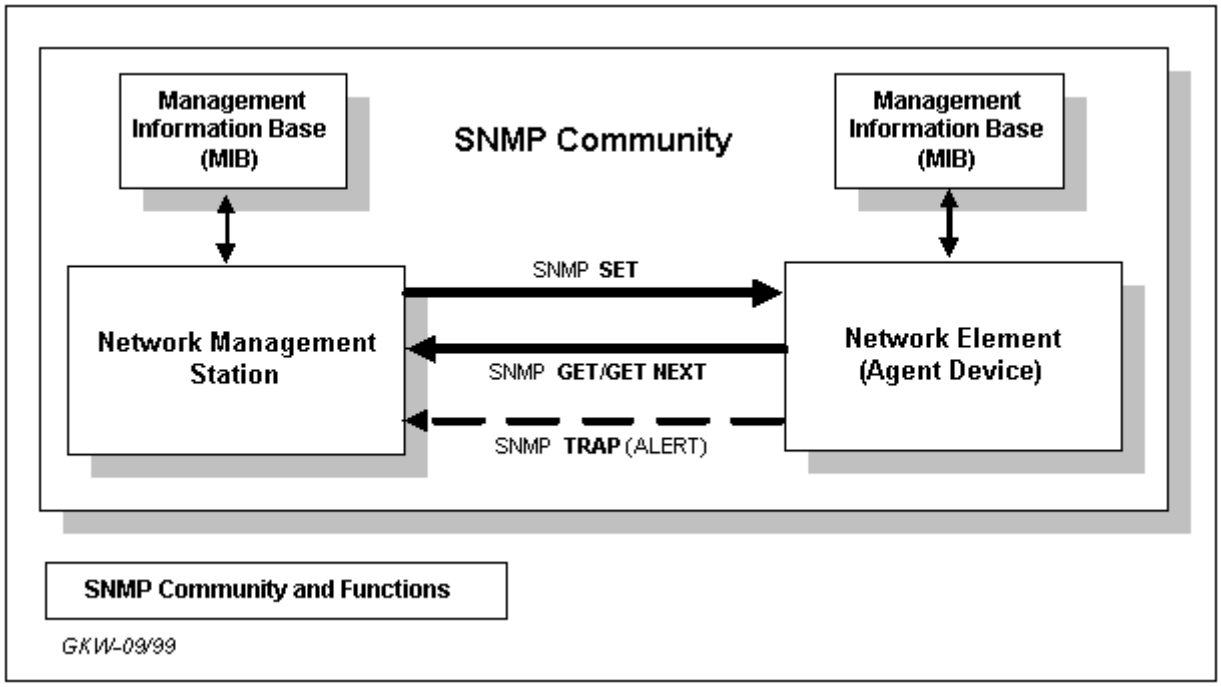

Figure 4: Simple Network Management Protocol Architecture

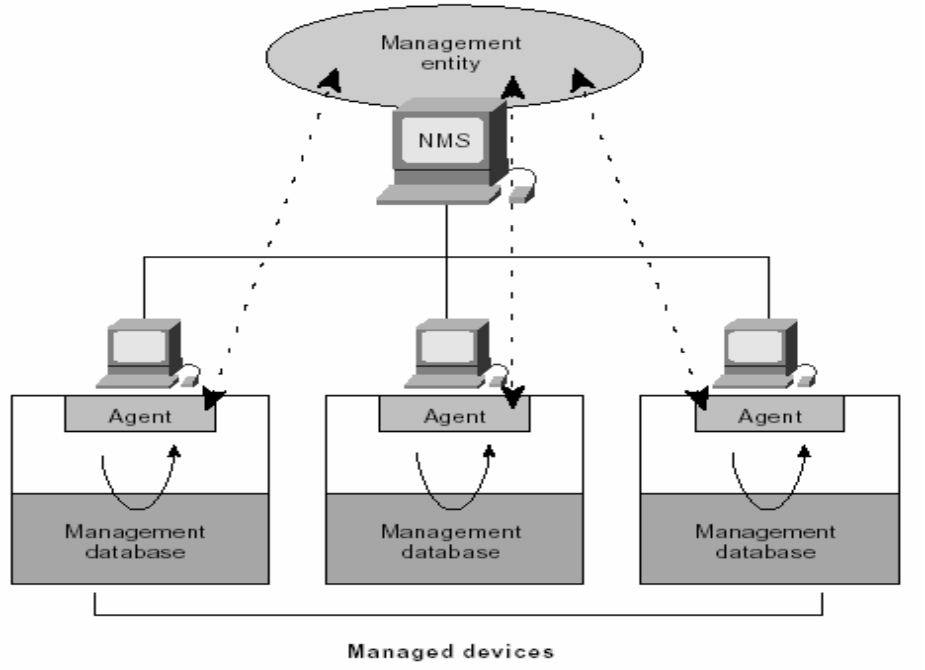

Figure 5: Simple Network Management Protocol Architecture

1.2.2.2 REMOTE NETWORK MONITORING (RMON)

RMON (Remote Network Monitoring) provides standard information that a network administrator can use to monitor, analyze, and troubleshoot a group of distributed local area networks (LANs) and interconnecting T-1/E-1 and T-2/E-3 lines from a central site. RMON specifically defines the information that any network monitoring system will be able to provide. It's specified as part of the Management Information Base (MIB) in RFC 1757 as an extension of the Simple Network Management Protocol (SNMP). The latest level is RMON Version 2 (sometimes referred to as "RMON 2" or "RMON2").

RMON can be supported by hardware monitoring devices (known as "probes") or through software or some combination. For example, Cisco's line of LAN switches includes software in each switch that can trap information as traffic flows through and record it in its MIB. A software agent can gather the information for presentation to the network administrator with a graphical user interface. A number of vendors provide products with various kinds of RMON support.

RMON collects nine kinds of information, including packets sent, bytes sent, packets dropped, statistics by host, by conversations between two sets of addresses, and certain kinds of events that have occurred. A network administrator can find out how much bandwidth or traffic each user is imposing on the network and what Web

sites are being accessed. Alarms can be set in order to be aware of impending problems.

1.2.2.3 INTERNET CONTROL MESSAGE PROTOCOL (ICMP)

The Internet Protocol (IP) is used for host-to-host datagram service in a system of interconnected networks called the Catenet. The network connecting devices are called Gateways. These gateways communicate between themselves for control purposes via a Gateway to Gateway Protocol (GGP). Occasionally a gateway or destination host will communicate with a source host, for example, to report an error in datagram processing. For such purposes this protocol, the Internet Control Message Protocol (ICMP), is used. ICMP, uses the basic support of IP as if it were a higher level protocol, however, ICMP is actually an integral part of IP, and must be implemented by every IP module. ICMP messages are sent in several situations: for example, when a datagram cannot reach its destination, when the gateway does not have the buffering capacity to forward a datagram, and when the gateway can direct the host to send traffic on a shorter route. The Internet Protocol is not designed to be absolutely reliable. The purpose of these control messages is to provide feedback about problems in the communication environment, not to make IP reliable. There are still no guarantees that a datagram will be delivered or a control message will be returned. Some datagram's may still be undelivered without any report of their loss. The higher level protocols that use IP must implement their own reliability procedures if reliable communication is required. The ICMP messages typically report errors in the processing of datagram's. To avoid the infinite regress of messages about messages

etc., no ICMP messages are sent about ICMP messages. Also ICMP messages are only sent about errors in handling fragment zero of fragmented datagram's. (Fragment zero has the fragment offset equal zero).

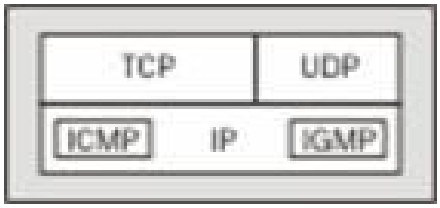

Figure 6: Internet Control Message Protocol in OSI Model

1.2.2.3.1 Packet Internet Groper (Ping)

Ping is a utility to determine whether a specific IP address is accessible. It works by sending a packet to the specified address and waiting for a reply. PING is used primarily to troubleshoot Internet connections. There are many freeware and shareware Ping utilities available for personal computers.

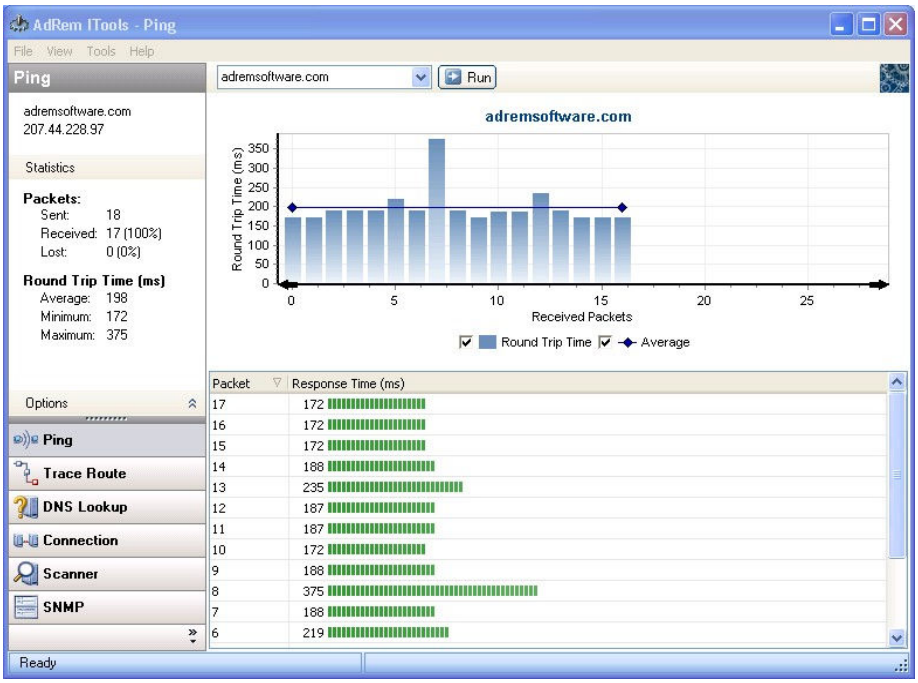

Figure 7 Working of PING

1.2.2.3.2 Trace Route

``Traceroute" is a network debugging utility that attempts to trace the path a packet takes through the network - its route. A key word here is ``attempts" by no means does traceroute work in all cases. If you've been paying attention, you already know that the only facilities TCP/IP provides for tracing packet routes are IP packet options (record route and its variants) that are poorly specified, rarely implemented in a useful way, and often disabled for security reasons. Trace route does not depend on any of these facilities. Traceroute, to put it simply, is a hack.

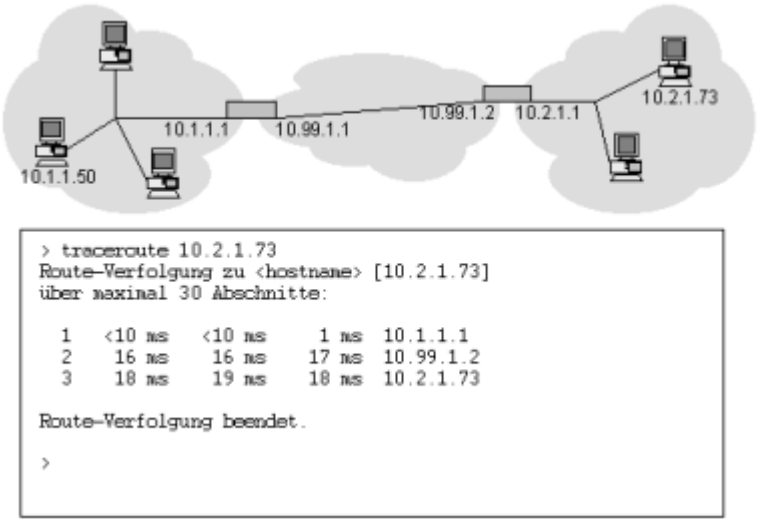

Figure 8: Running Traceroute Application

Traceroute transmits packets with small TTL values. Recall that the TTL (Time To Live) is an IP header field that is designed to prevent packets from running in loops. Every router that handles a packet subtracts one from the packet's TTL. If the TTL reaches zero, the packet has expired and is discarded. Traceroute depends on the common router practice of sending an ICMP Time Exceeded message, documented in RFC 792, back to the sender when this occurs. By using small TTL values which quickly expire, traceroute causes routers along a packet's normal delivery path to generate these ICMP messages which identify the router. A TTL value of one should produce a message from the first router; a TTL value of two generates a message from the second; etc.

In a typical traceroute session, a group of packets with TTL=1 are sent. A single router should respond, using the IP address of the interface it transmits the ICMP Timeout messages on, which should be the same as the interface it received the original packets on. The user is told this IP address, and DNS is used to convert this into a symbolic domain address. Also, round trip times are reported for each packet in the group. Traceroute reports any additional ICMP messages (such as destination unreachables) using a rather cryptic syntax. Once this first group of packets has been processed (this can take 10 seconds or no time at all), the second group (TTL=2) begins transmitting, and the whole process repeats.

Chapter 2

LITERATURE REVIEW

Network Topology discovery is a mechanism that enables us to keep track of the status of resources and their interconnections in a network. The real-time topology discovery of a network is crucial for management tasks such as resource management, resource discovery, congestion avoidance, failure detection and correction as well as network state visualization and analysis. In large, constantly growing networks, it is difficult to tell how exactly the network is laid out. This is extremely important when making decisions to troubleshoot problems and add new hardware to an existing network.

2.1 NETWORK TOPOLOGIES

Topology refers to the shape of a network, or the network's layout. How different nodes in a network are connected to each other and how they communicate is determined by the network's topology. Topologies are either physical or logical. There are multiple types of physical and logical network topologies.

2.1.1 Mesh Topology

Devices are connected with many redundant interconnections between network nodes. In a true mesh topology every node has a connection to every other node in the network.

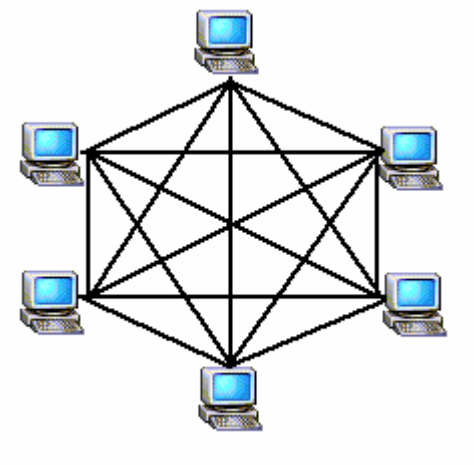

Figure 9 : Mesh Topology

2.1.2 Star Topology

All devices are connected to a central hub/switch. Nodes communicate across the network by passing data through the hub/switch.

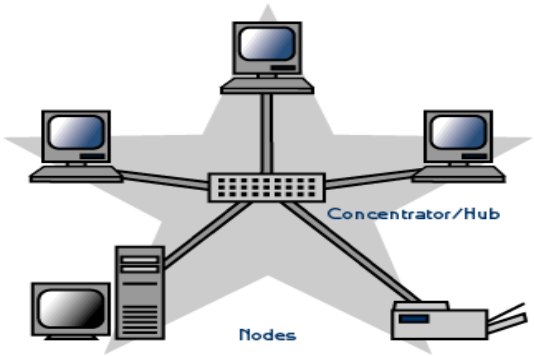

Figure 10 : Start Topology

2.1.3 Bus Topology

All devices are connected to a central cable, called the bus or backbone.

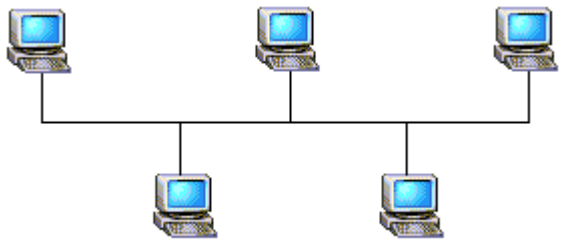

Figure 11 : Bus Topology

2.1.4 Ring Topology

All devices are connected to one another in the shape of a closed loop, so that each device is connected directly to two other devices, one on either side of it.

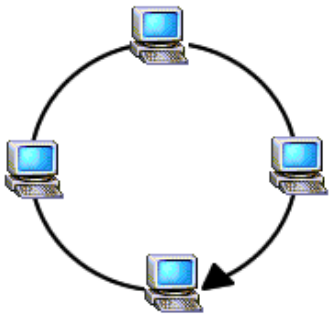

Figure 12 : Ring Topology

2.1.5 Tree Topology

Tree topology is a hybrid topology. Groups of star-configured networks are connected to a linear bus backbone.

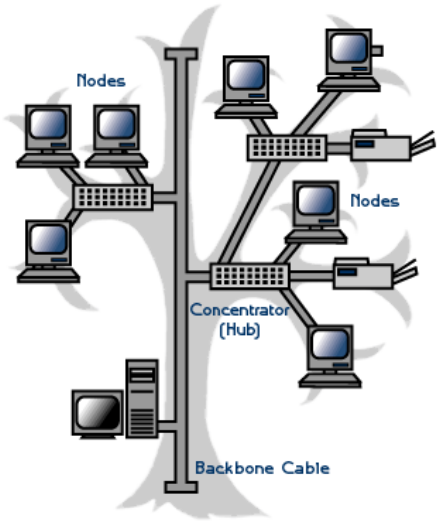

Figure 13 : Tree Topology

2.2 Network Topology Discovery

Active Network Topology discovery is a mechanism to keep track of the status of network resources and their interconnections.

2.2.1 Benefits of Network Topology Discovery

The knowledge of the real-time topology of a network is crucial for management tasks such as resource management, resource discovery, congestion avoidance, server siting, failure detection and correction as well as network state visualization and analysis.

Resource management: For any resource management software the information of the topology of the network is the most important thing.

Resource Discovery: An autonomic algorithm for topology discovery can help in discovering newly added resources in a network.

Congestion avoidance: An overall real-time picture of the network resources can help the system administrator to plan the network in a way that the traffic is distributed and the congestion is minimized.

Fault Detection and Correction: The network admin can visualize the links or servers that are down and can take corrective actions.

Siting: A topological map helps a user to know their current position in the network and to site the servers and important network machines to minimize latency.

Network State Visualization and Analysis: The discovered information gives the snapshot of the current network state and it is really helpful in analysis tasks such as resource utilization etc.

2.2.2 Characteristics of a Topology Discovery Algorithm

The prime objective of any network topology discovery solution is to develop algorithms with the following characteristics.

1 **Fast** – The Algorithms should be fast enough to provide near realtime mapping of the topology since if more time is consumed the data becomes obsolete.

2 **Complete** - to be useful, the algorithm should correctly discover the majority of the hosts and routers within a particular network, with a minimum of errors.

3 **Accurate** – The algorithm should produce minimal errors.

4 **Efficient** – The algorithm should not be resource intensive and should have least possible overhead on the network resources.

2.3 Related Works

There is a previous implementation available for the project termed as Constella Gold which could discover the topology the Network management station (NMS)'s subnet for a single interface of a single router.

The software could not be used if we needed to discover the topology of the network beyond one router or beyond one interface of the connected router. So in order to enable the software multi-subnet discovery we need to cater many major issues like catering for the transparency of layer 2 devices beyond an interface, automatic detection of routers and interfaces in the network.

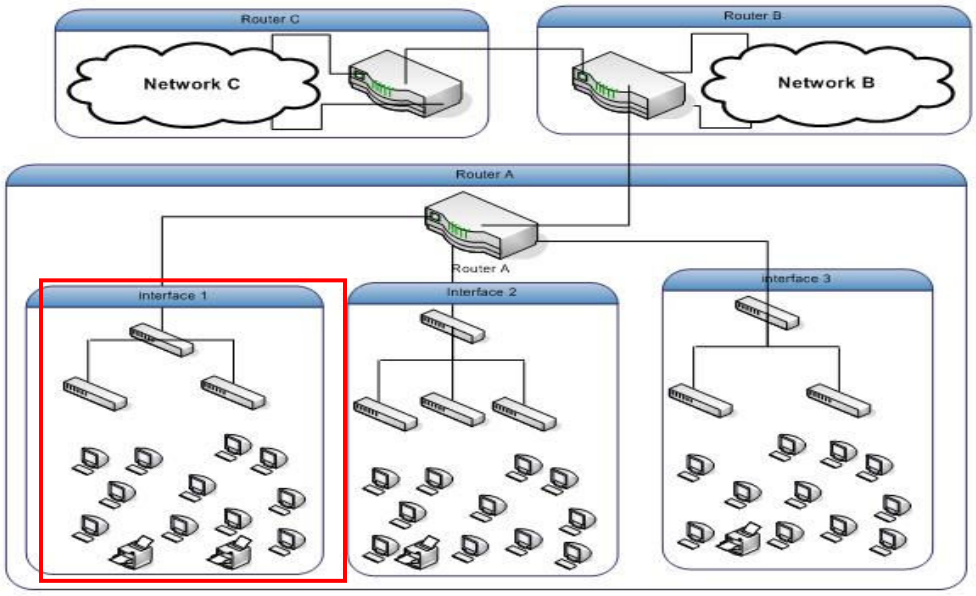

Figure 14: Single Interface discovery of Constella Gold in a sample network

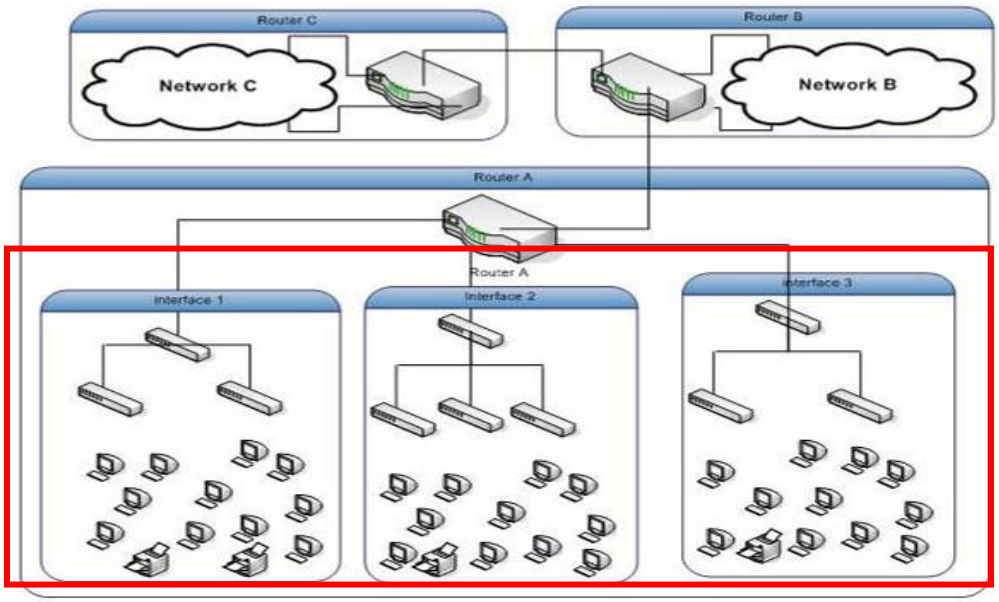

Figure 15: Single Router Multi Interface discovery not supported by Constella Gold

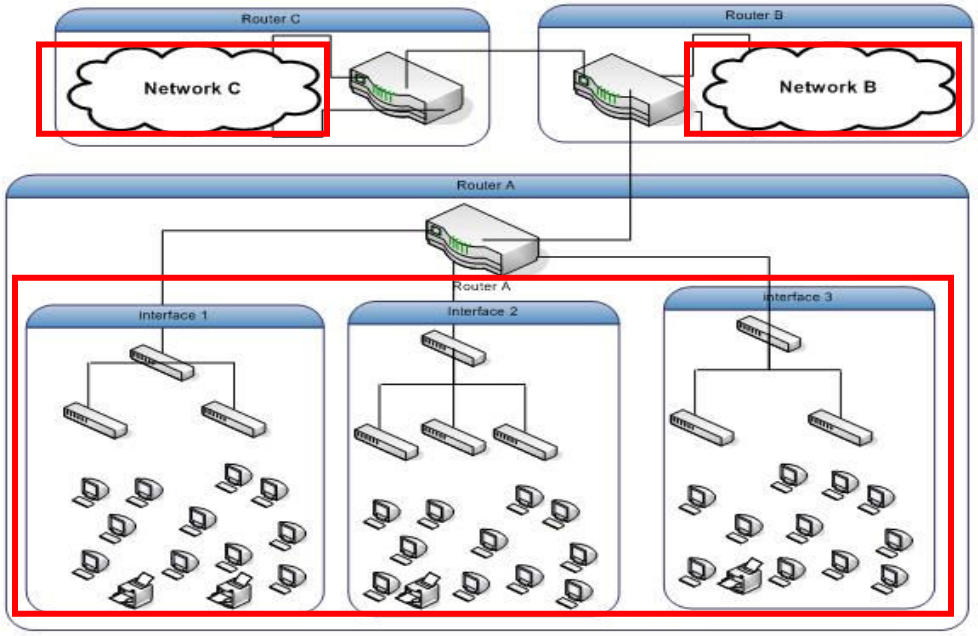

Figure 16: Multi Routers Multi Interface Discovery not supported by Constella Gold

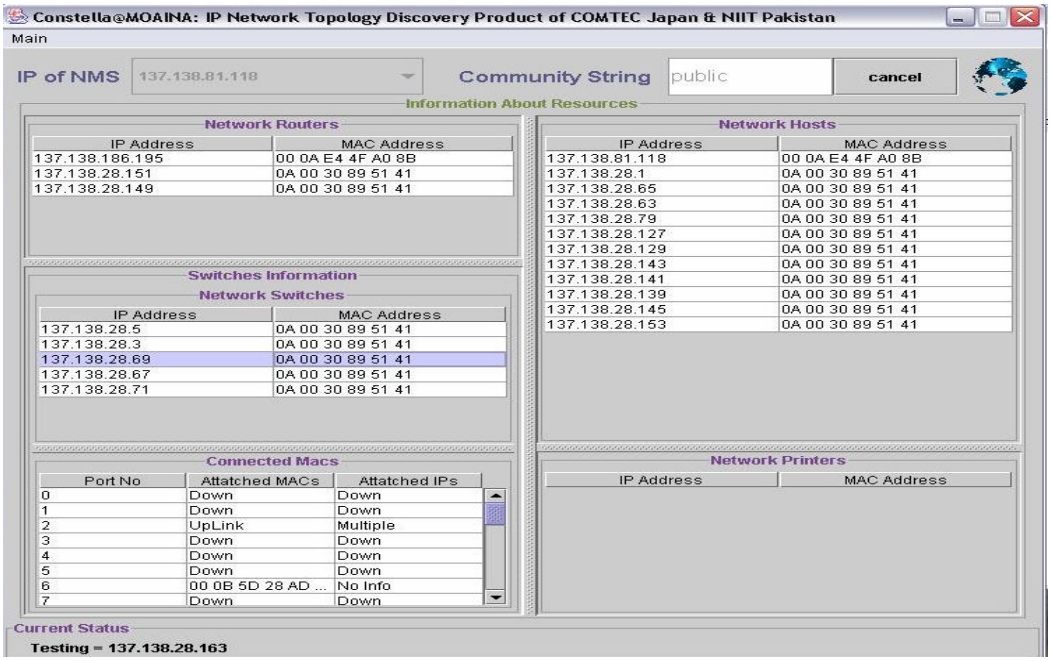

Figure 17: Snapshot of Constella Gold

2.3.1 Comparison with Constella Gold

No	Constella Gold	Constella Platinum
1.	*The software could only discover a single interface of a single router so topology discovery in most cases was not complete*	*The software can discover Multiple interfaces of multiple routers in a network. So topology information produced is complete.*
2.	This was an academic project where we had to just give a proof of concept.	This is an industrial project where we have to consider many user related issues, network efficiency etc which will involve a lot of testing.
3.	Only switch to node connections were found in that software.	Implementation of Router to Router, Router to Switch, Switch to Switch connections will also be done.
4.	Separate versions of windows and Linux.	Platform independent software.
5.	Probing is being done on the basis of Ping Utility which is slow and is filtered and blocked by many routers.	Probing being done using a combination multithreaded Ping and TPing (TCP based Ping) which uses concurrent probing and is not blocked by typical firewalls
6.	Not boundary aware discovery	The discovery can go up to the limit of the

	and discovery restricted to a single interface of the router to which the NMS is connected.	network no matter how big it is. And the user can specify the boundary by restricting subnets that are automatically detected for the user. The user can also specify the boundary conditions on the basis of no of Hops.
7.	The discovery algorithm was based on class full discovery.	The discovery algorithm is based on CIDR based classless discovery.
8.	Interconnection between switch and host are present only	A complete interconnection between router-to- router, router-to-switch, switch-to-switch, and switch-to-host is been implemented
9.	No standard output format was used for the output of the hierarchy mapping and interconnection.	XML standard output file will be use which is scalable and reusable in sense that I will also be used with other tools.
10.	The Hierarchy module was generating trees as a result of discovery.	The hierarchy output will be changed to standard XML format which can easily be integrated to the visualization module.
11.	The network could not be visualized	Constella Visualization has been implemented.

Table 1 : Comparison with Constella Gold

Chapter 3

SYSTEM ARCHITECTURE AND DESIGN

The project objective was to build a Software tool that can autonomously discover the active topology of the network with an algorithm that is fast, as well as efficient without any security concerns and with less dependency on Simple Network Management Protocol (SNMP).

The name Constella is taken from the concept of constellation of stars on the sky. The networks today are based on the same concept where each node acts like a star and each link acts like a path that the stars show us. We can see only few stars at a time on the sky likewise in the network we can retrieve only the active topology.

In today's dynamic and growing world need for Information Technology is increasing as well as the need of computers and there interconnection. This trend has increased the span of Local Area Network which is still increasing. It becomes a big task for the network administrator to keep all records of network manually. A bigger problem may arise when they have to debug a problem or to achieve scalability.

Constella is the automated tool that can discover whole of the physical network and there interconnection on a single click. It can help network administrator in monitoring and analyzing the existing network. They will get the exact snap shot of whole network by utilizing the information provided by "Constella" they can identify

where the actual problem lies. It can also help is load balancing by equally distributing the nodes on one side of the network to the other side. In other words “Constella” will change approach towards the network administration.

Constella Platinum is network topology discovery software which can autonomously detect the alive topology of the network and the interconnections of the network nodes and can also visualize the network.

3.1 System Architecture

The overall system consists of three modules.

Discovery Module

This module deals with the discovery of network resources in multi-subnet environments. The algorithm first discovers all the routers available in the network with an optional boundary based on the boundary conditions.

Hierarchy Module

This module deals with the device types and the interconnections of the discovered resources.

Visualization Module

This module visualizes the discovered data.

The overall architecture of the project is as follows.

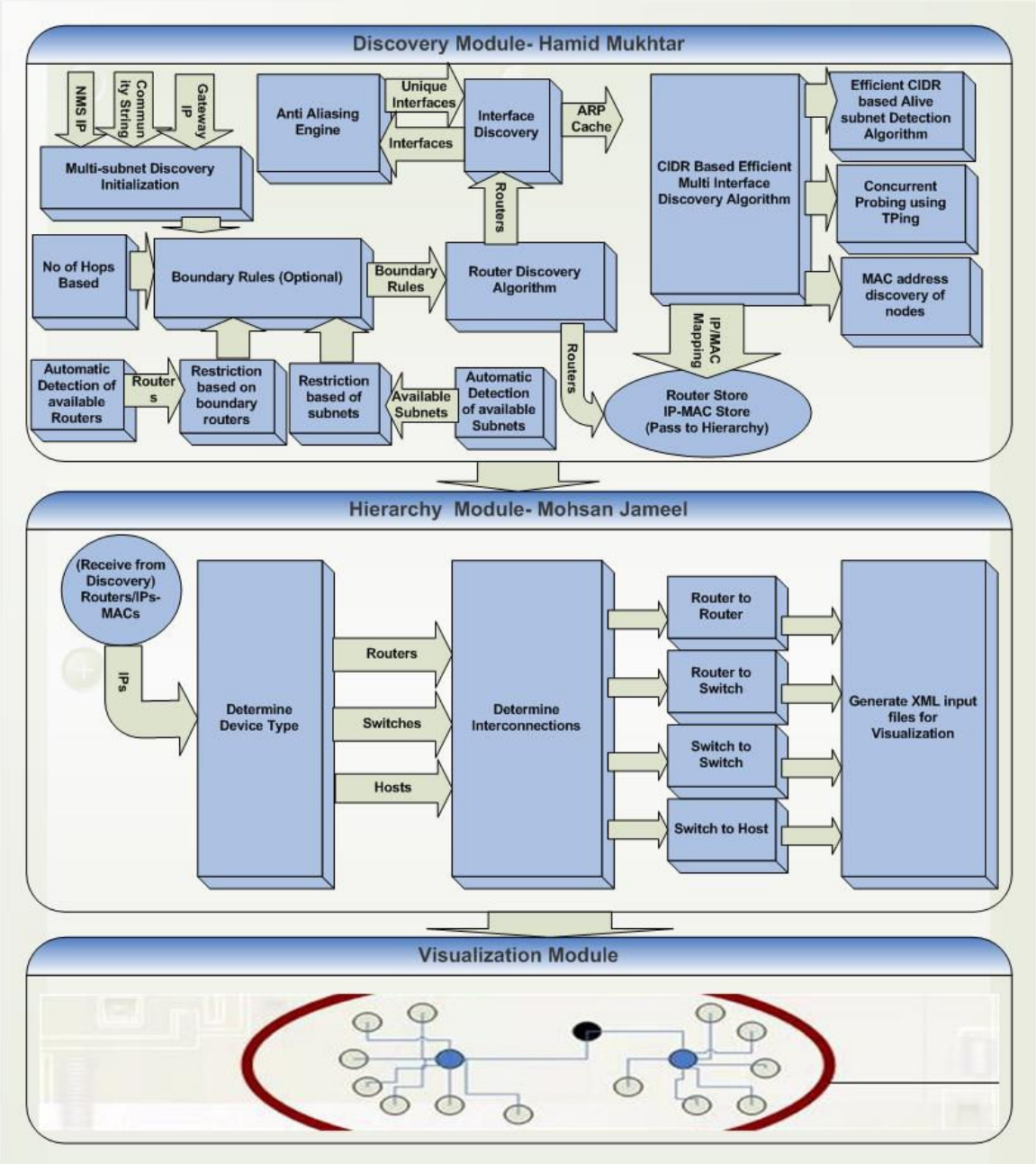

Figure 18: Constella Platinum Architecture

3.1.1 Constella Platinum Discovery Module

The Constella Platinum discovery module starts with automatically detecting the Network Management Station (NMS)'s IP address and the gateway's IP address. The user is allowed to set an optional boundary rules on the basis of subnets, the boundary and stub router combination or the number of hops specified. The boundary rules serve as an input to the Router discovery algorithm which then detects the routers within the conditions specified by the boundary rules specified.

After we get all the available routers we get their interfaces and perform Anti-aliasing to remove redundant routers from the list. At this stage we have all the unique routers and their interfaces present in the network. After this we get the ARP cache of each interface to apply the efficient subnet guessing algorithm on it. The subnet guessing algorithm first checks the addresses ranges generated on the basis of the ARP cache entries which are highly probable of being alive and then applies the guessing algorithm on other subnet present in the network so that the discovery can be complete and accurate.

We use concurrent TCP Ping (TPing) based probing to discover the network very rapidly with low blockage unlike ICMP Ping and our algorithm is able to detect the MAC addresses of the devices beyond the connected router's interface which are transparent in the network.

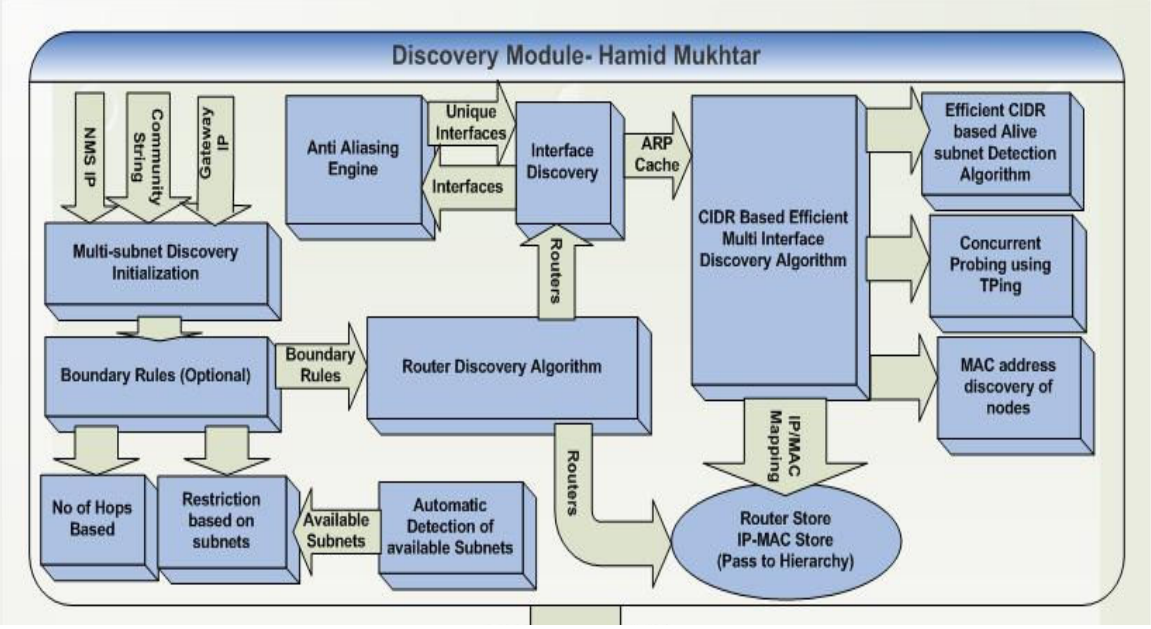

Figure 19: Constella Platinum Discovery Module Architecture

The details of salient features are as follows the whole algorithm will be explained in the algorithm section.

3.1.2 Network Boundary Awareness

Most of today's network topology discovery algorithms discover network topology in a best effort fashion. In case of using multiple network management systems for the distributed management of networks, the discovered topology performed in each network management system may include just parts of networks in interest or lots of redundant networks and nodes beyond the management scope. To facilitate distributed network topology discovery there is need to develop a boundary-aware network topology discovery algorithm, which can find networks within a

predefined management scope. There are many heuristic boundary rules that can be proposed to help determine the boundary of the network topology during discovery.

Effective network management and performance calls for efficient algorithms discovering network topology automatically, and also mandate for knowing the boundaries of the network. A network topology discovery algorithms need to be made aware of the boundary of the network or it will start discovering resources of the network beyond the required network. The boundary detection cannot be made automated since networks evolved haphazardly without any standard followed so it is nearly impossible to be able to have a fully autonomic algorithm for boundary awareness. For boundary awareness the possible and the most trivial solution is to ask the user to give the list of all the allowed IP address ranges or all the restricted IP address ranges. In both the cases automation cannot be ensured.

In the former case as the network grows the user of the software has to remember all the IP addresses in the network which at a stage becomes difficult for the user to provide remember all these address. Secondly if the user is being asked to put in all the possible IP address ranges then there is no need for automated software for discovery of resources since to use that software the user should first have a map of the entire network So there is no need for the automated discovery if the network map is required as a prerequisite. In the latter case it is impossible to give a list of all possible address ranges that are to be restricted since then if a new subnet is added in the network the user needs to be aware of that and there is no need for automation when user awareness of the network beforehand is a pre-requisite. Other solutions can be to

specify all the router IP addresses that are allowed or are to be restricted in the configuration files but these also needs information about the network from the user and thus are not suitable for the autonomic management and scalability issues.

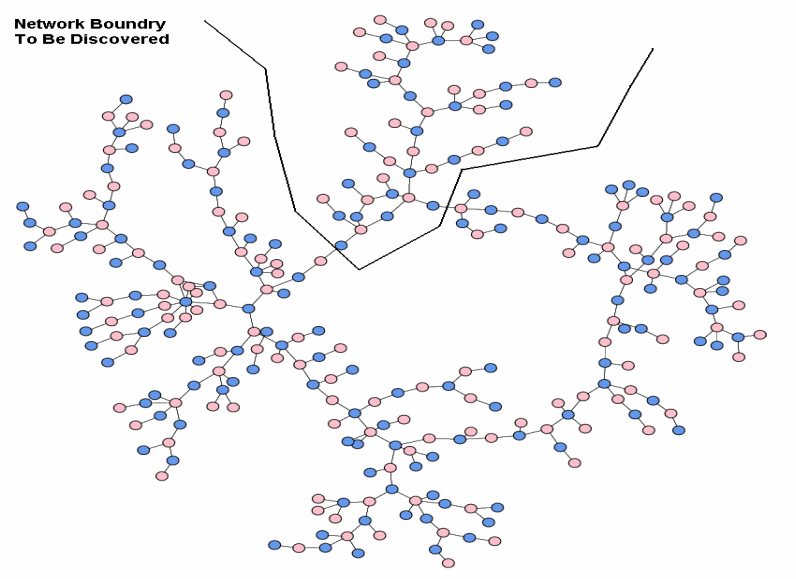

Figure 20: Boundary aware Topology discovery

As automation cannot be ensured due to lack of standards for network evolution a semi automated solution should also work which can minimize the human interaction and can ensure scalability at the same time. Therefore for boundary awareness we add two techniques.

3.1.2.1 Boundary on the basis of automatically detected subnets

First approach is to specify the network IP address and network masks of the allowed subnets in the configuration files but as this approach decreases the autonomy of the algorithm so to make it a more autonomous we first detect all the available

'subnets' in the network and give the user an option to allow or restrict them. In this way the network changes are automatically detected and there is no manual entry at the user's part and he/she just has to identify which subnets are not a part of their network and he can easily restrict them.

3.1.2.2 Boundary on the basis of Boundary and Stub Routers

In the second approach we can first detect all the available routers in the network and then the user has an option of selecting the boundary and the stub routers from the list of available routers. The discovery will not continue beyond these routers. To make it more user friendly an approach can be to discover and visualize and the routers and connections in the network and giving the user an option to select the boundary and the stub routers for the network to restrict the topology discovery within these routers so that other networks are not interfere. The discovery algorithm is restricted to till the boundary routers so it doesn't cross the boundary router and is also restricted till the stub router.

3.1.2.3 Boundary on the basis of No of Hops

In this criterion we will give user an option to specify the boundary of the network on the basis of no of hops which is a universally accepted and well known term for specifying boundary for networks. But the for boundary awareness it is not the best criteria since the results will change if we change our position within the same network. But still if the user does not want to check more devices on the basis of increasing distance he/she can limit the boundary on the basis of no of hops.

The user will specify the no of hops he wants to discover and the topology discovery will be restricted to the specified no of hops.

3.1.3 Unique Router Discovery using anti aliasing

This feature separates Constella Platinum from some other algorithms like the algorithm proposed by Cornell Research group [5]. In a network, routers may have multiple interfaces. The Cornell algorithm simply counts each interface as another router in the network. Hence the resulting map will have more than one node for the same router. In Constella Platinum we have introduced a concept of Anti-Aliasing to remove redundant router entries from the router discovery algorithm and for this we introduced a concept of Anti-aliasing. After the router discovery we detect the interfaces of each router in the Router Store. If any two nodes in the Router Store have the same list of interface IP addresses we merge the two Routers into a single router and thus the resulting map has unique routers. The router discovery algorithm is explained in detail in the algorithm section.

For discovery of routers we first get the gateway routers of the network management station (NMS) i.e. the machine on which the software is running and then get its next hop entries recursively till all the end point have SNMP disabled on them or their community string is unknown or the boundary conditions set by the users are reached

3.1.4 CIDR based discovery

Typical topology discovery solutions were made to handle the class full addressing schemes and are incompatible with the new CIDR based addressing scheme. Constella Platinum can handle addresses from both schemes and we have upgraded our software to use ipCidrRouteTable and ipRouterTable in the SNMP MIB II to make our software work in both the conditions.

3.1.5 Discovery of Transparent MAC addresses

Constella Platinum software can discover the MAC addresses of nodes in a multi-subnet environment which are transparent to the network. Beyond one router interface the MAC addresses of machines are transparent to the NMS because the MAC addresses are layer 2 addresses. So in order to get those MAC address Constella Platinum queries the router's interfaces after the probing and get MAC address entries from the routers.

3.1.6 TCP PING

TPing stands for TCP Ping (or Turbo Ping). As name suggests, it uses TCP instead of ICMP (that is used by ping). The requirement of accuracy for a topology discovery can be achieved by using TPing which is not a feature of many network management softwares and it separates Constella Platinum from other softwares. TPing is an opensource module available for Linux and we have developed our own Windows version of TPing i.e. **WinTPing** to make our software platform independent.

TPing can concurrently ping more than one host at a time. TPing sends a TCP SYN packet to a target host at some port and if there is any reply from the user i.e. port unreachable or the connection ACK packet we can conclude that the host was alive. Due to its innovative technique the ping is never blocked since the network administrators since they usually block ICMP packets and connection to open ports are always allowed and for closed ports the networks send Port Unreachable message. It probes one host and moves to the next one in the list in a round robin fashion. ICMP packets are filtered and blocked at a lot of routers and sites. For example, www.cnn.com won't reply to your ICMP probes. As TPing is based on TCP, its probes will not be blocked for the reason explained above. This would contribute a lot performance wise. In our earlier implementation of traditional ping we used to ping a pool of IP addresses one by one. We would have to suffer from the inherent delay of traditional ping while pinging such a host that is not alive. On the other hand, TPing does not wait for a host which is not alive, it moves on the next host to discover its status. This renders a great efficiency to our Topology Discovery Algorithm, Constella Platinum. We have modified the TPing to ping up to 255 hosts concurrently.

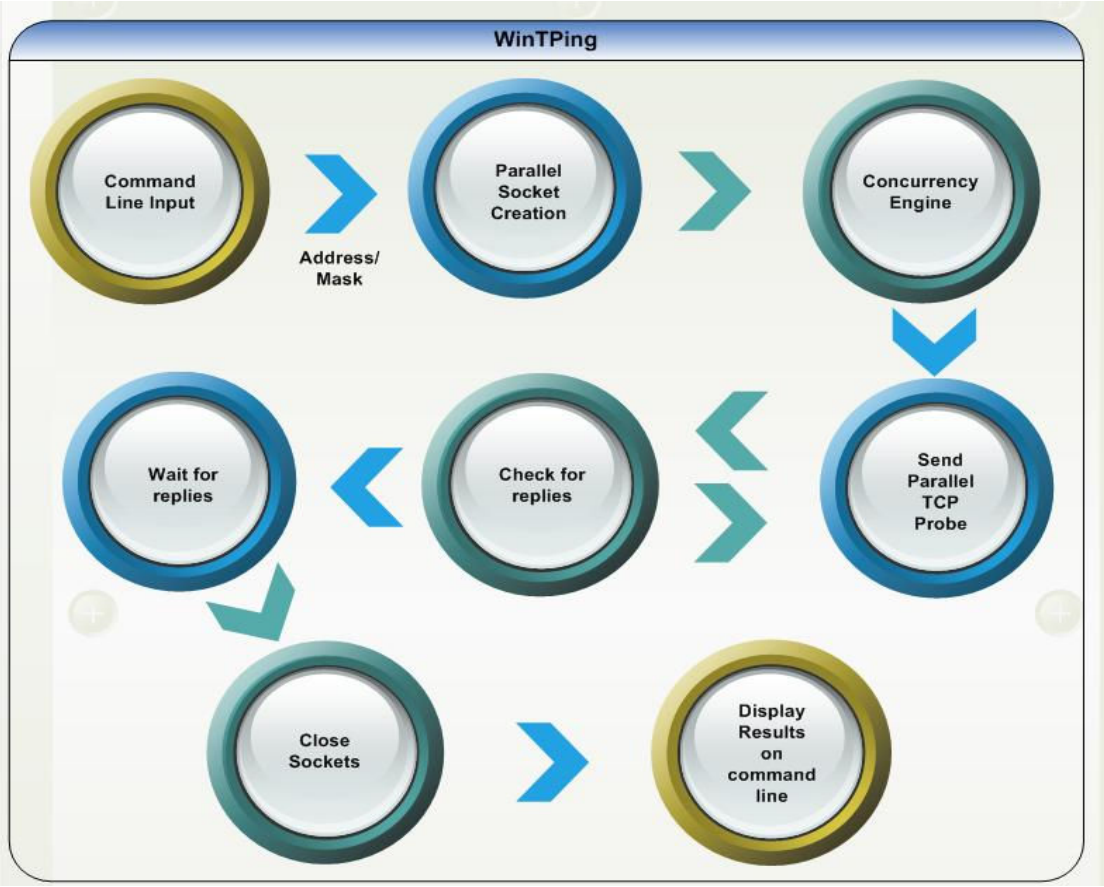

Figure 21 Win TPing Architecture

3.2 Semantics for Network Topology Discovery Data

Topology discovery is a mechanism to keep track of the status of the resources and how they are connected in a network. For typical topology discovery solutions the data is produced in semi-structured format. We proposed a semantic based approach to define elements in the network. To enable this semantic enrichment the data needs to be exported in a structured format. By defining shared and common vocabularies,

ontologies help both people and machines to communicate concisely, supporting the exchange of semantics instead of syntax.

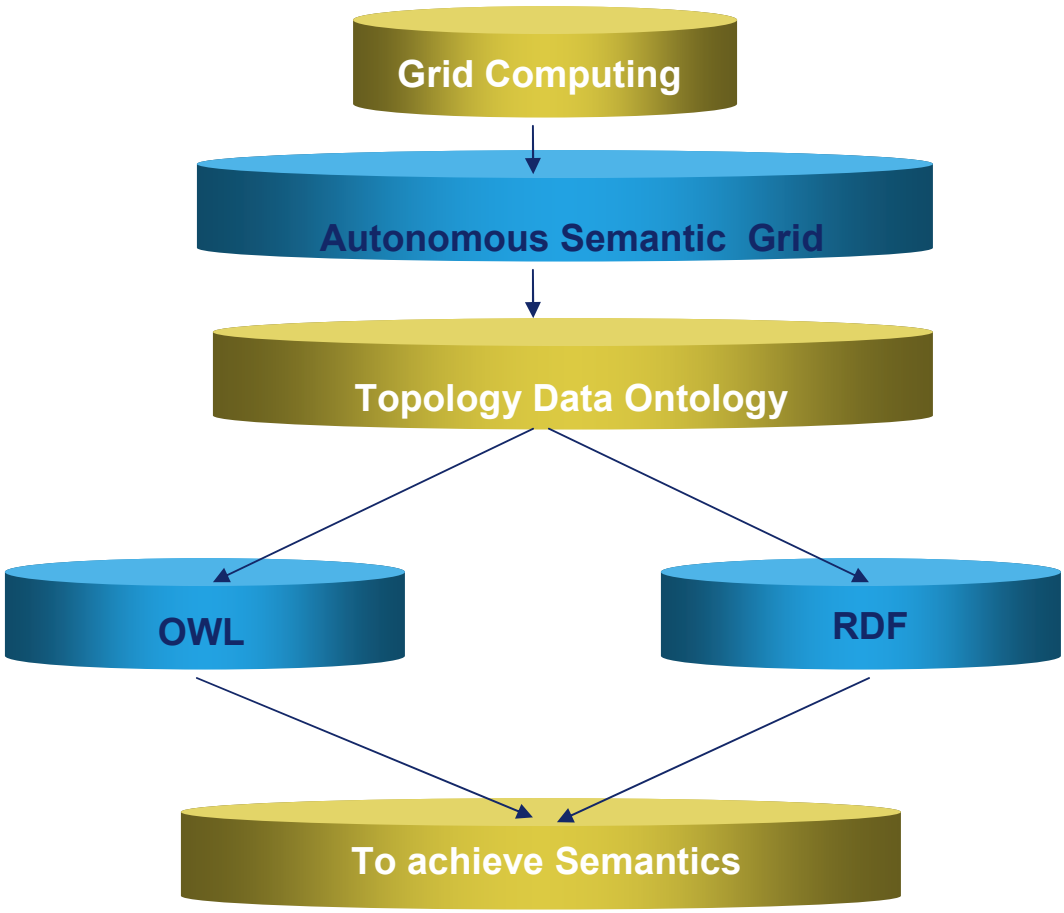

Figure 22: Relationship with Semantic Grid

We are using Resource Description Format (RDF) which is formal data model from the World Wide Web Consortium (W3C) for machine understandable metadata used to provide standard descriptions of web resources. This approach will reduce the human intervention for the information retrieval about a particular network so that it can be further used by FIPA compliant Multi-agent systems to provide an agent understandable format where agents will be acting on behalf of the user for

automatically discovering and querying the topology ontologies and retrieving the required data. The ontologies will provide the intelligence to the agents which will carry out intelligent tasks on behalf of the user.

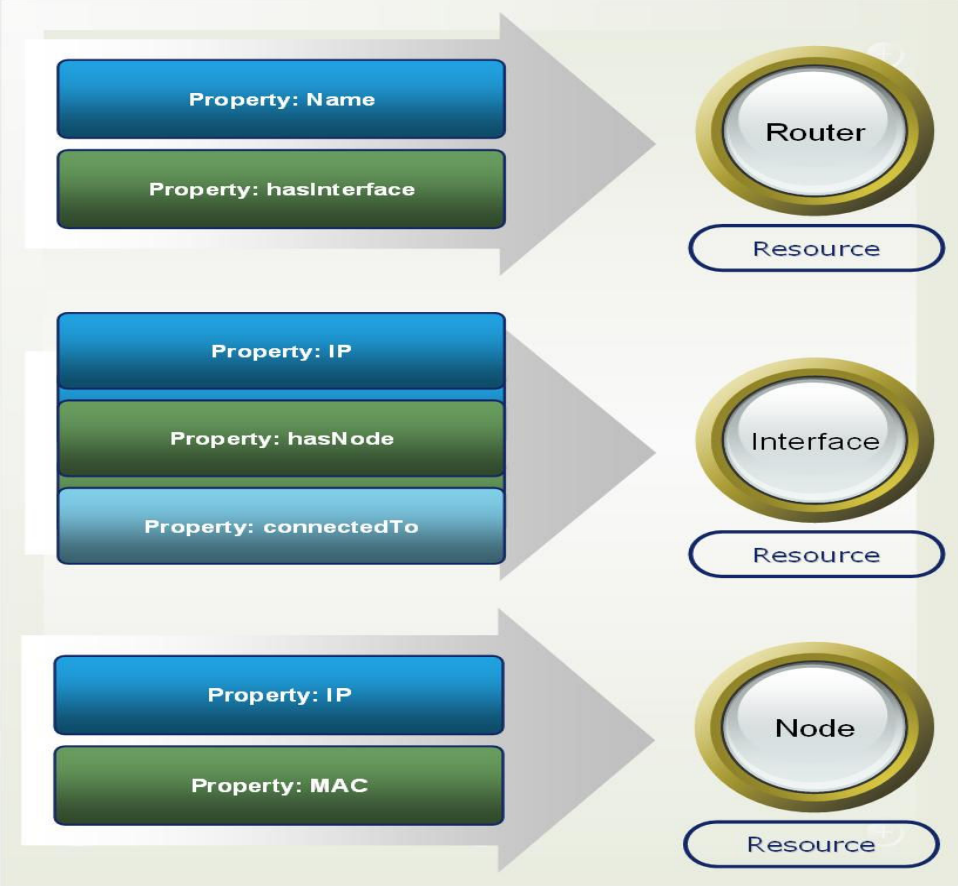

Figure 23: Network Topology Ontology Architecture

3.2.1 Advantages of RDF data Model

- The RDF model is made up of triples: as such, it can be efficiently implemented and stored; other models requiring variable-length fields would require a more cumbersome implementation.

- The RDF model is essentially the canonicalization of a (directed) graph, and so as such has all the advantages (and generality) of structuring information using graphs
- The basic RDF model can be processed even in absence of more detailed information (an "RDF schema") on the semantics: it already allows basic inferences to take place, since it can be logically seen as a *fact basis*
- The RDF model has the important property of being *modular*: the union of knowledge (directed graphs) is mapped into the union of the corresponding RDF structures; this means that:
    - information processing can be *fully parallelized*
    - in presence of *partial information* (an essential feature in a volatile environment like the web) the output is still a consistent RDF model, that can be succesfully processed

- RDF syntax is layered: the basic serialization syntax allows for quite a powerful encoding, while still being compact (@@this can be further elaborated, even in a mathematical sense).

3.3 System Design

3.3.1 Object Oriented Methodology

Object orientation makes it easy to simulate the task distribution among various modules, by the use of objects that can interact with each other and share & exchange information like the way it is done in real life. Therefore, the design is more realistic.

Thinking of objects doing their jobs and interacting with each other helps in understanding and developing the system more comprehensively.

The OOD approach also supports modularity, which in turn is required for reusability. Similarly, the concept of polymorphism simplifies the design to a great extent when a higher level of abstraction is to be presented to the user. This helps in hiding the underlying details.

3.3.2 Project Management

Due to the research oriented nature of the project, the project was managed as per the following task schedule:

- Research on concept evolution
- Development of Module
- System architecture and design
- System development
- Module integration
- System Testing

The most influential risk factor was time, as time to be spent on the research part could not be estimated in absolute values. To manage this situation, initial time allocation for the research part was more than that of the other modules. Apart from research, special attention was paid to the system architecture and implementation.

3.3.3 Class Diagrams

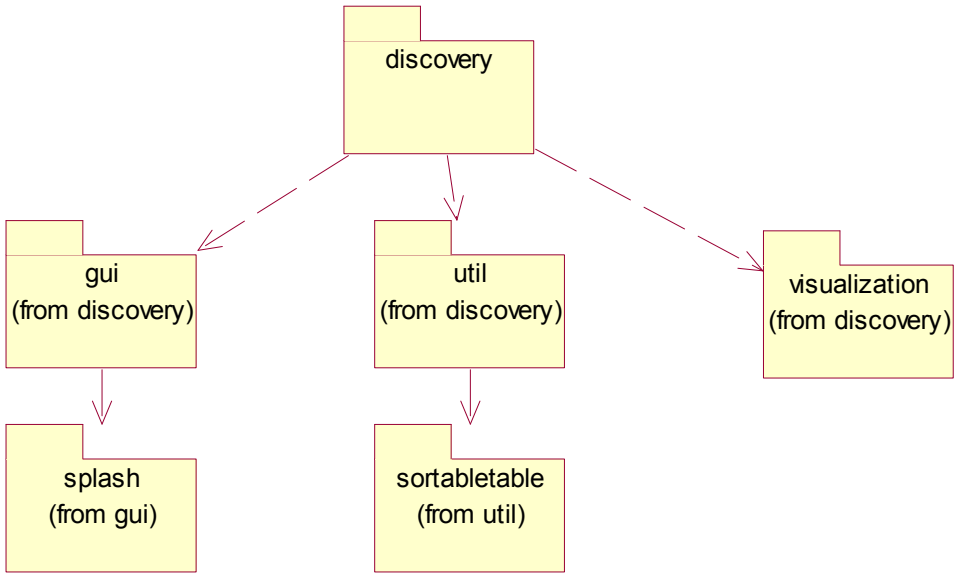

Figure 24: Constella Platinum Package Hierarchy

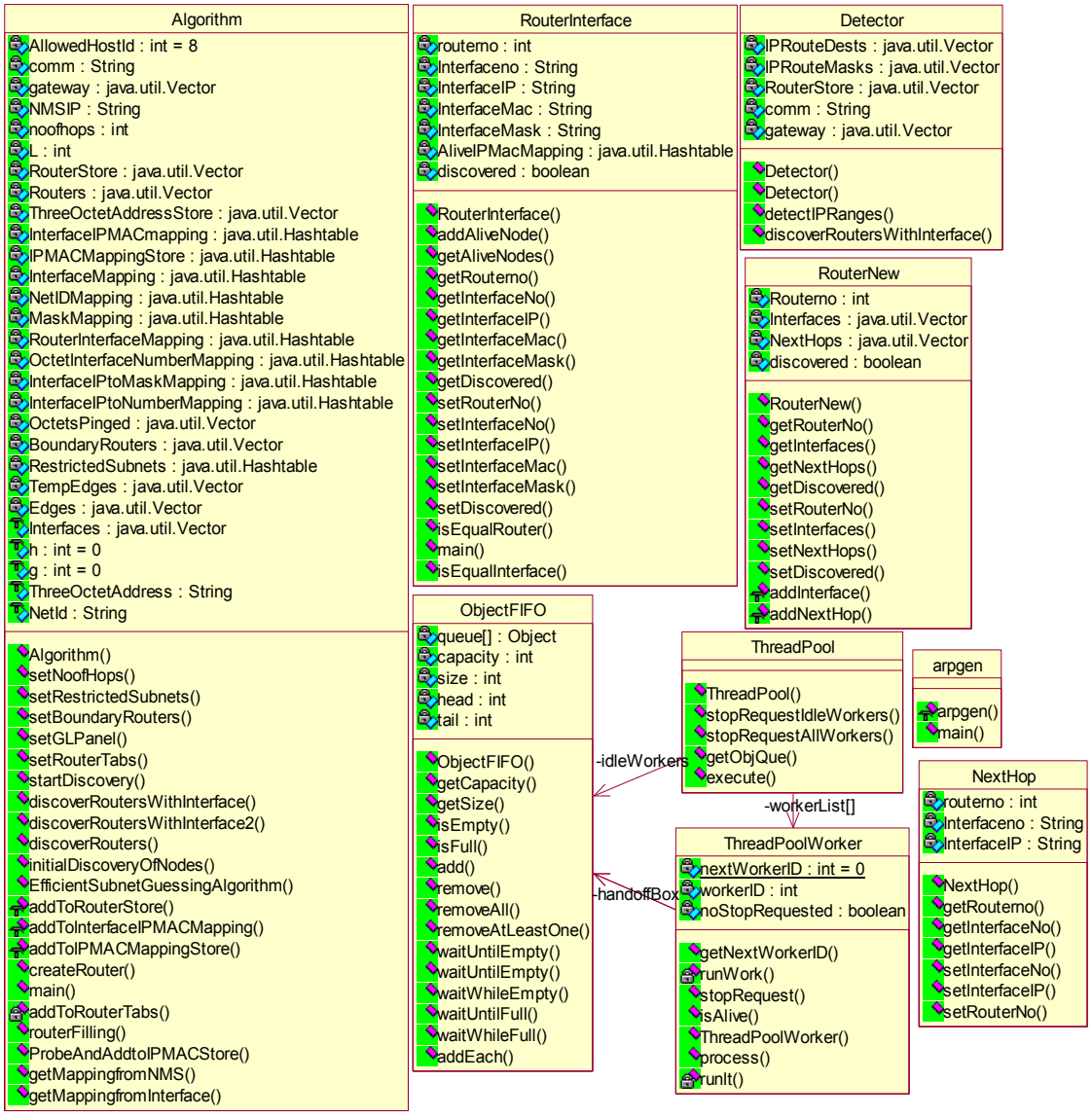

Figure 25: Class Diagram of Main Discovery Package

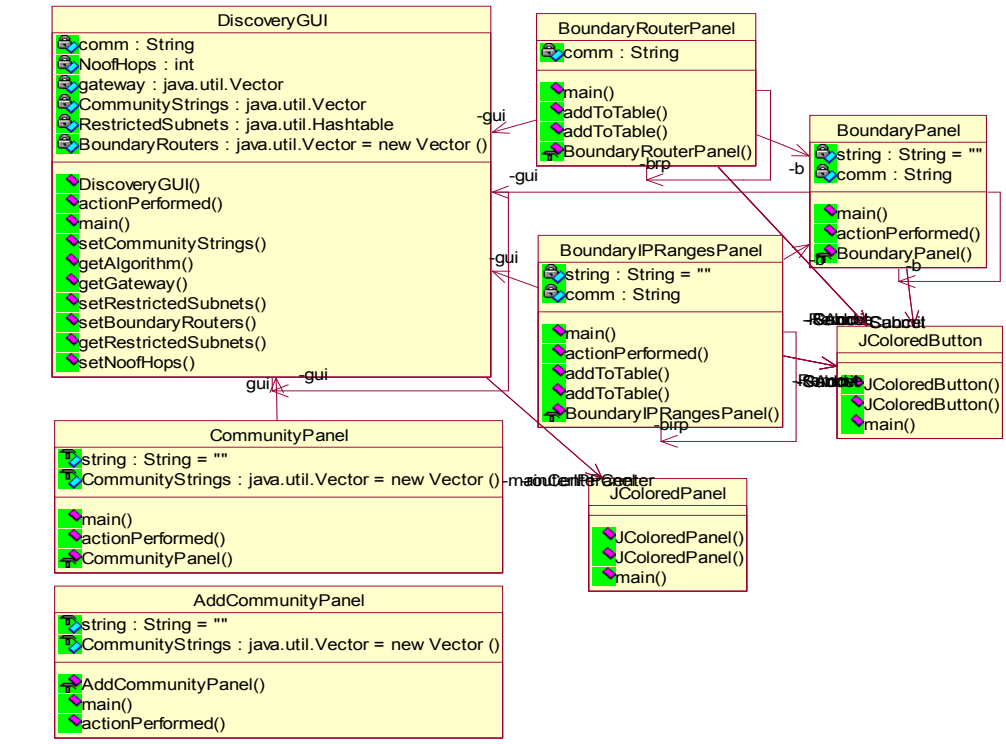

Figure 26: GUI Package (1/2)

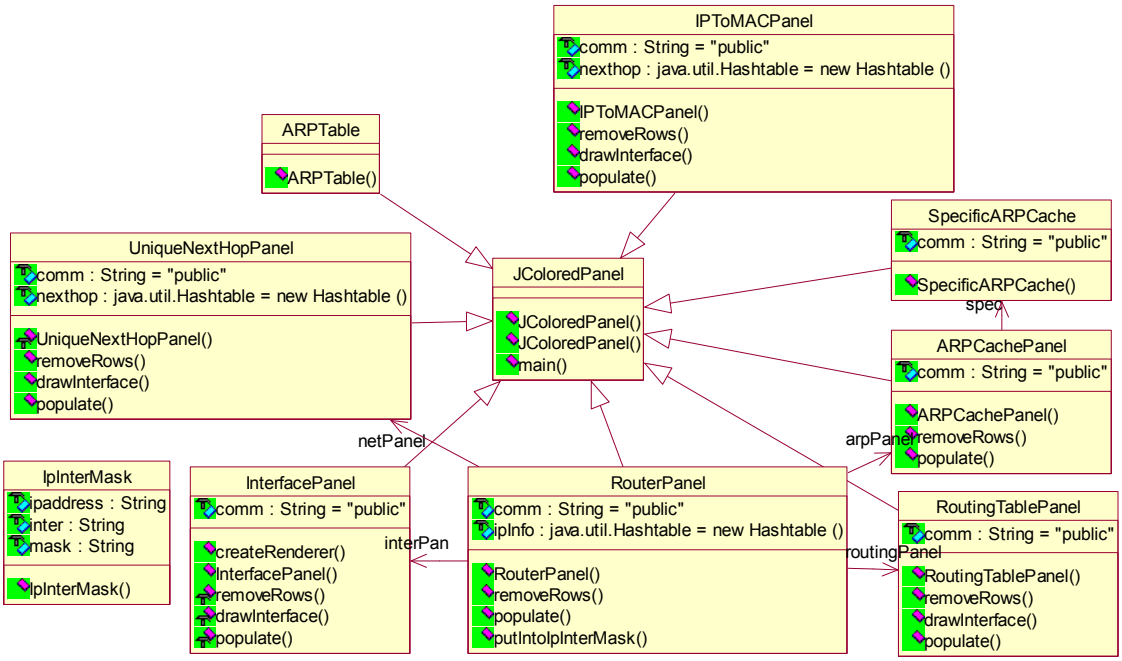

Figure 27: GUI Package (2/2)

Figure 28: Util Package

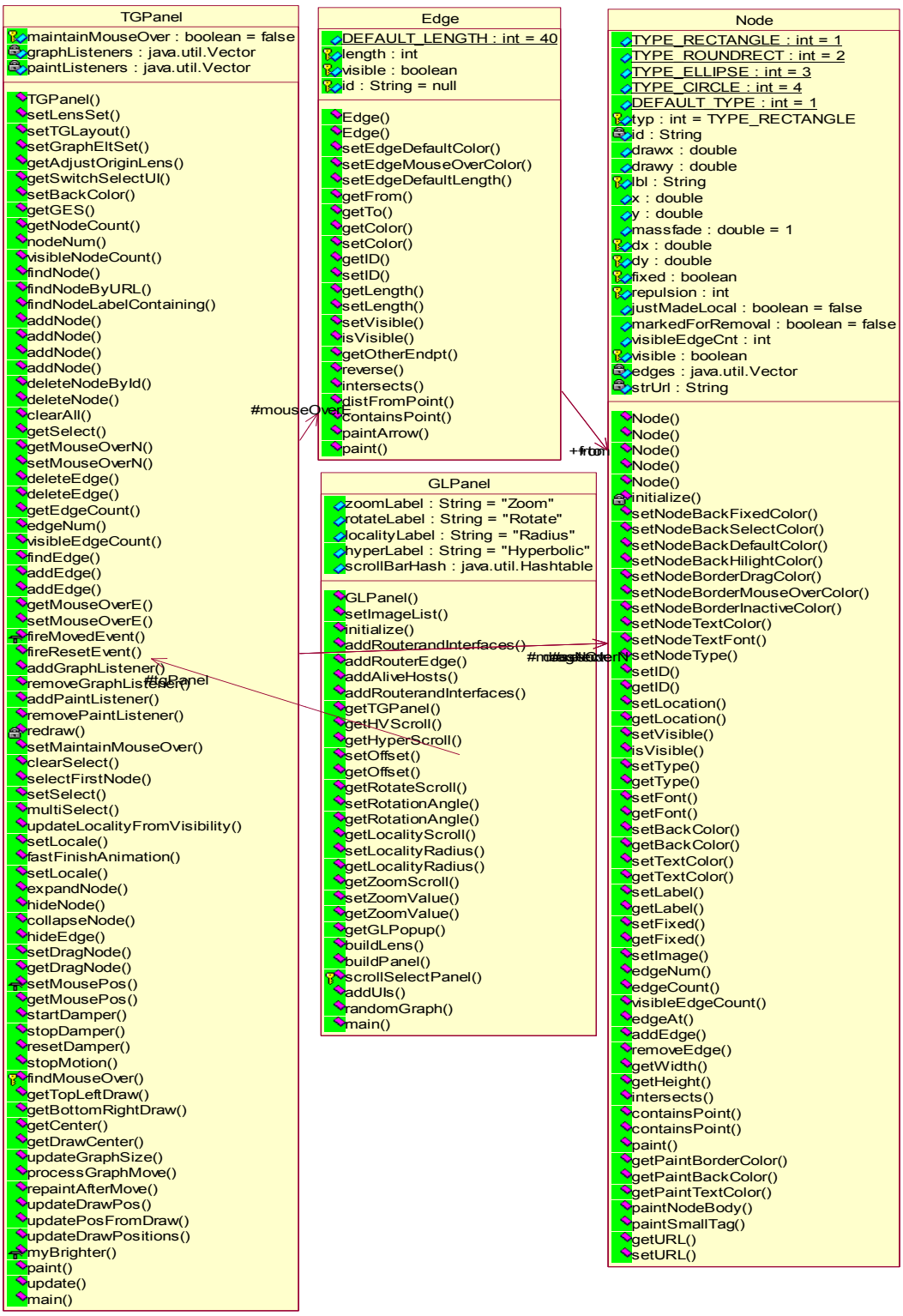
TGPanel
maintainMouseOver : boolean = false
graphListeners : java.util.Vector
paintListeners : java.util.Vector
TGPanel()
setLensSet()
setTGLayout()
setGraphEltSet()
getAdjustOriginLens()
getSwitchSelectUI()
setBackColor()
getGES()
getNodeCount()
nodeNum()
visibleNodeCount()
findNode()
findNodeByURL()
findNodeLabelContaining()
addNode()
addNode()
addNode()
addNode()
deleteNodeById()
deleteNode()
clearAll()
getSelect()
getMouseOverN()
setMouseOverN()
deleteEdge()
deleteEdge()
getEdgeCount()
edgeNum()
visibleEdgeCount()
findEdge()
addEdge()
addEdge()
getMouseOverE()
setMouseOverE()
fireMovedEvent()
fireResetEvent()
addGraphListener()
removeGraphListener()
addPaintListener()
removePaintListener()
redraw()
setMaintainMouseOver()
clearSelect()
selectFirstNode()
setSelect()
multiSelect()
updateLocalityFromVisibility()
setLocale()
fastFinishAnimation()
setLocale()
expandNode()
hideNode()
collapseNode()
hideEdge()
setDragNode()
getDragNode()
setMousePos()
getMousePos()
startDamper()
stopDamper()
resetDamper()
stopMotion()
findMouseOver()
getTopLeftDraw()
getBottomRightDraw()
getCenter()
getDrawCenter()
updateGraphSize()
processGraphMove()
repaintAfterMove()
updateDrawPos()
updatePosFromDraw()
updateDrawPositions()
myBrighter()
paint()
update()
main()
#mouseOverE
Edge
DEFAULT_LENGTH : int = 40
length : int
visible : boolean
id : String = null
Edge()
Edge()
setEdgeDefaultColor()
setEdgeMouseOverColor()
setEdgeDefaultLength()
getFrom()
getTo()
getColor()
setColor()
getID()
setID()
getLength()
setLength()
setVisible()
isVisible()
getOtherEndpt()
reverse()
intersects()
distFromPoint()
containsPoint()
paintArrow()
paint()
+from
GLPanel
zoomLabel : String = "Zoom"
rotateLabel : String = "Rotate"
localityLabel : String = "Radius"
hyperLabel : String = "Hyperbolic"
scrollBarHash : java.util.Hashtable
GLPanel()
setImageList()
initialize()
addRouterandInterfaces()
addRouterEdge()
addAliveHosts()
addRouterandInterfaces()
getTGPanel()
getHVScroll()
getHyperScroll()
setOffset()
getOffset()
getRotateScroll()
setRotationAngle()
getRotationAngle()
getLocalityScroll()
setLocalityRadius()
getLocalityRadius()
getZoomScroll()
setZoomValue()
getZoomValue()
getGLPopup()
buildLens()
buildPanel()
scrollSelectPanel()
addUIs()
randomGraph()
main()
Node
TYPE_RECTANGLE : int = 1
TYPE_ROUNDRECT : int = 2
TYPE_ELLIPSE : int = 3
TYPE_CIRCLE : int = 4
DEFAULT_TYPE : int = 1
typ : int = TYPE_RECTANGLE
id : String
drawx : double
drawy : double
lbl : String
x : double
y : double
massfade : double = 1
dx : double
dy : double
fixed : boolean
repulsion : int
justMadeLocal : boolean = false
markedForRemoval : boolean = false
visibleEdgeCnt : int
visible : boolean
edges : java.util.Vector
strUrl : String
Node()
Node()
Node()
Node()
Node()
initialize()
setNodeBackFixedColor()
setNodeBackSelectColor()
setNodeBackDefaultColor()
setNodeBackHilightColor()
setNodeBorderDragColor()
setNodeBorderMouseOverColor()
setNodeBorderInactiveColor()
setNodeTextColor()
setNodeTextFont()
setNodeType()
setID()
getID()
setLocation()
getLocation()
setVisible()
isVisible()
setType()
getType()
setFont()
getFont()
setBackColor()
getBackColor()
setTextColor()
getTextColor()
setLabel()
getLabel()
setFixed()
getFixed()
setImage()
edgeNum()
edgeCount()
visibleEdgeCount()
edgeAt()
addEdge()
removeEdge()
getWidth()
getHeight()
intersects()
containsPoint()
containsPoint()
paint()
getPaintBorderColor()
getPaintBackColor()
getPaintTextColor()
paintNodeBody()
paintSmallTag()
getURL()
setURL()

Figure 29: Visualisation Package

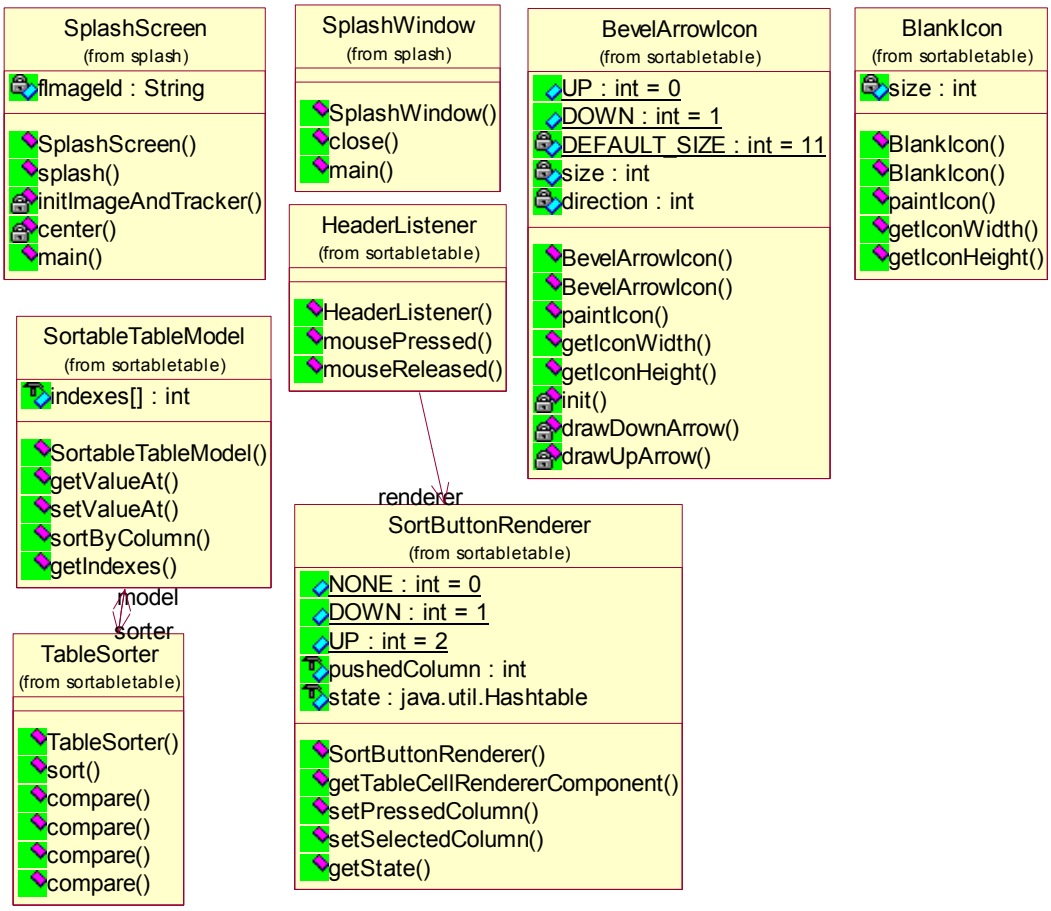

Figure 30: Other Classes

Chapter 4

ALGORITHM

Our algorithm completes the network discovery in two steps. In the first step all the available routers in the network are discovered and the user can optionally select the network boundary to restrict the discovery within the limits. We detect all the interfaces of the discovered routers and perform anti-aliasing to remove redundant routers. In the second step we get the ARP cache entries of every interface of each router and use our subnet guessing and hierarchy generation algorithms to discover the entire network topology. The algorithm detects the Network Management Station (NMS[1]) IP address and the gateway IP address to start the algorithm.

4.1 Router Discovery within the Network Boundary

For router discovery we use a router discovery algorithm similar to [5] where a Router Store (RS) is maintained. We first detect the gateway router and add it to the Router Store and then we recursively add the next hop routers to the Router Store till all end points have SNMP disabled on them or their SNMP community[2] is unknown. To restrict our algorithm from probing other subnets concept of boundary awareness needs to be introduced. Therefore for boundary awareness we add two techniques.

[1] NMS is the machine on which the software is running

[2] SNMP community is the password which is used for authentication between the management station and the SNMP agents running on machine.

First is to specify the network IP address and network masks of the allowed subnets in the configuration files but this approach decreases the autonomy of the algorithm so in order to make it a bit autonomous we first detect all the available subnets available in the network and give the user an option to allow or restrict them. The second approach can be to discover and visualize and the routers and connections in the network and giving the user an option to select the boundary and the stub routers for the network to restrict the topology discovery within these routers so that other networks are not interfere. The user can also restrict the discovery process on the basis of no of hops from the gateway router. To get the next hop entries of a router we use ipCidrRouteDest and ipCidrNextHop objects of the ipCidrRouteTable[3]. We check in the ipCidrRouteDest for the entries that are marked as next hop and pick the next hop entries from ipCidrNextHop object. The algorithm will detect all the routers within the optional boundary conditions.

Weakness of the Cornell algorithm described in [5] is that it is incapable of properly dealing with multiple interfaces on routers. In a network, routers may have multiple interfaces. The algorithm simply counts each interface as another router in the network. Hence the resulting map will have more than one node for the same router. For this a concept of Anti-aliasing needs to be introduced. After the router discovery we detect the interfaces of each router in the Router Store. If any two nodes in the

[3] Previously we used ipRouteDest and ipRouteNextHop object of the ipRouteTable but this table was obsoleted and replaced by ipForwardingTable by RFC 1351. RFC 2096 updated this table to ipCidrRouteTable

Router Store have the same list of interface IP addresses we merge the two Routers into a single router and thus the resulting map has unique routers.

For each interface of router we get information about its interface IP addresses, network masks and MAC addresses with the help of ipAdEntAddr, ipAdEntNetMask and ipAdEntIfIndex objects of the ipAddrTable of IP group and ifIndex and ifPhysAddress objects of the Interfaces group of SNMP MIB-II.

4.2 Intelligent Algorithm for generation of IP address Ranges

At this stage we have a set of available routers and their interfaces. We need to discover the alive switches, Hosts and printers connected to these routers. Sending ICMP ping request to all the possible IP addresses is not feasible in determining the devices that are alive in the network because of the large number of possible IP addresses. For example, there will be more than 16 million possible addresses for Class A networks. So we propose an intelligent and efficient algorithm for generating a list of IP addresses having a high probability of being assigned to devices in the network.

So for all the interfaces we get the network masks from which we find the number of bits reserved for assignment in the network i.e. the Host id. After extensive testing on various networks, we have concluded that using our modified T-Ping 254 or fewer nodes can be probed very efficiently. So If the number of bits for host id are less than or equal to eight, i.e. less than or equal to 254 nodes can be assigned IP addresses,

then we probe them concurrently using T-Ping. For example, if the network mask at that interface is 255.255.255.224 (255.255.255.11100000), Then we will only have to probe 2^5 = 32 nodes concurrently. We store the interface to mask mapping so that we can restrict the probing within the subnet.

If the number of bits for host id is greater than eight, then we apply our subnet guessing algorithm so that we do not have to probe all possible IP addresses within the large address space. The algorithm is as follows. We first get the list of IP addresses from the ARP cache of the interface through ipNetToMediaNetAddres object of the ipNetToMediaTable of SNMP MIB-II since ARP cache contains the IP addresses of the nodes which recently communicated with the router and thus they are highly probable to be alive. Then we convert each address in the interface ARP cache to a Three Octet Address (TOA) by converting the fourth octet to zero and add it to the Three Octet Address Store which contains the addresses with last octet as zero. For example, 11.12.13.14 in the ARP cache will be converted to 11.12.13.0. During the execution of the algorithm we store the TOA to interface IP address mapping as well as TOA to network id mapping for use in the Subnet Guessing Algorithm.

At this stage all small address spaces are checked for their status and for the interfaces involving Host ID bits greater than eight we apply the following algorithm.

4.3 Intelligent Subnet Guessing Algorithm

At this stage the Three Octet Address Store (TOAStore) for the interfaces which have large address space is filled and we apply the subnet guessing algorithm on these addresses since the small address spaces are already checked for their status. This algorithm runs iteratively. Each time an entry n.n.n.0 from the Three Octet Address store is taken after probing the address range n.n.n.0 and we add one to and subtract one from the binary value of the first 24 bits of this address. For simplicity we say that going in the right direction mean we are adding and going in the left direction means we are subtracting one from the address. For example, if we get 10.11.12.0 from the Three Octet Address store, going in the right direction means we keep adding one to the binary value of the first 24 bits of this address which yields 10.11.13.0, 10.11.14.0, 10.11.15.0 and so on. Going in the left direction means we subtract one from the binary value of the first 24 bits of the address which yields 10.11.11.0, 10.11.10.0, 10.11.9.0 and so on. We probe the last 8 bit address space of the newly formed addresses. We say that a failure has occurred if no host from the 8 bit address space sends a positive reply during probing. The addition and subtraction continues till the number of successive failures reaches a limit 'L'. For example, we kept the limit 'L' to 3 and probes to 10.11.13.0, 10.11.14.0 and 10.11.15.0 all resulted in a failure, then we will stop going right. Similarly while going left if there are 3 successive failures then we will stop subtracting. If we have stopped going left and right then we move to the next value in the Three Octet Address store and the same procedure is followed for each value in the Three Octet Address store. The algorithm makes sure

that the subtraction and addition does not cause the resultant address to go out of the IP address range for that subnet.

To remove redundancy we check if the generated address has already been probed or the address is already in the store or the generated address is outside the range of the Interface Mask. In all the three cases we stop the addition or subtraction in that direction because if an address has already been tested, then we don't have to test it again and if the address is already present in the store then it will be tested on its own turn and in the third case if the generated address space is outside the interface mask then we do not need to test that.

In the probing module we probe the entire address space that is passed as argument using T-Ping. The procedure T-Ping (address, mask) generates IP addresses within a range of the mask and then probes all the nodes concurrently. If there is no active node in the address range then it returns false. Otherwise it returns true. We add the probed subnets to a List named IP-MAC Mapping Store.

To get the link layer mapping we have two cases. If the interface is directly connected to NMS, then we can easily get the IP to MAC IP to MAC mapping of the alive nodes from the NMS ARP cache. If the interface is not directly connected to the NMS, then the IP to MAC mapping of the active nodes is acquired from that interface's ARP cache. We get the ARP cache MAC addresses through ipNetToMediaPhysAddress object of ipNetToMediaTable of SNMP MIB-II. After we

get the ARP cache mapping we add them all to the IP-MAC Mapping store. Following is the probing pseudo code. InterfaceIP is the interface IP address.

So through our algorithm we can also discover MAC addresses of the devices which are multi-hop away and are transparent from the user.

4.3 Probing Technique

ICMP ping works fine for nodes that are active but for inactive nodes it is a performance bottleneck since it has to wait for the response timeout. Our goal was to use a probing technique which can concurrently ping multiple nodes with time efficiency. One approach is to use broadcast ping, but it is usually blocked by routers. Second approach is to modify the ping code to enable concurrent pinging so that we don't create a new process each time but certain routers like cnn.com filter and block ICMP packets. The third and best approach is to use a fast TCP based ping which can concurrently ping multiple hosts and is not blocked by routers, since it would open TCP sockets and if some message is returned back it means that the node is alive. So we are using an open source implementation of TCP-Ping i.e. T-Ping [1] with a little modification for probing up to 254 nodes concurrently.

The algorithm gives a very fast initial discovery and can also be used in the resource discovery part of the grid computing environments as well as with topology discovery solutions. For very big address spaces the completeness of the algorithm

depends on the value of the limit of successive failures ‘L’. The larger the value of ‘L’, the larger the address space will be probed.

Chapter 5

IMPLEMENTATION AND PERFORMANCE RESULTS

The software Constella Platinum is implemented in Java and is fully platform independent and is tested on Windows and Linux systems. The software automatically detects the OS of the machine and runs the appropriate code.

5.1 TOOLS USED

5.1.1 Programming Language Java

Java language from Sun Microsystems is a general-purpose, object-oriented language that's been in the news quite a bit over the past couple of years. Most of Java's press coverage emphasizes its client/server role, as a language for writing ``applets'' that are downloaded from a server and executed locally. But Java isn't restricted to writing applets; it works just as well for writing traditional single-computer applications. Here we will examine the advantages and disadvantages of java language.

A. Simple

Although Java resembles C++, it omits many of C++'s more confusing features, including the ones most likely to cause problems for beginners: pointers, operator

overloading, multiple inheritance, and templates. Moreover, Java lacks many of the automatic type conversions that C++ performs.

At the same time, Java adds an important feature that simplifies programming: automatic garbage collection. Having the language handle storage management gives Java a big edge over languages such as C and C++, where releasing memory that's no longer needed requires programmer intervention. Garbage collection not only makes programming easier but also avoids the bugs caused by dangling pointers. In C++ programming, too much effort is spent on problems of memory allocation and reallocation.

B. Object-Oriented

The importance of introducing the object-oriented paradigm early in a student's program of study is increasingly being recognized. C++ is the most popular object-oriented language. Java also supports object-oriented programming, but with significant advantages over C++:

- Use of objects. Only the primitive types are not objects. There are no stand-alone functions; all functions must belong to a class.
- Objects are always allocated dynamically and manipulated through references, thereby simplifying their semantics.
- Storage management is handled automatically, significantly reducing the difficulty of writing many classes.

- Features, like operator overloading and multiple inheritance are not present in java.

C. Distributed

With the growing importance of networking in general and the Internet in particular. Java is unique among major languages in its support for networking, which includes classes for working with URLs and sockets. Java makes it easy for programs to access specific URLs on the Web, allowing gaining a better understanding of how the web works as well as being able to write some rather interesting programs.

D. Robust

A number of Java's properties are the result of making the language safe for transmitting executable content over the Internet. ``Java puts a lot of emphasis on early checking for possible problems, later dynamic (runtime) checking, and eliminating situations that are error prone."

Here are some of the measures that Java uses to achieve robustness:

- *No pointers.* Although Java uses pointers internally, no pointer operations are made available to programmers. There are no pointer variables, arrays can't be manipulated via pointers, and integers can't be converted into pointers.
- *Garbage collection.* Thanks to automatic garbage collection, there's no chance of a program corrupting memory via a dangling pointer.

- *Strict type checking***.** Java's type checking is much stricter than that in C or C++. In particular, casts are checked at both compile time and run time. As a bonus, type checking is repeated at link time to detect version errors.
- *Run-time error checking***.** Java performs a number of checks at run time, including checking that array subscripts are within bounds.

E. Secure

In addition to being *robust* (resistant to programmer error), Java programs are designed to be *secure* (safe against malicious attack). Java's run-time system performs checks to make sure that programs transmitted over a network have not been tampered with. The code produced by the Java compiler is checked for validity, and the program is prevented from performing unauthorized actions.

F. Architecture-Neutral

The Java language is completely architecture-neutral. As a result, programs written in Java will run on any platform that supports the Java run-time system. The significance of a multiplatform language like Java cannot be overstated. Sun's Java Development Kit is available for a variety of platforms, including Windows 95 and NT, Macintosh, and Sun Solaris, all of which are widely used.

G. Portable

Java programs are not only architecture-neutral but portable as well. One way in which Java achieves portability is by completely defining all aspects of the language, leaving no decisions to the compiler writer. Consider the issue of types. Most programming languages don't define the exact ranges of types, allowing for variations based on the computer's architecture. Java, on the other hand, completely defines the ranges and properties of all types. Values of the int type are always signed 32-bit integers; float values are stored in 32 bits using the IEEE 754 representation.

H. Interpreted

Java is usually an interpreted language. A Java compiler translates a program into byte codes, which can then be executed by an interpreter. Linking is done at run time, with code loaded dynamically by the run-time system as needed.

I. High-Performance

Programs written in interpreted, garbage-collected languages often don't execute at high speed. In Java, however, the performance penalty isn't as bad as in some languages. One reason for Java's superior performance is that garbage collection is done by a separate low-priority thread. That way, garbage collection takes place primarily when the program has nothing else useful to do.

Greater speed can be achieved by translating Java's bytecodes into native machine instructions. This can be done by translating the entire program to native code

prior to execution, or it can be done on the fly by a ``just-in-time" (JIT) compiler. JIT compilation is becoming a standard feature of commercial Java environments

J. Multithreaded

Unlike most major programming languages, Java has built-in support for multitasking. A Java program may create any number of threads, which appear to execute in parallel. Java support concurrency. Java's model of concurrency is simple enough that even beginners can use concurrency effectively.

K. Dynamic

Java is designed to accommodate the fast-paced, modern world of software development, in which components of a system may change on a regular basis. Java's run-time linking guarantees that a program always loads the most recent version of its library modules. It also reduces recompilation by making it possible to add methods and instance variables to a library without having to recompile its clients.

5.1.2 SNMP (Simple Network Management Protocol)

To discover the topology Simple Network Management Protocol (SNMP) can be used to gather information from the managed devices like network switches, printers, routers etc. This information is stored in Management Information Bases (MIB) that is stored in each of the SNMP daemon. All the network equipment that is designed to be managed by SNMP must implement a MIB, and the management

application must be told what can be managed on the agent. The collection of the descriptions of all the manageable features might be either a standard or custom MIB sub tree, which is described by a MIB module. MIB module files are loaded into the NMS so that the device can be managed. Now in the following sections we will discuss some of the important MIB that are used in the topology discovery algorithm which we have developed.

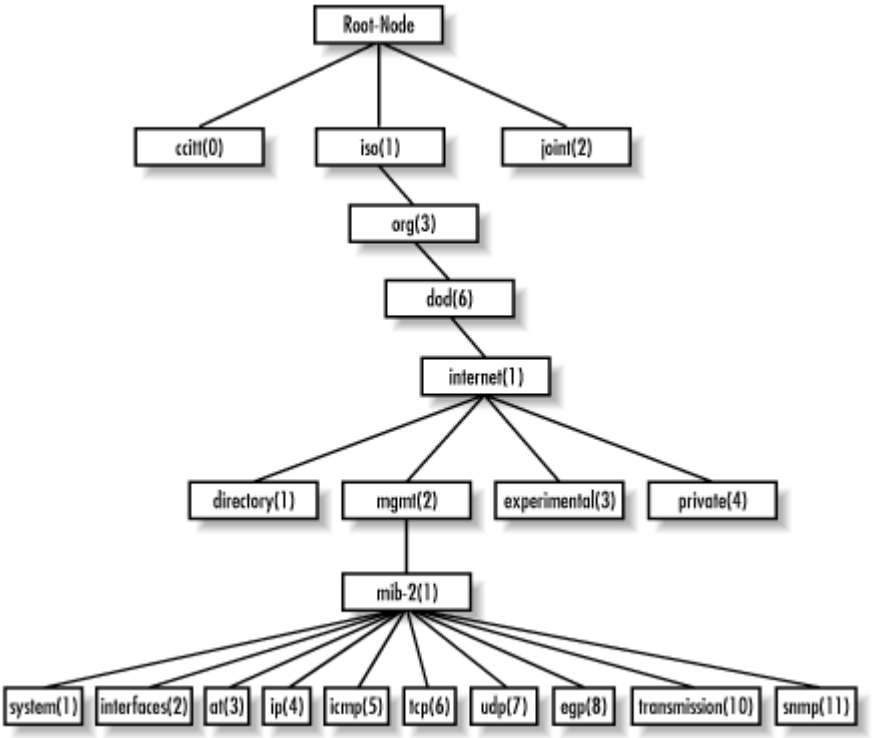

Figure 31 : SNMP MIB Tree

A. IP-MIB

IP-MIB is a Management Information Base which is related to the information

provided by the Internet Protocol Layer. This MIB will be present on all the devices which work on the IP Layer.

- ip(mib-2 4)

  - ipForwarding(ip 1)
  - ipDefaultTTL(ip 2)
  - ipInReceives(ip 3)
  - ipInHdrErrors(ip 4)
  - ipInAddrErrors(ip 5)
  - ipForwDatagrams(ip 6)
  - ipInUnknownProtos(ip 7)
  - ipInDiscards(ip 8)
  - ipInDelivers(ip 9)
  - ipOutRequests(ip 10)
  - ipOutDiscards(ip 11)
  - ipOutNoRoutes(ip 12)
  - ipReasmTimeout(ip 13)
  - ipReasmReqds(ip 14)
  - ipReasmOKs(ip 15)
  - ipReasmFails(ip 16)
  - ipFragOKs(ip 17)
  - ipFragFails(ip 18)

- ipFragCreates(ip 19)
- ipAddrTable(ip 20)
    - ipAddrEntry(ipAddrTable 1)
        - ipAdEntAddr(ipAddrEntry 1)
        - ipAdEntIfIndex(ipAddrEntry 2)
        - ipAdEntNetMask(ipAddrEntry 3)
        - ipAdEntBcastAddr(ipAddrEntry 4)
        - ipAdEntReasmMaxSize(ipAddrEntry 5)
- ipNetToMediaTable(ip 22)
    - ipNetToMediaEntry(ipNetToMediaTable 1)
        - ipNetToMediaIfIndex(ipNetToMediaEntry 1)
        - ipNetToMediaPhysAddress(ipNetToMediaEntry 2)
        - ipNetToMediaNetAddress(ipNetToMediaEntry 3)
        - ipNetToMediaType(ipNetToMediaEntry 4)
- ipRoutingDiscards(ip 23)

B. IF-MIB

Now we will discuss the managed objects used for managing Network Interfaces. Here in this section we will discusses the 'interfaces' group of MIB-II. We will discuss how the IF-MIB in definition of numerous media-specific MIB modules can be used in conjunction with the “interfaces” group for managing various sub-

layers beneath the internet work layer. It specifies clarifications to, and extensions of, the architectural issues within the MIB-II model of the 'interfaces' group. One of the strengths of internet work layer protocols such as IP is that they are designed to run over any network interface. In achieving this, IP considers any and all protocols it runs over as a single "network interface" layer. A similar view is taken by other internet work layer protocols. This concept is represented in MIB-II by the 'interfaces' group which defines a generic set of managed objects such that any network interface can be managed in an interface-independent manner through these managed objects. The 'interfaces' group provides the means for additional managed objects specific to particular types of network interface (e.g., a specific medium such as Ethernet) to be defined as extensions to the 'interfaces' group for media-specific management. Since the standardization of MIB-II, many such media-specific MIB modules have been defined. Experience in defining these media-specific MIB modules has shown that the model defined by MIB-II is too simplistic and/or static for some types of media-specific management. As a result, some of these media-specific MIB modules assume an evolution or loosening of the model. In information in this media specific IF-MIB can be used in connection with other standard MIB to discover the physical topology.

5.1.2.1 Address Resolution Protocol (ARP)

The address resolution protocol (ARP) is a protocol used by the Internet Protocol (IP), specifically IPv4, to map IP network addresses to the hardware addresses used by a data link protocol. The protocol operates below the network layer

as a part of the interface between the OSI network and OSI link layer. It is used when IPv4 is used over Ethernet.

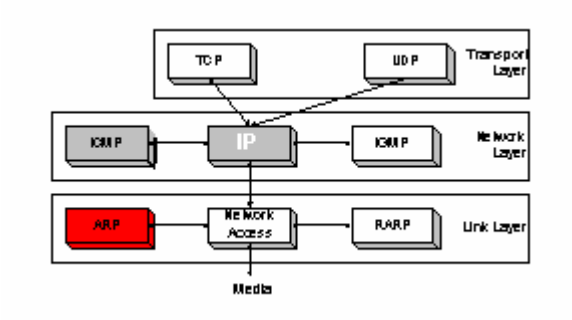

Figure 32 : Address Resolution Protocol

The term address resolution refers to the process of finding an address of a computer in a network. The address is "resolved" using a protocol in which a piece of information is sent by a client process executing on the local computer to a server process executing on a remote computer. The information received by the server allows the server to uniquely identify the network system for which the address was required and therefore to provide the required address. The address resolution procedure is completed when the client receives a response from the server containing the required address.

An Ethernet network uses two hardware addresses which identify the source and destination of each frame sent by the Ethernet. The destination address (all 1's) may also identify a broadcast packet (to be sent to all connected computers). The hardware address is also known as the Medium Access Control (MAC) address, in reference to the standards which define Ethernet. Each computer network interface

card is allocated a globally unique 6 byte link address when the factory manufactures the card (stored in a PROM). This is the normal link source address used by an interface. A computer sends all packets which it creates with its own hardware source link address, and receives all packets which match the same hardware address in the destination field or one (or more) pre-selected broadcast/multicast addresses.

The Ethernet address is a link layer address and is dependent on the interface card which is used. IP operates at the network layer and is not concerned with the link addresses of individual nodes which are to be used .The address resolution protocol (ARP) is therefore used to translate between the two types of address. The ARP client and server processes operate on all computers using IP over Ethernet. The processes are normally implemented as part of the software driver that drives the network interface card.

There are four types of ARP messages that may be sent by the ARP protocol. These are identified by four values in the "operation" field of an ARP message. The types of message are:

- ARP request
- ARP reply
- RARP request
- RARP reply

5.2 *Performance Results*

This system has been implemented and extensively tested in the labs at NUST Institute of information technology Pakistan.

This system has been implemented and extensively tested in the labs at NUST Institute of Information Technology Pakistan (NIIT) (Test Case A,B,C) which has a total of 21 subnets and Comtec Japan (Test Case D,E) with a network containing 4 subnets.

Initial computation of the algorithm is fast since it has to consult the NMS ARP cache to get the MAC addresses. Then after we start discovering the IP addresses beyond the gateway the algorithm slows down due to the latency and the computation time of the SNMP queries since SNMP queries to check the router's ARP cache take more time then consultation of the NMS ARP cache. The accuracy of the algorithm depends upon the value of Limit 'L' which is the number of allowed successive failures in each direction in my algorithm. The more the value of 'L' the more is the accuracy since with more value of 'L' we have to check more nodes.

Test cases:

1. Test case A,B,C: Network = NIIT: L=1, L=2,L=3
2. Test case D,E,F: Network = Comtec, L=1, L=2

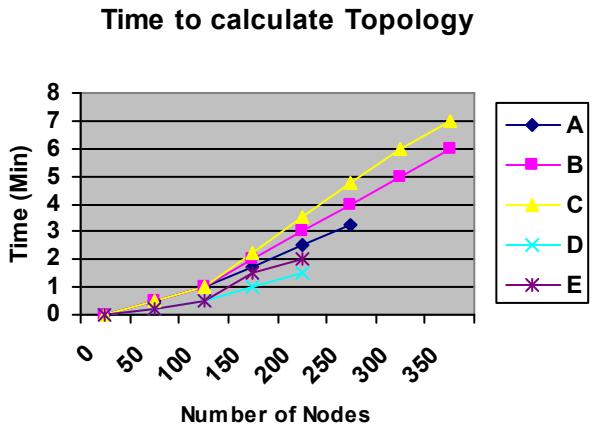

Figure 33: Time Efficiency of Constella Platinum

Figure 32 shows result of the test cases with respect to time. The increase in time from test case A to C is due to increase in the no of dead address ranges. Similar is the cases from D and E, but the variation is very small because the network is small. This shows that probability of getting complete network is more if we keep the L>1. But it will slightly decrease the time efficiency of the algorithm.

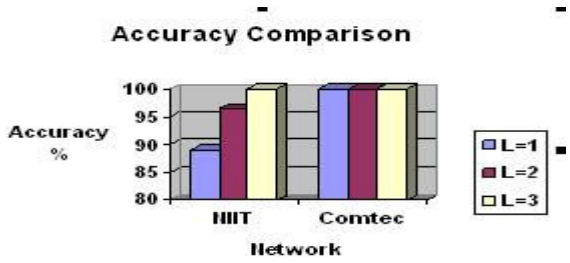

Figure 34: Algorithm Accuracy in different test cases

Figure 33 shows relative comparison of accuracy of test case A to E. In a network like NIIT we have to keep the value of L more to ensure completeness and at Comtec as we had a small network so the results were accurate even at low values of L.

5.3 *Snapshots*

Figure 35: Initial View

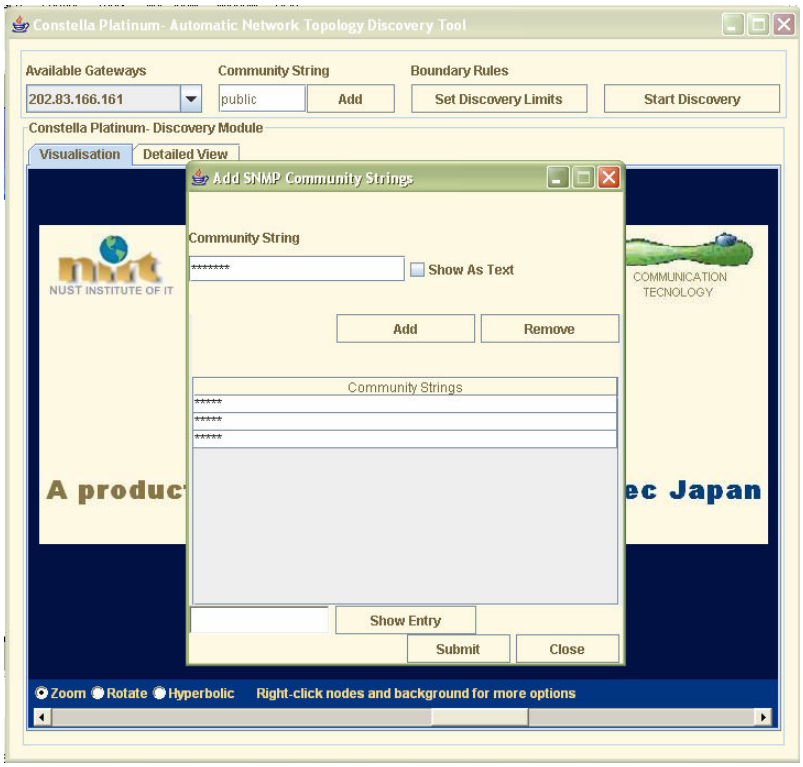

Figure 36: Add Community String Panel

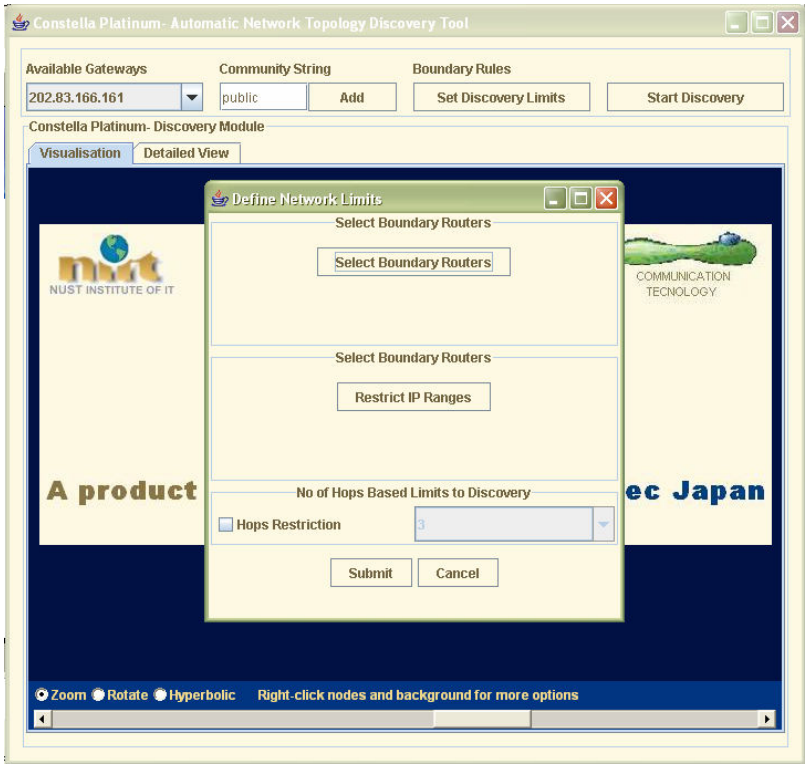

Figure 37: Optional Boundary Selection

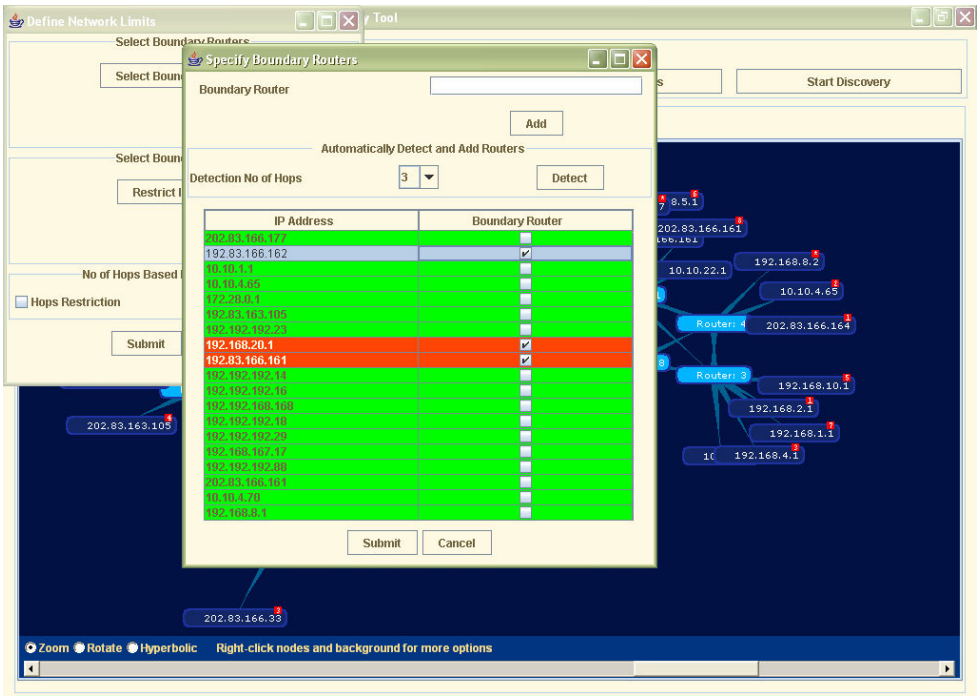

Figure 38: Selection of Auto detected Boundary Routers

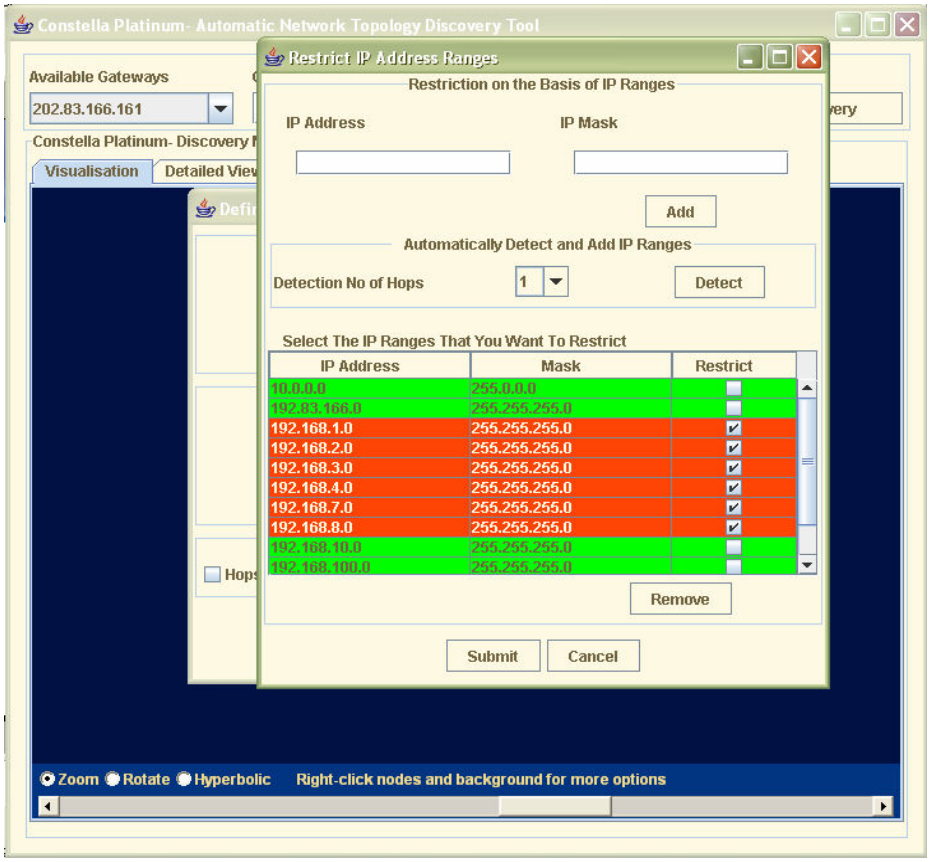

Figure 39: Restriction on the basis of IP Address Ranges

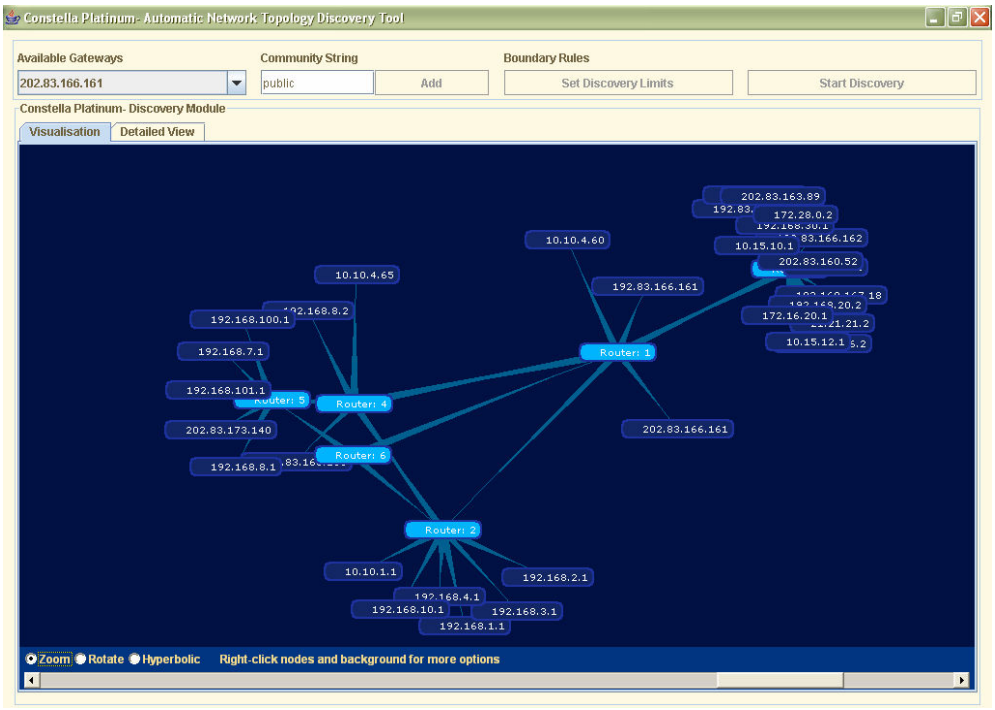

Figure 40: Discovered Routers and their Interconnections

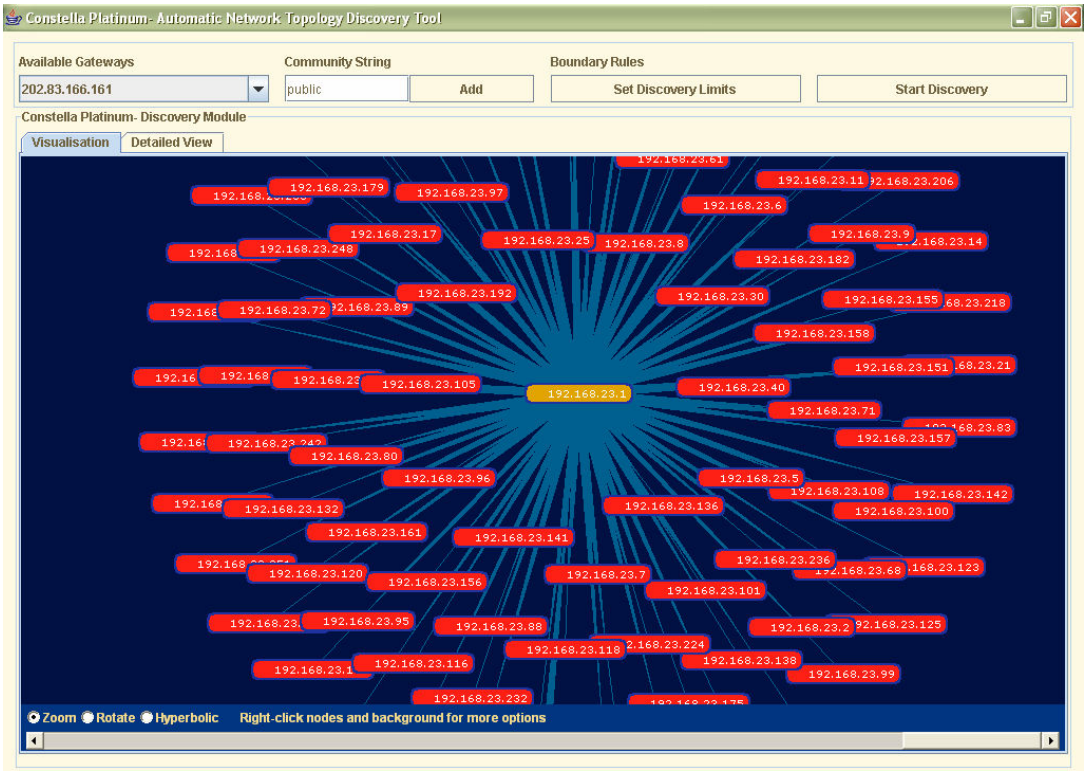

Figure 41: Nodes under a subnet

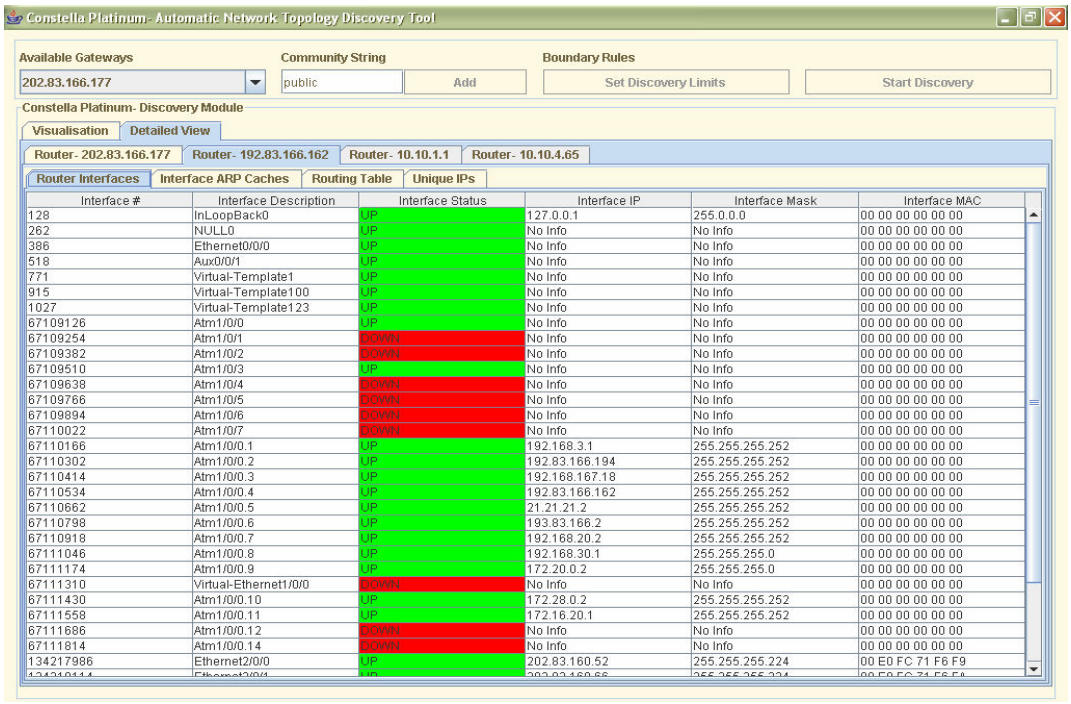

Figure 42: Interface Management

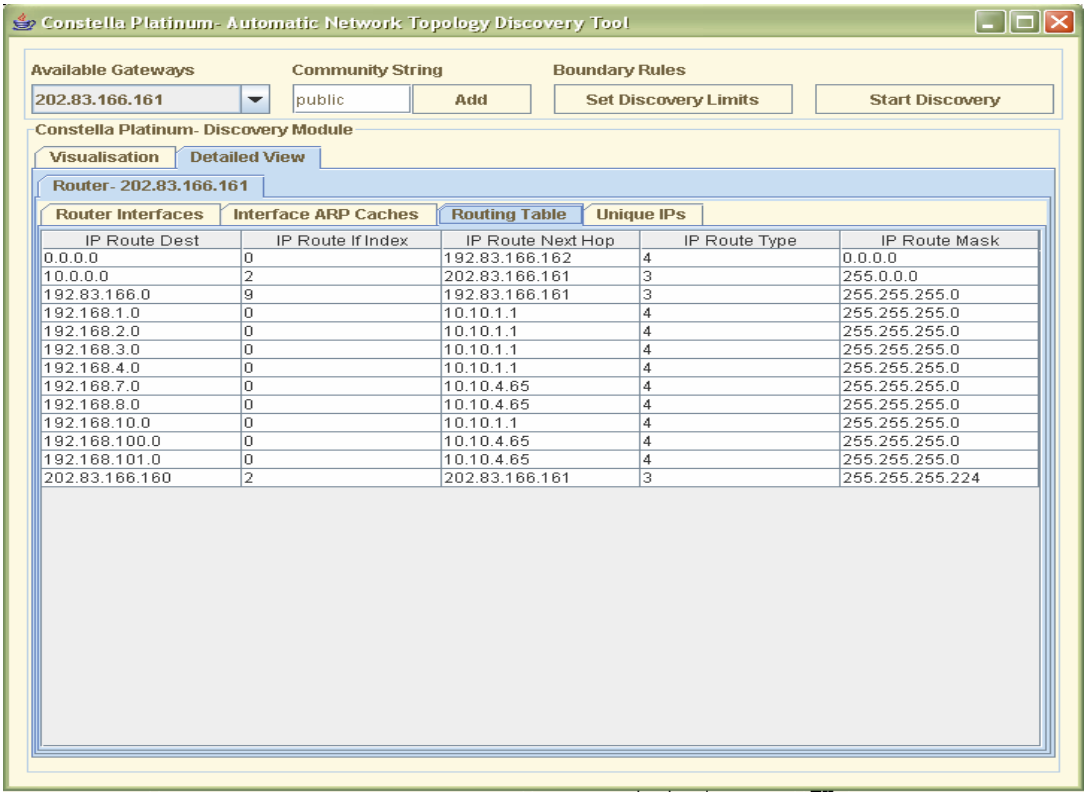

Figure 43: Detailed View of a router

Chapter 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

Autonomous discovery of active network plays an important role in enhancing the network management and analysis. This algorithm makes use of TCP based ping which is not blocked by routers. The algorithm only requires SNMP to be supported on routers and switches and uses MIB-II information which is mandatory for SNMP enabled devices. In this paper we proposed a technique to discover the MAC addresses of devices that are multi-hop away by first probing and then getting the values through SNMP. We have developed ARP cache based intelligent alive subnet guessing algorithm to determine the IP ranges which have greater probability of being assigned, thus reducing the number of queries required to discover the devices in the network. A special characteristic of our algorithm is that the discovery process yields a quick initial response. Our discovery includes routers, computers, switches, printers and other IP enabled devices, along with physical connectivity relationships that exist among entities in a communication network. We have implemented our algorithm which has been experimentally tested over our research network. The results clearly validate our methodology, demonstrating the accuracy and practicality of the proposed algorithms. The algorithm exports the data in RDF format which is makes the data semantically enriched.

6.2 Future Directions

The future directions can be as follows.

- Semantic FIPA agent based discovery
- ATM Networks
- IPv6 topology discovery
- Constella web interface
- Constella Lite (PDA version)
- Wireless Networks
- MPLS/ASON networks
- 3-D Visualisation

6.3 Publications

- Loosely Coupled Architecture for Integrating Network Topology Discovery Algorithms

  Faran Javed, Hamid Mukhtar, Fawad Nazir, Hafiz Farooq Ahmed, Hiroki Suguri, Arshad Ali

  (Accepted Honet 05)

- IP Network Topology Discovery for Large and Multi subnet using Mobile "Service" Agents

  Mohsan Jameel, Hamid Mukhtar, Hafiz Farooq Ahmad, Arshad Ali, Hiroki Suguri

- (Accepted Honet 06)

- Autonomous Network Topology Discovery of Large Multi-subnet Networks using TCP based probing IEEE Infocom 07

  Hamid Mukhtar, Mohsan Jameel, Hafiz Farooq Ahmad, Hiroki Suguri, Arshad Ali

  (submitted for acceptance)

- Ontology Based Semantics for Network Topology Discovery Data

  Hamid Mukhtar, Maruf Pasha, Mohsan Jameel, Hafiz Farooq Ahmad, Hiroki Suguri, Arshad Ali

  (Submitted for acceptance)

APPENDICES

Appendix – A

OSI Layering

The ISO (International Standards Organization) has created a layered model, called the OSI (Open Systems Interconnect) model, to describe defined layers in a network operating system. The purpose of the layers is to provide clearly defined functions that can improve Internet works connectivity between "computer" manufacturing companies. Each layer has a standard defined input and a standard defined output

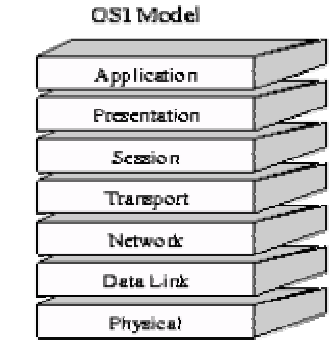

Figure 44: OSI Model

These are the 7 Layers of the OSI model:

- Application Layer (User Interface)
- Presentation Layer (Translation)

- Session Layer (Syncs and Sessions)
- Transport Layer (Packets; flow control & error handling)
- Network Layer (addressing, routing)
- Data Link Layer (data frames to bits)
- Physical Layer (hardware and raw bit streams)

Brief description of functionality of each layer is as follows

Physical Layer – Layer 1

The physical layer is responsible for passing bits onto and receiving them from the connecting medium. This layer has no understanding of the meaning of the bits, but deals with the electrical and mechanical characteristics of the signals and signaling methods. For example, it comprises the RTS and CTS signals in an RS-232 environment, as well as TDM and FDM techniques for multiplexing data on a line. SONET also provides layer 1 capability.

Data Link Layer - Layer 2

The data link is responsible for node to node validity and integrity of the transmission. The transmitted bits are divided into frames; for example, an Ethernet, Token Ring or FDDI frame in local area networks (LANs). Frame relay and ATM are also at Layer 2. Layers 1 and 2 are required for every type of communications.

Network Layer – Layer 3

The network layer establishes the route between the sender and receiver across

switching points, which are typically routers. The most ubiquitous example of this layer is the IP protocol in TCP/IP. IPX, SNA and AppleTalk are other examples of routable protocols, which mean that they include a network address and a station address in their addressing system. This layer is also the switching function of the dial-up telephone system. If all stations are contained within a single network segment, then the routing capability in this layer is not required.

Layers 1 through 3 are responsible for moving packets from the sending station to the receiving station.

Transport Layer – Layer 4

This layer is responsible for overall end to end validity and integrity of the transmission. The lower layers may drop packets, but the transport layer performs a sequence check on the data and ensures that if a 12MB file is sent, the full 12MB is received.

"OSI transport services" include layers 1 through 4, collectively responsible for delivering a complete message or file from sending to receiving station without error.

Session Layer - Layer 5

This Layer provides coordination of the communications in an orderly manner. It determines one-way or two-way communications and manages the dialog between both parties; for example, making sure that the previous request has been fulfilled before the next one is sent. It also marks significant parts of the transmitted data with checkpoints to allow for fast recovery in the event of a connection failure.

In practice, this layer is often not used or services within this layer are sometimes incorporated into the transport layer.

Presentation Layer – Layer 6

When data are transmitted between different types of computer systems, the presentation layer negotiates and manages the way data are represented and encoded. For example, it provides a common denominator between ASCII and EBCDIC machines as well as between different floating point and binary formats. Sun's XDR and OSI's ASN.1 are two protocols used for this purpose. This layer is also used for encryption and decryption

Application Layer – Layer 7

This top layer defines the language and syntax that programs use to communicate with other programs. The application layer represents the purpose of communicating in the first place. For example, a program in a client workstation uses commands to request data from a program in the server. Common functions at this layer are opening, closing, reading and writing files, transferring files and e-mail messages, executing remote jobs and obtaining directory information about network resources.

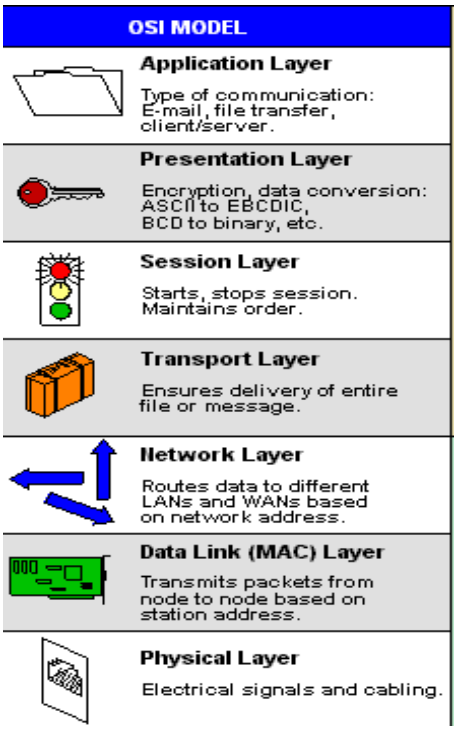

Figure 45: OSI Seven Layer Model

Lower Layer Devices

1 Repeaters (Layer 1 Devices)

Devices that forward frames at the OSI physical layer are called repeaters, hubs or concentrators depending on the technology. These devices are used to extend the span of a physical network and might be required for network operations, depending on the network technology. A repeater has two or more network interfaces, which are also called ports. A repeater does not resolve conflicts that can occur in some technologies when devices try to simultaneously transmit a frame. Such a conflict is called collisions.

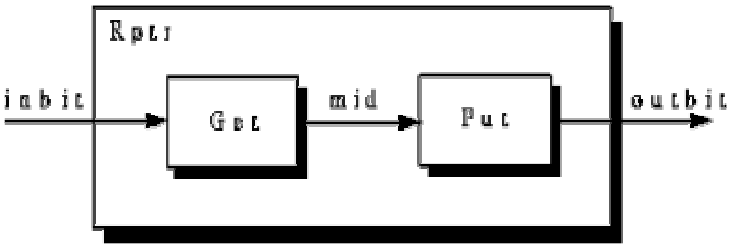

Figure 46 : Repeater Logical Architecture

Technologies such as Ethernet are designed to operate with collisions. The portion of an Ethernet network connected by repeaters is called a collision domain, because a collision event can occur due to transmissions of device in that portion of the network. Repeaters are implemented with hardware circuits and do not need a processor to function. To allow a repeater to be managed, vendor has added a management processor with a network interface whose single purpose is to provide management of repeater.

2 Bridges (Layer 2 Devices)

A bridge device filters data traffic at a network boundary. Bridges reduce the amount of traffic on a LAN by dividing it into two segments. Bridges operate at the data link layer (Layer 2) of the OSI model. Bridges inspect incoming traffic and decide whether to forward or discard it. An Ethernet bridge, for example, inspects each incoming Ethernet frame - including the source and destination MAC addresses, and sometimes the frame size - in making individual forwarding decisions. Bridges serve a similar function as switches that also operate at Layer 2. Traditional bridges, though, support one network boundary, whereas switches usually offer four or more hardware ports. Switches are sometimes called "multi-port bridges" for this reason.

The ports of a bridge operate in promiscuous mode because they must receive all frames and not just those addressed to the network interfaces of the bridge. A bridge revives a complete frame before forwarding it. A forwarded frame used the original source MAC address from a received frame and not the MAC address of the bridge network interface.

There are two major types of bridges, which are transparent and source routing. A transparent bridges uses the destination MAC address of a received frame to decide whether to forward a frame, and if so, then to which port or ports. A source routing bridge used a path contained in a frame to direct its forwarding decisions.

3 Routers (Layer 3 Devices

Network router is a device or a piece of software in a computer that forwards and routes data packets along networks. A network router connects at least two networks, commonly two LANs or WANs or a LAN and its ISP network. A router is often included as part of a network switch. A router is located at any where one network meets another, including each point-of-presence on the Internet. A router has two key jobs: The router ensures that information doesn't go where it's not needed. This is crucial for keeping large volumes of data from clogging the network. The router makes sure that information does make it to the intended destination. In performing these two jobs, a router joins the two networks, passing information from one to the other and, in some cases, performing translations of various protocols between the two networks. It also protects the networks from one another, preventing the traffic on one from unnecessarily spilling over to the other. This process is known

as routing. Routing is a function associated with the Network layer (layer 3) in the Open Systems Interconnection (OSI) model. Routers use network layer protocol headers, such as IP header where the source and destination addresses are included and routing tables to determine the best path to forward the packets. For the communication among routers and decide the best route between any two hosts, routing protocols such as ICMP are used. Actually, routers are specialized computers that send messages speeding to their destinations along thousands of possible pathways. One of the tools a router uses to decide which path a packet should go is a routing table. A routing table contains a collection of information, including: Information on which connections lead to particular groups of addresses, Priorities for connections to be used, Rules for handling both routine and special cases of traffic Information in the routing tables can be static (with routes manually entered by the network administrator) or dynamic (where routers communicate to exchange connection and route information using various routing protocols). A routing table can be as simple as a few lines in the smallest routers, but can grow to massive size and complexity in the very large routers that handle the bulk of Internet messages. As the number of networks attached to one another grows, the routing table for handling traffic among them grows, and the processing power of the router is increased

Appendix – B

SIMPLE NETWORK MANAGEMENT PROTOCOL (SNMP)

SNMP is based on the manager/agent model. SNMP is referred to as "simple" because the agent requires minimal software. Most of the processing power and the data storage resides on the management system, while a complementary subset of those functions resides in the managed system.

To achieve its goal of being simple, SNMP includes a limited set of management commands and responses. The management system issues Get, GetNext and Set messages to retrieve single or multiple object variables or to establish the value of a single variable. The managed agent sends a Response message to complete the Get, GetNext or Set. The managed agent sends an event notification, called a *trap* to the management system to identify the occurrence of conditions such as threshold that exceeds a predetermined value. In short there are only five primitive operations:

- get (retrieve operation)
- get next (traversal operation)
- get response (indicative operation)
- set (alter operation)
- trap (asynchronous trap operation)

SNMP Message Construct

Each SNMP message has the format:

- Version Number
- Community Name - kind of a password
- One or more SNMP PDUs - assuming trivial authentication

Each SNMP PDU except trap has the following format:

- request id - request sequence number
- error status - zero if no error otherwise one of a small set
- error index - if non zero indicates which of the OIDs in the PDU caused the error2
- list of OIDs and values - values are null for get and get next

Trap PDUs have the following format:

- enterprise - identifies the type of object causing the trap
- agent address - IP address of agent which sent the trap
- generic trap id - the common standard traps
- specific trap id - proprietary or enterprise trap
- time stamp - when trap occurred in time ticks
- list of OIDs and values - OIDs that may be relevant to send to the NMS

Outline of the SNMP protocol

- Each SNMP managed object belongs to a community
- NMS station may belong to multiple communities
- A community is defined by a community name which is an OctetString with 0 to 255 octets in length.
- Each SNMP message consists of three components
    - version number
    - community name
    - data - a sequence of PDUs associated with the request

Security levels with basic SNMP

Authentication

- trivial authentication based on plain text community name exchanged in SNMP messages
- authentication is based on the assumption that the message is not tampered with or interrogated

Authorization

- Once community name is validated then agent or manager checks to see if sending address is permitted or has the rights for the requested operation
- "View" or "Cut" of the objects together with permitted access rights is then derived for that pair (community name, sending address)

Summary

- Not very secure
- SNMP version 2 is addressing this
- Extended security is possible with current protocol (example: DES and MD5)
- Does not reduce its power for monitoring

What does SNMP access

- SNMP access particular instances of an object
- All instances of an object in the MIB reside at the leaf nodes of the MIB tree
- SNMP Protocol access objects by forming an Object identifier of form x.y

  where x is the "true" OID for the object in the MIB, and y is a suffix specified by the protocol that uniquely identifies a particular instance (e.g.. when accessing a table)
- For primitive single instance leaf objects use y=0

  for example: sysDescr (OID: 1.3.6.1.2.1.1.1) would be referenced in the SNMP protocol by 1.3.6.1.2.1.1.1.0 (i.e sysDescr.0)
- For single instance of columnar leaf objects (i.e one instance from a table type of object) use y=I1.I2.I3.... (Ii are table indexes)

MIB traversal using GetNext operation

SNMP assumes that the communication path is a connectionless communication sub network. In other words, no prearranged communication path is established prior to the transmission of data. As a result, SNMP makes no guarantees about the reliable delivery of the data. Although in practice most messages get through, and those that don't can be retransmitted. The primary protocols that SNMP implements are the *User Datagram Protocol* (UDP) and the *Internet Protocol* (IP). SNMP also requires Data Link Layer protocols such as Ethernet or Token Ring to implement the communication channel from the management to the managed agent.

SNMP's simplicity and connectionless communication also produce a degree of robustness. Neither the manager nor the agent relies on the other for its operation. Thus, a manager may continue to function even if a remote agent fails. When the agent resumes functioning, it can send a trap to the manager, notifying it of its change in operational status. The connectionless nature of SNMP leaves the recovery and error detection up to the NMS (Network Management Station) and even up to the agent. However keep in mind that SNMP is actually transport independent (although original design was connectionless transport function, which corresponds to the UDP protocol) and can be implemented on other transports as well:

- TCP (Connected approach)
- Direct mapping onto Ethernet MAC level
- Encapsulation onto X25 protocol
- Encapsulation onto ATM Cell
- and so on.....

The following figure describes the Transport Mechanism used in SNMP over UDP:

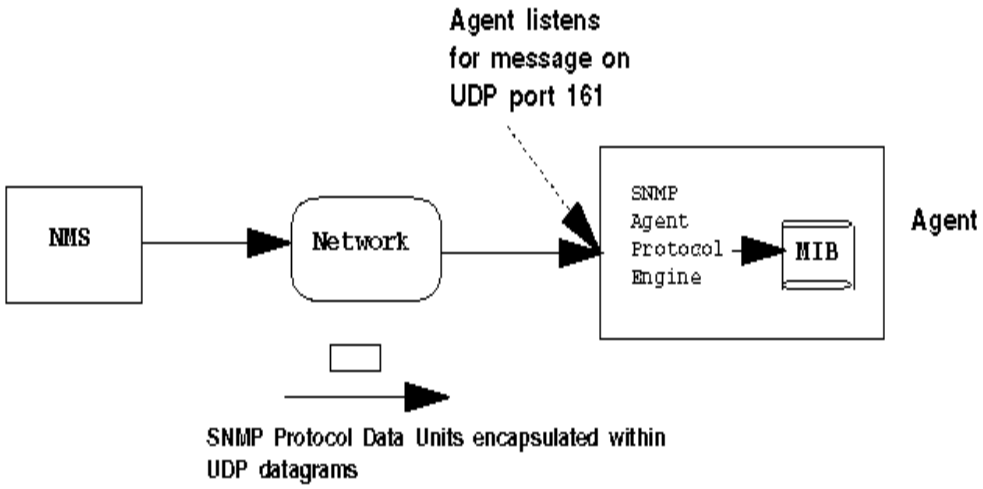

Figure 47: SNMP Transport Mechanism

UDP Transport

- Agent listen on UDP port 161
- Responses are sent back to the originating NMS port from a dynamic port , although many agents use port 161 also for this target.
- Maximum SNMP message size is limited by maximum UDP message size (i.e 65507 octets)
- All SNMP implementations have to> receive packets at least 484 octets in length
- Some SNMP implementation will incorrectly or not handle packets exceeding 484 octets

- Asynchronous Traps are received on port 162 of the NMS
- UDP more suitable than TCP when dynamic route changes occur often (e.g. when there are problems in the network)
- UDP packets minimize the demands placed on the network(no resource tied up as with connection mode)

Agent and NMS are responsible for determining error recovery

Appendix – C

MANAGEMENT INFORMATION BASE (MIB)

Managment information bases (MIBs) are a collection of definitions, which define the properties of the managed object within the device to be managed. Every managed device keeps a database of values for each of the definitions written in the MIB. It is not the actual database itself - it is implementation dependant. Definition of the MIB conforms to the SMI given in RFC 1155. Latest Internet MIB is given in RFC 1213 sometimes called the MIB-II. You can think of a MIB as an information warehouse

Criteria and Philosophy for standardized MIB

- Objects have to be uniquely named
- Objects have to be essential
- Abstract structure of the MIB needed to be universal
- For the standard MIB maintain only a small number of objects
- Allow for private extensions
- Object must be general and not too device dependant
- Objects can not be easily derivable from their objects
- If agent is to be SNMP manageable then it is mandatory to implement the Internet MIB (currently given as MIB-II in RFC 1157)

Naming an Object

- Universal unambiguous identification of arbitrary objects
- Can be achieved by using an hierarchical tree
- Based on the Object Identification Scheme defined by OSI

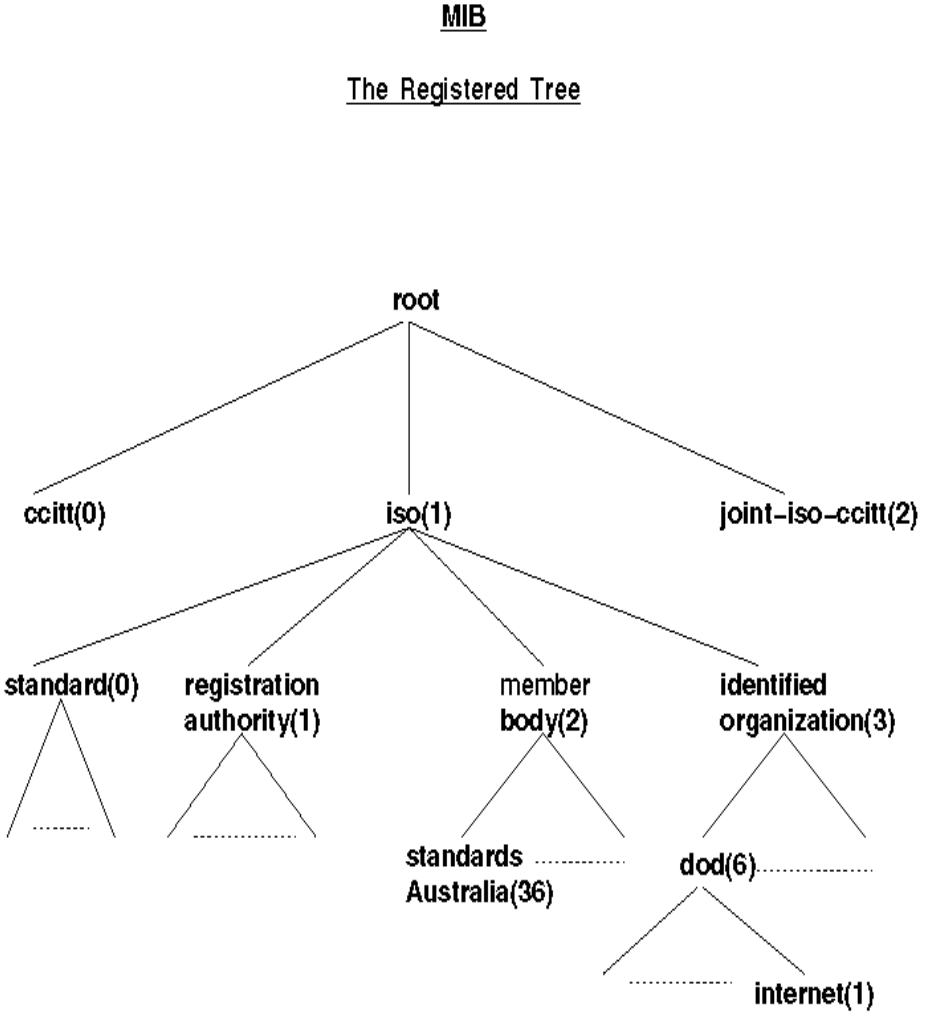

Figure 48: OID Object Tree

Object identifiers

- Object name is given by its name in the tree
- All child nodes are given unique integer values within that new sub-tree
- Children can be parents of further child sub-tree(i.e they have subordinates) where the numbering scheme is recursively applied
- The Object Identifier (or name) of an object is the sequence of non-negative Integer values traversing the tree to the node required.
- Allocation of an integer value for a node in the tree is an act of registration by whoever has delegated authority for that sub tree
- This process can go to an arbitrary depth
- If a node has children then it is an aggregate node
- Children of the same parent cannot have the same integer value

Object and Object identifiers

- Object is named or identified by the sequence of integers in traversing the tree to the object type required
- This does notidentify an instance of the object
- The Object Identifier (OID) is shown in a few ways with a.b.c.d.e being the preferred
- OIDs can name many types of objects:

  Standard documents (e.g.: FTAM)

  people (e.g.: Tax file numbers in Sweden are OIDs)

Organizations (e.g.: RAD or LANNET)

Computing network nodes (e.g.: workstations)

Dumb devices (e.g.: toasters)

... in fact anything at all ...

- For SNMP it is the Internet sub-tree for constructing OIDs for SNMP manageable agents

The Internet Sub-tree

- Directory sub-tree if for future directory services
- Experimental sub-tree is for experimental MIB work - still has to be registered with the authority (IESG)
- MIB sub-tree is the actual mandatory Internet MIB for all agents to implement (currently MIB-II RFC 1156 - this is the only sub tree of mgmt)
- Enterprise sub-tree (of private) are MIBs of proprietary objects and are of course not mandatory (sub-tree registered with Internet Assigned Numbers Authority) for example: Cisco router OID: 1.3.6.1.4.1.9.1.1
- SNMP management nearly always interest in MIB and specific enterprises MIBs

MIB-II Standard Internet MIB

- Definition follows structure given in SMI

- MIB-II (RFC 1213) is current standard definition of the virtual file store for SNMP manageable objects
- Has 10 basic groups
    - system
    - interfaces
    - at
    - ip
    - icmp
    - tcp
    - udp
    - egp
    - transmission
    - snmp
- If agent implements any group then is has to implement all of the managed objects within that group
- An agent does not have to implement all groups
- Note: MIB-i and MIB-II have same OID (position in the internet sub-tree)

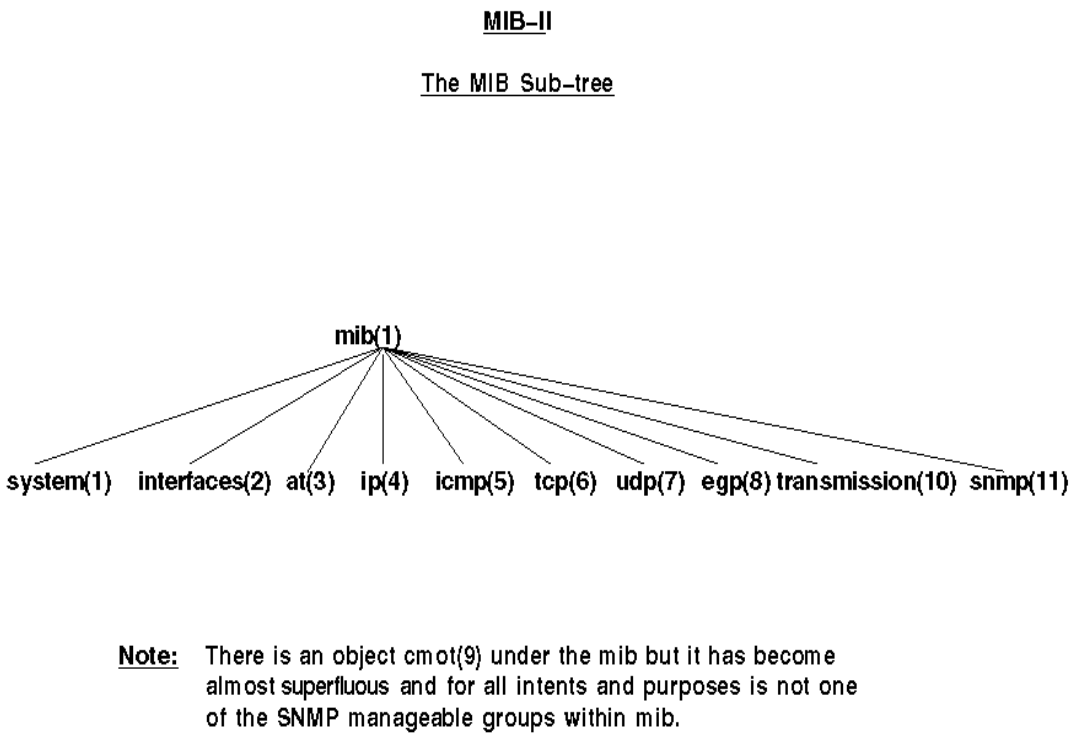

Figure 49: MIB-II Structure

Appendix – D

Routing Protocols

Routing is the act of moving information across an internetwork from a source to a destination. Along the way, at least one intermediate node typically is encountered. Routing is often contrasted with bridging, which might seem to accomplish precisely the same thing to the casual observer. The primary difference between the two is that bridging occurs at Layer 2 (the link layer) of the OSI reference model, whereas routing occurs at Layer 3 (the network layer). This distinction provides routing and bridging with different information to use in the process of

moving information from source to destination, so the two functions accomplish their tasks in different ways.

The Routing Information Protocol, or RIP, as it is more commonly called, is one of the most enduring of all routing protocols. RIP is also one of the more easily confused protocols because a variety of RIP-like routing protocols proliferated, some of which even used the same name! RIP and the myriad RIP-like protocols were based on the same set of algorithms that use distance vectors to mathematically compare routes to identify the best path to any given destination address. These algorithms emerged from academic research that dates back to 1957.

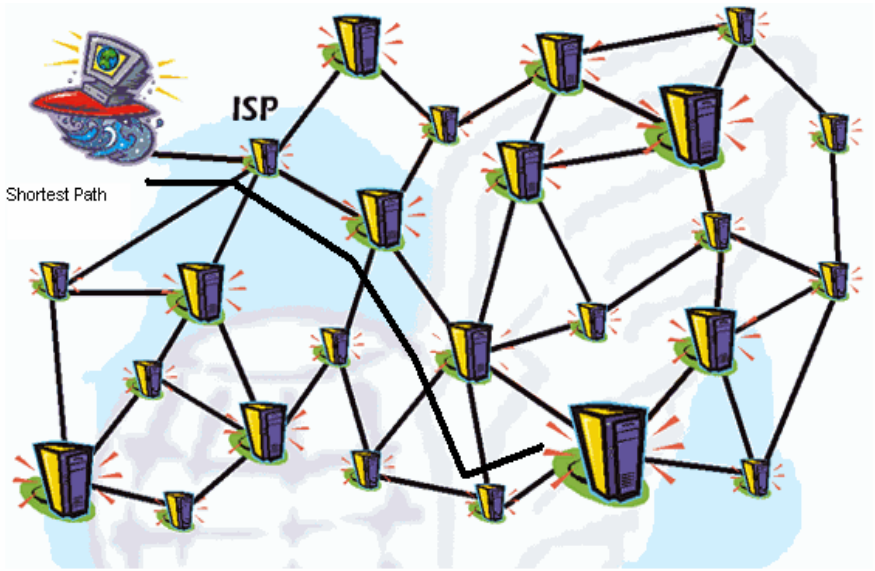

Figure 50 : Routing Protocols , Shortest Path Selection

Today's open standard version of RIP, sometimes referred to as IP RIP, is formally defined in two documents: Request For Comments (RFC) 1058 and Internet

Standard (STD) 56. As IP-based networks became both more numerous and greater in size, it became apparent to the Internet Engineering Task Force (IETF) that RIP needed to be updated. Consequently, the IETF released RFC 1388 in January 1993, which was then superceded in November 1994 by RFC 1723, which describes RIP 2 (the second version of RIP). These RFCs described an extension of RIP's capabilities but did not attempt to obsolete the previous version of RIP. RIP 2 enabled RIP messages to carry more information, which permitted the use of a simple authentication mechanism to secure table updates. More importantly, RIP 2 supported subnet masks, a critical feature that was not available in RIP.

OSPF protocol was developed due to a need in the internet community to introduce a high functionality non-proprietary Internal Gateway Protocol (IGP) for the TCP/IP protocol family. The discussion of creating a common interoperable IGP for the Internet started in 1988 and did not get formalized until 1991. At that time the OSPF Working Group requested that OSPF be considered for advancement to Draft Internet Standard.

The OSPF protocol is based on link-state technology which is a departure from the Bellman-Ford vector based algorithms used in traditional Internet routing protocols such as RIP. OSPF has introduced new concepts such as authentication of routing updates, Variable Length Subnet Masks (VLSM), route summarization, etc.

Appendix – E

Resource Description Framework (RDF)

The Resource Description Framework (RDF) is an infrastructure that enables the encoding, exchange and reuse of structured metadata. RDF is an application of XML that imposes needed structural constraints to provide unambiguous methods of expressing semantics. RDF additionally provides a means for publishing both human-readable and machine-processable vocabularies designed to encourage the reuse and extension of metadata semantics among disparate information communities. The structural constraints RDF imposes to support the consistent encoding and exchange of standardized metadata provides for the interchangeability of separate packages of metadata defined by different resource description communities.

Introduction

The World Wide Web affords unprecedented access to globally distributed information. Metadata, or structured data about data, improves discovery of and access to such information. The effective use of metadata among applications, however, requires common conventions about semantics, syntax, and structure. Individual resource description communities define the semantics, or meaning, of metadata that address their particular needs. Syntax, the systematic arrangement of data elements for machine-processing, facilitates the exchange and use of metadata among multiple

applications. Structure can be thought of as a formal constraint on the syntax for the consistent representation of semantics.

The Resource Description Framework (RDF), developed under the auspices of the World Wide Web Consortium (W3C), is an infrastructure that enables the encoding, exchange, and reuse of structured metadata. This infrastructure enables metadata interoperability through the design of mechanisms that support common conventions of semantics, syntax, and structure. RDF does not stipulate semantics for each resource description community, but rather provides the ability for these communities to define metadata elements as needed. RDF uses XML (eXtensible Markup Language) as a common syntax for the exchange and processing of metadata. The XML syntax is a subset of the international text processing standard SGML (Standard Generalized Markup Language (SGML)) specifically intended for use on the Web. The XML syntax provides vendor independence, user extensibility, validation, human readability, and the ability to represent complex structures. By exploiting the features of XML, RDF imposes structure that provides for the unambiguous expression of semantics and, as such, enables consistent encoding, exchange, and machine-processing of standardized metadata.

RDF supports the use of conventions that will facilitate modular interoperability among separate metadata element sets. These conventions include standard mechanisms for representing semantics that are grounded in a simple, yet powerful, data model discussed below. RDF additionally provides a means for publishing both human-readable and machine-processable vocabularies. Vocabularies

are the set of properties, or metadata elements, defined by resource description communities. The ability to standardize the declaration of vocabularies is anticipated to encourage the reuse and extension of semantics among disparate information communities..

The goals of RDF are broad, and the potential opportunities are enormous. This introduction to RDF begins by discussing the background context of the RDF initiative and relates it to other metadata activities. A discussion of the functionality of RDF and an overview of the model, schema and syntactic considerations of this framework follow.

Background

The history of metadata at the W3C began in 1995 with PICS, the Platform for Internet Content Selection. PICS is a mechanism for communicating ratings of web pages from a server to clients. These ratings, or rating labels, contain information about the content of web pages: for example, whether a particular page contains a peer-reviewed research article, or was authored by an accredited researcher, or contains sex, nudity, violence, foul language, etc. Instead of being a fixed set of criteria, PICS introduced a general mechanism for creating rating systems. Different organizations could rate content based on their own objectives and values, and users -- for example, parents worried about their children's web usage -- could set their browsers to filter out any web pages not matching their own criteria. Development of

PICS was motivated by the anticipation of restrictions on Internet content in the USA and elsewhere.

Through a series of meetings with the digital library community, limitations in the PICS specifications were identified, and functional requirements were outlined to address the more general problem of associating descriptive information with Internet resources based on the PICS architecture. As a result of these discussions, the W3C formed a new working group, PICS-NG Next Generation to address the more general issues of resource description.

Shortly after the PICS-NG working group was chartered, it became clear that the infrastructure designed in the early document specifications was applicable in several additional applications. As a result, the W3C consolidated these applications as the W3C Resource Description Framework working group.

RDF is the result of a number of metadata communities bringing together their needs to provide a robust and flexible architecture for supporting metadata on the web. While the development of RDF as a general metadata framework, and as such, a simple knowledge representation mechanism for the web, was heavily inspired by the PICS specification, no one individual or organization invented RDF. RDF is a collaborative design effort. Several W3C Member companies are contributing intellectual resources. It is drawing upon the XML design as well as proposals submitted by Microsoft's and Netscape. Other metadata efforts, such as the Dublin Core and the Warwick Framework have also influenced the design of the RDF.

The RDF Data Model

RDF provides a model for describing resources. Resources have properties (attributes or characteristics). RDF defines a *resource* as any object that is uniquely identifiable by an Uniform Resource Identifier (URI). The properties associated with resources are identified by property-types, and property-types have corresponding values. *Property-types* express the relationships of values associated with resources. In RDF, *values* may be atomic in nature (text strings, numbers, etc.) or other resources, which in turn may have their own properties. A collection of these properties that refers to the same resource is called a *description*. At the core of RDF is a syntax-independent model for representing resources and their corresponding descriptions (SPEC). The following graphic (Figure 1) illustrates a generic RDF description.

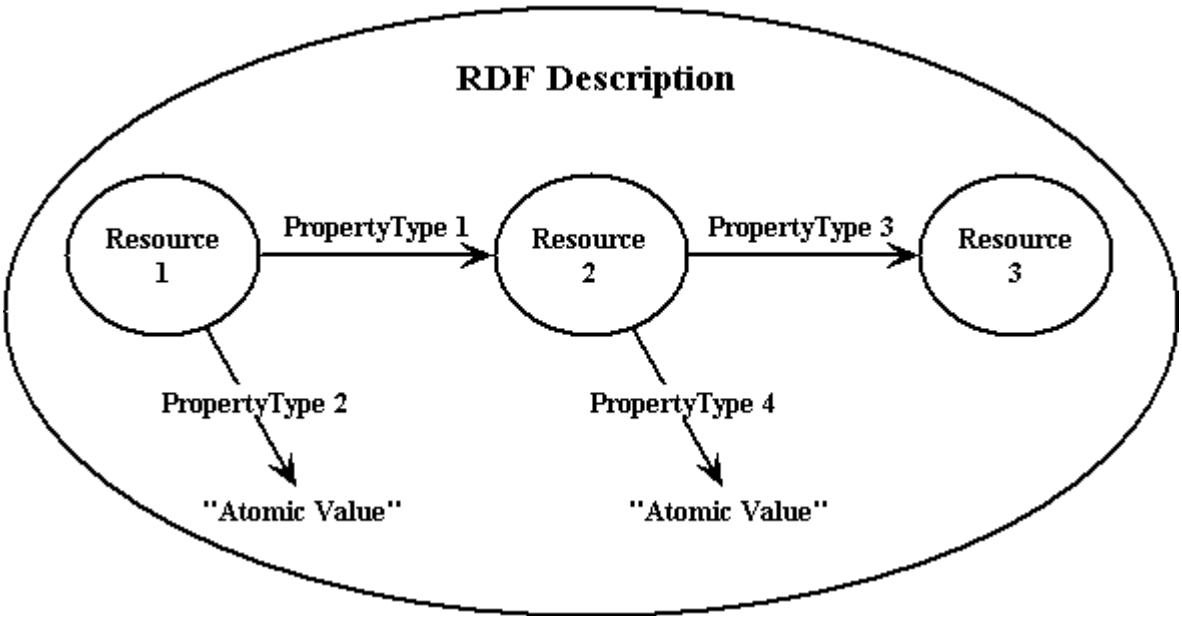

Figure 51: RDF Description

The application and use of the RDF data model can be illustrated by concrete examples. Consider the following statements:

1. "The author of Document 1 is John Smith"
2. "John Smith is the author of Document 1"

To humans, these statements convey the same meaning (that is, John Smith is the author of a particular document). To a machine, however, these are completely different strings. Whereas humans are extremely adept at extracting meaning from differing syntactic constructs, machines remain grossly inept. Using a triadic model of resources, property-types and corresponding values, RDF attempts to provide an unambiguous method of expressing semantics in a machine-readable encoding.

RDF provides a mechanism for associating properties with resources. So, before anything *about* Document 1 can be said, the data model requires the declaration of a resource *representing* Document 1. Thus, the data model corresponding to the statement "the author of Document 1 is John Smith" has a single resource Document 1, a property-type of author and a corresponding value of John Smith. To distinguish characteristics of the data model, the RDF Model and Syntax specification (SPEC) represents the relationships among resources, property-types, and values in a directed labeled graph. In this case, resources are identified as nodes, property-types are defined as directed label arcs, and string values are quoted. Given this representation, the data model corresponding to the statement is graphically expressed as (Figure 2):

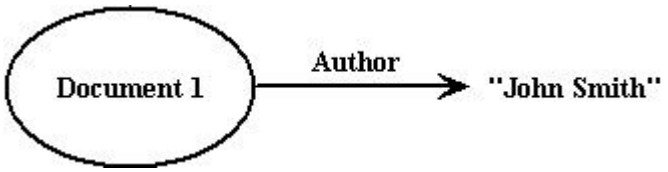

Figure 52: RDF Example (1/4)

If additional descriptive information regarding the author were desired, e.g., the author's email address and affiliation, an elaboration on the previous example would be required. In this case, descriptive information *about* John Smith is desired. As was discussed in the first example, before descriptive properties can be expressed about the *person* John Smith, there needs to be a unique identifiable resource *representing* him. Given the directed label graph notation in the previous example, the data model corresponding to this description is graphically represented as (Figure 3):

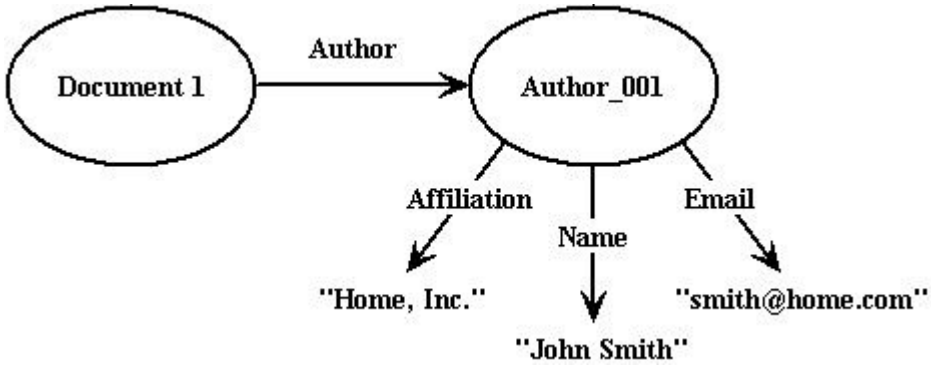

Figure 53: RDF Example (2/4)

In this case, "John Smith" the string is replaced by a uniquely identified resource denoted by Author_001 with the associated property-types of name, email

and affiliation. The use of unique identifiers for resources allows for the unambiguous association of properties. This is an important point, as the person John Smith may be the value of several different property-types. John Smith may be the author of Document 1, but also may be the value of a particular company describing the set of current employees. The unambiguous identification of resources provides for the reuse of explicit, descriptive information.

In the previous example the unique identifiable resource for the author was created, but not for the author's name, email or affiliation. The RDF model allows for the creation of resources at multiple levels. Concerning the representation of personal names, for example, the creation of a resource representing the author's name could have additionally been described using "firstname", "middlename" and "surname" property-types. Clearly, this iterative descriptive process could continue down many levels. What, however, are the practical and logical limits of these iterations?

There is no one right answer to this question. The answer is dependent on the domain requirements. These issues must be addressed and decided upon in the standard practice of individual resource description communities. In short, experience and knowledge of the domain dictate which distinctions should be captured and reflected in the data model.

The RDF data model additionally provides for the description of other descriptions. For instance, often it is important to assess the credibility of a particular description (e.g., "The Library of Congress told us that John Smith is the author of

Document 1"). In this case the description tells us something about the statement "John Smith is the author of Document 1", specifically, that the Library of Congress asserts this to be true. Similar constructs are additionally useful for the description of collections of resources. For instance, "John Smith is the author of Documents 1, 2, and 3". While these statements are significantly more complex, the same data model is applicable. A more detailed discussion of these issues is outside the scope of this overview, but more information is available in the RDF Model and Syntax Specification (SPEC).

The RDF Syntax

RDF defines a simple, yet powerful model for describing resources. A syntax representing this model is required to store instances of this model into machine-readable files and to communicate these instances among applications. XML is this syntax. RDF imposes formal structure on XML to support the consistent representation of semantics.

RDF provides the ability for resource description communities to define semantics. It is important, however, to disambiguate these semantics among communities. The property-type "author", for example, may have broader or narrower meaning depending on different community needs. As such, it is problematic if multiple communities use the same property-type to mean very different things. To prevent this, RDF uniquely identifies property-types by using the XML namespace mechanism. XML namespaces provide a method for unambiguously identifying the

semantics and conventions governing the particular use of property-types by uniquely identifying the governing authority of the vocabulary. For example, the property-type "author" defined by the Dublin Core Initiative as the "person or organization responsible for the creation of the intellectual content of the resource" and is specified by the Dublin Core CREATOR element. An XML namespace is used to unambiguously identify the Schema for the Dublin Core vocabulary by pointing to the definitive Dublin Core resource that defines the corresponding semantics. Additional information on RDF Schemas is discussed latter. If the Dublin Core RDF Schema, however, is abbreviated as "DC", the data model representation for this example would be (Figure 4):

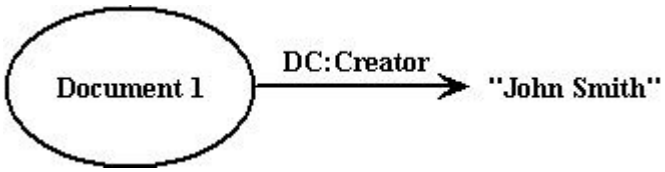

Figure 54: RDF Example (3/4)

This more explicit declaration identifies a resource Document 1 with the semantics of property-type Creator unambiguously defined in the context of DC (the Dublin Core vocabulary). The value of this property-type is John Smith.

The corresponding syntactic way of expressing this statement using XML namespaces to identify the use of the Dublin Core Schema is:

<?xml:namespace ns = "http://www.w3.org/RDF/RDF/" prefix ="RDF" ?>

```
<?xml:namespace ns = "http://purl.oclc.org/DC/" prefix = "DC" ?>

<RDF:RDF>

  <RDF:Description RDF:HREF = "http://uri-of-Document-1">

    <DC:Creator>John Smith</DC:Creator>

  </RDF:Description>

</RDF:RDF>
```

In this case, both the RDF and Dublin Core schemas are declared and abbreviated as "RDF" and "DC" respectively. The RDF Schema is declared as a boot-strapping mechanism for the declaration of the necessary vocabulary needed for expressing the data model. The Dublin Core Schema is declared in order to utilize the vocabulary defined by this community.

The URI associated with the namespace declaration references the corresponding schemas. The element <RDF:RDF> (which can be interpreted as the element RDF in the context of the RDF namespace) is a simple wrapper that marks the boundaries in an XML document where the content is explicitly intended to be mappable into an RDF data model instance (SPEC). The element <RDF:Description> (the element Description in the context of the RDF namespace) is correspondingly used to denote or instantiate a resource with the corresponding URI http://uri-of-Document-1. And the element <DC:Creator> in the context of the <RDF:Description>

represents a property-type DC:Creator and a value of "John Smith". The syntactic representation is designed to reflect the corresponding data model.

In the more advanced example, where additional descriptive information regarding the author is required, similar syntactic constructs are used. In this case, while it may still be desirable to use the Dublin Core CREATOR property-type to represent the person responsible for the creation of the intellectual content, additional property-types "name", "email" and "affiliation" are required. For this case, since the semantics for these elements are not defined in Dublin Core, an additional resource description standard may be utilized. It is feasible to assume the creation of an RDF schema with the semantics similar to the vCard specification designed to automate the exchange of personal information typically found on a traditional business card, could be introduced to describe the author of the document. The data model representation for this example with the corresponding business card schema defined as CARD would be (Figure 5):

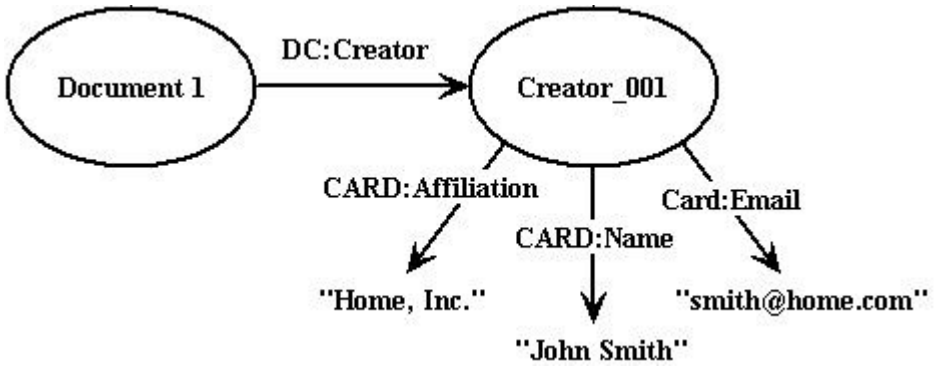

Figure 55: RDF Example (4/4)

This, in turn, could be syntactically represented as

```
 <?xml:namespace ns = "http://www.w3.org/RDF/RDF/" prefix = "RDF" ?>

<?xml:namespace ns = "http://purl.oclc.org/DC/" prefix = "DC" ?>

<?xml:namespace ns = "http://person.org/BusinessCard/" prefix = "CARD" ?>

<RDF:RDF>

  <RDF:Description RDF:HREF = "http://uri-of-Document-1">

    <DC:Creator RDF:HREF = "#Creator_001"/>

  </RDF:Description>

  <RDF:Description ID="Creator_001">

    <CARD:Name>John Smith</CARD:Name>

    <CARD:Email>smith@home.net</CARD:Email>

    <CARD:Affiliation>Home, Inc.</CARD:Affiliation>

  </RDF:Description>

</RDF:RDF>
```

in which the RDF, Dublin Core, and the "Business Card" schemas are declared and abbreviated as "RDF", "DC" and "CARD" respectively. In this case, the value associated with the property-type DC:Creator is now a resource. While the reference to the resource is an internal identifier, an external URI, for example, to a controlled authority of names, could have been used as well. Additionally, in this example, the semantics of the Dublin Core CREATOR element have been refined by the semantics defined by the schema referenced by CARD. This construct is similar to the Warwick Framework, a recognition of separate maintainable and interchangeable packages of descriptive information used in the description of resources. The structural constraints RDF imposes to support the consistent encoding and exchange of standardized metadata provides for the interchangeability of separate packages of metadata defined by different resource description communities.

The RDF Schema

RDF Schemas are used to declare vocabularies, the sets of semantics property-types defined by a particular community. RDF schemas define the valid properties in a given RDF description, as well as any characteristics or restrictions of the property-type values themselves. The XML namespace mechanism serves to identify RDF Schemas.

A human and machine-processable description of an RDF schema may be accessed by de-referencing the schema URI. If the schema is machine-processable, it may be possible for an application to learn some of the semantics of the property-types

named in the schema. To understand a particular RDF schema is to understand the semantics of each of the properties in that description. RDF schemas are structured based on the RDF data model. Therefore, an application that has no understanding of a particular schema will still be able to parse the description into the property-type and corresponding values and will be able to transport the description intact (e.g., to a cache or to another application).

The exact details of RDF schemas are currently being discussed in the W3C RDF Schema working group (SCHEMA). It is anticipated, however, that the ability to formalize human-readable and machine-processable vocabularies will encourage the exchange, use, and extension of metadata vocabularies among disparate information communities. RDF schemas are being designed to provide this type of formalization.

REFERENCES

[1] TPing www.eecs.umich.edu/~azeitoun/tools.html Accessed on:15-04-06

[2] Lowekamp, D. R. O'Hallaron, and T. R. Gross, "Topology Discovery for Large Ethernet Networks", in *Proceedings of ACM SIGCOMM*, San Diego, California, Aug. 2001.

[3] Y. Bejerano, Y. Breitbart, M. Garofalakis, and R. Rastogi, "Physical Topology Discovery for Large Multi-Subnet Networks", July 2002, Bell Labs Tech. Memorandum.

[4] T. Stott, "Snmp-based layer-3 path discovery," Tech. Rep. ALR-2002-005, Avaya Labs Research, Avaya Inc., Basking Ridge, NJ, 2002.

[5] R. Siamwalla, R. Sharma, and S. Keshav, "Discovering Internet topology." http://www.cs.cornell.edu/skeshav/papers/discovery.pdf, 1999.

[6] Yuri Breitbart, Minos Garofalakis, Ben Jai, Cliff Martin, Rajeev Rastogi, and Avi Silberschatz "Topology Discovery in Heterogeneous IP Networks: The *NetInventory* System"

[7] David T. Stott "Layer-2 Path Discovery Using Spanning Tree MIBs" Avaya Labs Research, Avaya Inc. 233 Mount Airy Road Basking Ridge, NJ 07920 March 7, 2002.

[8] Akshay Adhikari, Lorraine Denby, Jean Meloche,Balaji Rao "Operational Layer 3 Topology" Avaya Labs Research, Basking Ridge, NJ _ akshay, ld, jmeloche, August 12, 2003

[9] Kapil Bajaj D. Manjunath "Intranet Topology Discovery Using Untwine" Indian Institute of Technology, Bombay Powai Mumbai 400 076 INDIA

[10] R. Govindan and H. Tangmunarunkit, "Heuristics for Internet map discovery," in *Proc. of the 2000IEEE Computer and Communications Societies Conf. on Computer Communications (INFOCOM-00)*, (Los Alamitos, CA), IEEE, Mar. 26-30, 2000.

[11] SNMP MIB Browser http://www.snmplink.org/ Accessed on:15-03-06

[12] Network View Software http://www.networkview.com/index.html

[13] Types of Networks Accessed on: 19-07-06 aocomputerscience.tripod.com/networks.html

[14] Importance of Networks Accessed on:19-07-06 http://www.ictglobal.com/imp_networks.html

[15] Introduction to RDF Accessed on :15-06-06 www.dlib.org/dlib/may98/miller/05miller.html

[16] RDF Intro http://www.xml.com/pub/a/2001/01/24/rdf.html Accessed on:15-06-06

[17] Bierman and K. Jones, "Physical Topology MIB", Sept. 2000, Internet RFC-2922 (available from http://www.ietf.org/rfc/).

[18] S. Keshav, *"An Engineering Approach to Computer Networking"*. Addison-Wesley Professional Computing Series, 1997.

[19] J. Case, M. Fedor, M. Schoffstall, and J. Davin, "A Simple Network Management Protocol (SNMP)", May 1990, Internet RFC-1157 (available from http://www.ietf.org/rfc/).

[20] W. Stallings, *"SNMP, SNMPv2, SNMPv3, and RMON 1 and 2"*. Addison-Wesley Longman, Inc., 1999, (Third Edition).

[21] Lowekamp, D. R. O'Hallaron, and T. R. Gross, "Topology Discovery for Large Ethernet Networks", in *Proceedings of ACM SIGCOMM*, San Diego, California, Aug. 2001.

[22] Y. Breitbart, M. Garofalakis, C. Martin, R. Rastogi, S. Seshadri, and A. Silberschatz, "Topology Discovery in Heterogeneous IP Networks", in *Proceedings of IEEE INFOCOM'2000*, Tel Aviv, Israel, Mar. 2000.

[23] Y. Bejerano, Y. Breitbart, M. Garofalakis, and R. Rastogi, "Physical Topology Discovery for Large Multi-Subnet Networks", July 2002, Bell Labs Tech. Memorandum.

[24] Youssef Azzana, Fabrice Guillemin, Philippe Robert A Stochastic Model for Topology Discovery of Tree Networks

[25] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos, ìOn power-law relationships of the Internet topology, in *Proceedings of the Conference on Applications, Technologies, Architectures and Protocols for Computer Communications*. 1999, pp. 251,262, ACM Press.

[26] P. Radoslavov, H. Tangmunarunkit, R. Govindan H. Yu, S. Shenker, and D. Estrin, On characterizing network topologies and analyzing their impact on protocol design, Tech. Rep. 03-782, Univ. South. Cal., 2001.

[27] L. Dall'Astra, I. Alvarez-Hameli, A. Barrat, A. Vasquez, and A. Vespignani, ìA statistical approach to the traceroute exploration of

networks: theory and simulations, Available at arXiv:cond-mat/0406404, June 2004.

[28] Anukool Lakhina, John Byers, Mark Crovella, and Peng Xie, Sampling biases in IP topology measurements, in *IEEE Infocom*, San Francisco, 2003.

[29] Internet Topology: Discovery and Policy Impact, Hongsuda Tangmunarunkit, USC-ISI, Ramesh Govindan ICSI , Scott Shenker ACIRI/ICSI

[30] Jangwon Lee, Gustavo de Veciana, Department of Electrical and Computer Engineering University of Texas at Austin, Resource and Topology Discovery for IP Multicast using a Fan-out Decrement Mechanism

[31] V. Paxson. End-to-end Routing Behavior in the Internet. In *Proceedings of the ACM SIGCOMM Symposium on Communication Architectures and Protocols*, San Francisco, CA, September 1996.

[32] V. Paxson. End-to-end Internet Packet Dynamics. In *Proceedings of the 1997 ACM SIGCOMM Conference on Communication Architectures and Protocols*, September 1997.

[33] R. Siamwalla and R. Sharma and S. Keshav. Discovering internet topology.

[34] http://www.cs.cornell.edu/skeshav/papers/discovery.pdf. Accessed on:20-02-06

[35] J. D. Case, M. Fedor, M. Schoffstall, and C. Davin. Simple Network Management Protocol (SNMP). Request for Comments 1157, Internic Directory Services, May 1990.

[36] K. C. Claffy and D. McRobb. Measurement and Visualization of Internet Connectivity and Performance. http://www.caida.org/Tools/Skitter/. Accessed on:19-04-06

[37] Dartmouth University. Intermapper: An intranet mapping and snmp monitoring program for the macintosh. http://www.dartmouth.edu/netsoftware/intermapper. Accessed on:15-04-06

[38] Deborah Estrin, Mark Handley, John Heidemann, Steven McCanne, Ya Xu, and Haobo Yu. Network visualization with the VINT network animator nam. Technical Report 99-703, University of Southern California, March 1999.

[39] Faloutsos, M. Faloutsos, and P. Faloutsos. What does Internet look like? Empirical Laws of the Internet Topology. In Proceedings of ACM SIGCOMM 1999, Boston, MA, September 1999.

[40] V. Fuller, T. Li, J. Yu, and K. Varadhan. Classless Inter-Domain Routing (CIDR): An Address Assignment and Aggregation Strategy. Request for Comments 1519, Internic Directory Services, September 1993.

[41] Ramesh Govindan and Hongsuda Tangmunarunkit. Heuristics for Internet Map Discovery. In *Proceedings of IEEE Infocom*, Tel-aviv, Israel, April 2000.

[42] T. Stott, "Snmp-based layer-3 path discovery," Tech. Rep. ALR-2002-005, Avaya Labs Research, Avaya Inc., Basking Ridge, NJ, 2002.

[43] R. Siamwalla, R. Sharma, and S. Keshav, "Discovering Internet topology." http://www.cs.cornell.edu/skeshav/papers/discovery.pdf, 1999.

[44] W. Zhao, J. J. Li, and D. T. Stott, "A method for heterogeneous network discovery." Internal Technical Report, Avaya Labs Research, Avaya Inc., Dec. 2001.

[45] David T. Stott "Layer-2 Path Discovery Using Spanning Tree MIBs" Avaya Labs Research, Avaya Inc. 233 Mount Airy Road Basking Ridge, NJ 07920 March 7, 2002.

[46] Akshay Adhikari, Lorraine Denby, Jean Meloche,Balaji Rao "Operational Layer 3 Topology" Avaya Labs Research, Basking Ridge, NJ _ akshay, ld, jmeloche, August 12, 2003

[47] R. Siamwalla, R. Sharma, and S. Keshav "Discovering Internet Topology" Cornell Network Research Group Department of Computer Science Cornell University, Ithaca, NY 14853 Yigal Bejerano, Yuri Breitbart_ , Minos Garofalakis, Rajeev Rastogi

[48] "Physical Topology Discovery for Large Multi-Subnet Networks" Bell Labs, Lucent Technologies 600 Mountain Ave., Murray Hill, NJ 07974.

[49] Bruce Lowekamp David R. O'Hallaron Thomas R. Gross "Topology Discovery for Large Ethernet Networks"

[50] Kapil Bajaj D. Manjunath "Intranet Topology Discovery Using Untwine" Indian Institute of Technology, Bombay Powai Mumbai 400 076 INDIA

[51] Java Comparison http://www2.gsu.edu/~matknk/java/reg97-2.htm

[52] Yuri Breitbart, Minos Garofalakis, Ben Jai, Cliff Martin, Rajeev Rastogi, and Avi Silberschatz "Topology Discovery in Heterogeneous IP Networks: The *NetInventory* System"

[53] Dartware http://www.dartware.com/pricing/index.html Accessed on: 27-02-2006

[54] LanSurveyor https://secure.neon.com/order_form.html Accessed on: 15-02-2006

[55] Solar Windshttp://www.asl-solarwinds.co.uk/ Accessed on: 18-02-2006

[56] http://www.networkview.com/index.html Accessed on: 23-02-2006

[57] CISCO SNMP Reference Accessed on: 24-02-2006 http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/snmp.htm

[58] SNMP MIB Descriptions http://www.snmplink.org/ Accessed on: 24-02-2006

[59] SNMP Tutorial http://www2.rad.com/networks/1995/snmp/snmp.htm Accessed on: 23-02-2006

[60] Project Octopus: Network Topology Discovery, http://www.cs.cornell.edu/cnrg/topology_aware/topology/Default.html Accessed on:15-04-06

[61] Overview of Topology Discovery : http://www.cisco.com/en/US/products/sw/cscowork/ps4879/products_user_guide_chapter09186a00801761c2.html Accessed on:15-07-06

[62] Network topology discovery : http://www2.rad.com/networks/2003/rpr/topol.htm Accessed on:15-04-06

[63] http://www.cse.iitb.ac.in/~kapil/untwine/ :Untwine Intranet Topology Discovery Tool Accessed on:15-04-06

[64] Indian Institute of Technology Bombay, India. http://www.eecs.umich.edu/~msim/project_pages/finite_mixture_models.html Accessed on:15-04-06