S N M P – INTRODUCTION

Network Management Requirements:

- Fault Management:
  - Determine the location of fault
  - Isolate the fault and continue working
  - Minimize the impact of the fault
  - Repair or replace the failed component
- Account Management :
  - Established charges for use of services and charge accordingly.
- Configuration and Name Management :
  - Initializing a network and shutting down gracefully
  - Maintaining, adding and updating the relationship among the components and status of the components.
- Performance Management;
  - Facility needed to evaluate the behaviour of managed objects and the effectiveness of the communication activities.
  - Control effectiveness of communication activities at various levels.
- Security Management:
  - Address the security aspects essential for network management and to protect managed objects
  - Protection of target network security, access control of facilities, generating, storing, distributing encryption keys, passwords, authorization control information etc.

Network Management System:

A Network Management System is a collection of tools (hardware and software) for network monitoring and control. It is the incremental hardware and software additions implemented among the existing network components. The software is used in accomplishing the network management tasks residing the host computers and communication processors (bridges, routers, front end processors, cluster controller terminals etc)

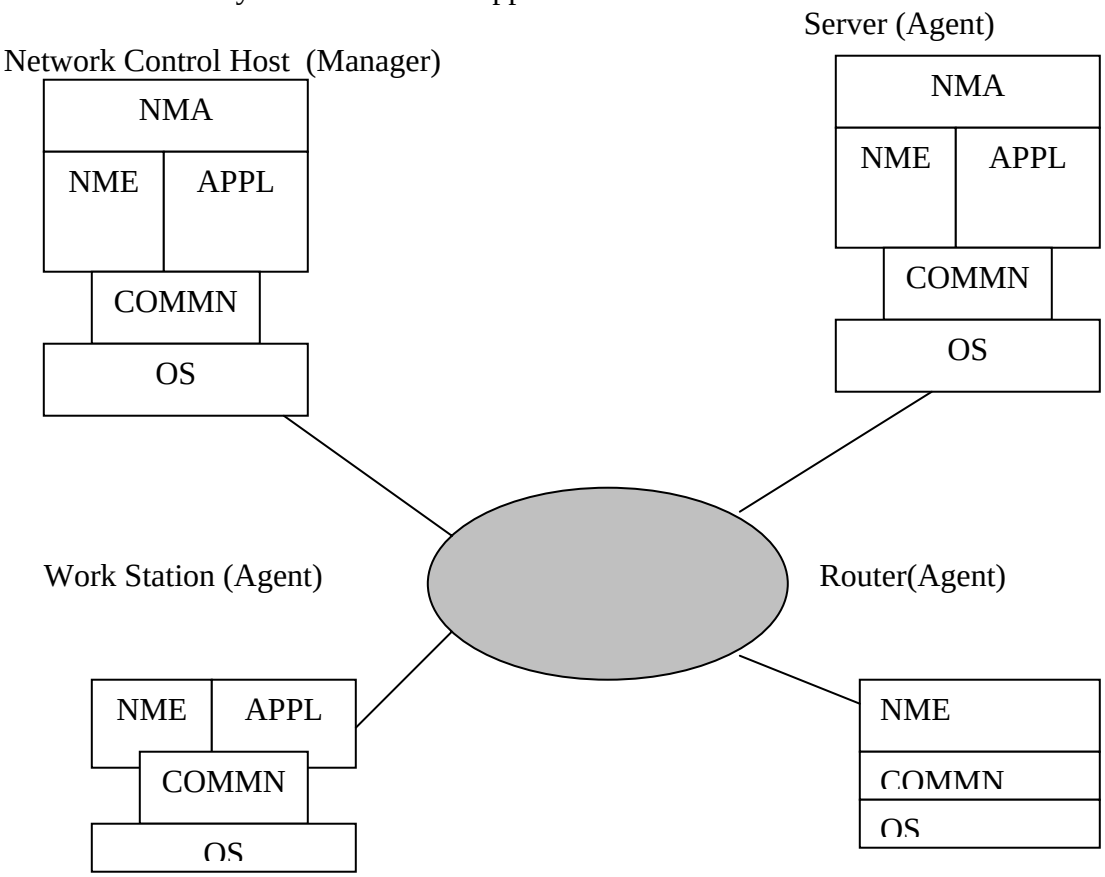
Network Management Configuration:

The architecture of the network management system is shown in the next page.
Each node contains a collection of software devoted to the network management tasks referred as NME – Network Management Entity. Each NME performs the following tasks:

- Collect statistics on communication and network related activities.
- Store statistics locally
- Respond to commands from the **network control center**, including commands to :
  - Transmit collected statistics to the network control center
  - Change parameter
  - Provide status information (parameter values. Active links etc)
  - Generate artificial traffic to perform a test.
- Send message to the network control center when local conditions undergo significant change.

At least one node in the network is designated as the network control host, or manager. IN addition to NME, the network control host includes a collection of software called network

management application (NMA). NMA includes operators interface to allow an authorized user to manage the network. NMA responds to user commands by displaying information and / or by issuing commands to NME through the network. This communication is carried out through an application level network management protocol that employs the communication architecture in the same fashion as any other distributed application.



NMA : Network Management Application; Appl: Application;
NME Network Management Entity ; Comm : Communication software ; OS Operating System.

Important points to remember are:

- Network Management software relies on the host operating system and on the communication architecture.
- Network Control Host communicates with and controls the NMEs in other systems.
- For maintaining high availability on the network management functions, two or more network control hosts are used. One may be collecting statistics or even idle where as other may be used for control. IF the primary network fails, the back ups system can be used.

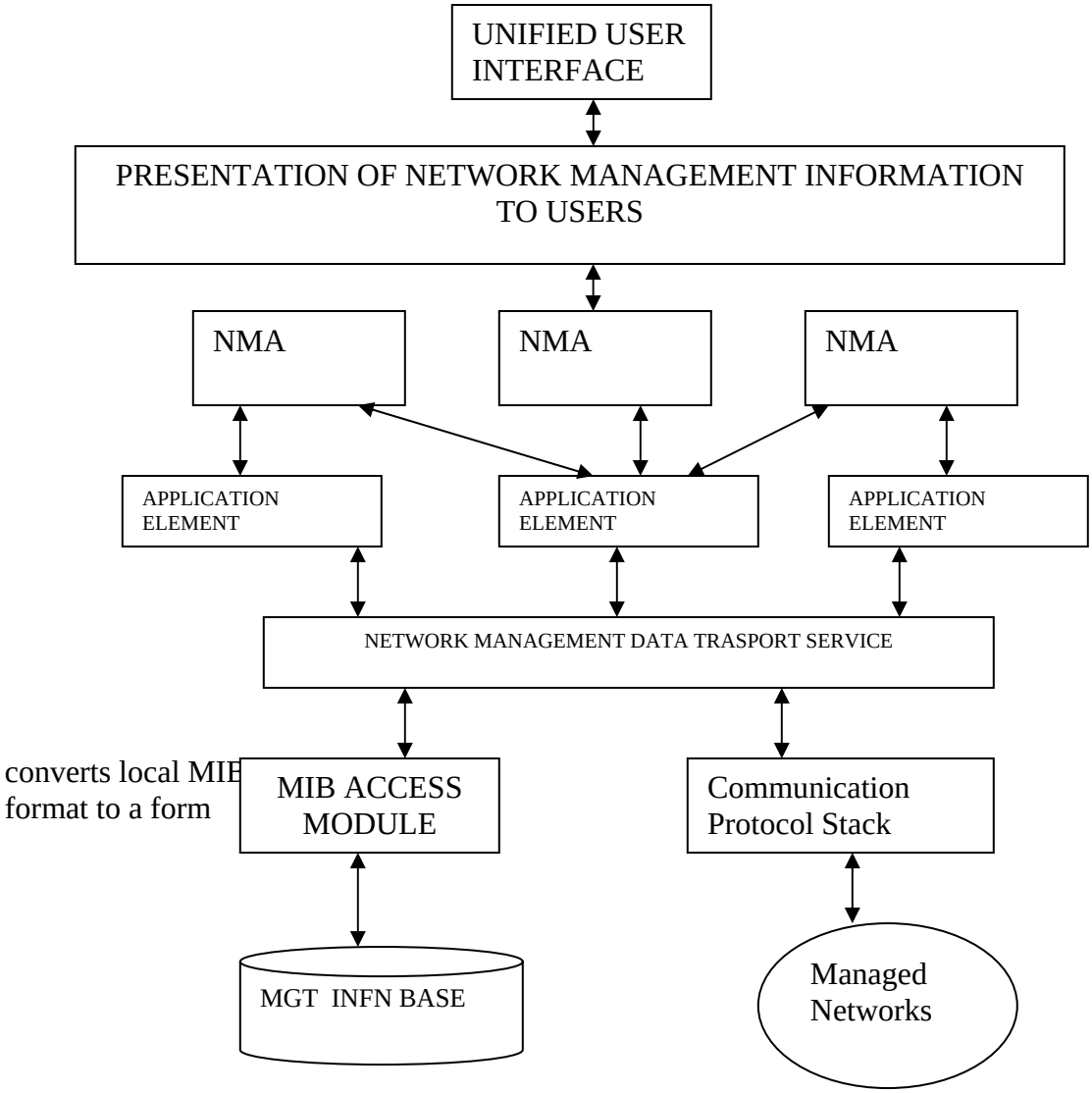
Network Management Software Architecture can be divided into three broad categories;

- User Presentation software:

Interaction between user and NM Software takes place across user interface. Such interface allows the manager to monitor and control the network

Such interface has to unified user interface at any node regardless of vendor.

- Network Management Software:
  This normally consists of three layers. Top layer consists of collection of network management application that provide the services of interest to users – like fault management, configuration management, performance management and security
- The small number of network management applications is supported by a larger number of application elements. There are the modules that implement more [primitive and more general purpose network management functions such a as generating alarms summarizing data etc. The application elements implements basic tools that are of use tone or more of the network management applications.
- Lowest level of management specific software is a network management data transport service. This module consists of a network management protocol used exchange management information among managers and agents and service interface to the application elements



ARCHITECTURAL MODEL OF NETWORK MANAGEMENT SYSTEM

S N M P - NETWORK MANAGEMENT CONCEPTS

Network Management Architecture:

The model of network management that is used for TCP / IP network management includes the following key elements.

- Management Station
- Management Agent
- Management Information Base
- Network Management protocol

Management Station: It is a stand alone device but it may be a capability implemented on a shared platform. IN either case Management Station serves as the interface for the hu8man network manager into the network management system. At a minimum the station will consists of the following:

- A set of management application for data analysis, fault recovery and so on.
- An interface by which the network manager my monitor and control the network.
- The capability of translating the network manager's requirenment into the actual monitoring and control of remote elements in the network
- A database of information extracted from the MIB of all the managed entities in the network.

Management Agent: This is other active element. Platforms like bridges, hosts, routers and hubs may be equipped with SNMP agents so that they may be managed from the management stations. The management agent responds to request for information and actions from the management stations and may asynchronously provide the management stations with important but unsolicited information.

Management Information Base :

Resources in the network may be managed by representing those resources as objects. Each objects is essentially a data variable that represents one aspects of management agent. The collection of object is referred to as a **Management Information Base (MIB)** The MIB functions as a collection of access points at the agent for the management station. These objects are standardized across system of a particular class (common set of objects is used for the management of various bridges). A management station performs the monitoring functions by retrieving the value of MIB objects. An Management station can cause an action to take place at an agent or can change the configuration setting at an agent by modifying the value of specific variables.

Network Management Protocol :

The management stations and agents are linked by Network Management Protocol. The protocol used for the management of TCP / IP networks is SNMP, which includes the following key capabilities:

get : enables the management station to retrieve the value of the objects at the agent.

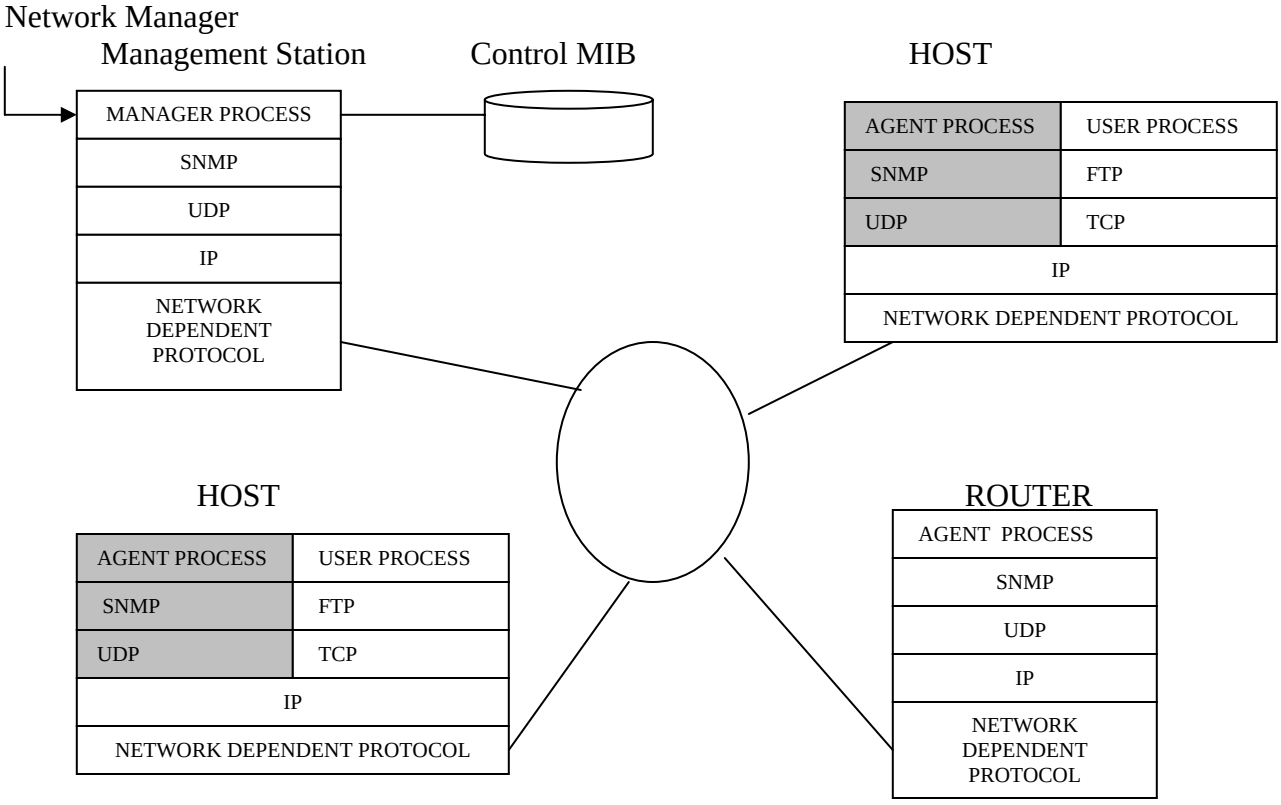
set : Enables the management station to set the4 value of objects at the agent.

trap: enables an agent to notify the management station of significant events.

It is prudent to have two system to perform Management station functions to provide redundancy.

Network Management Protocol Architecture:

SNMP was designed to be an application level protocol that is part of TCP / IP protocol suite. It operates over UDP. Following figure shows the typical configuration of SNMP Protocol:



CONFIGURATION OF SNMP

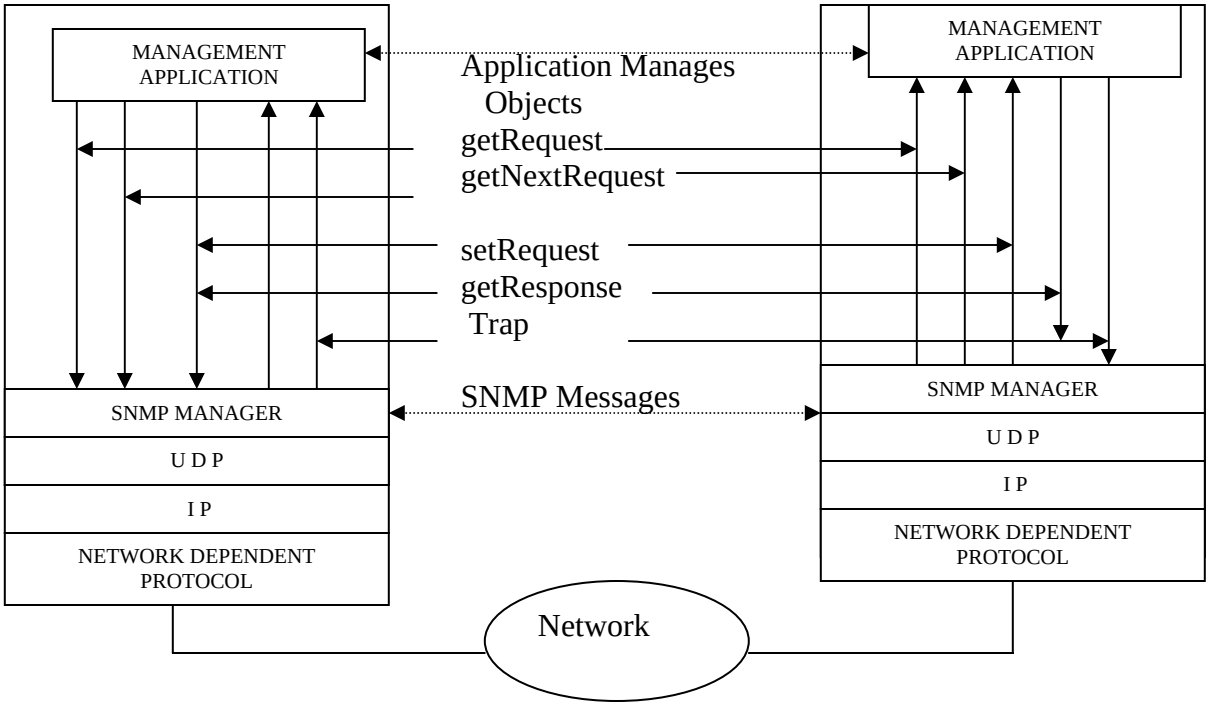
For a stand alone management station, a manager process controls access to a central MIB and provides interface to the network manager. The manager process achieves network management by using SNMP which is implemented on top of UDP, IP and the relevant network dependent protocol (Ethernet, FDDI, and X.25).

Each agent implements SNMP, UDP and IP. For an agent device that supports other application such as FTP, both TCP and UDP are required. Shaded area represents the support provided to network management functions.

Following figure provides further elaborate look at the protocol context on SNMP. From management station three types of SNMP messages are issued on behalf of a management application : getRequest, getNextRequest, setRequest. The first two are variation of get function. All the three messages are acknowledged by the agent in the form of getResponse message. In addition a agent may issue a tap message in response to an event that affects the MIB and the underlying managed resources.

As SNMP relies on UDP, SNMP by itself is connectionless protocol. No ongoing connections are maintained between a management station and its agents. Instead, each exchange is a separate transaction between a management station and an agent.

Role of SNMP



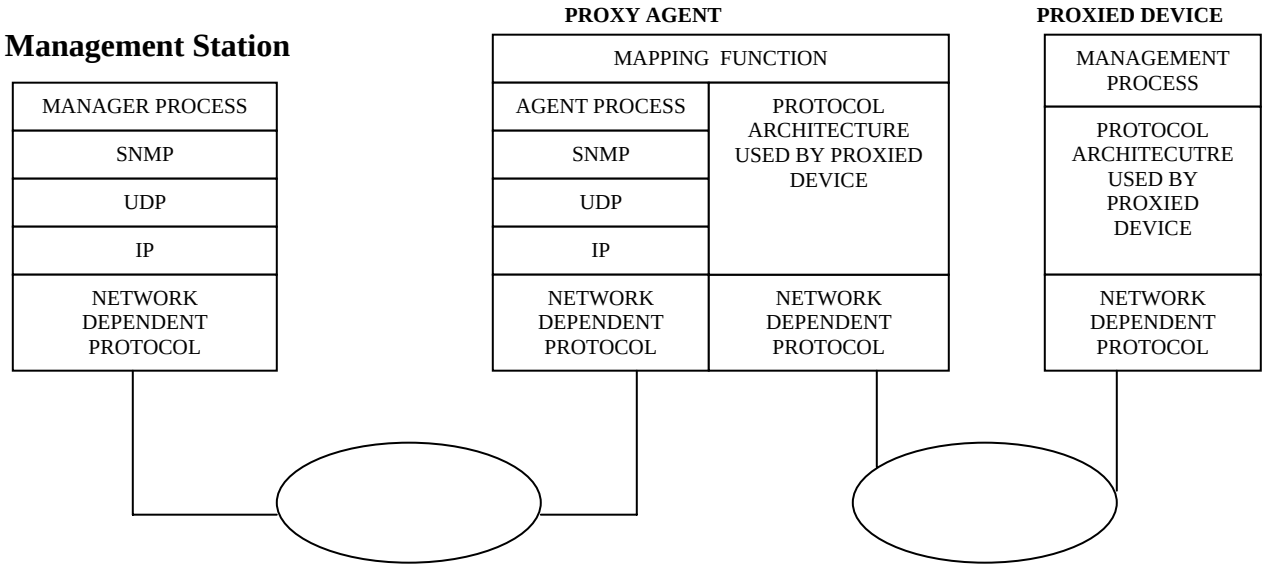
Trap Directed Polling :

If a management station is responsible for a large number of agents and if each agent maintains a large number of objects, then it becomes difficult for the management station to regularly poll agents for all of their readable data. Instead SNMP and associated MIB are designed to encourage the manager to use a technique referred to as ***trap directed polling.***

In this at initialization time, and at frequent intervals (Once a day) a management station can poll all of the agents it knows for some key information such as interface characteristics and for some baseline performance statistics (average number of packets sent) and received over a given period of time. Once this baseline is established, the management station refrains from polling. However any unusual incident is notified by the each agent for example agent crashing and rebooting, link failure, overload condition etc. These events are communicated in SNMP messages known as ***traps***.

Proxies: The use of SNMP requires that all agents, as well as management stations must support a common protocol suite, such as UDP and IP. This limits direct management to such devices such as some bridges , modems that are not part of TCP / IP protocol suit. Similarly a number of PCs, Programmable controllers that do not implement TCP/ IP for which it is not desirable to add SNMP, agent logic and MIB maintenance etc.

To accommodate devices that do not implement SNMP, the concept of proxy was developed. In this scheme an SNMP agent acts as a proxy for one or more other devices. This is shown in the following figure.



Above figure shows that management stations sends queries concerning a device to its proxy agent. The proxy agent converts each query into the management protocol that the device is using. When the agent receives a reply to a query, it passes that reply back to the management station. Similarly, if an event notification of some sort from which the device is transmitted to the proxy, the proxy sends that on to the management station in the form of trap message.

S N M P MANAGEMENT INFORMATION

- The foundation of Network Management System is creation of database that contains information about the elements to be managed.
- An MIB is an structured collection of information about objects that are part of the network(servers, workstations, routers, bridges etc.)
- Each system in a network maintains a MIB that reflects the status of the managed resources at that system. NME (Network Management Entity) can monitor the resources at that system by reading the values of objects in the MIB and may control the resources at that system by modifying those values.

To serve these needs, the MIB must meet certain objectives:

- Data base where manageable objects are defined.
- The objects or objects used to represent a particular resources must be the same at each system. (Keeping data regarding active, passive or total open connections with any two of these data which must be uniform)
- A common scheme for representation must be used to support interoperability.

The first points details the objects types and the second point details the type of structure for uniformity.

Structure Management Information

Structured Management Information explains "How to write and define MIB". The SMI defines the general framework within which a MIB can be defined and constructed . The SMI identifies the data type that can be used in the MIB and specifies how within MIB are represented and named.

For the sake of simplicity, SMI must do the following:

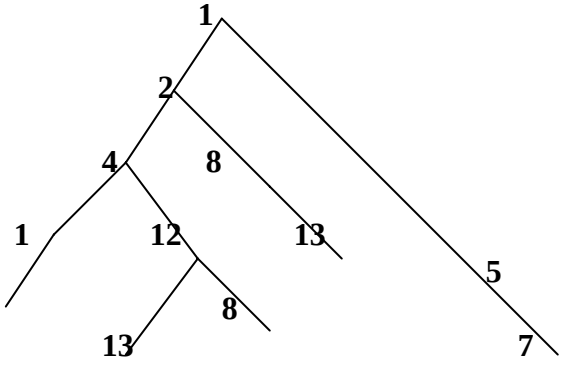
- Provide a standardized technique for defining the structure of a particular MIB
- Provide standardized technique for defining individual objects, including the systems and the value of each object.
- Provide a standardized technique for encoding object values.

MIB Structure

- The Internet Naming Hierarchy
- Objects Types
- Simple/Tabular Objects
- Instances Identification

The Internet Naming Hierarchy

All managed objects in the SNMP environment are arranged in a hierarchical or tree structure. The leaf objects of the tree are the actual managed objects, each of which represents some resources, activity or related information that is to be managed. The tree structure itself defines a grouping of objects into logically related sets. Each object is named by the sequence of the identifiers from the root to the object



The object identifier is : 1.2.4.12.3

Object Types:

A restricted subset of ASN.1 is used to describe objects types

Two ASN.1 classes are used :

Universal Types Application Independent

Application-Wide Types :

- Defined in the context of a particular application
- Each application, including SNMP, is responsible for defining its own application-wide data types

The

Following data types are permitted :

Integer (ex. : 5, - 10)
Octet string (ex. : protocol)
Null (object with no value associated)
Object identifier (ex. : 1.3.6.1.2)
And the constructor type (used to build tables) : Sequence, Sequence- of

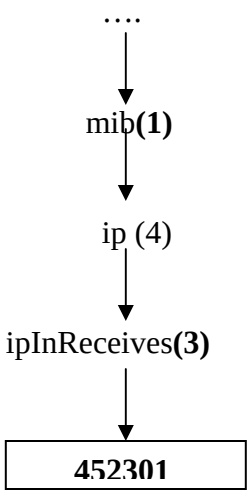
RFC 1155 defines the following application-wide data types :

Network address , IP address : Internet 32- bit address
Counter : Non- negative integer (can be incremented but not decremented)
Gauge : Non- negative integer that may increase or decrease
Timeticks : Non- negative integer counting the time in hundredths of second
Opaque : Arbitrary data transmitted in the form of an octet string

Simpler And Tabular Objects

Simple Objects : Object with a unique instance within the agent.
Its type is one of the following : integer, octet string, null, object identifier, network address, IP address, counter, gauge, time ticks or opaque.

Example: The ipInreceives object has one instance

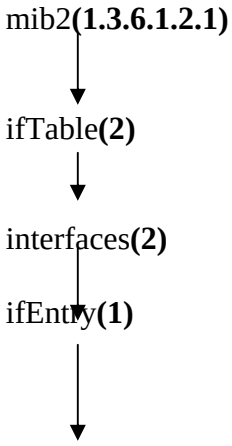


ipInReceives object has one instance.

Tabular Objects :

Two-dimensional table containing zero or more rows .
Each row is made of one or more simple objects (components).
One or more components are used as indexes to unambiguously identifying the rows
The definition of tables is based on ASN.1 types “Sequence" and "Sequence- of "ASN.1 type.

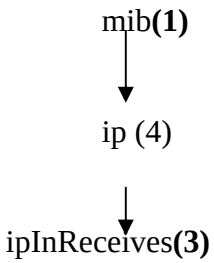
• The table is indexed by ifIndex.
•Each row is an instance of the ifIndex, ifPhysAddress and ifAdminStatus objects



ifIndex**(1)**	ifPhysAddress**(6)**	ifAdminStatus**(7)**	
1	**00:00:39:20:04**	**1 (up)**	row **1**
2	**08:00:56:16:11**	**3 (testing)**	**r**ow **2**
3	**00:00:b4:02:33**	**2 (down)**	row **3**

Instance Identifier:

Instance identifier = Object identifier + 0



...

Object	**Instance identifier**
ipInReceives	mib.4.3.0

The internet node has the object identifier value of 1.3.6.1. This value serves as the prefix for the nodes at the next lower level of the tree.

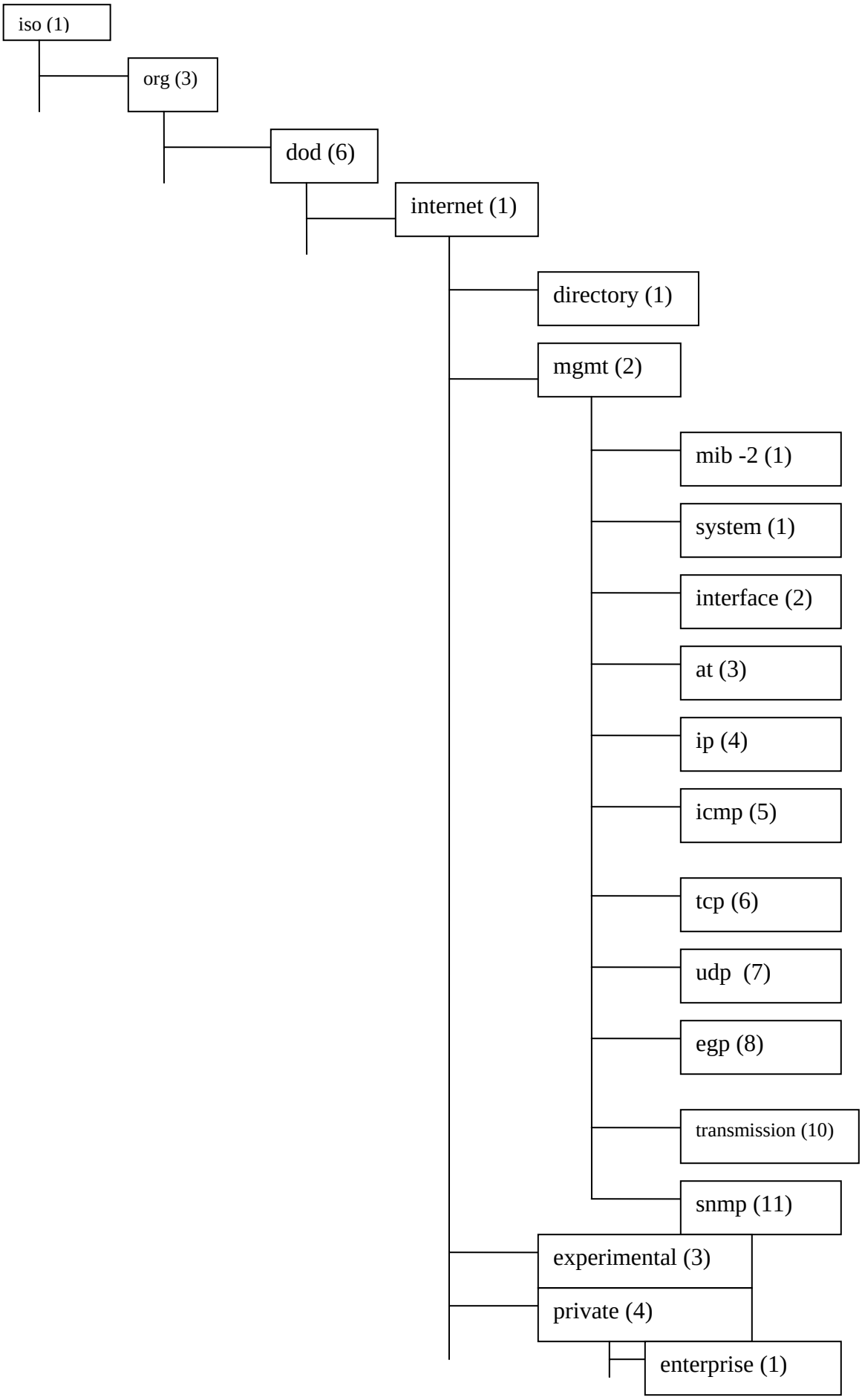
The SMI document defines four nodes under the internet node:

directory: reserved for future use with OSI directory

mgmt : used for objects defined in IAB approved document. (Internet Activity Board)

experimental : Used to identify objects used in internet experiments.

private : used to identify objects defined unilaterally.


iso (1)
org (3)
dod (6)
internet (1)
directory (1)
mgmt (2)
mib -2 (1)
system (1)
interface (2)
at (3)
ip (4)
icmp (5)
tcp (6)
udp (7)
egp (8)
transmission (10)
snmp (11)
experimental (3)
private (4)
enterprise (1)

The **mgmt** subtree consists of the definition of management information Bases that have been approved by the IAB (Internet Activity Board). At present two version of the MIB have been developed, **mib -1** and **mib-2.** The second MIB is an extension of the first. Both are provided with the same object identifier in the subtree . Additional objects can be defined by

- By expanding mib-2
- By creating experimental mib
- By creating private extensions under private tree structure.

MIB I defined 114 objects in 8 groups where as MIB II defined 173 objects with 10 groups.

MIB II object grouping is given above in the tree format: The only addition with respect to MIB I were addition **transmission** and snmp **node** objects as shown. MIB I was issued as RFC 1156 and the MIB II was defined in the RFC 1213. In this some additional object group are added. The mib II group is sub divided into the following groups.

- **system**: overall information about the system.
- **interfaces**: information about each of the interfaces from the system to a sub network
- **at** : address translation: description of address translation table for the internet to subnet address mapping
- **ip** : information related to the implementation and execution experience of the IP on this system.
- **icmp**: information related to the implementation and execution experience of ICMP on this system.
- **tcp**: information related to the implementation and execution experience of UDP on this sytem.
- **egp**: information related to the implementation and execution experience of EGP (External Gateway Protocol) on this system.
- **dot3 (transmission)**: information about the transmission schemes and access protocols at each system interface.
- **snmp**: information related to the implementation and execution experience of SNMP on this system.

SIMPLE NETWORK MANAGEMENT PROTOCOL

The operation that are supported in SNMP are the alteration and inspection of variables. The three general purpose operations may be performed on scalar objects.

- Get : A management station retrieves a scalar object value from a amanged station.
- Set: A management station updates a a scalar object value in a managed station.
- Tap: A managed station sends an unsolicited scalar object value to a management station.

Few points to understand in this respect are :

- It is not possible to change the structure of a MIB by adding or deleting object instances – that is addition and deletion of a row of a table is not possible.
- It is not possible to issue commands for an action to be performed.
- Access is provided only to an leaf object in the object identifiertree.
- It is not possible to access an entire table or row of a table with one atomic action.

Communities and Community Names:

In SNMP network management, there are a number of managed station that control its won MIB and there are a number of management stations that access some of these agents' MIB as per its requirement. Each MIB managed station controls its won local MIB and must be able to control the use of that MIB by a number of management stations. There are three aspects to this control.

- Authentication service : The managed station may wish to limit the access to the MIB to authorized management stations only.
- Access Policy : The managed stations may wish to give different access privileges to different management stations.
- Proxy Service : A managed station may act as a proxy to other managed stations. This may involve implementing the authentication service and / or access policy for the other managed systems on the proxy system.

An SNMP community is a relationship between an SNMP agent and a set of SNMP managers that defines authentication, access control and proxy characteristics. The managed system establishes one community for each desired combination of authentication, access contro9l and proxy characteristics. Each community is given a unique community name and the management station within that community are provided with and must employ the community name in all get and set opetations. The agent may establish a number of communities with overlapping management station membership.

Authentication Service: In the case of SNMP message, the function of an authentication service would be to assure the recipient that the message is from the source from which it claims to be. The scheme of authentication is that the management station includes the community name which functions like a password. This is although very trivial. But for sensitive application like **set ,** this may trigger an authentication procedure which may involve encryption and decryption procedure.

Access Policy : BY defining a community, the agent limits access to its MIB to a selected set of management stations. BY the use of more than one community, the agent can provide different categories of MIB access to different management stations. The two aspects of this control are :

- SNMP MIB view: a subset of objects with an MIB. Different MIB views may be defined for each community.
- SNMP access control: an element of the set {READ-ONLY, READ-WRITE}. An access mode is defined for each community.

The combination of a MIB view and an access mode is referred to as an SNMP community profile. Thus a community profile consists of a defined subset of the MIB at the agent, plus an access mode for those objects.

MIB ACCESS category	SNMP access Mode	
	Read Only	READ-WRITE
Read only	Available for get and trap operation	
Read write	Available for get and trap operations	Available for get, set, and trap operations
Write only	Available for get and trap operations, but the value is implementation specific	Available for get, set and trap operations, but the value is implementation specific for get and trap operations.
Not accessible	Unavailable	

Proxy Service : Proxy is an SNMP agent that acts on behalf of other devices. Typically other devices do not support SNMP. For each devices that the proxy system represents, it maintains an S?NMP access policy.

Protocol Specification:

With SNMP, information is exchanged between a management station and agent in the form of an SNMP message. Each message includes a version number indicating the version of SNM, a community name to be used for this exchange and one of five types of protocol data units as shown below:

Version	Community	SNMP PDU

SNMP Message Format

PDU Type	**request ID**	**0**	**0**	**variablebindings**

GetRequest PDU, GetNextRequest PDU and setRequest PDU

PDU type	**Request Id**	**Error status**	**Error index**	**Variable bindings**

GetResponse PDU

PDU type	**Enterprise**	**Agent addr**	**Generic trap**	**Specific trap**	**Timestamp**	**Variablebindings**

Trap PDU

Name1	**Value 1**	**Name2**	**Value2**	**…….**	**Name n**	**Value n**

Variable bindings

GetRequest, getNextRequest and setRequest PDU 's have the same format as the getResponse PDU with error-status and error-index fields set to zero. This convention reduces by one the number of different PDU format with which the SNMP entity must deal.

Details of fields are given below:

Version : SNMP version (RFC 1157 is version 1)

Community : A pairing of an SNMP agent with some arbitrary set of SN?MP application entities

Request-id : Unique ID is provided for each request

Error-status: Indicates occurrence of exception while processing a request.
noError (0), tooBIg(1), noSuchNAme(2), badValues(3), readOnly(4), genErr(5)

error-index: When error status is nonzero, may provide additional information by indicating which variable in a list caused the exception.

Variable bindings: A list of variable names and correcsponding values

Enterprise : Type of object generating tap: based on sysObjectID

Agent-addr: Address of object generating trap.

Generic trap: generic trap type: values are coldStart(0), warmStart(1), linkDown(2), linkUp(3), authentication-Failure(4), egpNeighborLoss (5), enterprise-Specific(6)

SpecificTrap: Specific trap code;

Time-stamp: Time elapsed between the last initialization of the network entity and the generation of the trap.

Transmission of an SNMP message: Steps involved:

1. The PDU is constructed, using the ASN.1 structure defined in RFC 1157
2. This PDU is then passed to an authentication service, together with the source and destination transport address and community name. The authentication performs any required transformations for this exchange, such as encryption or the inclusion of authentication code, and returns the result.
3. The protocol entity then constructs a message, consisting of a version field, the community name and the result from step 2
4. The new ASN.1 object is then encoded using the basic encoding rules and passed to the transport service.

REMOTE NETWORK MONITORING(R M O N)

With MIB II, the network manager can obtain information that is purely local to individual devices. That is, in a LAN with a number of devices on it each with an SNMP agent, the manager can learn of the amount of traffic into and out of each device, but cannot easily learn about the traffic on the LAN as a whole. Network monitors are the devices that are traditionally employed to study the network traffic as a whole. They are also called network analyzers, probes etc. They may be standalone devices or may be workstations a server, or a router with additional monitoring functionality. The monitor can produce summary information, including error statistics, such as count of undersized packets and the number of collision performance statistics (packet delivered per second) etc. The monitor may also store packet for later analysis.

For effective network management, they need to communicate with a central network management station. In this context, they are referred to as remote monitors.

RMON Goals:

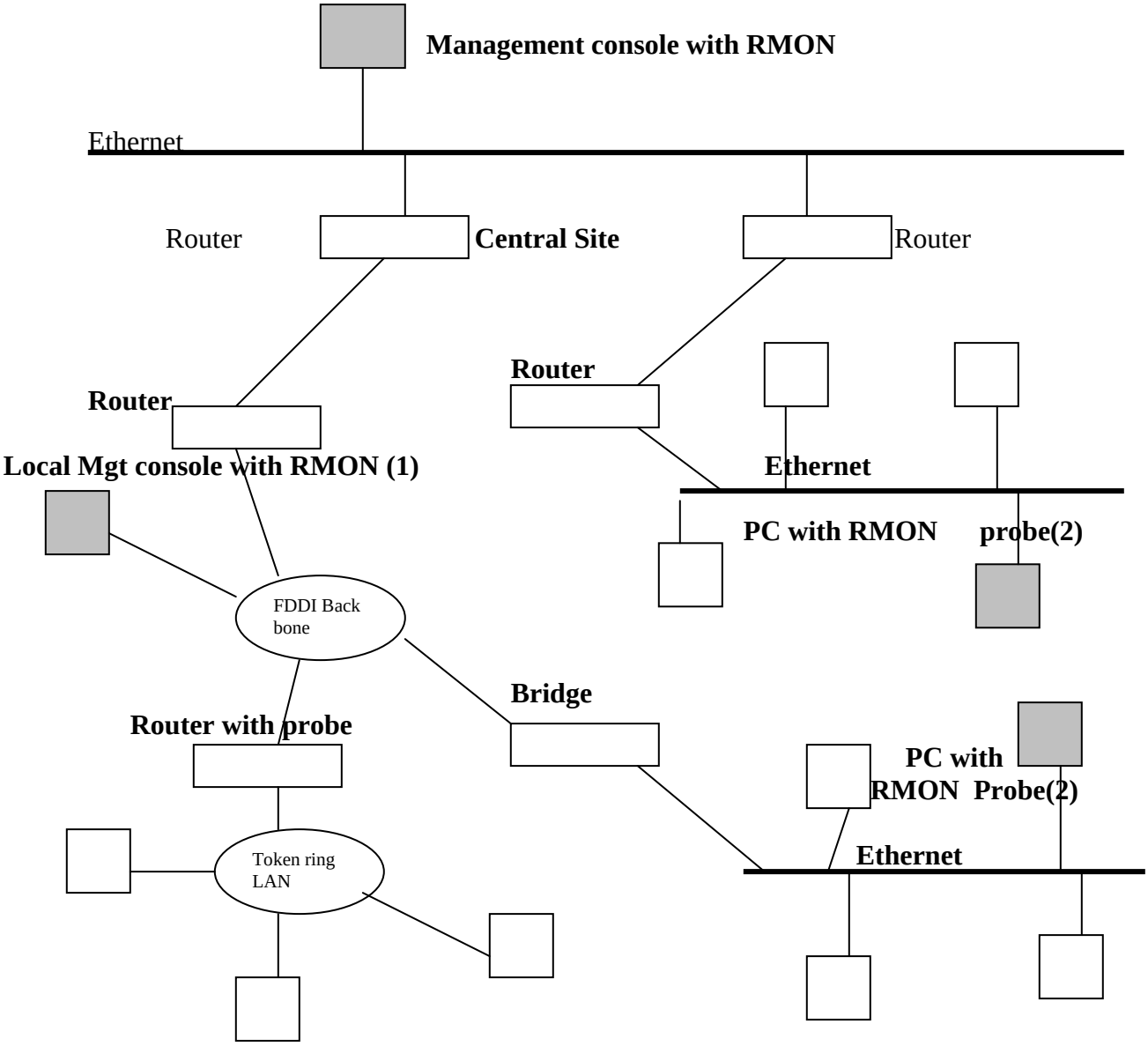
RMON specification is a definition of a MIB. The effect is to define standards network monitoring functions and interfaces for communicating between SNMP based management consoles and remote monitors.

RMON goals as listed in RFC 1757 are given below:

- **Off line operation**: The monitor should collect fault, performance and configuration information continuously. It accumulates statistics that may be retrieved by the manager at a later time. It may also intimate the manager if an exception occurs.
- **Proactive monitoring:** If it is not disruptive, the monitor can run diagnostic and log network performance. In case of failure in the net, the monitor may be able to notify the management station of the failure and provide the management station with information useful in diagnosing the failure.
- **Problem detection and reporting:** Instead of proactive monitoring by running diagnostic tools, it may check for congestion and error condition passively**.** When one of these conditions occur, the monitor can log this condition and attempt to intimate the management station.

- **Value-added data:** The monitor may reduce the work load of management station by performing analysis specific to the data collected on its subnet work.
- **Multiple managers:** An internetworking configuration may have more than one management station to take care of improved reliability, to perform different functions etc. The monitor can be configured to deal with more than one management station concurrently.

Configuration using RMON : Following figure depicts the configuration using RMON.



The top of the figure contains a management station with dedicated RMON capability which is attached to the central LAN. ON the two sub network, RMON MIB is implemented in personal computers.(1 & 2). It may be used exclusively for network performance monitoring or may share other functions. There are other RMON elements that are installed on router as well.

A system that implements the RMON MIB is referred to as RMON probe. The probe has an agent similar to SNMP agent. The probe is capable of reading writing the local RMON MIB in response to management action .

RMON MIB:

The RMON MIB is divided into ten groups

1. **statistics :** Maintains low level utilization and error statistics for each sub net work monitored.
2. **history:** records periodical statistical samples form information available in the statistics group.
3. **alarm**: Allows the management consol user to set a sampling interval and alarm threshold for any counter or integer recorded by the RMON.
4. **host:** contains counters for various types of traffic to and from hosts attached to the subnetwork.
5. **hostTopN** : Contains sorted host statistics that report on the hosts that top a list based on some parameter in the host table.
6. **matrix:** shows error and utilization information in matrix form, so the operator can retrieve information for any pair of network addresses.
7. **filter**: allows the monitor to observe packets that match a filter
8. **packet captures**: governs how data is sent to a management console.
9. **event** : gives a table of all events generated by the RMON probe.
10. **tokenRing :** maintains statistics and configuration information for token ring subnetwork.

SNMP v2 MANAGEMENT INFORMATION

SNMPv2 is a major upgrade over SNMPv1 that expands functionality of SNMP and broadens its applicability to include OSI based as well as TCP/IP based networks. SNMPv1 based on SMI and MIB was quite simple and easy to implement. However, it was not able to take care of the more complex environment containing arbitrary resources and had very low security features. Grouping based on the community name alone for security was inadequate as an attacker can easily observe the message content and find the community name. Because of this SNMPv1 was vulnerable to attacks that can modify or disable a network configuration.

Two different working groups were formed - one to deal with security aspects and other to deal with other aspects including protocol management information. However, over a period of 4 eight years – 1992 - 1996 the issue of security implementation could not be worked out satisfactorily. Hence the SNMPv2 was released without security enhancement.

SNMPv2 can support either a highly centralized network management strategy or a distributed one. IN the latter case, some systems operate in the role of both manager and agent. In its agent role, such a system will accept commands from a superior management system; these commands may deal with access to information stored locally at the intermediate manager or may require the intermediate manager to provide summary information about agents subordinate to itself.

The key enhancements to SNMPv2 are in the following category:

- **Structure of Management Information (SMI)** : This deals with Object definition, Conceptual Tables, Notification definition and information modules.
- **Manager to Manager to capability**.
- **Protocol Operation.**

IN this the macro used to define objects types has been expanded to include several new data types and to enhance the documentation associated with an object. A new convention has been provided for creating and deleting conceptual rows in a table. SNMPv2 MIB contains basic traffic information about operation of the SNMPv2 protocol. It includes two new PDUs.

SNMPv3 : The final form of SNMPv2 contains no provision for security. This deficiency is removed in SNMPv3. The documents (RFC 2271,72,73,74, &75) define a set of security capability and a framework that enables that set to be used with the SNMPv2 or SNMPv1.

ABSTRACT SYNTAX NOTATION ONE (ASN.1)

ABSTRACT SYNTAX NOTATION ONE (ASN.1) IS A FORMAL LANGUAGE FOR ABSTRACTLY DESCRIBING MESSAGES TO BE EXCHANGED AMONG AN EXTENSIVE RANGE OF APPLICATIONS INVOLVING THE INTERNET, INTELLIGENT NETWORK, CELLULAR PHONES, GROUND-TO-AIR COMMUNICATIONS, ELECTRONIC COMMERCE, SECURE ELECTRONIC SERVICES, INTERACTIVE TELEVISION, INTELLIGENT TRANSPORTATION SYSTEMS, VOICE OVER IP AND OTHERS. DUE TO ITS STREAMLINED ENCODING RULES, ASN.1 IS ALSO RELIABLE AND IDEAL FOR WIRELESS BROADBAND AND OTHER RESOURCE-CONSTRAINED ENVIRONMENTS.

EXAMPLE OF A MESSAGE DEFINITION SPECIFIED WITH ASN.1 NOTATION:

```
    Report ::= SEQUENCE {
        author       OCTET STRING,
        title        OCTET STRING,
        body         OCTET STRING,
       }
```

REPORT" IS THE NAME OF THIS TYPE OF MESSAGE. SEQUENCE INDICATES THAT THE MESSAGE IS A SEQUENCE OF DATA ITEMS.

THE FUNDAMENTAL UNIT OF ASN.1 IS THE MODULE. THE SOLE PURPOSE OF A MODULE IS TO NAME A COLLECTION OF TYPE DEFINITIONS AND/OR VALUE DEFINITIONS (ASSIGNMENTS) THAT CONSTITUTE A DATA SPECIFICATION. A TYPE DEFINITION IS USED TO DEFINE AND NAME A NEW TYPE BY MEANS OF A TYPE ASSIGNMENT

AND A VALUE DEFINITION IS USED TO DEFINE AND NAME A SPECIFIC VALUE, WHEN IT IS NECESSARY, BY MEANS OF A VALUE ASSIGNMENT.

THE FIGURE BELOW CONTAINS AN EXAMPLE MODULE. IT IS DEFINED AS A *MODULE REFERENCE* INVENTORYLIST, FOLLOWED BY AN OPTIONAL *OBJECT IDENTIFIER* VALUE 1 2 0 0 6 1 (SEE THE SIMPLE TYPES SECTION.), FOLLOWED BY THE KEYWORD DEFINITIONS, FOLLOWED BY THE OPTIONAL *TAG DEFAULT* (NOT INCLUDED IN THE EXAMPLE), FOLLOWED BY THE ASSIGNMENT CHARACTER SEQUENCE ::= , FOLLOWED BY THE KEYWORDS BEGIN AND END BRACKETING THE *MODULE BODY*.

```
InventoryList {1 2 0 0 6 1} DEFINITIONS ::=
  BEGIN
    {
     ItemId ::= SEQUENCE
       {
        partnumber IA5String,
        quantity INTEGER,
        wholesaleprice REAL,
        saleprice REAL
       }
     StoreLocation ::= ENUMERATED
       {
        Baltimore (0),
        Philadelphia (1),
        Washington (2)
       }
    }
   END
```

Figure: Example of an ASN.1 module.

TYPE ASSIGNMENT:

A TYPE ASSIGNMENT CONSISTS OF A TYPE REFERENCE (THE NAME OF THE TYPE), THE CHARACTER SEQUENCE ::= (‘‘IS DEFINED AS’’), AND THE APPROPRIATE TYPE.

VALUE ASSIGNEMENT

```
gadget  ItemId  ::=
  {
   partnumber      "7685B2",
   quantity        73,
   wholesaleprice  13.50,
   saleprice       24.95
```

```
)
```

BUILT IN TYPE:

SN.1'S BUILT-IN SIMPLE TYPES ARE SHOWN IN THE FOLLOWING TABLE . THE UNIVERSAL CLASS NUMBER (TAG) AND A TYPICAL USE OF EACH TYPE ARE ALSO INCLUDED.

Simple Types	Tag	Typical Use
BOOLEAN	1	Model logical, two-state variable values
INTEGER	2	Model integer variable values
BIT STRING	3	Model binary data of arbitrary length
OCTET STRING	4	Model binary data whose length is a multiple of eight
NULL	5	Indicate effective absence of a sequence element
OBJECT IDENTIFIER	6	Name information objects
REAL	9	Model real variable values
ENUMERATED	10	Model values of variables with at least three states
CHARACTER STRING	*	Models values that are strings of characters from a specified characterset

Type BOOLEAN takes values TRUE and FALSE. Usually, the type reference for BOOLEAN describes the true state.

TYPE INTEGER TAKES ANY OF THE INFINITE SET OF INTEGER VALUES. ITS SYNTAX IS SIMILAR TO PROGRAMMING LANGUAGES SUCH AS C OR PASCAL. IT HAS AN ADDITIONAL NOTATION THAT NAMES SOME OF THE POSSIBLE VALUES OF THE INTEGER. FOR EXAMPLE,

```
ColorType ::= INTEGER
    {
       red      (0)
       white    (1)
       blue     (2)
    }
```

TYPE BIT STRING TAKES VALUES THAT ARE AN ORDERED SEQUENCE OF ZERO OR MORE BITS.

```
OCCUPATION  ::=  BIT STRING
         {
             clerk      (0)
```

```
        editor     (1)
        artist     (2)
        publisher  (3)
    }
```

NAMES THE FIRST BIT ``CLERK", THE SECOND BIT ``EDITOR", AND SO ON. STRINGS OF BITS CAN THEN BE WRITTEN BY LISTING THE NAMED BITS THAT ARE SET TO 1. FOR EXAMPLE, (EDITOR, ARTIST) AND '0110'B ARE TWO REPRESENTATIONS FOR THE SAME VALUE OF ``OCCUPATION".

TYPE OCTET STRING TAKES VALUES THAT ARE AN ORDERED SEQUENCE OF ZERO OR MORE EIGHT-BIT OCTETS.

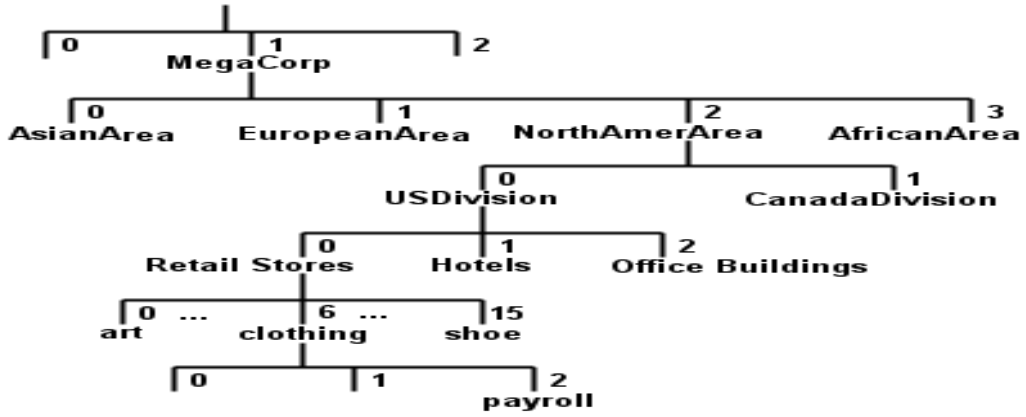
TYPE NULL TAKES ONLY ONE VALUE, NULL. IT CAN BE USED AS A PLACE MARKER, BUT OTHER ALTERNATIVES ARE MORE COMMON.

TYPE OBJECT IDENTIFIER NAMES INFORMATION OBJECTS (FOR EXAMPLE, ABSTRACT SYNTAXES OR ASN.1 MODULES). THE TYPE NOTATION REQUIRES THE KEYWORDS OBJECT IDENTIFIER. THE NAMED INFORMATION OBJECT IS A NODE ON AN OBJECT IDENTIFIER TREE THAT IS MANAGED AT THE INTERNATIONAL LEVEL. ISO, CCITT, OR ANY OTHER ORGANIZATION IS ALLOWED A SUBTREE WHICH THE ORGANIZATION DEFINES. ON EACH LEVEL J OF THE OBJECT IDENTIFIER TREE, NODES ARE NUMBERED 0,1,2,.... A LIST OF POSITIVE NUMBERS, ENCLOSED IN BRACES AND ORDERED BY LEVEL STARTING FROM THE ROOT, UNIQUELY IDENTIFIES AN INFORMATION OBJECT AT A NODE OF THE TREE. THIS ORDERED LIST OF POSITIVE NUMBERS DELIMITED BY BRACES IS THE VALUE NOTATION FOR TYPE OBJECT IDENTIFIER.

THE FOLLOWINGFIGURE ILLUSTRATES THE CONCEPT OF AN OBJECT IDENTIFIER TREE. FOR EXAMPLE, IN THE SUBTREE WITH ROOT ``RETAILSTORES'' THE INFORMATION OBJECT ``PAYROLL'' HAS LOCAL VALUE 0 6 2. MORE FORMALLY, IF

```
CLOTHINGTYPE  ::=  OBJECT IDENTIFIER    THEN
PAYROLL  CLOTHINGTYPE  ::=  {0 6 2}.
```

IF THE RETAIL STORES ARE CONSIDERED AS PART OF AN INTERNATIONAL ``MEGACORP'' THEN 1 2 0 0 6 2 UNIQUELY IDENTIFIES ``PAYROLL''.



TYPE ENUMERATED IS SIMILAR TO THE INTEGER TYPE, BUT NAMES SPECIFIC VALUES ONLY. FOR EXAMPLE,

```
ColorType ::= ENUMERATED
    {
       red      (0)
       white    (1)
       blue     (2)
    }
```

HAS THE SAME INTERPRETATION AS IN THE TYPE INTEGER EXAMPLE NEAR THE BEGINNING OF THIS SECTION, EXCEPT THAT COLORTYPE CAN TAKE ONLY THE VALUES SPECIFICALLY IN THE LIST; THAT IS, NO OTHER VALUES THAN 0 FOR ``RED", 1 FOR ``WHITE", OR 2 FOR ``BLUE".

TYPE CHARACTER STRING TAKES VALUES THAT ARE STRINGS OF CHARACTERS FROM SOME DEFINED (ISO- OR CCITT-REGISTERED) CHARACTER SET.

Character String Type	Tag	Character Set
`NumericString`	18	0,1,2,3,4,5,6,7,8,9, and space
`PrintableString`	19	Upper and lower case letters, digits, space, apostrophe, left/right parenthesis, plus sign, comma, hyphen, full stop, solidus, colon, equal sign, question mark
`TeletexString (T61String)`	20	The Teletex character set in CCITT's T61, space, and delete
`VideotexString`	21	The Videotex character set in CCITT's T.100 and T.101, space, and delete

VisibleString (ISO646String)	26	Printing character sets of international ASCII, and space
IA5String	22	International Alphabet 5 (International ASCII)
GraphicString	25	All registered G sets, and space
GraphicString	27	All registered C and G sets, space and delete

STRUCTURED TYPES

ASN.1'S BUILT-IN STRUCTURED TYPES ARE SHOWN IN THE FOLLOWING TABLE. THE UNIVERSAL CLASS NUMBER (TAG) AND A TYPICAL USE OF EACH TYPE ARE ALSO INCLUDED.

Structured Types	Tag	Typical Use
SEQUENCE	16	Models an ordered collection of variables of different type
SEQUENCE OF	16	Models an ordered collection of variables of the same type
SET	17	Model an unordered collection of variables of different types
SET OF	17	Model an unordered collection of variables of the same type
CHOICE	*	Specify a collection of distinct types from which to choose one type
SELECTION	*	Select a component type from a specified CHOICE type
ANY	*	Enable an application to specify the type Note: ANY is a deprecated ASN.1 Structured Type. It has been replaced with X.680 Open Type.

Type SEQUENCE is an ordered list of zero or more component types. The type notation requires braces around the list and permits a local identifier preceding the list to act as the name of the sequence type.

```
AirlineFlight  ::=  SEQUENCE
   {
    airline   IA5String,
    flight    NumericString,
```

```
seats     SEQUENCE
              {
               maximum   INTEGER,
               occupied  INTEGER,
               vacant    INTEGER
              },
airport   SEQUENCE
              {
               origin              IA5String,
               stop1       [0]     IA5String  OPTIONAL,
               stop2       [1]     IA5String  OPTIONAL,
               destination         IA5String
              },
 crewsize ENUMERATED
              {
                six    (6),
                eight  (8),
                ten    (10)
              },
 cancel   BOOLEAN    DEFAULT FALSE
}.
```

THIS INSTANCE OF AIRLINEFLIGHT INDICATES THAT AMERICAN AIRLINES FLIGHT 1106 FLIES NON-STOP FROM BALTIMORE-WASHINGTON AIRPORT TO LOS ANGELES. THE AIRPLANE REQUIRES A CREW OF 10 PEOPLE, HAS 320 SEATS, OF WHICH 107 ARE FILLED AND 213 ARE EMPTY. THE FLIGHT IS NOT CANCELED. TWO COMPONENTS, STOP1 AND STOP2 OF THE SEQUENCE TYPE AIRPORT ARE TAGGED WITH THE CONTEXT-SPECIFIC TAGS [0] AND [1] TO AVOID AMBIGUITY DUE TO CONSECUTIVE OPTIONAL COMPONENTS NOT HAVING DISTINCT TYPES. WITHOUT THE TAGS, THE DEFINITION OF AIRPORT WOULD BE INVALID IN ASN.1.

TYPE SEQUENCE OF IS SIMILAR TO SEQUENCE, EXCEPT THAT ALL VALUES IN THE ORDERED LIST MUST BE OF THE SAME TYPE. FOR EXAMPLE, THE SEATS TYPE IN THE ABOVE EXAMPLE COULD BE SEQUENCE OF INTEGER INSTEAD OF SEQUENCE.

TYPE SET TAKES VALUES THAT ARE UNORDERED LISTS OF COMPONENT TYPES. THE TYPE AND VALUE NOTATIONS FOR SET ARE SIMILAR TO SEQUENCE, EXCEPT THAT THE TYPE OF EACH COMPONENT MUST BE DISTINCT FROM ALL OTHERS AND THE VALUES CAN BE IN ANY ORDER.

```
Person  ::=  SET
```

```
{
 name      IA5String,
 age       INTEGER,
 female    BOOLEAN
}.
```

TYPE SET OF TAKES VALUES THAT ARE UNORDERED LISTS OF A SINGLE TYPE. THE SEQUENCE TYPE EXAMPLE ABOVE WOULD BE VALID IF THE SEATS TYPE WERE SET OF INTEGER INSTEAD OF SEQUENCE.

Type CHOICE takes one value from a specified list of distinct types.

```
Prize  ::=  CHOICE
  {
   car        IA5String,
   cash       INTEGER,
   nothing    BOOLEAN
  }.
```

Type SELECTION enables the user to choose a component type from a specified CHOICE type.

```
Winner  ::=  SEQUENCE
  {
   lastName    VisibleString,
   ssn         VisibleString,
   cash   <    Prize
  }
```

TAGGED

TYPE TAGGED IS USED TO ENABLE THE RECEIVING SYSTEM TO CORRECTLY DECODE VALUES FROM SEVERAL DATATYPES THAT A PROTOCOL DETERMINES MAY BE TRANSMITTED AT ANY GIVEN TIME.

ITS TYPE NOTATION CONSISTS OF THREE ELEMENTS: A USER-DEFINED TAG, FOLLOWED BY THE VALUE NOTATION OF THE TYPE BEING TAGGED.

THE USER-DEFINED TAG CONSISTS OF A CLASS AND CLASS NUMBER CONTAINED IN BRACES. CLASS IS UNIVERSAL, APPLICATION, PRIVATE, OR CONTEXT-SPECIFIC. THE UNIVERSAL CLASS IS RESTRICTED TO THE ASN.1 BUILT-IN TYPES. IT DEFINES AN APPLICATION-INDEPENDENT DATA TYPE THAT MUST BE DISTINGUISHABLE FROM ALL OTHER DATA TYPES. THE OTHER THREE CLASSES ARE USER DEFINED. THE APPLICATION CLASS DISTINGUISHES DATA TYPES THAT HAVE A WIDE, SCATTERED USE WITHIN A PARTICULAR PRESENTATION CONTEXT. PRIVATE DISTINGUISHES DATA TYPES WITHIN A

PARTICULAR ORGANIZATION OR COUNTRY. CONTEXT-SPECIFIC DISTINGUISHES MEMBERS OF A SEQUENCE OR SET, THE ALTERNATIVES OF A CHOICE, OR UNIVERSALLY TAGGED SET MEMBERS. ONLY THE CLASS NUMBER APPEARS IN BRACES FOR THIS DATA TYPE; THE TERM COONTEXT-SPECIFIC DOES NOT APPEAR.

Character String Types

The most common Character String Types are listed as follows:

Character Type	Tag	Description
`BMPString`	30	Basic Multilingual Plane of ISO/IEC/ITU 10646-1
`IA5String`	22	International ASCII characters (International Alphabet 5)
`GeneralString`	27	all registered graphic and character sets plus SPACE and DELETE
`GraphicString`	25	all registered G sets and SPACE
`NumericString`	18	1, 2, 3, 4, 5, 6, 7, 8, 9, 0, and SPACE
`PrintableString`	19	a-z, A-Z, ' () +,-.?:/= and SPACE
`TeletexString`	20	CCITT and T.101 character sets
`UniversalString`	28	ISO10646 character set
`UTF8String`	12	any character from a recognized alphabet (including ASCII control characters)
`VideotexString`	21	CCITT's T.100 and T.101 character sets
`VisibleString`	26	International ASCII printing character sets

Macros in ASN.1 are similar to macros in application software, they provide the capability of defining types and values that are not included in the standard repertoire One significant use of ASN.1 macros is in OSI application protocol standards, specifically for defining remote operations and object classes. In this section, we include two macros, ERROR and OPERATOR, that appear in the common service elements

```
<macro name> MACRO ::=
     BEGIN
```

```
  TYPE NOTATION    ::=   <user-defined type notation>
  VALUE NOTATION   ::=   <user-defined value notation>
<supporting syntax>
END
```

THE FOLLOWING ERROR MACRO DEFINED IN X.219 PROVIDES A SPECIFIC INSTANCE OF THE GENERAL TEMPLATE.

```
ERROR  MACRO  ::=
BEGIN
  TYPE NOTATION    ::=    Parameter
  VALUE NOTATION   ::=    value (VALUE CHOICE
                                 {
                                  localValue   INTEGER,
                                  globalValue  OBJECT IDENTIFIER
                                 })
  Parameter        ::=    ``PARAMETER'' NamedType | empty
  NamedType        ::=    identifier type  |  type
END
```

IN THIS DEFINITION, DETAILS OF PARAMETER AND NAMEDTYPE ARE IN THE SUPPORTING SYNTAX. PARAMETER CONSISTS OF THE KEYWORD ``PARAMETER'' FOLLOWED BY A NAMED TYPE; IT MAY NOT HAVE AN ENTRY. THE VALUE NOTATION IS A CHOICE OF INTEGER OR OBJECT IDENTIFIER. THE DEFINITION ALLOWS USERS TO DEFINE OPERATION ERRORS. FOR EXAMPLE, THE ERROR MACRO IS USED IN THE REMOTE OPERATIONS SERVICE ELEMENT (ROSE) OF A FOLLOWING CHAPTER TO DEFINE BADQUEUENAME AS FOLLOWS:

```
BadQueueName   ERROR
               PARAMETER   QueueName
                 ::= 0
```

BadQueueName HAS TYPE ERROR, ONE PARAMETER ``queuename'' (IDENTIFIED ELSEWHERE AS TYPE IA5STRING), AND VALUE 0. IN THE REMOTE OPERATION, ONLY THE VALUE 0 IS TRANSMITTED, THE OTHER TERMS IN THE DEFINITION ARE FOR THE USER'S BENEFIT.