ESCUELA POLITECNICA NACIONAL
FACULTAD DE INGENIERIA ELECTRICA

LA TECNICA DE OBJETOS APLICADA A LA GESTION DE REDES DE TELECOMUNICACIONES

Tesis previa a la obtención del Título de
Ingeniero en Electrónica y Telecomunicaciones

Carlos Alfonso Herrera Muñoz

Quito, Octubre de 1999

Certifico que el presente trabajo ha sido realizado en su totalidad por el señor Carlos Alfonso Herrera Muñoz.

Ing. Carlos Egas A.
DIRECTOR DE TESIS

AGRADECIMIENTO

A todas las personas que contribuyeron directa o indirectamente en la realización de la presente tesis, especialmente al Ing. Carlos Egas por su desinteresada y decidida colaboración y además a todos mis compañeros de trabajo que me motivaron a la culminación de este trabajo.

INDICE

INTRODUCCION

La tendencia que en la actualidad predomina en el campo de la gestión de redes es la optimización de los recursos de la red, tanto en hardware como en software. La integración de redes que utilizan equipos de diferentes fabricantes complica la realización de una adecuada administración, debiendo tomarse en cuenta la diversidad de productos y herramientas que se utilizan para este propósito. Por esta razón es necesario estandarizar los elementos de hardware y software que permita optimizar la tarea de gestión de redes.

Actualmente las empresas poseen redes dedicadas a la transmisión de: voz, datos y vídeo. Estas redes cuentan con sistemas dedicados e individuales de gestión, lo cual genera costos elevados en la realización de las tareas de gestión y dificulta procesos tales como: monitoreo, configuración, planificación, y detección de fallas en las redes de telecomunicaciones.

El propósito de este trabajo es utilizar la técnica de objetos para desarrollar una base de información de gestión utilizando en el modelo de gestión Internet, y dejar abierto el campo de aplicación de esta técnica en otras aplicaciones que requieren gestión de recursos de red.

En el capítulo 1 se pone a consideración todas las definiciones básicas de gestión de red, como son conceptos básicos, el modelo de gestión de red sobre el cual se

realiza la descripción de una base de información de gestión indicando el protocolo que se va a utilizar.

En el capítulo 2 se describe los conceptos que maneja la técnica orientada a objetos y su aplicabilidad a las arquitecturas de gestión de red.

En el capítulo 3 se describe las definiciones de una base de información de gestión utilizada en la arquitectura Internet, además de su estructura y el estándar ASN.1 que permite describir la sintaxis de una MIB. También se explica la ubicación de las MIB privadas dentro del nodo Internet.

En el capítulo 4 se desarrolla una base de información de gestión privada, que permite administrar un Repetidor de Radioenlace, en el cual se aplican todos los conceptos definidos en los capítulos anteriores.

diferentes estándares de gestión de red para la Interconexión de Sistemas Abiertos (O.S.I.)

Otra de las organizaciones es la Junta de la Arquitectura de Internet (Internet Architective Board I.A.B), la cual se encarga de las políticas técnicas para la estandarización en Internet, resultado de su labor es entre otras cosas, los estándares para gestión de red en la arquitectura TCP/IP, tales como el Protocolo Simple de Gestión de Red (SNMP) y el Protocolo CMIS/CMIP. Todos los estándares propuestos son publicados en archivos de texto denominados "Requerimientos para Comentar" o sus siglas en inglés RFC (Request for Comments).

Para realizar las tareas de gestión de redes es necesario definir una arquitectura de gestión de red y definir los estándares protocolares, para obtener, entre otras cosas, herramientas de trabajo dinámicas que faciliten y optimicen los procesos de gestión de redes.

Aunque algún trabajo ya había sido realizado por la ISO, con el propósito de producir normas de gestión de red, no fueron aceptadas ni utilizadas por la mayoría de organismos y empresas como un estándar para gestionar redes debido, entre otras cosas, a la gran cantidad de maneras de realizar gestión de redes.

En la actualidad se está llegando a una masiva utilización de las normas y de los protocolos estándares, que se requieren para realizar una adecuada

gestión de redes, debido fundamentalmente a la gran cantidad de elementos que pueden formar parte de una red, y por lo complejo que se sería administrar redes de diferentes fabricantes, por lo que la estandarización de los procesos de gestión resulta ventajosa para los administradores y para los usuarios de red.

El término **GESTION** involucra la planificación, monitoreo, contabilidad y control de las actividades y los recursos de una red. Esta definición puede ser plenamente aplicada a la definición de gestión de red.[1]

Sin embargo las estructuras de gestión de red OSI e INTERNET están enfocadas principalmente en el monitoreo, contabilidad y control de las actividades y recursos de la red.

Por lo tanto los aspectos como la planificación y la organización no son involucrados en la arquitectura de gestión INTERNET.

La gestión de red incluye las siguientes funciones: operaciones, administración, mantenimiento y aprovisionamiento (OAM&P, Operation, Administration, Mantenance and Provisioning), funciones que son necesarias para interpretar, controlar, supervisar la red y el transporte de los servicios.[2]

---

[1] BLACK Uyless; Network Management Standars; Second Edition;McGraw-Hill; USA 1995

[2] AIDAROUS Salah, Telecomunication Network Managenent into the 21ST Century, IEEE 1996

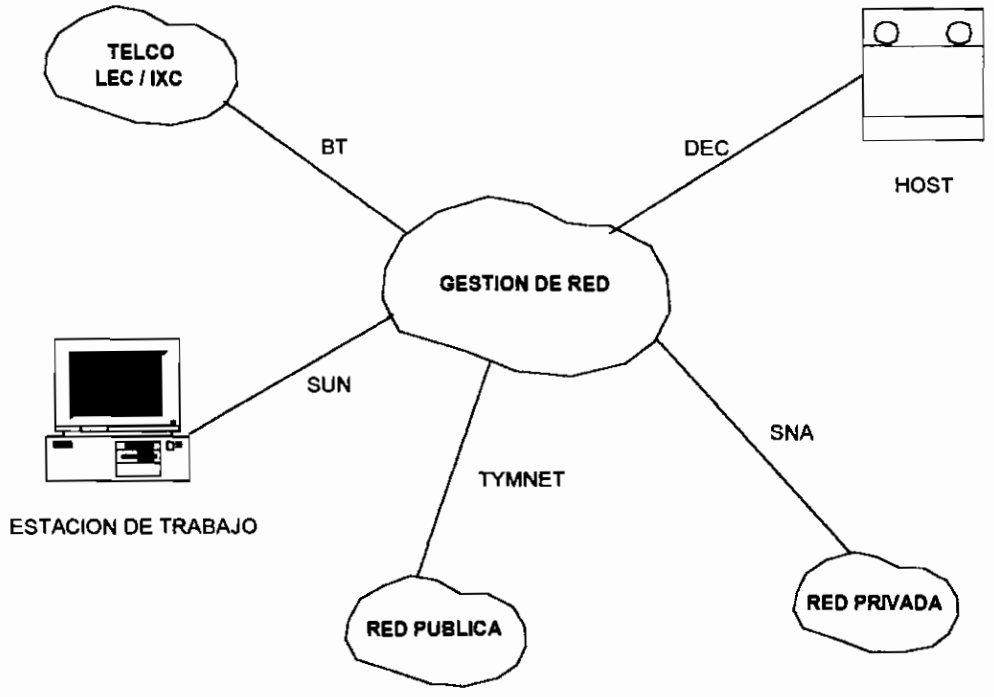


Figura 1.1: **Gestión de Redes utilizando diferentes estándares de gestión.**

La figura 1.2 en cambio nos indica la ventaja de utilizar estándares de gestión de red, lo cual permite que el centro de control de red utilice un solo tipo de software para interactuar con las diferentes redes a ser gestionadas, dando lugar a la utilización del mismo software para cada uno de los elementos de la red gestionada.

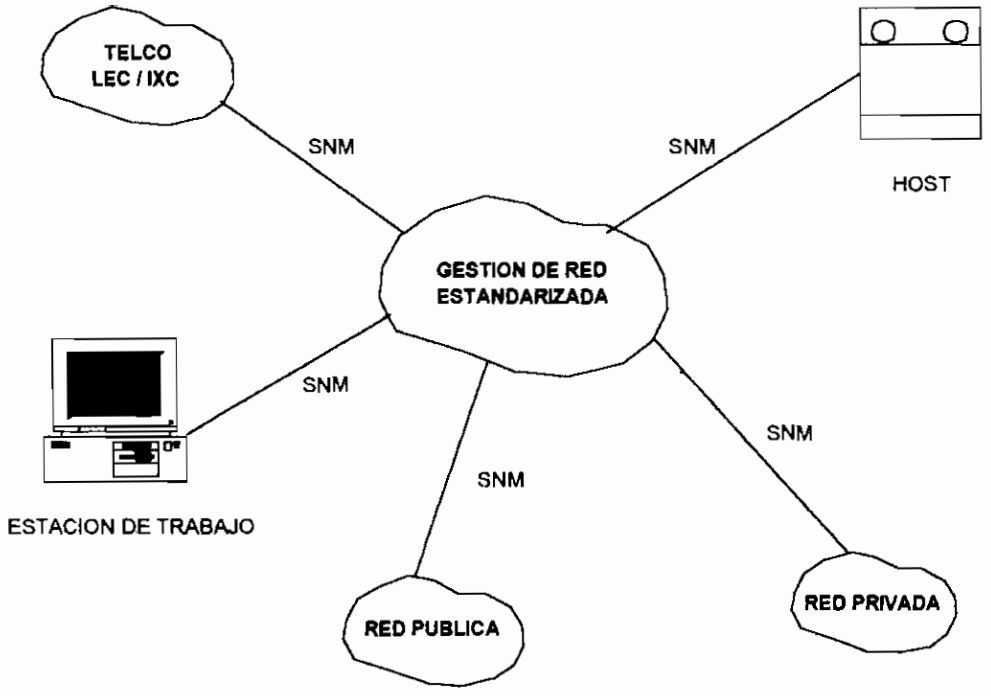


Figura 1.2: **Gestión de Redes utilizando los mismos estándares de gestión.**

Muchas organizaciones o entidades públicas o privadas operan en ambientes heterogéneos y utilizan una amplia variedad de componentes de hardware, así como también diferentes protocolos de comunicación. Actualmente estas organizaciones o entidades están enfrentando el gran problema técnico que es, la construcción de un único paquete de interconexión para diferentes componentes y protocolos redes de telecomunicaciones.

Otros objetivos de la gestión de red se pueden resumir en los siguientes aspectos:

- Garantía de un servicio continuado
- Capacidad para reparar y evitar fallas

ellos. Los protocolos de la arquitectura de gestión de red INTERNET son los que más prevalecen en la industria de las telecomunicaciones y los esfuerzos de la IEEE están reforzando el camino para estandarizar los sistemas de gestión en redes LAN y MAN.

La arquitectura TMN es una red que esta conformada por interfaces, protocolos estándar y dispositivos, que describen una arquitectura para la gestión de redes de telecomunicaciones y además habilita una infraestructura adecuada para la gestión de redes de área local LAN y de área extendida WAN.[3]

La utilización de protocolos estándares no solamente facilitan el trabajo de los diferentes elementos de hardware de red tales como: estaciones de trabajo, host, ruteadores, multiplexores, repetidores, etc, si no que también proveen a la red una flexibilidad en la selección del hardware y software de red, porque ellos definen las interfaces y los protocolos estándares entre los diferentes fabricantes.

Otro de los objetivos de éstos estándares es permitir una plataforma común para la implantación de las aplicaciones de usuario, el software de soporte y las aplicaciones de gestión de red. Estos objetivos simplifican y reducen las interfaces entre el centro de control de red y los recursos de gestión de red del usuario.[4]

[3] AIDAROUS Salah; Telecomunication Network Managenent into the 21ST Century; IEEE 1996

[4] BLACK Uyless; Network Management Standars; Second Edition;McGraw-Hill, USA 1995

Los estándares de gestión de red OSI están organizados alrededor de la técnica del diseño orientado a objetos, que es un concepto moderno de diseño de sistemas de gestión de red.

En el centro de control de red, la técnica de diseño orientado a objetos proporciona a los ingenieros de redes una considerable flexibilidad en el manejo de los esquemas de gestión de red y provee de una herramienta efectiva para manejar el software costoso de gestión de red.

1.4 GESTION DE REDES INTEGRADAS[5]

En los procesos de gestión de red, se pueden encontrar dos tipos de redes a ser gestionadas: las redes heterogéneas y las redes integradas.

Una **red heterogénea** es aquella red que posee diferentes interfaces de usuario y sus propios sistemas de gestión de red para cada uno de sus componentes de red, en cambio una **red integrada** es aquella que posee una sola interfaz de usuario, así como también un solo sistema de gestión llamado integrado, para todos los componentes de red.

Una de las metas importantes de la gestión de red es apoyar el acercamiento integrado en la gestión de red o redes que contienen diferentes tipos de dispositivos y paquetes de software de gestión.

[5] BLACK Uyless; Network Management Standars; Second Edition;McGraw-Hill; USA 1995

Los objetivos de la gestión de redes integradas son los siguientes:

- La gestión de redes integradas es importante porque permite reducir el costo de las diferentes interfaces de un sistema de gestión.

- La integración es necesaria porque la gestión de red requiere uniformidad para el intercambio de información de gestión entre los diferentes fabricantes de dispositivos red.

- La gestión de redes integradas no implica que los servicios son los mismos para todos lo usuarios de la red, es decir que, la gestión de redes integradas significa que el servicio de gestión de red es el mismo para las definiciones de funcionalidad de la red, la contabilidad, la configuración, la seguridad, criterios de falla y el intercambio de protocolos comunes de unidades de datos.

Una de las metas importantes de los estándares de gestión de red es desarrollar e integrar un conjunto de procedimientos y estándares que se apliquen igualmente en diferentes dispositivos de red.

1.5.2 Agente de Gestión

Se lo considera como el elemento activo de la red porque es un software que responde a los requerimientos para informar y actuar desde una estación de gestión y poder entregar datos importantes, así como también datos o información que no es solicitada.

Los dispositivos de red tales como host, puentes, ruteadores, hubs, repetidores, etc, pueden ser equipados con agentes SNMP, lo cual facilita la gestión de estos elementos desde una estación de gestión que trabaje con protocolo SNMP.

En conclusión, el Agente de Gestión reporta a la Estación de Gestión sobre el estado de los elementos de red gestionados y recibe indicaciones de la Estación de Gestión para mejorar la funcionalidad de esos elementos.

1.5.3 Estación de Gestión

La estación de gestión puede actuar, tomar o poner datos en el agente, así como también puede cambiar los parámetros de configuración en un agente, para modificar los valores de variables específicas.

La Estación de Gestión actúa como una interfaz entre la persona que gestiona o administra la red y el sistema de gestión de red.

Una MIB contiene información, que esta estructurada utilizando una organización lógica definida llamada **estructura de información de gestión (SMI).** Esta SMI esta definida como una estructura de árbol, empezando desde la raíz, la cual contiene ramales, que organizan los objetos gestionados mediante categorías lógicas. La MIB representa a los objetos gestionados como hojas en los ramales.[8]

En general se puede indicar que una colección o conjunto de objetos gestionados se conoce con el nombre de Base de Información de Gestión (M.I.B.), la cual no significa que tenga las mismas características de una base de datos numérica normal ya que estas, se almacenan con una estructura diferente.

La principal función de una MIB es la de proporcionar puntos de acceso o caminos de acceso a la información de gestión, para que el agente pueda acceder a la misma de una manera rápida.

En general, las Bases de Información de Gestión son utilizadas por el agente y la estación de gestión para determinar la estructura y el contenido de la información de gestión.

[8] MILLER Mark; Managing Internetworks with SNMP; M&T Books; USA 1993

1.5.5 Protocolos de Gestión de Red.

Un protocolo es un conjunto de reglas que gobierna el formato y el significado de las tramas, paquetes o mensajes que son intercambiados por las entidades corresponsales dentro de una capa.[9]

Los protocolos de gestión de red son utilizados para enlazar la estación de gestión y el agente de gestión en el nivel de la capa de aplicación.

Entre los principales objetivos de los protocolos de gestión de red es la de proporcionar la habilidad para obtener información de los dispositivos de red, así como también permitir efectuar intercambio de información.

Existen varias categorías de protocolos de gestión de red, los cuales son los siguientes:[10]

a. Un protocolo simple de gestión de red, el cual define un formato común de datos y parámetros, además permite una fácil recuperación de la información.

b. Un protocolo de gestión de red más complejo, el cual añade capacidades de cambio y mecanismos de seguridad, para proteger la información y prevenir para que no cualquiera pueda realizar esos cambios.

---

[9] TANENBAUM Andrew; Redes de Ordenadores; Segunda Edición; Prentice Hall; México 1991

[10] LEINWAND Allan; Network Management; Second Edition; Addison Wesley; USA 1996

c. Un protocolo de gestión de red avanzado, el cual tiene la habilidad de ejecutar gestión de red remota, similar a un procedimiento remoto y estar totalmente independiente de las capas de protocolos de red de todos los dispositivos de la red.

En la actualidad se trata de estandarizar los protocolos de gestión de red para ayudar al administrador de red, obtener la información de gestión necesaria de todos los dispositivos de la red.

Los protocolos más comunes para realizar la gestión de redes de telecomunicaciones son los siguientes:

1. El Protocolo Simple de Gestión de Red (SNMP, Simple Network Management Protocol)
2. SNMPv2 (SNMP versión 2)
3. Servicios Comunes de Información de Gestión(CMIS, Common Management Information Services)
4. El Protocolo de Información de Gestión Común(CMIP, Common Management Information Protocol)

La Junta de la Arquitectura de Internet(IAB), había propuesto una estrategia muy práctica para lograr una estandarización:[11]

- Que se adopte SNMP como una solución a corto plazo.
- Que los protocolos CMIS y CMIP sean una solución a largo plazo.

---

[11] RFC 1155

Cabe indicar que los protocolos de gestión de red proporcionan simplemente métodos para monitorear y configurar dispositivos de red.

1.5.6 Ubicación de los Componentes de Gestión de Red

Los estándares de gestión de red OSI, INTERNET y IEEE no requieren que los componentes estén ubicados en una localidad en particular dentro de una red.

En la figura 1.3 se toma como referencia una típica red LAN, en la cual se indica la ubicación de la estación de gestión, el agente de gestión y la MIB.[12]

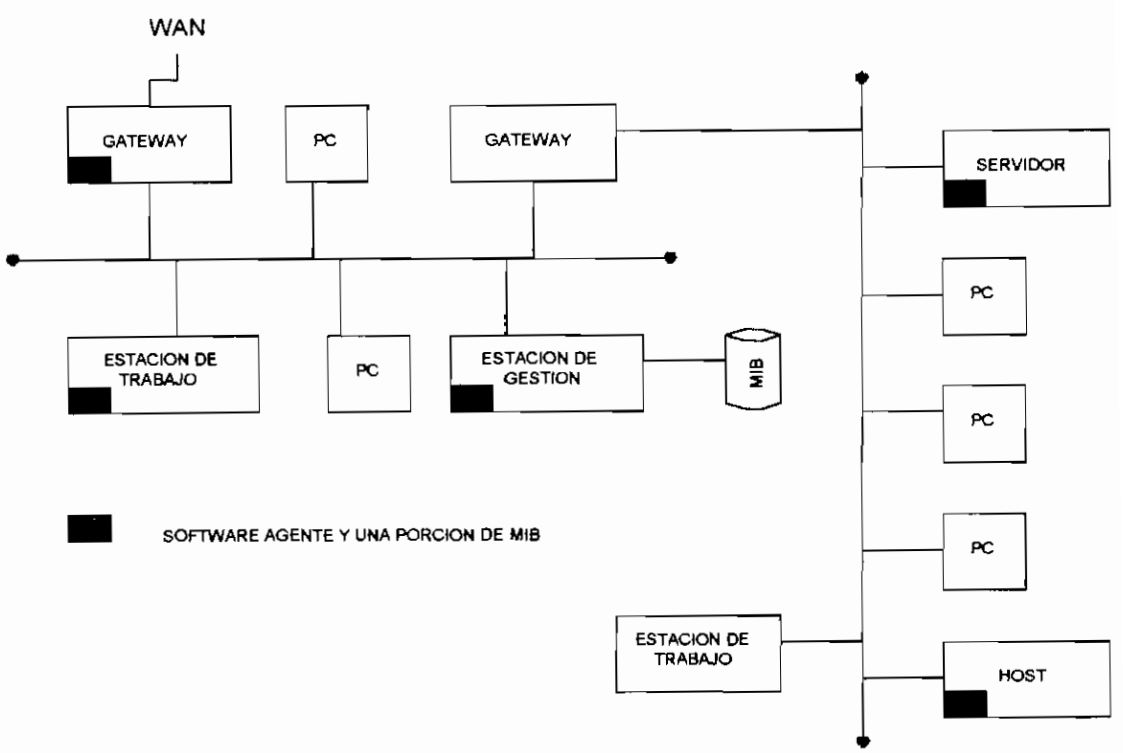


Figura 1.3: **Componentes de una red LAN**

[12] BLACK Uyless; Network Management Standars; Second Edition;McGraw-Hill; USA 1995

En la actualidad, el software agente esta ubicado en los componentes de red, tales como servidores, gateways, puentes, ruteadores, repetidores.

La Base de Información de Gestión MIB está generalmente almacenada en la estación de gestión de red y otra parte de la MIB que es necesaria para el agente está ubicada en el dispositivo de red a ser gestionado.

En la figura 1.4 se indica otra forma de ubicar los componentes de gestión de red, en la cual se identifica la estación de gestión (Manager Network) y los diferentes dispositivos de red y también se pueden identificar sus respectivos agentes de red y sus MIBs.[13]

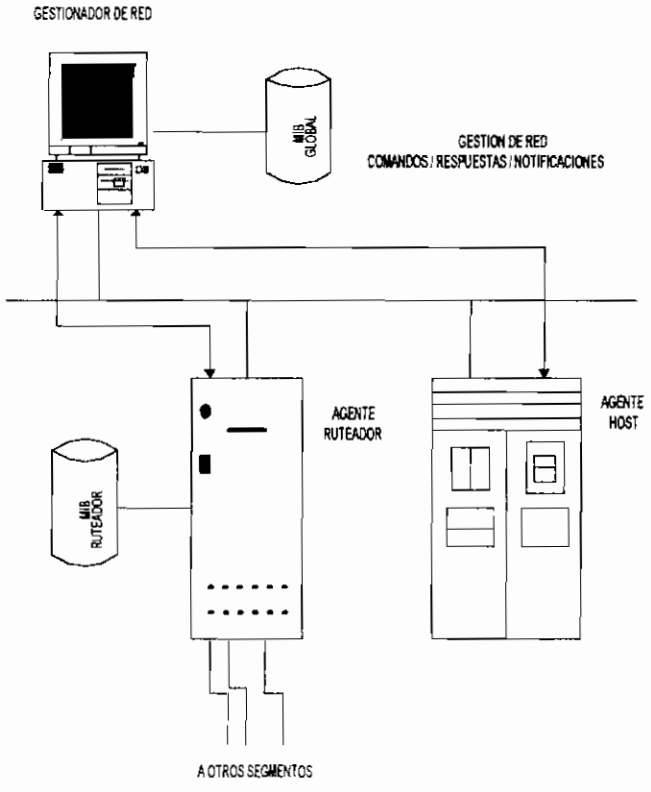


Figura 1.4: **Componentes de gestión de red**

[13] MILLER Mark; Managing Internetworks with SNMP; M&T Books; USA 1993

1.6 PROTOCOLO SNMP (Simple Network Management Protocol)

SNMP es un protocolo de manejo de red que fue diseñado para cumplir con los requerimientos de TCP/IP. Este protocolo posee una adecuada capacidad para monitorear una red y también efectuar cambios en la red.[14]

La RFC 1098 reconoce al protocolo SNMP como un protocolo estándar y en la RFC 1157 se describe el modelo estación/agente utilizado en el protocolo SNMP.

SNMP opera en la capa aplicación, utilizando el protocolo de control de transmisión/protocolo Internet(TCP/IP, Transmision Control Protocol/Internet Protocol), por lo que no es necesario conocer aspectos específicos del hardware sobre el que funciona.

Este protocolo utiliza el modelo estación/agente, se lo considera "simple" porque el agente requiere de un mínimo software para realizar el proceso de gestión. En general este protocolo utiliza un limitado conjunto de comandos de gestión y respuestas.

Un **agente SNMP** es un software capaz de contestar preguntas válidas de una **estación SNMP,** en el caso de un sistema de gestión de red, será la información definida en la MIB.

[14] CASE JD; SNMP Network Management; Second Edition; McGraw-Hill; USA 1996

Un dispositivo de red que entrega información de la MIB a la estación de gestión necesariamente tendrá un agente SNMP. El agente y la estación de gestión hablan el mismo lenguaje en el modelo estación/agente.

El agente y la estación SNMP se comunican a través de mensajes estándares. Un mensaje puede ser enviado en un simple paquete entre la estación y el agente. Cada paquete contiene todo o parte del mensaje de intercambio, el cual es llamado Unidades de Datos de Protocolo (PDU Protocol Data Unit).

El protocolo SNMP tiene cinco tipos de mensajes:[15]

1. Get-Request
2. Get-Response
3. Get-Next-Request
4. Set-Request
5. Trap

La estación SNMP utiliza el comando o mensaje Get-Request para recuperar información de un dispositivo de red que posee un agente SNMP. El agente SNMP, en cambio, responde al mensaje Get-Request con un mensaje Get-Response.

[15] LEIWAND Allan, Network Management; Second Edition; Addison Wesley; USA 1996

La información que retorna incluye el nombre del sistema, el tiempo que el sistema esta operando y el número de interfaces de red en el sistema.

Un Get-Next-Request es utilizado conjuntamente con un Get-Request para obtener información una tabla de objetos. El Get-Request recupera un objeto específico; el Get-Next-Request pide por el próximo objeto específico en una tabla en orden específico Un agente responde a un Get-Next-Request con un mensaje Get-Response. El Set-Request permite configurar remotamente los parámetros de un dispositivo.

El Trap SNMP es un mensaje no solicitado que un agente SNMP envía a la estación de gestión. Estos mensajes informan a la estación de gestión la ocurrencia de un evento específico, por ejemplo un Trap SNMP puede ser utilizado para informar al sistema de gestión de red que un circuito a fallado, o que el espacio de disco de un dispositivo esta próximo a completar su capacidad o que un usuario no permitido esta en un host.

El agente debe estar configurando con la dirección de la estación de gestión y saber a donde enviar el mensaje del Trap SNMP.

En la figura 1.5 se puede observar una configuración SNMP.

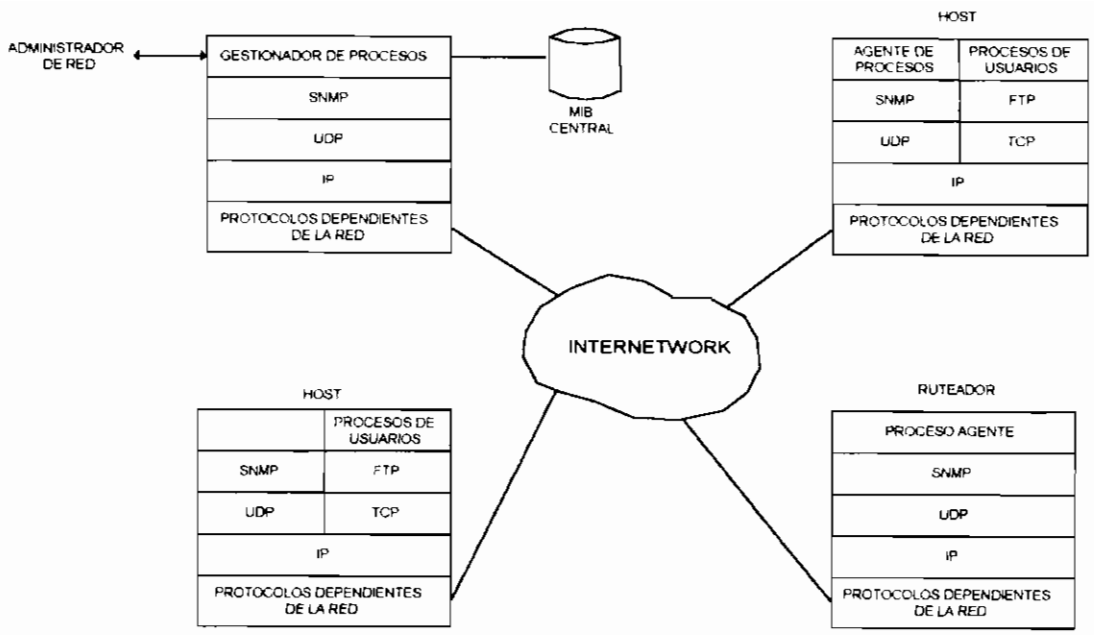


Figura 1.5: **Configuración SNMP**

El protocolo SNMP utiliza UDP (Protocolos de datagramas de usuarios) para ejecutar las funciones de la capa transporte. UDP provee un menor servicio de conexión, además SNMP no puede mantener una conexión entre el agente y la estación para transmitir un mensaje.

UDP provee rapidez en el servicio de la capa transporte con una mínima cantidad de asignación de recursos. Sin embargo UDP no provee un método exacto para el intercambio de mensajes entre el agente y la estación.

1.7 PROTOCOLO SNMPv2

Este protocolo fue desarrollado para ayudar a resolver algunas deficiencias del SNMPv1, tales como aquellas que se refieren al caso de que este protocolo fue estandarizado únicamente para redes IP y además son inseguras.

SNMPv2 provee de herramientas y mecanismos de seguridad para poder entregar un sistema de autenticación el usuario y determinar el nivel de acceso del usuario a los recursos de la red y dar privacidad a los datos intercambiados.[16]

1.8 EL PROTOCOLO CMIS/CMIP

Este protocolo fue recomendado por la IAB como un protocolo básico para satisfacer requerimientos futuros y fue desarrollado por la ISO, con diferentes metas del protocolo SNMP.

CMIP fue diseñado tomando en cuenta todos los errores que posee el protocolo SNMP, pero también utiliza los PDU como variables para monitorear la red.

[16] Curso de redes de Computadora, Protocolos de gestión de redes, http://roblepntic.mec.es/atei

2.2 CONCEPTOS BÁSICOS

2.2.1. OBJETOS

Un objeto representa una unidad o una entidad individual e identificable sea real o abstracta que cumple un papel bien definido en un determinado problema o proceso. Un objeto tiene un estado, un comportamiento y una identidad.

El estado de un objeto se refiere a todas las propiedades que tiene el objeto, así como también los valores que tienen cada una de estas propiedades.

El comportamiento es la forma como el objeto actúa y reacciona, en función de sus cambios de estado y del paso de mensajes. Es decir que el comportamiento de un objeto es la actividad que puede realizar sobre el mismo o sobre uno o más objetos.

La identidad del objeto se refiere al nombre que tiene cada objeto para poder ser invocado a realizar una tarea en un determinado proceso.[17]

En un sistema basado en computadora, un objeto típicamente es un productor o consumidor de información o simplemente un elemento de información.

Como ejemplos típicos de objetos se pueden considerar los siguientes: máquinas, órdenes, archivos, conmutadores, señales, cadenas alfanuméricas, algún elemento que conforma una red, etc.[18]

---

[17] CORDOVA Raúl; Técnica Orientada a Objetos; Folleto UASB; ECUADOR 1998

[18] PRESMAN Roger; Ingeniería de Software, Segunda edición; McGraw-Hill ;México 1990

Un objeto es un elemento que puede ser claramente identificado.

2.2.2 MENSAJES Y METODOS

En la metodología orientada a objetos, la acción se inicia mediante la transmisión de un mensaje o solicitud a un objeto responsable de la acción. El mensaje tiene codificada la petición de una acción y se acompaña de cualquier información adicional necesaria para llevar a cabo la petición.

El receptor es el agente al cual se envía el mensaje. Si el receptor acepta el mensaje, acepta la responsabilidad de llevar a cabo la acción indicada. En respuesta a un mensaje el receptor ejecutará algún método para satisfacer la petición.

Por lo tanto, un mensaje es una petición al objeto para que ejecute una de sus operaciones o métodos. Los métodos especifican la forma en que se controlan los datos de un objeto.[19]

2.2.3 CLASES

Las Clases son el conjunto o agrupamientos de objetos que tiene la misma estructura o atributos, así como también el mismo comportamiento.

---

[19] PRESMAN Roger; Ingeniería de Software; Segunda edición; McGraw-Hill ;México 1990

Una clase es un conjunto de objetos que comparten una estructura común y un comportamiento común.[20]

Todos los objetos de una clase dada utilizan los mismos métodos en respuesta a mensajes similares.

2.2.4 HERENCIA

Heredar significa compartir los atributos y las operaciones entre las clases con base a un relacionamiento jerárquico.[21]

La Herencia es una herramienta para organizar, construir y emplear clases reutilizables. Sin herencia, cada clase sería una unidad independiente y cada una se desarrollaría empezando desde cero.

La herencia es una característica por la cual se puede crear o definir una nueva clase a partir de otra clase y se la puede identificar cuando dos objetos se relacionan con la palabra "es un". Las clases se pueden organizar en una estructura de herencia jerárquica.

Una clase heredará propiedades de una superclase que esté mas arriba del árbol. Estas propiedades pueden ser la estructura de los datos y los métodos o

[20] CORDOVA Raúl; Técnica Orientada a Objetos, Folleto UASB; ECUADOR 1998

[21] CORDOVA Raúl; Técnica Orientada a Objetos; Folleto UASB; ECUADOR 1998

parte de ellos. A veces una clase hereda propiedades de más de una superclase, esto recibe el nombre de **herencia múltiple.**

En la siguiente figura se indica en forma gráfica el concepto de herencia:

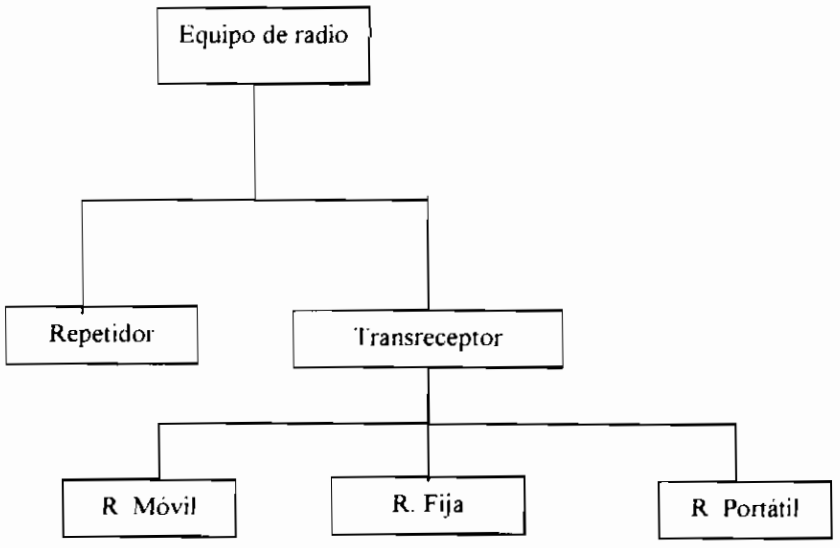


Figura 2.1: **La herencia (es un) aplicada al objeto Equipo de radio**

2.2.5 ENCAPSULAMIENTO

El empaque de un conjunto de datos y métodos se denominan encapsulamiento. El objeto esconde sus datos de los demás objetos y permite el acceso a los datos mediante sus propios métodos, esto recibe el nombre de ocultamiento de la información.

El encapsulamiento oculta detalles de su implantación interna a los usuarios de un determinado objeto.

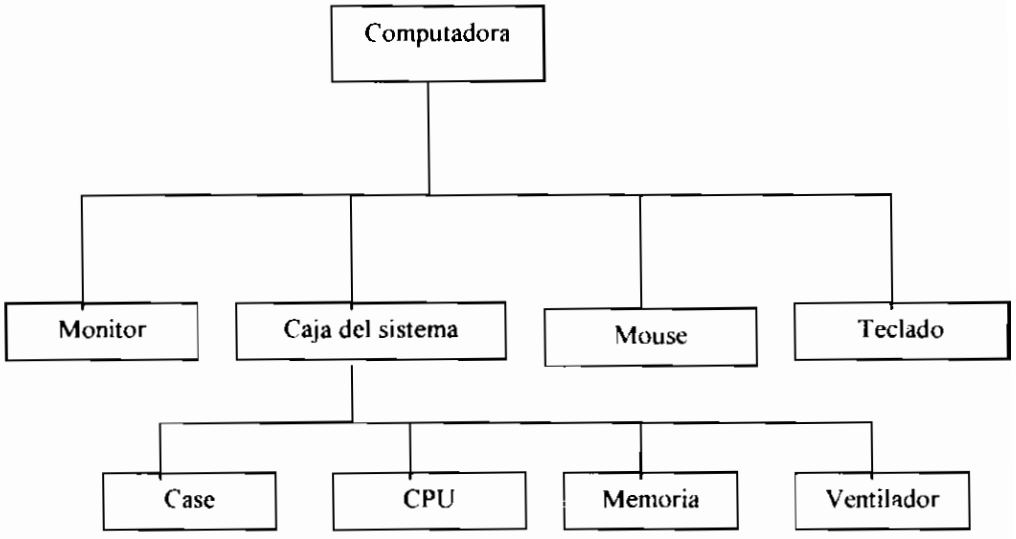


Figura 2.2: **La agregación aplicada al objeto Computadora**

2.3 RECURSOS A SER GESTIONADOS[22]

Existen varios recursos de red que pueden ser gestionados, éstos se los puede identificar en dos grupos:

- La gestión de red involucra monitoreo y control de diferentes componentes de hardware y software de red.

- El espectro de gestión de red abarca la gestión de productos heterogéneos, multivendedores, multiprotocolos geográficamente dispersos y recursos humanos.

[22] EGAS Carlos, Gestion de Redes, CLEI 98

Estas arquitecturas de gestión de red utilizan los conceptos de la técnica orientada a objetos ya sea en forma total o únicamente determinados conceptos, para la ejecución de las tareas de gestión.

El modelo de gestión de red OSI utiliza ampliamente todos los conceptos de la Técnica Orientado a Objetos, en cambio los modelos INTERNET y IEEE solo utilizan parte de estos conceptos.

La arquitectura de gestión de red INTERNET no emplean por ejemplo las operaciones de creación y eliminación de objetos declarados.

La idea de diseño orientado a objetos aplicado a la gestión de redes, se refleja en que los encargados de la operación de gestión o administradores de red interactúan con el objeto gestionado sin el conocimiento de las operaciones internas que realiza el mismo.

2.4.1 Objetos Gestionados[23]

Los recursos que son supervisados y gestionados por la Gestión de Red son llamados **Objetos Gestionados**.

Un objeto gestionado puede ser cualquier elemento importante que se utilice en una arquitectura de gestión de red.

[23] BLACK Uyless; Network Management Standars, Second Edition;McGraw-Hill, USA 1995

Los elementos de hardware tales como switches, ruteadores, estaciones de trabajo, PBX, repetidores, multiplexores, etc pueden ser identificados como objetos gestionados.

En lo referente al software, tales como los programas, algoritmos de rutinas y rutinas de gestión, también pueden ser considerados como objetos gestionados.

Las características más importantes de los objetos gestionados son:

- Las operaciones de gestión permitidas que pueden ser realizadas en un objeto gestionado deben formar parte de su definición.

- En la definición de los objetos gestionados se deben considerar las operaciones que tiene relación a los recursos del sistema.

- El estado de los objetos gestionados o sus propiedades pueden determinar el tipo de operaciones que pueden ser realizadas en un objeto gestionado.

2.4.2 Objetos Gestionados en la Arquitectura de Red OSI

En este modelo de gestión de red, los objetos gestionados están descritos y definidos mediante cuatro aspectos de gestión de red:

- **Atributos:** o llamadas características, las cuales son conocidos en su interface(límite visible).

- **Operaciones:** es decir las acciones que pueden ser realizadas en el objeto.
- **Notificaciones:** o también llamadas reportes que son permitidos realizar.
- **Comportamiento:** es decir las respuestas a las operaciones realizadas en el objeto.

En la figura 2.3 se indica a un objeto gestionado en el modelo OSI con sus correspondientes aspectos de gestión de red.

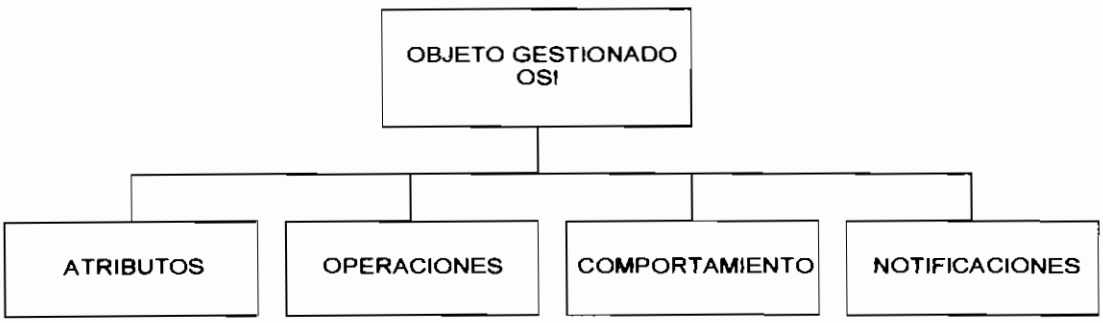


Figura 2.3: **Objeto gestionado en el modelo OSI**

2.4.3 Objetos Gestionados en la Arquitectura de Red IEEE

Los estándares IEEE utilizan los mismos aspectos de gestión de red de los objetos gestionados en el modelo OSI.

Sin embargo existen algunas diferencias, fundamentalmente en lo que se refiere a las operaciones que se pueden realizar en un objeto gestionado, tales como las operaciones de lectura, escritura, comparación y escritura, acción y eventos.

2.4.4 Objetos Gestionados en la Arquitectura de Red Internet

Los objetos gestionados en el modelo Internet están descritos en una manera menos abstracta, los cuales contienen los siguientes aspectos:

- **Sintaxis:** la cual es utilizada para modelar al objeto.
- **Acceso:** el cual define nivel de acceso permitido del objeto.
- **Estado:** que son los requerimientos para la implantación del objeto.
- **Nombre:** que sirve para identificar al objeto

En la siguiente figura 2.4 se indica a un objeto gestionado en el modelo Internet con sus correspondientes aspectos:

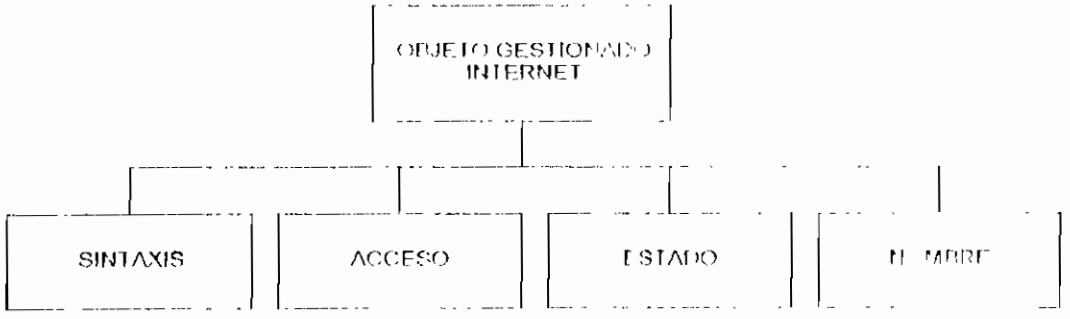


Figura 2.4: **Objeto gestionado en el modelo Internet**

La **Sintaxis** define el tipo de dato, para lo cual se utiliza la Notación de Sintaxis Abstracta ASN.1 que se describirá en el capítulo siguiente.

Los tipos de datos pueden ser enteros (INTEGER), cadenas de octetos (OCTECT STRING), secuencia (SEQUENCE) y otros tipos especiales.

El **Acceso** proporciona información de cómo un objeto gestionado puede ser accedido, cuyas operaciones permitidas son: solamente de lectura, lectura/escritura, solamente de escritura y no accesible.

El **Estado** permite definir el estado mandatorio(mandatory), mediante el cual cada nodo gestionado es requerido para implementar su objeto gestionado; el estado opcional (optional), en el cual el nodo gestionado es opcional para implementar el objeto y el estado obsoleto (obsolete), en el cual los objetos gestionados no están completamente definidos y los nodos no necesitan implementarlos.

El **Nombre** del objeto gestionado se refiere a un Objeto Identificador ASN.1, el cual sirve para identificar a un objeto gestionado.

Una vez que se ha indicado las características más importantes de los objetos gestionados para cada arquitectura de gestión de red, cabe indicar que en el presente trabajo se utilizará el modelo Internet para determinar cómo la técnica de objetos permite identificar los objetos gestionados. Una vez identificados los objetos a ser gestionado, se implementará una base de información de gestión, la cual será descrita en el capítulo tres.

Como se indicó anteriormente, la técnica orientada a objetos involucra muchos conceptos los cuales se utilizan de acuerdo al campo de aplicación. Ya sea en el campo relacionado al hardware o al software. Es necesario mencionar que en la mayoría de las áreas de su posible aplicación, depende de la arquitectura

de gestión el que se aplique todos o parte de los conceptos de la técnica de objetos.

En el modelo Internet no se hace una distinción entre objetos y los atributos, por esta razón, no es permitida la utilización de los atributos de un determinado objeto en otros objetos, esto da lugar a que no se pueda utilizar la característica de herencia que se menciona en la técnica de objetos. A diferencia con el modelo OSI, en la cual si es permitida la función de herencia, ya que este modelo si utiliza todos los conceptos de la técnica orientada al objeto.

El modelo OSI se utiliza el concepto de clases, en el cual están agrupados todos los objetos que tiene similares características, tales como atributos, operaciones y notificaciones. En cambio que en el modelo Internet no se emplea este concepto de clases si no los llamados grupos, los cuales cumplen la misma función que las clases, en los cuales están agrupados todos los objetos que proporciona similar información.

Por ejemplo, en la arquitectura Internet, se define un objeto denominado grupo interfaz, en el cual se encuentran todos los objetos que pertenecen a la capa física y enlace, así como también todos los objetos que pertenecen a los protocolos de gestión de red están clasificados dentro del grupo SNMP.

Por lo tanto, el modelo Internet utiliza el concepto de Grupo, el cual cumple la misma función que las clases que se define en la técnica de objetos.

En el modelo Internet también se emplea los términos de tipos de objetos y variables objetos.

Un tipo de objeto es la definición de un objeto gestionado en el cual se debe incluir su sintaxis, acceso, estado y su nombre.

La variable objeto es una ocurrencia o instancia del tipo de objeto. Por ejemplo, un tipo de objeto puede ser tabla de datos y las entradas a la tabla son las variables del tipo de objeto.

CAPITULO 3: BASES DE INFORMACION DE GESTION(M.I.B.)

3.1. GENERALIDADES

Como se indicó en el capítulo 1, existen diferentes tipos de arquitectura de gestión de red, el objetivo del presente trabajo es tomar como referencia la arquitectura de gestión de red Internet, para desarrollar el estudio de las Bases de Información de Gestión (MIB) que se pueden gestionar utilizando el protocolo SNMP.

En este capítulo se realizara una descripción completa de la estructura de las bases de información de gestión y la forma como desarrollar MIBs privadas utilizando un lenguaje estándar.

En la RFC 1052, la IAB recomendó unas mejoras a ser colocadas en la definición y extensión MIB para usar con SNMP y CMIS/CMIP. La RFC 1065 describe la sintaxis y el tipo de información disponible en una MIB para la gestión de redes TCP/IP. La RFC 1065 fue adoptada por la IAB como un estándar completo en la RFC 1155.

Usando las reglas de la Estructura de Información de Gestión(SMI), la RFC 1066 presento la primera versión de las MIB para uso con el protocolo TCP/IP. Este estándar, conocido ahora como MIB-I explica y define la información

necesaria para el monitoreo y control de las redes de Internet basadas en el protocolo TCP/IP.

La RFC 1066 fue aceptada por la IAB como un estándar completo en el RFC 1156. La RFC 1158 propuso una segunda MIB, la MIB-II, para ser utilizada con el protocolo TCP/IP. Esta propuesta, formalizada como un estándar y aprobada por la IAB en la RFC 1213, la cual amplía mas información de las bases definidas en la MIB-I.

3.2 CONCEPTO MIB

En los ambientes TCP/IP y OSI hay una base de datos que contiene información relacionada a los elementos que pueden ser gestionados, a ésta base de datos que se hace referencia, se conoce como **Base de Información de Gestión (M.I.B.)**[24]

La Base de Información de Gestión(MIB) es una de las partes más importantes de un sistema de gestión de red. La MIB identifica a los elementos de una red (objetos gestionados)que van a ser gestionados.

La MIB también define nombres no ambiguos que están asociados con cada objeto gestionado. En general una MIB es un almacén virtual que provee un modelo de información de gestión.[25]

[24] CASE JD; SNMP Network Management; Second Edition; McGraw-Hill, USA 1996

La Base de Información de Gestión es aquella información que es accesible a través de los protocolos de gestión de red tales como el SNMP.

La MIB define el contenido de la información que es transportada con los protocolos de gestión de red. Esta información también describe la habilidad de los usuarios de gestión de red para acceder a los elementos de la MIB, por ejemplo, un usuario A puede tener capacidad solamente de lectura en la MIB, en cambio otro usuario puede tener capacidad de leer y escribir información en la MIB.

Cada recurso a ser gestionado es representado mediante un **objeto.** Por lo tanto se puede afirmar que una MIB es una colección estructurada de objetos.

Los protocolos de gestión de red no operan directamente en los objetos gestionados, sino que ellos operan en la MIB. En redes con SNMP, a una MIB se considera como una estructura de base de datos en forma de árbol.

Cada elemento de red, tales como una estación de trabajo, servidor, ruteador, puente, repetidor, etc, contiene una MIB que reflejan el estado de los recursos a ser gestionados en el sistema. Una entidad de gestión de red está en la capacidad de monitorear los recursos de la red mediante la lectura de los valores de los objetos en la MIB y también puede controlar los recursos en ese sistema mediante la modificación de esos valores. [26]

[25] BLACK Uyless; Network Management Standards; Second Edition; McGraw-Hill; USA 1995
[26] CASE JD; SNMP Network Management; Second Edition; McGraw-Hill, USA 1996

3.3 OBJETIVOS DE LA MIB[27]

- El objeto o los objetos utilizados para representar un recurso particular de una red debe ser el mismo en cada sistema.

- Se debe utilizar un esquema común para la representación.

El primer objetivo esta encaminado a definir los objetos y especificar la estructura de estos objetos dentro de la MIB.

El segundo objetivo se refiere a la utilización de una misma estructura de información de gestión.

3.4 ESTRUCTURA DE INFORMACION DE GESTION

(Structure of Management Information S.M.I.)

La Estructura de Información de Gestión (SMI) está especificada en la RFC 1155, 1212 y 1215, en la cual se define una estructura general para una MIB, para que pueda ser definida y construida.

La SMI permite identificar los tipos de datos que se pueden utilizar en la MIB y también especifica que recursos pueden ser representados y nombrados dentro de la MIB. Por lo tanto, una MIB puede almacenar solamente simples

---

[27] CASE JD; SNMP Network Management, Second Edition; McGraw-Hill, USA 1996

tipos de datos, éstos son de dos tipos, los llamados escalares y los arreglos de escalares de dos dimensiones.

En redes que utilizan el protocolo SNMP, solo se pueden recuperar escalares y también entradas individuales en una tabla de datos. La SMI no soporta la creación o recuperación de datos complejos.[28]

Esta característica esta en contraste con el modelo de gestión OSI, el cuál si provee una estructura de datos compleja y modos de recuperación para soportar una funcionalidad más amplia.

El SMI evita los tipos de datos complejos, para simplificar el trabajo de implantación y para mejorar la interoperabilidad, pero las MIBs inevitablemente contendrán tipos de datos creados por el usuario, a menos que se pongan ciertas restricciones en las definiciones de determinados datos.

3.4.1 Funciones de la Estructura de Información de Gestión[29]

Para obtener una estandarización para la representación de la información de gestión, la estructura de la información de gestión debe cumplir con lo siguiente aspectos:

[28] CASE JD; SNMP Network Management; Second Edition; McGraw-Hill; USA 1996

[29] CASE JD, SNMP Network Management; Second Edition; McGraw-Hill; USA 1996

- Proporcionar una técnica estandarizada para definir la estructura de una MIB particular.

- Proporcionar una técnica estandarizada para definir objetos individuales, incluyendo la sintaxis y el valor de cada objeto.

- Proporcionar una técnica estandarizada para codificar los valores de los objetos.

La Estructura de Información de Gestión establece que cada objeto gestionado debe tener los siguientes elementos:

a) Nombre
b) Sintaxis
c) Codificación.

El **nombre** es el Identificador de objeto (object identifier OID), únicamente identifica el objeto.

La **sintaxis** define el tipo de dato, tales como entero (integer), cadena de octetos(string of octets), etc.

La **codificación** describe la forma como la información asociada a cada objeto gestionado es serializada para la transmisión en la red.[30]

Los estándares de Internet utilizan el código ASN.1, que es un lenguaje para la construcción de las MIBs, la cuál permite describir la sintaxis de los tipos de objetos.

3.5 LA NOTACIÓN DE SINTAXIS ABSTRACTA ASN.1(Abstract Syntax Notation One)

La estructura de información de gestión Internet utiliza el estándar ASN.1, el cual define la sintaxis abstracta para una información estructurada conocida como mensajes, es decir que ASN.1 define elementos básicos del lenguaje y proporciona reglas para combinar estos elementos en los mensajes.

La ASN.1 especifica una gran cantidad de tipos de datos para diferentes aplicaciones, de los cuales, se tomaran en cuenta aquellos tipos de datos que pueden ser utilizados en la definición de objetos de las MIBs.

La ASN.1 define tipos básicos de datos, tales como enteros o caracteres y tipos nuevos de datos que están basados en combinaciones de los tipos básicos.

Se debe diferenciar dos términos que son estandarizados, la sintaxis abstracta la cual especifica la notación de los datos por medio del código ASN.1, y la sintaxis de transferencia la cual permite convertir las definiciones ASN.1 en un patrón de bits para que se puedan transmitir a los elementos de red, mediante la utilización de las reglas de codificación básica llamada BER(Basic Encoding

[30] RFC 1155

Rules)[31]. Las reglas de codificación básicas son utilizadas para convertir la información de la MIB en un formato que pueda ser entendido por el agente.

ASN.1 utiliza algunos términos únicos para definir sus procedimientos, incluyendo tipos de definiciones, asignación de valores, definiciones y evocaciones de macros y definición de módulos.

La ASN.1 especifica algunas palabras como palabras claves o secuencias de caracteres reservados. Palabras claves tales como INTEGER, OBJECT y NULL tiene un especial significado y se representan en letras mayúsculas.

La ASN.1 define la sintaxis para la descripción de MIBs y utiliza una estructura de árbol para organizar toda la información disponible. Cada pieza de información en el árbol es un nodo etiquetado, el cual posee un identificador de objeto y un pequeño texto para descripción.

El identificador de objeto es una serie de enteros separados por puntos y el texto describe al nodo etiquetado.

Un nodo etiquetado puede tener subárboles que contienen otros nodos etiquetados, cada nodo etiquetado en un subárbol es numerado en orden ascendente. Si un nodo etiquetado no tiene subárboles o nodos hojas, éste contiene un valor que es conocido como objeto.[32]

---

[31]MILLER Mark; Managing Internetworks with SNMP; M&T Books; USA 1993

[32]LEIWAND Allan; Network Management, Second Edition; Addison Wesley; USA 1996

En la figura 3.1 se indica la estructura de árbol definida en el código ASN.1.

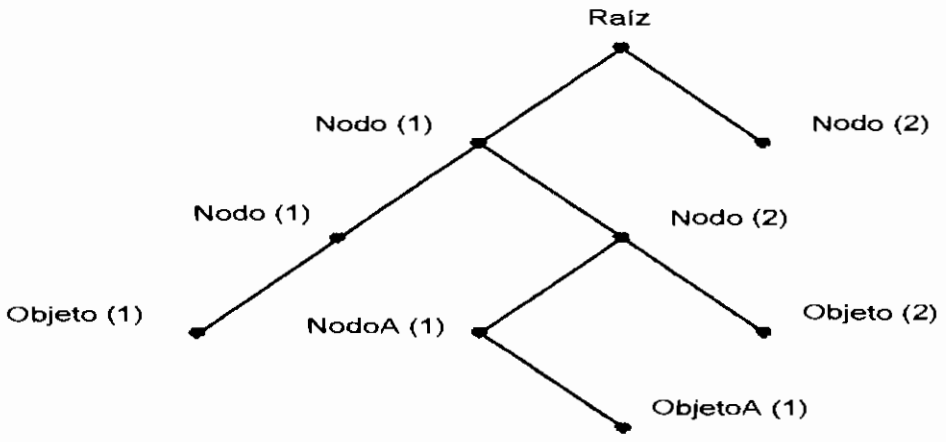


Figura 3.1: **Estructura del árbol ASN.1.**

3.5.1 Reglas de la ASN.1[33]

La ASN.1 impone un conjunto de reglas para el diseñador de MIBs. Estas reglas son simples, pero muy importantes y son las siguientes:

- Diferentes tipos de datos están incluidos en el estándar.

- Los nombres de todos los tipos de datos deben empezar con una letra mayúscula.

- Las palabras reservadas están todas en mayúscula y tiene un significado especial dentro del estándar.

❖ Ciertos nombres de los objetos deben empezar con una letra minúscula.

3.5.2 Símbolos ASN.1

En el código ASN.1 se hace una diferencia entre las letras mayúsculas y minúsculas que se utilizan para representar tipos o valores de datos.

En la siguiente tabla se indica la convención ASN.1 para los nombres de los tipos, valores, macros, módulos y palabras claves:

Item	*Convención*
Tipos	La letra inicial es mayúscula
Valores	La letra inicial es minúscula
Macros	Todas las letras son mayúsculas
Módulos	La letra inicial es mayúscula
Palabras claves ASN.1	Todas las letras son mayúsculas

Tabla 3.1: **Los nombres ASN.1**

Las palabras claves que frecuentemente se utilizan en la descripción de bases de información de gestión son: BEGIN, CHOICE, DEFINED, END, EXPORTS, IDENTIFIER, IMPORTS, , INTEGER, NULL, OBJECT, OCTET, SEQUENCE.

El código ASN.1 también da un significado especial a ciertos caracteres que se indican en la siguiente tabla:

[33] BLACK Uyless; Network Management Standards, Second Edition ; McGraw-Hill; USA 1995

Caracter	Nombre
-	Número asignado
--	Comentario
::=	Asignado(Definido como)
|	Alternativa(opción de una lista)
{ }	Inicio y fin de una lista
( )	Inicio y fin de una expresión de subtipo
..	Indicación de rango

Tabla 3.2: **Significado de caracteres en ASN.1**

3.5.3 Tipos y valores de los datos

Un **tipo** es una clase de dato. El tipo define la estructura del dato que una computadora necesita para entender y procesar la información de gestión.

El **tipo** es una clase de información, tales como enteros, booleanos, octetos, etc. Un tipo puede ser utilizado para describir una colección o un grupo de valores. Por ejemplo, el tipo entero, describe todos los valores que son números enteros (no decimales). El término **tipo de dato** es sinónimo de tipo.[34]

El **valor** cuantifica el tipo de dato. El valor proporciona una instancia o modelo específico para cada tipo. Por ejemplo, un valor podría ser un número o un conjunto de textos alfabéticos. Por convención, los valores empiezan con una letra minúscula.[35]

[34] BLACK Uyless, Network Management Standards; Second Edition; McGraw-Hill; USA 1995

[35] MILLER Mark, Managing Internetworks with SNMP; M&T Books; USA 1993

Algunas aplicaciones permiten solamente un subconjunto de posibles tipos de valores. La especificación del subtipo aparece después del tipo e indica el valor o los valores permisibles, llamados **valores subtipo** y deben estar entre paréntesis. Ejemplo: INTEGER(0..255)

La estructura de información de gestión SMI define tres tipos de datos:[36]

1. Los tipos Simples o Primitivos (Simple types).
2. Los tipos Constructor (Construtor types)
3. Los tipos Definidos (Defined types)

3.5.3.1 Tipos Simples

La ASN.1 define diferentes tipos Simples (conocidos como tipos Primitivos) que incluyen los siguientes: INTEGER, OCTET STRING, OBJECT IDENTIFIER y NULL.

Por convención, los tipos de datos empiezan con una letra mayúscula, la ASN.1 define a estos cuatro tipos de datos como una secuencia de caracteres reservados y se los representa todo su nombre en letras mayúsculas.

3.5.3.1.1 Tipo Entero(INTEGER TYPE)

Es un tipo primitivo, en el cual el objeto puede contener números enteros positivos o negativos, incluyendo el cero.

El tipo entero(INTEGER) tiene dos casos especiales:

1. El tipo **entero numerado**(enumerated integer), en el cual los objetos tienen un número específico(no cero) tal como 1,2 o 3.

2. El tipo **entero cadena de bits**(integer bitstring), el cual es utilizado para pequeñas cadenas de bits tales como (0..127) y que permite mostrar su valor en formato hexadecimal.

Ejemplo de la definición de un objeto entero:

IpDefaultTTL OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-write

STATUS mandatory

DESCRIPCION

" Descripción textual del objeto "

::= {ip 2}

3.5.3.1.2 Tipo Cadena de octetos(OCTET STRING TYPE)

Es un tipo primitivo, en el cual los objetos contienen una ordenada secuencia de cero, uno o más octetos.

SNMP utiliza tres casos especiales de tipos Cadenas de Octetos(OCTECT STRING), éstos son: DisplayString, el octectBitstring y el PhysAddress.

[36] RFC 1155

En el tipo DisplayString, todos los octetos son caracteres ASCII que se pueden imprimir. El tipo octectBitstring es utilizado para cadenas de bits que exceden el ancho de los 32 bits. El tipo PhysAddress es utilizado para representar una dirección media o una capa física.

Un ejemplo de uso del tipo DisplayString es

SysContact OBJECT-TYPE

SYNTAX DisplayString (SIZE(0..255))

ACCESS read-write

STATUS mandatory

DESCRIPTION

"Descripción textual del objeto"

::= {system 4}

Note que en la indicación del subtipo, el tamaño permisible está entre 0 y 255 octetos.

3.5.3.1.3 Tipo Identificador de Objeto(OBJECT IDENTIFIER TYPE)

El tipo Identificador de Objeto es un conjunto de valores de todos los identificadores de objetos asignados de acuerdo a ciertas reglas.

El tipo ObjectName es un caso especial que utiliza SNMP, está restringido a los identificador de objetos, de los objetos y subárboles dentro de una MIB.

Ejemplo del tipo OBJECT IDENTIFIER:

IpRouteInfo OBJECT-TYPE

SYNTAX OBJECT IDENTIFIER

ACCESS read-only

STATUS mandatory

DESCRIPCION

"Descripción textual del objeto"

::= {ipRouteEntry 13}

3.5.3.1.4 Tipo Nulo(NULL TYPE)

Es un tipo con un simple valor. Este es utilizado mantener una ubicación, pero en la actualidad no es utilizado para los objetos SNMP.

El NULL esta asignado a ser el valor para variable desconocida, que es el valor de búsqueda del GetRequest PDU.

3.5.3.2 Tipos Construídos

Los tipos Construidos, a veces también llamados Estructurados o Agregados, tales como el SEQUENCE y el SEQUENCE OF, definen tablas y filas (entradas) dentro de estas tablas.

Por convención, los nombres para las tablas de objetos terminan con el sufijo Table y los nombres para las filas terminan con el sufijo Entry.

3.5.3.2.1 Tipo Secuencia(SEQUENCE TYPE)

Es un tipo construido, el cual esta definido mediante la referencia a una establecida y ordenada lista de tipos. Algunos de los tipos pueden ser opcionales y solo pueden ser diferentes tipos ASN.1. Cada valor del nuevo tipo consiste de una ordenada lista de valores, una para cada tipo de componente.

La SEQUENCE, como un entero, define una fila dentro de una tabla. Cada entrada en la SEQUENCE especifica a una columna dentro de la fila.

3.5.3.2.2 Tipo Secuencia de(SEQUENCE OF TYPE)

Es un tipo que está definido mediante la referencia a un tipo simple existente, cada valor en el nuevo tipo es una ordenada lista de cero, uno o más valores de esos tipos existentes

La SEQUENCE y la SEQUENCE OF definen las filas en la tabla.

3.5.3.3 Tipos definidos

Estos tipos de datos son utilizados principalmente en la estructura de gestión de red Internet, a los cuales también se los identifica como tipos de datos de aplicaciones amplias.

Los tipos definidos o de aplicaciones amplias definidos en la RFC 1155 son los siguientes: NetworkAddress, IpAddress, Counter, Gauge, TimeTicks, y Opaque.

El tipo **NetworkAddress** fue diseñado para representar una dirección de una de las diferentes familias de protocolos. Un CHOICE es un tipo primitivo que proporciona alternativas entre otros tipos. Actualmente solo una familia, la familia Internet(llamada internet IpAddress en la definición SMI), a sido definida para este CHOICE.

Por ejemplo:

AtNetAddress OBJECT-TYPE

SYNTAX NetworkAddress

ACCESS read-write

STATUS deprecated

DESCRIPTION

" Descripción textual del objeto"

::= { atEntry 3}

El tipo **IpAddress** es un tipo que representa una dirección de Internet de 32 bits. Se lo representa como un OCTECT STRING de ancho 4(octetos).

Por ejemplo:

TcpConnRemAddress OBJECT-TYPE

SYNTAX IpAddress

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Descripción textual del objeto"

::= {tcpConnEntry 4}

El **tipo Counter** en un tipo dato que representa un entero no negativo que puede ser incrementado pero no decrementado. Cuando el Counter alcanza su máximo valor, inicia de nuevo incrementándose desde cero. El valor máximo del Counter es $2^{32} - 1$, o 4294967295 en decimal. Por lo tanto un objeto que esta definido como un tipo de dato Counter es un número entero sin signo de 32 bits, a diferencia de un tipo entero que es un número con signo de 32 bits.

El tipo Counter es uno de los tipos más utilizados en la definición de objetos en Internet. Su principal aplicación es la contar el número de paquetes u octetos enviados o recibidos.

Ejemplo de utilización de este tipo de dato:

IcmpInDestUnreachs OBJECT-TYPE

SYNTAX Counter

ACCES read-only

STATUS mandatory

DESCRIPTION

"Descripción textual del objeto"

::= { icmp 3}

El tipo **Gauge** representa un entero no negativo. Puede incrementar o decrementar, con un valor máximo de $2^{32} - 1$. Si se alcanza el máximo valor, el Gauge almacena ese valor hasta que se vuelva a inicializar el sistema.

El tipo Gauge es utilizado para medir valores actuales de algunas entidades, tales como numero actual de paquetes almacenados en una cola.

Un tipo Gauge también puede ser utilizado para almacenar la diferencia entre el valor de una entidad desde el inicio hasta el final en un intervalo de tiempo, esta es la razón por la que se lo utiliza para monitorear las tasa de cambio de los valores de una entidad. El tipo Gauge es referido para almacenar valores.

Ejemplo de utilización de este tipo de dato:

```
ifSpeed OBJECT-TYPE
        SYNTAX Gauge
        ACCESS read-only
        STATUS mandatory
        DESCRIPTION
                "Descripción textual del objeto"
        ::= {ifEntry 5}
```

El **tipo TimeTicks** representa un entero no negativo, que cuenta el tiempo en centésimas de segundo desde algún intervalo o lapso de tiempo.

Cuando un determinado objeto definido en una MIB utiliza este tipo de dato, se debe indicar la referencia del intervalo o lapso de tiempo.

Por ejemplo:

```
SysUpTime OBJECT-TYPE
        SYNTAX TimeTicks
        ACCESS read-only
        STATUS mandatory
        DESCRIPTION
                "Descripción textual del objeto"
        ::= {system 3}
```

El tipo **Opaque** permite el paso de sintaxis arbitraria ASN.1, es decir que este tipo soporta la capacidad de pasar datos arbitrarios. El dato es codificado como un OCTEC STRING para la transmisión. Actualmente el SNMP no utiliza el tipo Opaque, pero se podría encontrar en MIBs privadas.

3.5.3.4 Módulos

Una colección de descripciones ASN.1, relacionadas a un tema común se conoce con el nombre de módulo. Los módulos empiezan con el nombre del módulo. Los nombres de los módulos empiezan con una letra mayúscula. Las declaraciones BEGIN y END encierran el cuerpo del módulo.

La siguiente notación describe las reglas para la representación de un módulo:

< nombre del módulo> DEFINITIONS : : = BEGIN

< cuerpo del módulo> END.

El nombre del módulo identifica al módulo. Este es un identificador ASN.1. La definición (DEFINITIONS) indica el módulo que está definido y las definiciones ASN.1 están ubicadas entre las palabras BEGIN y END.

Dentro de un módulo están las definiciones de los tipos (definitions of types). Estos tiene la siguiente forma:

<nombre del tipo> : : = < definición del tipo>

Esta notación es un ejemplo de un tipo simple. Este es también nombrado(o tiene su nombre) porque el directamente especifica el conjunto de sus valores.

El nombre del tipo es el identificador del tipo. La definición del tipo describe la clase y otros diferentes atributos.

El siguiente ejemplo muestra como tres tipos de definiciones están codificadas dentro e un módulo:

< nombre del módulo> DEFINITIONS : : = BEGIN

< nombre del tipo1> : : = <definición del tipo1>

< nombre del tipo2> : : = <definición del tipo2>

< nombre del tipo3> : : = <definición del tipo3>

END

Este ejemplo es conocido como un "tipo estructurado". Este contiene una referencia de uno o más tipos. Los tipos dentro de un tipo estructurado son llamados "tipos componentes".

Un módulo puede exportar(EXPORT) definiciones para uso de otros módulos, los cuales deben importar (IMPORT) esas definiciones.

El cuerpo puede contener IMPORT, en el cual están los nombres de los tipos, valores, macros y módulos en los cuales ellos están declarados.

Las declaraciones en los módulos, contienen las actuales definiciones ASN.1.

<< módulo>> DEFINITIONS : : = BEGIN

<< enlaces>>

<< declaraciones>>

END

3.5.3.5 Macros

Los macros son herramientas que son utilizadas para procesar las definiciones de una MIB. Por convención, una referencia de macro aparece completamente en letras mayúsculas, para lo cual se utiliza el identificador ASN.1 "OBJECT-TYPE". El macro OBJECT-TYPE define objetos en un formato estándar que son consistentes a través de varios MIBs públicos o privados.

Cada objeto dentro de un macro puede tener los siguientes atributos: SYNTAX, ACCESS, STATUS, DESCRIPTION, REFERENCE, INDEX.

La **Sintaxis**(SYNTAX) define la el tipo de dato que posee el objeto. Los cuales pueden ser los tipos simples tales como INTEGER, OCTECT STRING o NULL. También pueden ser casos especiales de tipos objetos simples, incluyendo un entero enumerado que define un valor entero y un DisplayString restringido para caracteres imprimibles ASCII. Una Tabla de Objetos utiliza la sintaxis SEQUENCE OF.

El **Acceso**(ACCESS) define el nivel mínimo de acceso a un objeto, y sus valores pueden ser: solamente de lectura(read-only), lectura y escritura(read-write) y no accesible(not-accesible).

El **Estado**(STATUS) define el soporte de implantación del objeto, el cual puede ser principal(mandatory), opcional(optional), discontinuado(deprecated) u

obsoleto(obsolete). Cuando el Estado(STATUS) define un nivel de soporte para un grupo particular, ese nivel aplica a todos los objetos dentro el grupo.

Cabe indicar que los objetos que han sido reemplazados por objetos compatibles son llamados discontinuados(deprecated). Los objetos que no son aceptados son llamados obsoletos(obsolete).

La **Descripción**(DESCRIPTION), proporciona una definición textual de un tipo de objeto, la cual no necesariamente puede estar presente.

La **Referencia**(REFERENCE), es un texto en el que se indica la referencia cruzada para un objeto definido en otro módulo MIB.

El **Indice**(INDEX) se emplea solamente con filas de objetos. El Indice indica el orden en el cual los objetos aparecen en una fila, a los que se conoce con el nombre de columna ordenada.

Por ejemplo:

```
TcpInSegs OBJECT TYPE
        SYNTAX Counter
        ACCESS read-only
        STATUS mandatory
        ::= { tcp 10}
```

Para el caso de este ejemplo, se define un objeto llamado tcpInSegs, el cual tiene definido el tipo de dato Counter, Este objeto es solamente de lectura y es principal para todos los dispositivos gestionados que soportan el nodo

principal mib.tcp. Cuando un protocolo accede a este objeto, el protocolo utiliza el nombre{tcp 10}, el cual identifica el décimo objeto definido dentro el grupo tcp.

3.6 ESTRUCTURA DE LA MIB

Una base de información de gestión, contiene un conjunto los objetos a ser gestionados, la cual esta estructurada mediante la notación ASN.1.

Una base de información de gestión consiste de un conjunto de objetos. Cada objeto tiene un tipo y un valor.

Para definir los objetos que van a ser incluidos en una MIB, se utiliza la estructura de información de gestión y la notación ASN.1, la cual incluye un número tipos de datos definidos y una gramática para definir nuevos tipos, los cuales se derivan de tipos existentes.

Para definir un objeto de una MIB, se utiliza un macro, el cual define un conjunto de tipos relacionados utilizados para definir objetos gestionados.[37]

Cada objeto dentro de una MIB SNMP está definido en un formato común. En esta definición se especifica el tipo de dato del objeto, su forma admisible, rangos de valores y su relación con otros objetos dentro de la MIB.

La notación ASN.1 es utilizada para definir cada objeto individual y también para definir la estructura completa de una MIB.

[37] CASE JD; SNMP Network Management; Second Edition; McGraw-Hill; USA 1996

Todos los objetos gestionados en el ambiente SNMP están colocados en jerarquía o estructura de árbol.

Los objetos hoja del árbol son los actuales objetos gestionados, cada uno de los cuales representan algún recurso, actividad o alguna información que tiene relación con la gestión que se está realizando.

Asociado con cada tipo de objeto en una MIB está un identificador de objeto del tipo ASN.1, llamado IDENTIFICADOR DE OBJETO (OBJECT IDENTIFIER).

El identificador de objetos es único para cada objeto, consiste de una secuencia de enteros conocidos como subidentificadores. La secuencia se lee de izquierda a derecha y define la localización del objeto en la estructura de árbol de la MIB.

Si por ejemplo el identificador de objeto para tcpConnTable es derivado como sigue:

Iso	*org*	*dod*	*internet*	*mgmt*	*mib-2*	*tcp*	*tcpConnTable*
1	*3*	*6*	*1*	*2*	*1*	*6*	*13*

La identificación normalmente deberá ser escrita como la siguiente secuencia de enteros: 1.3.6.1.2.1.6.13.

En la figura 3.2 se indica el árbol principal en el cual están los diferentes grupos que constituyen el marco de trabajo de gestión de red.

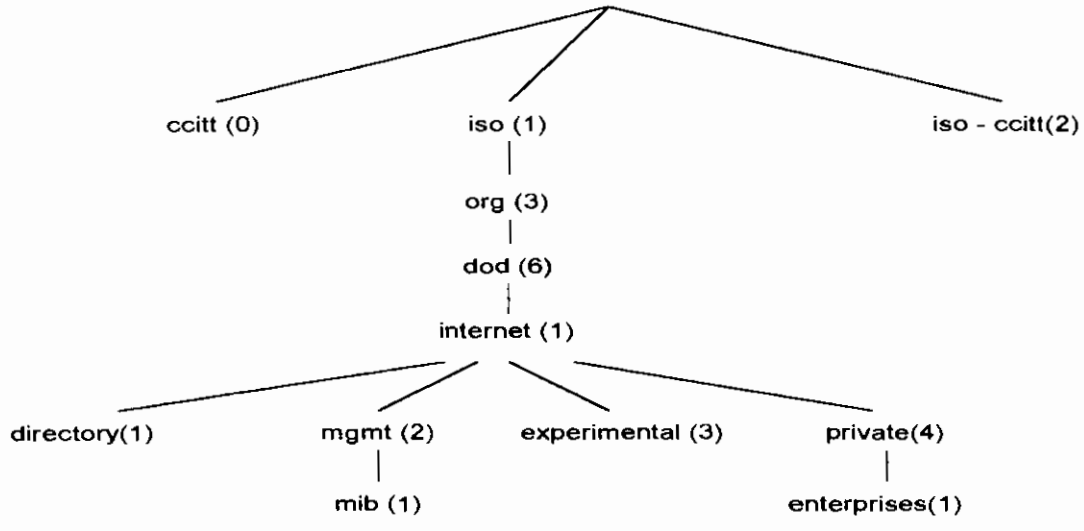


Figura 3.2: **El árbol principal para los procesos de gestión.**

Empezando desde la raíz, hay tres nodos en el primer nivel: iso, ccitt y el joint-iso-ccitt. La CCITT administra el nodo etiquetado como ccitt cuyo número de referencia es el 0, en cambio que la ISO administra el nodo etiquetado como iso cuyo número de referencia es el 1y conjuntamente la ISO y la CCITT administran el nodo etiquetado como joint-iso-ccitt cuyo número de ramal es el 2.

Continuando con los diferentes subramales de la estructura principal, se puede identificar que el nodo iso incluye algunos subramales, entre los cuales el de interés para este presente trabajo, es el ramal definido por la ISO para otras organizaciones, cuya etiqueta es org y su número de referencia es el ramal 3.

Bajo el subárbol org(3) se encuentra el nodo dod(6), el cual es administrado por el Departamento de Defensa de los Estados Unidos(DOD USA). Toda la información de los dispositivos de comunicación por medio de los protocolos dod, tales como TCP/IP, residen en el subárbol que tiene el objeto identificador de 1.3.6.1. Este objeto identificador es conocido como nodo internet. El texto descriptor para este identificador es {iso org(3) dod(6),1}. El nodo Internet es administrado por la IAB.

La ubicación del ramal mib internet tendrá la siguiente secuencia de enteros:

1.3.6.1.2.1, cuyo texto descritor es {iso(1) org(3) dod(6) internet(1) mgmt(2) 1}

3.7 **MIB INTERNET**

La estructura de árbol para el nodo internet es la siguiente:

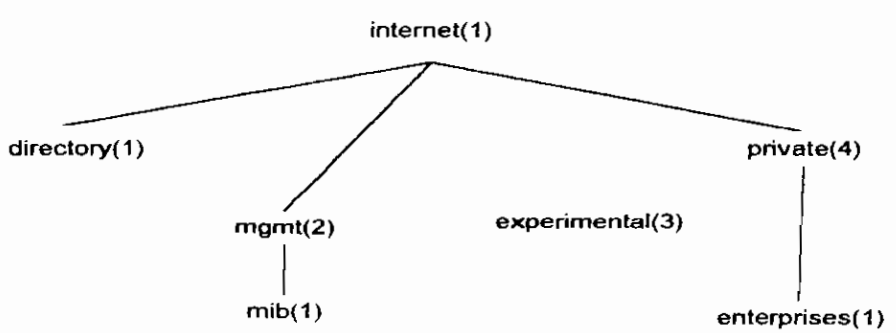


Figura 3.3: **El árbol para el nodo Internet**

El nodo Internet tiene el valor de identificador de objeto 1.3.6.1. y utilizando la notación ASN.1 se lo identifica como:

internet OBJECT IDENTIFIER ::= { iso (1) org (3) dod(6) 1}

La Estructura de Información de Gestión se define cuatro nodos bajo el nodo internet, los cuales son:

1. Directorio(directory)
2. Gestión(mgmt):
3. Experimental(experimental): utilizado para identificar objetos usados en experimentos de Internet
4. Privado(private): utilizado para identificar objetos definidos unilateralmente.

3.7.1 Subárbol Directorio(directory(1))

Este subárbol está reservado para un uso futuro con el directorio OSI dentro de Internet. También se lo conoce como el nodo {internet 1} o 1.3.6.1

3.7.2 Subárbol Gestión(Mgmt(2))

Este nodo está administrado por la Autoridad de Asignación de Números en Internet (Internet Assigned Numbers Authority IANA), en el cual están los objetos que son utilizados para obtener información específica de los dispositivos de red. Como un ramal dentro de este nodo, está el nodo mib, cuyo identificador de objeto es 1.3.6.1.2.1 o {mib 1}.

Los objetos que contiene el nodo mib entregan la información estándar básica que se requiere en los procesos de gestión, por lo que la mayoría de los dispositivos de red incluyen MIB que permiten obtener la información de gestión, además MIB que entreguen información específica sobre el dispositivo.

La estructura de gestión de red Internet está organizada alrededor de grupos de objetos, por lo que una MIB Internet se considera a aquella MIB que posee 10 grupos, a cada uno de los cuales a los que se los conoce como miembros de la MIB. La RFC 1213 entrega una descripción detallada para cada uno de los grupos de objetos.

En la figura 3.4 se indica los diferentes grupos de objetos que conforman una MIB Internet:

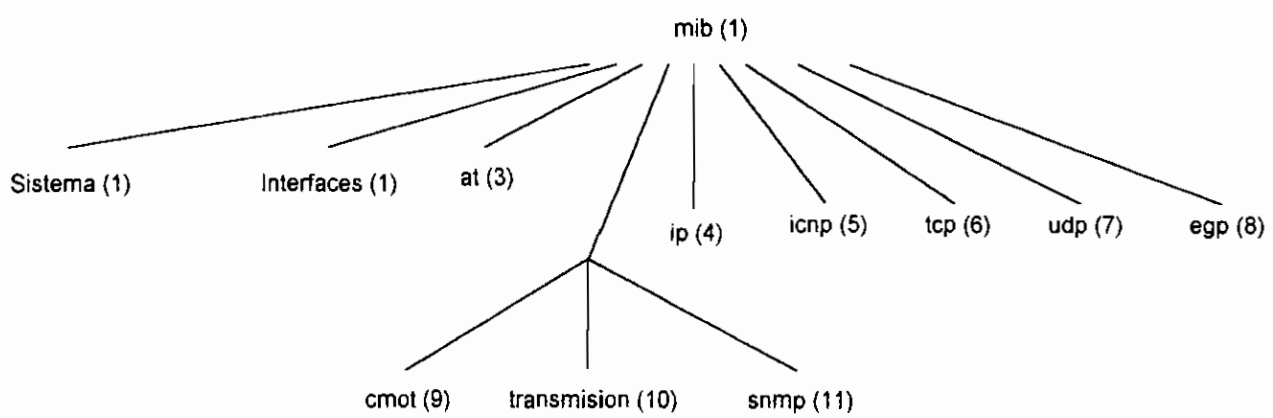


Figura 3.4: **El árbol para la mib Internet.**

3.7.3 Subárbol Experimental(experimental(3))

Es utilizado para experimentos de Internet, en lo relacionado a protocolos y también al desarrollo de MIBs, para que sean aceptados como estándares de gestión. Se lo conoce como el nodo {internet 3} o 1.3.6.1.3

3.7.4 Subárbol Privado(private(4))

Este subárbol permite a los fabricantes registrar sus propios objetos o definir sus Mibs privadas. Se lo conoce con el nombre de {internet 4} o 1.3.6.1.4

El subramal private(4) contiene un subramal etiquetado como enterprises(1), el cual mantiene o administra el registro para diferentes fabricantes de dispositivos de red. Este nodo empresarial(enterprises(1)), permite registrar a cada fabricante sus propias MIBs para sus equipos. Para muchos sistemas de gestión de red la porción más accedida es el nodo empresarial.

3.8 MIB PRIVADAS

Como se indicó anteriormente, el nodo empresarial (enterprises(1)), permite a los diferentes fabricantes de dispositivos de red definir sus propios grupos de objetos que mejoran la gestión de sus dispositivos y además poder distribuir esta información a otros usuarios que sean compatibles con el protocolo SNMP.

Por lo tanto cada fabricante debe registrarse, es decir que la IAB asigne un número identificador de objeto dentro del nodo empresarial.

Por ejemplo: Si una compañía llamada Zeus recibe una designación de subárbol con el número de referencia 22, su representación del objeto identificador deberá ser 1.3.6.1.4.22 o {enterprises 22}. Si esta compañía fabrica un nuevo producto, éste deberá tener el objeto identificador 1.3.6.1.4.22.1 y así sucesivamente con otros productos.

3.9 PROCEDIMIENTOS PARA LA IMPLANTACION Y UTILIZACION DE UNA MIB PRIVADA.

Una vez que se ha definido la forma como estructurar una MIB utilizando sus requisitos básicos para su descripción, tales como la identificación de los objetos gestionados que pertenecen a un determinando grupo, la definición de sus tipos de datos representados mediante el código ASN.1 y su correspondiente identificación dentro del árbol de gestión, se debe indicar el proceso para implantar una MIB privada la cual contiene información relacionada a un dispositivo de red perteneciente a un fabricante particular.

La descripción de la MIB se hace utilizando las reglas definidas en la estructura de información de gestión y el código ASN.1, esta descripción es un archivo de texto es decir que no tiene ningún formato especial, el cual contiene el conjunto de definiciones escrito en una lenguaje de alto nivel es decir que es entendible por los usuarios.

Esta descripción de la MIB en código ASN.1 debe ser traducida a un lenguaje de bajo nivel o de máquina, razón por la cual a una MIB también se lo conoce

como librería objeto, la cual es compilada con un compilador de MIBs, para que la información contenida en la MIB pueda ser entendida por los agentes de gestión SNMP y poder realizar procesos de gestión.

Existen dos compiladores de MIBs que pueden ser utilizados el compilador Mosy y el compilador Smic.

Por lo tanto el proceso de implantación de una MIB se puede resumir en los siguientes pasos:

1. Descripción de la MIB en código ASN 1 en un archivo de texto
2. Compilación del archivo de texto.
3. Instalación del archivo compilado en la Estación de Gestión
4. El sistema de gestión de red debe soportar el protocolo SNMP

CAPITULO 4: IMPLEMENTACION DE UNA BASE DE INFORMACION DE GESTION PRIVADA PARA UN REPETIDOR DE RADIOENLACE

INTRODUCCION

Aplicando los conocimientos descritos en los capítulos anteriores, tales como la utilización de la notación ASN.1 para la descripción de las Bases de Gestión de Información, así como también la estructura de la información y los diferentes tipos de datos y valores para los objetos gestionados, se procederá a la implantación de una base de información de gestión para un dispositivo de red típico utilizado en redes de telecomunicaciones.

Se tomara como referencia el dispositivo de red conocido como Repetidor de Radioenlace, para el cual se desarrollará su correspondiente MIB haciendo una descripción de los objetos más importantes que se requieren gestionar para este tipo de dispositivos y así poder realizar una óptima gestión de red.

En redes de telecomunicaciones, existen muchos dispositivos que requieren ser gestionados, pero el objetivo de este capítulo, es describir la MIB de un dispositivo particular, en este caso un Repetidor de Radioenlace, la cual puede ser considera como una recomendación para que los posibles fabricantes de repetidores de radioenlace estandaricen los objetos que pueden ser gestionados en un repetidor. Es decir que la MIB a desarrollarse pueda ser considerada como una RFC para repetidores de radioenlace.

4.1 DESCRIPCION DE LAS CARACTERISTICAS DE UN REPETIDOR

Del análisis de los diferentes equipos repetidores de radioenlace que existen el mercado de las telecomunicaciones, se tomará como referencia el Repetidor de Radioenlace Emisor y Receptor de la casa fabricante OMB de la serie OMB-PTRL 1-2, el cual es el más representativo de todos los repetidores que se consideraron para determinar la información necesaria para gestionar repetidores de RF.

Los datos técnicos del mismo se especifican a continuación:

EQUIPO DE RADIOENLACE EMISOR Y RECEPTOR

OMB-PTRL 1-2[38]

Datos técnicos comunes

Campo de frecuencia: 760 – 1100 MHz.

Impedancia de entrada/salida en R.F: 50 Ohms.

Dimensiones: 425 mm. Ancho y 84 mm. Grueso x 350 mm. Fondo

Adecuado para colocar en Rack estándar de 19"

Frecuencia de la red: 50 – 60 Hz.

Altura máxima de trabajo: 4000 m.

Datos técnicos del Transmisor

Estabilidad: ± 7 p.p.m.

Supresión de armónicos y espúreas: mejor que FCC y CCIR

[38] Manual de Especificaciones Técnicas OMB

Potencia de salida: 3 vatios

Relación señal/ruido: - 68 dB a 400 Hz.

Preéfasis: 50 a 75 uS.

Peso: 8 Kg.

Datos técnicos del Receptor

Distorsión armónica: < 0.5%

Nivel de Baja Frecuencia: 3.5 V.p.p max.

Sensibilidad R.F.: 50 uV x 20 dB. S/R

Deénfasis: 50 o 75 uS.

Impedancia de salida B.F.: 1 Kohm.

Peso: 8 Kg.

Las características técnicas proporcionadas por los fabricantes facilitan a los desarrolladores de MIBs la obtención de información necesaria a ser considerada en la tarea de gestión y además nos permiten definir aquellos elementos que pueden ser considerados como posibles objetos gestionados de una MIB, con el propósito de efectuar una adecuada gestión para este tipo de dispositivos.

4.2 DETERMINACION DE LOS OBJETOS GESTIONADOS

Considerando todas las definiciones dadas en los capítulos anteriores para el diseño de bases de información de gestión y basados en los conceptos de la técnica de objetos, se procederá a explicar los pasos que vamos a seguir para

llegar a la implantación de una base de información de gestión para un Repetidor de Radioenlace.

No todas las características técnicas especificadas por el fabricante pueden son consideradas como objetos gestionados, ya que dependen de la información que un usuario o administrador de red requiera para gestionar una red particular de una manera eficiente.

Tomando como referencia las definiciones para describir una MIB, en la cual se considera que los objetos a ser gestionados deben estar agrupados de acuerdo a una información común que ellos puedan proporcionar, los grupos se establecen de acuerdo a la información de gestión que se necesita obtener y además porque facilitan la descripción de la MIB.

Para que los objetos gestionados entreguen una determinada información de gestión se requiere que el objeto gestionado incluya un conjunto de elementos de hardware y software que permitan por ejemplo, sensar la potencia de transmisión del repetidor para que sea almacenada en la MIB.

En primer lugar, es necesario identificar los recursos del dispositivo que son necesarios gestionar en un repetidor, lo cual se lo hace describiendo que es lo que se necesita gestionar.

De un repetidor es necesario obtener la información básica que permita identificar al repetidor a ser gestionado como por ejemplo el rango de frecuencia, la potencia de salida y sensibilidad del receptor entre otras cosas.

También es muy importante conocer el estado actual de funcionamiento del equipo para determinar entre otras cosas si el repetidor está operando con alguna falla.

Otra tarea importante que se debe realizar es el monitoreo remoto del repetidor, tanto de las variables del transmisor como en el receptor, para obtener información tal como la potencia de salida, nivel de la señal de entrada, los voltajes de polarización, la frecuencia de operación del transmisor.

A más del monitoreo, una tarea muy frecuente es la de configurar remota mente el equipo, es decir tener la posibilidad de cambiar los parámetros de funcionamiento del equipo tales como la potencia del transmisor, la frecuencia de la portadora, etc.

4.3 ESTRUCTURA DE LA MIB

Para el caso particular de la MIB del Repetidor de Radioenlace recomendamos considerar cuatro grupos, los cuales van a facilitar la estructuración de la MIB para un mejor entendimiento tanto para el usuario así como también para las personas encargadas de realizar la gestión de red.

1. Grupo Básico
2. Grupo Operacional
3. Grupo Monitoreo
4. Grupo Configuración

4.3.1 Grupo Básico

Se recomienda considerar dentro de este grupo todos aquellos objetos que son aplicables a los repetidores de radioenlace en general, es decir contendrá los objetos que se puedan aplicar a un Repetidor sin importar la marca o características especiales que tengan estos dispositivos. Todos los objetos gestionados que pertenezcan a este grupo deben ser considerados en su descripción como objetos con estado mandatorio o principal.

4.3.2 Grupo Operacional

Los objetos que pertenecen a este grupo permiten entregar información relacionada al estado operacional actual del dispositivo, permitir inicializar el dispositivo y también realizar un autotest para comprobar el funcionamiento de sus partes.

4.3.3 Grupo Monitoreo

Se recomienda que los objetos pertenecientes a este grupo son aquellos que interesa monitorear para tener un buen control de la red, por lo que no es necesario realizar el monitoreo de todos los objetos gestionados que se definan para estos dispositivos. Por lo tanto en este grupo estarán todos aquellos objetos gestionados que permitan entregar estadísticas de monitoreo para un Repetidor de Radioenlace.

4.3.4 Grupo Configuración

Los objetos que pertenecen a este grupo deben permitir realizar la configuración remota de ciertos parámetros importantes del dispositivo sin necesidad de tener contacto físico con el equipo

4.3.5 Definición de los nombres para los grupos y objetos

Cada uno de los grupos, así como también los objetos que lo conforman deben tener su correspondiente nombre y su respectivo identificador de objeto dentro del árbol de la MIB particular. Cabe indicar que la MIB que se va a desarrollar esta ubicada en el nodo internet, bajo el nodo privado y bajo el nodo empresarial, para lo cual será necesario que la IAB designe un numero de referencia para ubicar esta MIB particular y describir sus grupos y los correspondientes objetos.

En la figura 4.1 se indica los cuatro grupos que conforman la MIB del Repetidor de Radioenlace,

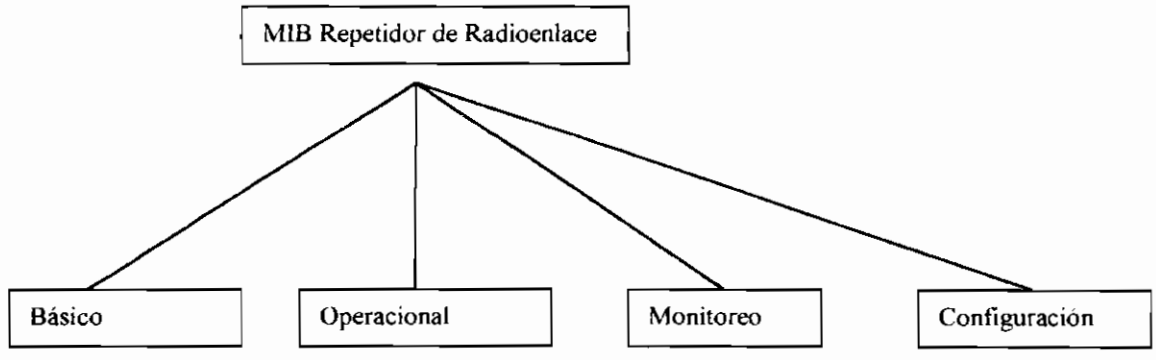


Figura 4.1: **Grupos de la MIB Repetidor de Radioenlace.**

El nombre que hemos seleccionado para identificar a la MIB del Repetidor de Radioenlace dentro del nodo empresarial es **snmpRpRadEnl** y para identificarla se asumirá el número de referencia 45 dentro del nodo empresarial, con la indicación de que este número solamente se asume para efecto de ubicar esta MIB particular, ya que este número debe ser autorizado y asignado por la IAB.

El árbol de ubicación de la MIB Repetidor de Radioenlace es el siguiente:

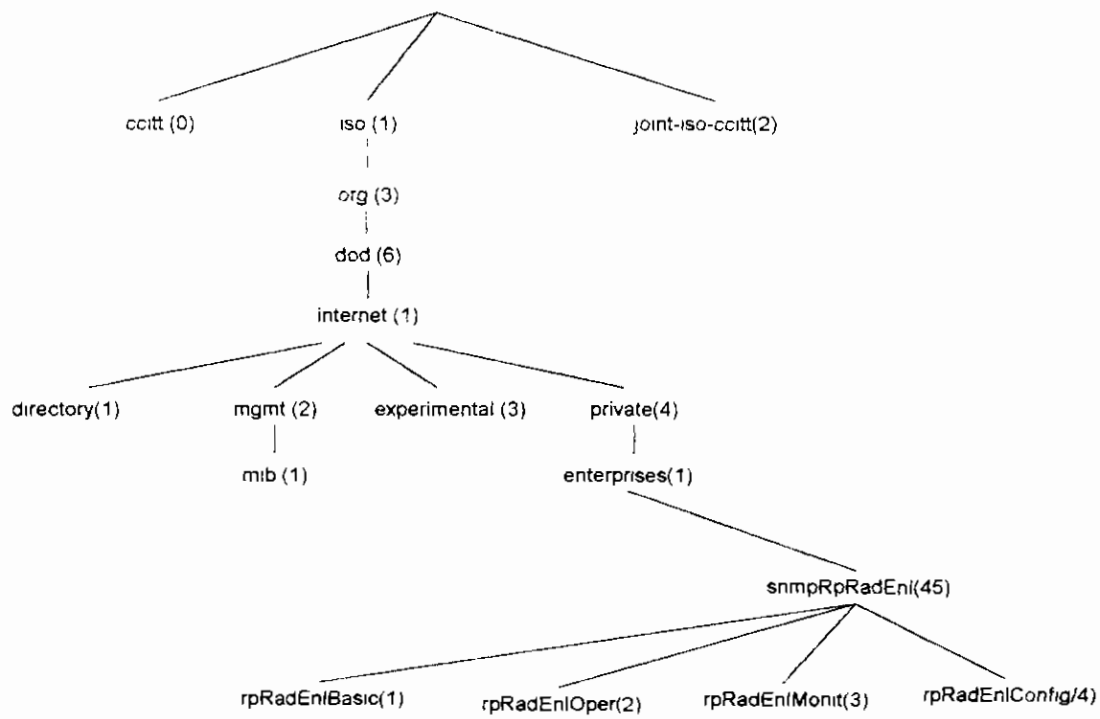


Figura 4.2: **Ubicación de la MIB Repetidor de Radioenlace y sus grupos**

4.3.6 Componentes del grupo Básico

El grupo Básico esta identificado como rpRadEnlBasic(1), el cual contiene tres subgrupos:

1. Grupo Información General: rpInfoGen(1)
2. Grupo Información del Transmisor: rpInfoTx(2)
3. Grupo Información del Receptor: rpInfoRx(3)

El árbol correspondiente al grupo Básico es el siguiente:

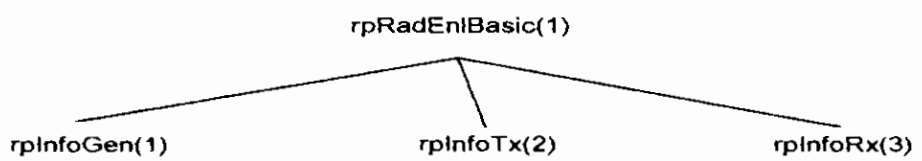


Figura 4.3: **Componentes del grupo Básico.**

Los objetos que pertenecen al grupo Información General son los siguientes:

- Rango de frecuencia de operación: ranFrecOp(1)
- Impedancia de entrada/salida: impEntSal(2)
- Dimensiones físicas: dimFis(3)
- Frecuencia de la red: frecRed(4)
- Altura máxima de trabajo: altMax(5)
- Peso del equipo: pesEqui(6)

El árbol correspondiente al grupo Información General es el siguiente:

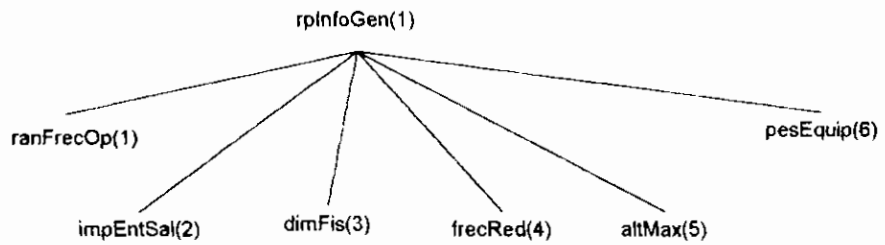


Figura 4.4: **Componentes del grupo Información General.**

Los objetos que pertenecen al grupo Información del Transmisor son los siguientes:

- Potencia de salida: potSal(1)
- Tipo de Transmisor: tipTrans(2)
- Estabilidad: estabTrans(3)
- Relación señal/ruido: relSenRuid(4)

El árbol correspondiente al grupo Información de Transmisor es el siguiente:

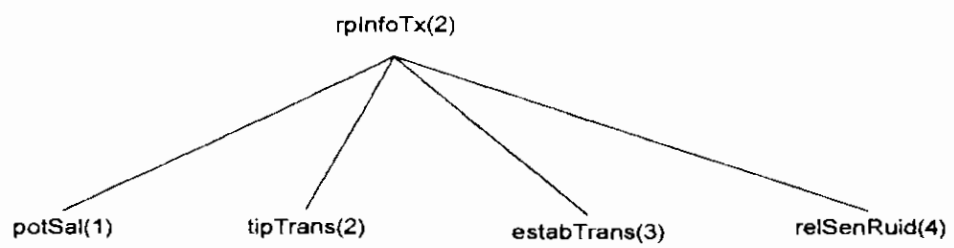


Figura 4.5: **Componentes del grupo Información del Transmisor.**

Los objetos que pertenecen al grupo Información del Receptor son los siguientes:

- Sensibilidad de RF: sensiRecep(1)
- Impedancia de salida: impSalRecep(2)

El árbol correspondiente al grupo Información del Receptor es:

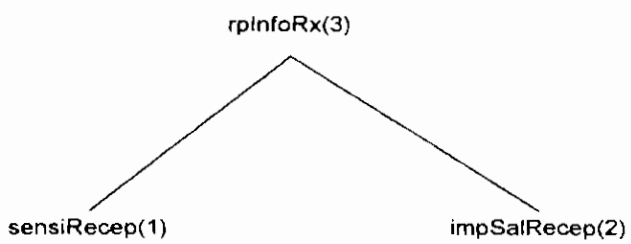


Figura 4.6: **Componentes del grupo Información del Receptor.**

4.3.7 Componentes del Grupo Operacional

El grupo Operacional esta identificado como rpRadEnlOper(1), el cual contiene tiene tres objetos:

- Estado operacional actual: rpRadEnlOperAct(1)
- Inicialización del equipo: rpRadEnlReset(2)
- Autotest del equipo: rpRadEnlAutotest(3)

El árbol correspondiente al grupo Operacional es el siguiente:

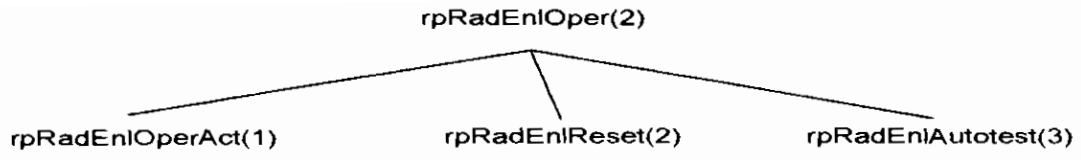


Figura 4.7: **Componentes del grupo Operacional.**

4.3.8 Componentes del grupo Monitoreo

El grupo Monitoreo esta identificado como rpRadEnlMonitor(2), contendrá dos subgrupos:

1. Grupo monitoreo del Transmisor: monitTrans(1)
2. Grupo monitoreo del Receptor: monitRecep(2)

Los objetos que pertenecen al grupo monitoreo del Transmisor son los siguientes:

- Potencia de salida: potSalTrans(1)
- Desviación de frecuencia: desvFrecTrans(2)
- Voltajes internos D.C.: voltIntDC(3)
- Temperatura: tempTrans(4)
- Frecuencia de operación: frecOpeTrans(5)
- Relación de onda estacionaria SWR: relOndEst(6)
- Modo de operación: modOperTrans(6)

El árbol correspondiente al grupo Monitoreo del Transmisor es el siguiente:

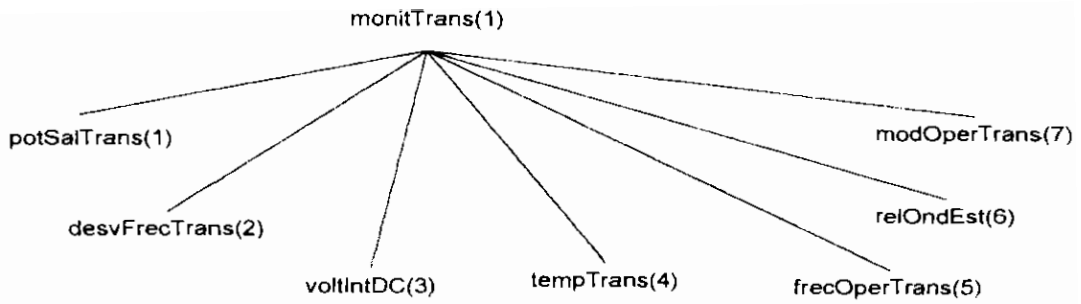


Figura 4.8: **Componentes del grupo Monitoreo del Transmisor.**

Los objetos que pertenecen al grupo monitoreo del Receptor son los siguientes:

- Nivel de RF de entrada: nivRFEnt(1)
- Indice de modulación: indMod(2)
- Temperatura: tempRecep(3)
- Modo de operación: modOperRecep(4)
- Desviación de frecuencia: desvFrecRecep(5)

El árbol correspondiente al grupo Monitoreo del Receptor es el siguiente:

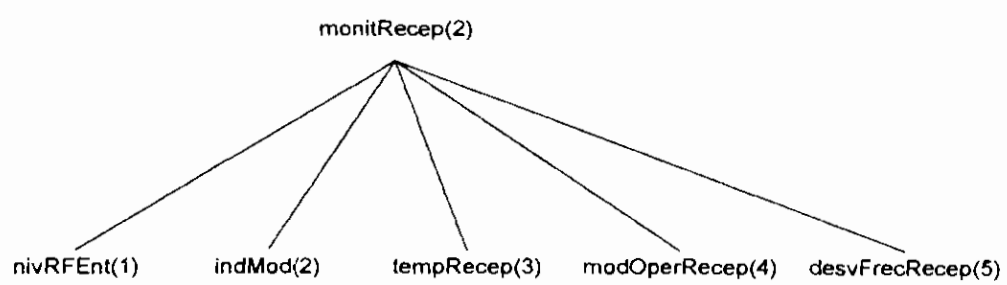


Figura 4.9: **Componentes del grupo Monitoreo del Receptor.**

4.3.9 Componentes del grupo Configuración

El grupo Configuración esta identificado como rpRadEnlConfig(3), el cual contiene dos subgrupos:

1. Grupo Configuración del Transmisor: configTrans(1)
2. Grupo Configuración del Receptor: configRecep(2)

Los objetos que pertenecen al grupo Configuración del Transmisor son los siguientes:

- Frecuencia de operación: configFrecTrans(1)
- Potencia del transmisor: configPotenTrans(2)
- Modo de operación: configModOperTrans(3)

Los objetos que pertenecen al grupo Configuración del Receptor son los siguientes:

- Frecuencia de operación: configFrecRecep(1)
- Amplitud de la señal de salida: configAmpRecep(2)
- Modo de operación: configModOperRecep(3)

El árbol correspondiente al grupo Configuración es el siguiente:

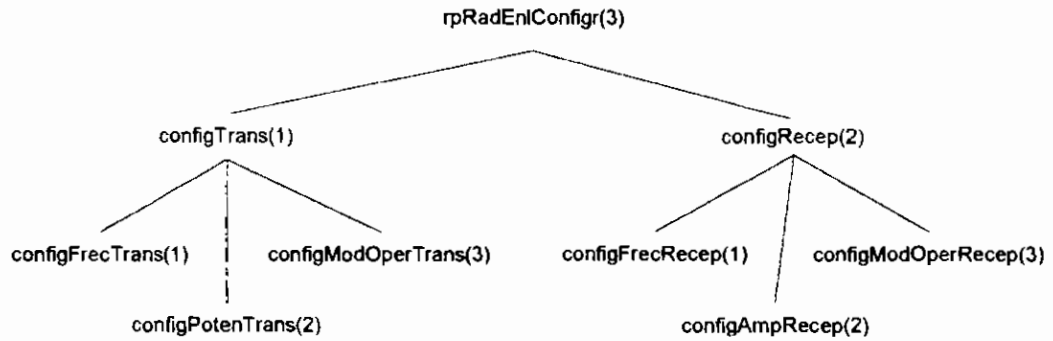


Figura 4.10: **Componentes del grupo Configuración.**

4.4 DESARROLLO DE LA MIB

Utilizando los conceptos básicos para estructurar una base de información de gestión, en la cual se emplea la estructura de información de gestión así como también la notación ASN.1 se procede al desarrollo de una base de información de gestión para un Repetidor de Radioenlace, para lo cual se considerará todos lo grupos y los objetos definidos anteriormente.

A continuación se realiza la descripción de la MIB Repetidor de Radioenlace:

```
-- MIB REPETIDOR DE RADIOENELACES
-- Desarrollado por: Carlos Herrera Muñoz
-- Escuela Politécnica Nacional
-- Octubre/1999

-- Definiciones

REPETIDOR-RADIOENLACES-MIB DEFINITIONS ::= BEGIN

  IMPORTS
    Counter, TimeTicks, Gauge
                              FROM RFC1155-SMI
    mib-1, DisplayString      FROM RFC1213-MIB
```

```
-- Definición del grupo principal y su correspondiente
-- ubicación dentro de la MIB Internet
  snmpRpRadEnl OBJECT IDENTIFIER ::= { mib-1 45 }

-- Definición de los grupos que conforman el grupo principal
-- El grupo principal tiene cuatro grupos: grupo Básico,
-- grupo Operacional, grupo Monitor y el grupo Configuración

   rpRadEnlBasic   OBJECT IDENTIFIER ::= { snmpRpRadEnl 1 }
   rpRadEnlOper   OBJECT IDENTIFIER ::= { snmpRpRadEnl 2 }
   rpRadEnlMonit OBJECT IDENTIFIER ::= { snmpRpRadEnl 3 }
   rpRadEnlConfig  OBJECT IDENTIFIER ::= { snmpRpRadEnl 4 }

-- Definición de los grupos pertenecientes al
-- grupo Básico

   rpInfoGen   OBJECT IDENTIFIER ::= { rpRadEnlBasic 1 }
   rpInfoTx     OBJECT IDENTIFIER ::= { rpRadEnlBasic 2 }
   rpInfoRx     OBJECT IDENTIFIER ::= { rpRadEnlBasic 3 }

-- Definición de los objetos pertenecientes al
-- grupo Operacional

  rpRadEnlOperAct     OBJECT IDENTIFIER ::= { rpRadEnlOper 1 }
  rpRadEnlReset       OBJECT IDENTIFIER ::= { rpRadEnlOper 2 }
  rpRadEnlautotest     OBJECT IDENTIFIER ::= { rpRadEnlOper 3 }
```

```
-- Definición de los grupos pertenecientes al
-- grupo Monitor

  monitTrans    OBJECT IDENTIFIER ::= { rpRadEnlMonit 1 }
  monitRecep    OBJECT IDENTIFIER ::= { rpRadEnlMonit 2 }


--Definición de los grupos pertenecientes al
-- grupo Configuración

 configTrans   OBJECT IDENTIFIER ::= { rpRadEnlConfig 1 }
 configRecep   OBJECT IDENTIFIER ::= { rpRadEnlConfig 2 }

-- Definición de los objetos pertenecientes a cada grupo
-- GRUPO BASICO
-- Definición de los objetos del grupo Información General
ranFrecOp    OBJECT IDENTIFIER ::= { rpInfoGen 1 }
impEntSal    OBJECT IDENTIFIER ::= { rpInfoGen 2 }
dimFis       OBJECT IDENTIFIER ::= { rpInfoGen 3 }
frecRed      OBJECT IDENTIFIER ::= { rpInfoGen 4 }
altMax       OBJECT IDENTIFIER ::= { rpInfoGen 5 }
pesEquip     OBJECT IDENTIFIER ::= { rpInfoGen 6 }
--
-- Definición de los objetos del grupo Información del Transmisor
potSal       OBJECT IDENTIFIER ::= { rpInfoTx 1 }
tipTrans     OBJECT IDENTIFIER ::= { rpInfoTx 2 }
estabTrans OBJECT IDENTIFIER ::= { rpInfoTX 3 }
relSenRuid OBJECT IDENTIFIER ::= { rpInfoTx 4 }
```

```
--
-- Definición de los objetos del grupo Información del Receptor

sensiRecep     OBJECT IDENTIFIER ::= { rpInfoRx 1 }

impSalRecep    OBJECT IDENTIFIER ::= { rpInfoRx 2 }


-- GRUPO MONITOREO
--
-- Definición de los objetos del grupo monitoreo del Transmisor

potSalTrans       OBJECT IDENTIFIER ::= { monitTrans 1 }

desvFrecTrans   OBJECT IDENTIFIER ::= { monitTrans 2 }

voltIntDC          OBJECT IDENTIFIER ::= { monitTrans 3 }

tempTrans         OBJECT IDENTIFIER ::= { monitTrans 4 }

frecOperTrans     OBJECT IDENTIFIER ::= { monitTrans 5 }

relOndEst         OBJECT IDENTIFIER ::= { monitTrans 6 }

modOperTrans    OBJECT IDENTIFIER ::= { monitTrans 7 }


--
-- Definición de los objetos del grupo monitoreo del Receptor

nivRFEnt          OBJECT IDENTIFIER ::= { monitRecep 1 }

indMod            OBJECT IDENTIFIER ::= { monitRecep 2 }

tempRecep         OBJECT IDENTIFIER ::= { monitRecep 3 }

modOperRecep  OBJECT IDENTIFIER ::= { monitRecep 4 }

desvFrecRecep  OBJECT IDENTIFIER ::= { monitRecep 5 }


--
-- GRUPO CONFIGURACION
--
-- Definición de los objetos del grupo Configuración del transmisor
```

```
configFrecTrans        OBJECT IDENTIFIER ::= { configTrans 1 }

configPotenTrans       OBJECT IDENTIFIER ::= { configTrans 2 }

configModOperTrans   OBJECT IDENTIFIER ::= { configTrans 3 }


--
-- Definición de los objetos del grupo Configuración del Receptor

configFrecRecep        OBJECT IDENTIFIER ::= { configRecep 1 }

configAmpRecep         OBJECT IDENTIFIER ::= { configRecep 2 }

configModOperRecep   OBJECT IDENTIFIER ::= { configRecep 3 }

--

-- Descripción de cada uno de los objetos

-- pertenecientes a  los diferentes grupos

--

-- GRUPO BASICO
--
-- Descripción de los objetos del grupo Información General
--

ranFrecOp          OBJECT-TYPE
        SYNTAX     DisplayString(SIZE(0.255))
        ACCESS     read-only
        STATUS     mandatory
        DESCRIPTION
                   " Este objeto proporciona información acerca del rango de
                    frecuencia de operación del equipo, la información esta en
                    formato cadena de caracteres"
: : = {rpInfoGen 1}
```

impEntSal OBJECT-TYPE

SYNTAX DisplayString(SIZE(0.255))

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información acerca de la impedancia de entrada del equipo, la información está en formato cadena de caracteres"

: : = {rpInfoGen 2}

dimFis OBJECT-TYPE

SYNTAX DisplayString(SIZE(0.255))

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información acerca de las dimensiones físicas del equipo, la información esta en formato cadena de caracteres"

: : = {rpInfoGen 3}

frecRed OBJECT-TYPE

SYNTAX DisplayString(SIZE(0.255))

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información acerca de la frecuencia de la red sobre la cual opera el equipo, la información esta en formato cadena de caracteres"

: : = {rpInfoGen 4}

altMax OBJECT-TYPE

SYNTAX DisplayString(SIZE(0.255))

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información acerca de la altura máxima permisible bajo la cual el equipo puede operar, la información esta en formato cadena de caracteres"

: : = {rpInfoGen 5}

pesEquip OBJECT-TYPE

SYNTAX DisplayString(SIZE(0.255))

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información acerca del peso de todo el equipo, la información esta en formato cadena de caracteres"

: : = {rpInfoGen 6}

--
-- Objetos del grupo Información del Transmisor
--

potSal OBJECT-TYPE

SYNTAX DisplayString(SIZE(0.255))

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información acerca de la potencia de salida del transmisor, la información esta en formato cadena de caracteres"

: : = {rpInfoTx 1}

tipTrans OBJECT-TYPE

SYNTAX DisplayString(SIZE(0.255))

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información acerca del tipo de transmisor sea sintetizado o no sintetizado, la información esta en formato cadena de caracteres"

: : = {rpInfoTx 2}

estabTrans OBJECT-TYPE

SYNTAX DisplayString(SIZE(0.255))

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información acerca los datos técnicos de la estabilidad del transmisor, la información esta en formato cadena de caracteres"

: : = {rpInfoTx 3}

relsenRuid OBJECT-TYPE

SYNTAX DisplayString(SIZE(0.255))

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información acerca los datos técnicos de la relación señal/ruido del transmisor, la información esta en formato de cadena de caracteres"

: : = {rpInfoTx 4}

--
-- Objetos del grupo Información del Receptor
--

sensiRecep OBJECT-TYPE

SYNTAX DisplayString(SIZE(0.255))

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información acerca de la sensibilidad de radio frecuencia del receptor, la información esta en formato cadena de caracteres"

: = {rpInfoRx 1}

```
impSalRecep                OBJECT-TYPE
        SYNTAX      DisplayString(SIZE(0.255))
        ACCESS      read-only
        STATUS      mandatory
        DESCRIPTION
                    " Este objeto proporciona información acerca de la
                    impedancia de salida del receptor, la información esta en
                    formato cadena de caracteres"
: : = {rpInfoRx 2}

--
-- GRUPO OPERACIONAL
--
-- Objetos del grupo Operacional
--

rpRadEnlOperAct   OBJECT-TYPE
        SYNTAX      INTEGER {
                            otros(1),
                            ok(2),
                            fallarep(3),
                            fallagrupo(4)
                            fallageneral(5)
                            }
        ACCESS      read-only
        STATUS      mandatory
        DESCRIPTION
```

" Este objeto proporciona información del estado operacional actual del repetidor, entregando reporte de fallas de acuerdo a los códigos definidos en la sintaxis del objeto"

: : = {rpRadEnlOper 1}

```
rpradEnlReset      OBJECT-TYPE
        SYNTAX     INTEGER {
                        noReset(1),
                        reset(2)
                              }
        ACCESS     read-write
        STATUS     mandatory
        DESCRIPTION
                   " Este objeto permite controlar el reseteo o no reseteo del
                   equipo, mediante una lectura y escritura de valores enteros
                   definidos como: 1 no resetea el equipo y 2 resetea el
                   equipo"
 : = {rpRadEnlOper 2}


rpRadEnlAutotest   OBJECT-TYPE
        SYNTAX     INTEGER {
                        noautotest(1),
                        autotest(2)
```

```
                                    }
        ACCESS      read-write
        STATUS      mandatory
        DESCRIPTION
                    " Este objeto permite controlar el equipo para realizar
                    tareas de autotest y verificar su estado actual, mediante la
                    utilización de los códigos definidos en la sintaxis"
: : = {rpRadEnlOper 3}

--
-- GRUPO MONITOREO
--
-- Objetos del grupo Monitoreo del Transmisor
--

potSalTrans         OBJECT-TYPE
        SYNTAX      INTEGER
        ACCESS      read-only
        STATUS      mandatory
        DESCRIPTION
                    " Este objeto proporciona información sobre actual potencia
                    de la potencia de salida del transmisor. El dato de potencia
                    puede no necesariamente puede ser entero por lo que se
                    debe considerar un formato de cuantos enteros y cuantos
                    decimales entrega la cadena de enteros, para discriminar
                    su valor real"
: : = {monitTrans 1}
```

desvfrecTrans OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información sobre la desviación de frecuencia del transmisor. Este dato no necesariamente puede ser entero por lo que se debe considerar un formato de cuantos enteros y cuantos decimales entrega la cadena de enteros, para discriminar su valor real"

: : = {monitTrans 2}

voltIntDC OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información acerca de los voltajes D.C. internos de polarización. Este dato no necesariamente puede ser entero por lo que se debe considerar un formato de cuantos enteros y cuantos decimales entrega la cadena de enteros, para discriminar su valor real"

: : = {monitTrans 3}

tempTrans OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información acerca de la temperatura interna a la que se encuentra el transmisor. Este dato no necesariamente puede ser entero por lo que se debe considerar un formato de cuantos enteros y cuantos decimales entrega la cadena de enteros, para discriminar su valor real"

: : = {monitTrans 4}

frecOperTrans OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información de la actual frecuencia a la que esta operando el transmisor. Este dato no necesariamente puede ser entero por lo que se debe considerar un formato de cuantos enteros y cuantos decimales entrega la cadena de enteros, para discriminar su valor real "

: : = {monitTrans 5}

```
relOndEst          OBJECT-TYPE
        SYNTAX     INTEGER
        ACCESS     read-only
        STATUS     mandatory
        DESCRIPTION
                   " Este objeto proporciona información sobre el SWR actual
                   del transmisor. Este dato no necesariamente puede ser
                   entero por lo que se debe considerar un formato de cuantos
                   enteros y cuantos decimales entrega la cadena de enteros,
                   para discriminar su valor real"
: : = {monitTrans 6}

modOperTrans       OBJECT-TYPE
        SYNTAX     INTEGER {
                           monofo(1),
                           estereo(2)
                                 }
        ACCESS     read-only
        STATUS     mandatory
        DESCRIPTION
                   " Este objeto proporciona información sobre el modo de
                   operación del transmisor el cual puede ser monofónico o
                   estereofónico, los cuales están definidos mediante códigos
                   en la sintaxis del objeto"
: : = {monitTrans 7}
```

```
--
-- Objetos del grupo Monitoreo del Receptor
--

nivRFEnt        OBJECT-TYPE
        SYNTAX  INTEGER
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                " Este objeto proporciona información relacionada al nivel
                de RF de entrada del receptor. Este dato no
                necesariamente puede ser entero por lo que se debe
                considerar un formato de cuantos enteros y cuantos
                decimales entrega la cadena de enteros, para discriminar
                su valor real"
: : = {monitRecep 1}

indMod          OBJECT-TYPE
        SYNTAX  INTEGER
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
                " Este objeto proporciona información relacionada al actual
                índice de modulación que tiene el receptor. Este dato no
                necesariamente puede ser entero por lo que se debe
                considerar un formato de cuantos enteros y cuantos
```

decimales entrega la cadena de enteros, para discriminar su valor real"

: : = {monitRecep 2}

tempRecep OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información sobre la temperatura actual que se encuentra el receptor. Este dato no necesariamente puede ser entero por lo que se debe considerar un formato de cuantos enteros y cuantos decimales entrega la cadena de enteros, para discriminar su valor real"

: : = {monitRecep 3}

modOperRecep OBJECT-TYPE

SYNTAX INTEGER {

mono(1),

estereo(2)

}

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información sobre el modo actual de operación del receptor, sea este monofónico o estereofónico, los cuales están definidos mediante códigos en la sintaxis del objeto "

: : = {monitRecep 4}

desvFrecRecep OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-only

STATUS mandatory

DESCRIPTION

" Este objeto proporciona información sobre la desviación de frecuencia del receptor. Este dato no necesariamente puede ser entero por lo que se debe considerar un formato de cuantos enteros y cuantos decimales entrega la cadena de enteros, para discriminar su valor real "

: : = {monitRecep 5}

--
-- **GRUPO CONFIGURACION**
--
-- Objetos del grupo Configuración del Transmisor
--

configFrecTrans OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-write

STATUS mandatory

DESCRIPTION

" Este objeto permite controlar la frecuencia de operación del transmisor, la cual puede ser leída, así como también modificada desde una estación de gestión. Este dato no necesariamente puede ser entero por lo que se debe considerar un formato de cuantos enteros y cuantos decimales entrega la cadena de enteros, para discriminar su valor real"

: : = {configTrans 1}

configPotenTrans OBJECT-TYPE

SYNTAX INTEGER

ACCESS read-write

STATUS mandatory

DESCRIPTION

" Este objeto permite controlar la potencia de salida del transmisor, la cual puede ser leída, así como también modificada desde una estación de gestión. Este dato no necesariamente puede ser entero por lo que se debe considerar un formato de cuantos enteros y cuantos decimales entrega la cadena de enteros, para discriminar su valor real"

: : = {configTrans 2}

configModOperTrans OBJECT-TYPE

SYNTAX INTEGER {

monofo(1),

```
                    estereo(2)
                        }
    ACCESS      read-write
    STATUS      mandatory
    DESCRIPTION
                " Este objeto permite controlar el modo de operación del
                transmisor, sea este monofónico o estereofónico, los cuales
                están definidos mediante códigos en la sintaxis del objeto"
: : = {configTrans 3}


--
-- Objetos del grupo Configuración del Receptor
--

configFrecRecep   OBJECT-TYPE
    SYNTAX      INTEGER
    ACCESS      read-write
    STATUS      mandatory
    DESCRIPTION
                " Este objeto permite controlar la frecuencia de operación
                del receptor, la cual puede ser leída, así como también
                modificada desde una estación de gestión. Este dato no
                necesariamente puede ser entero por lo que se debe
                considerar un formato de cuantos enteros y cuantos
                decimales entrega la cadena de enteros, para discriminar
                su valor real"
: : = {configRecep 1}
```

```
configAmpRecep    OBJECT-TYPE
        SYNTAX    INTEGER
        ACCESS    read-write
        STATUS    mandatory
        DESCRIPTION
                  " Este objeto permite controlar la amplitud de la señal de
                  recepción, la cual puede ser leída, así como también
                  modificada desde una estación de gestión. Este dato no
                  necesariamente puede ser entero por lo que se debe
                  considerar un formato de cuantos enteros y cuantos
                  decimales entrega la cadena de enteros, para discriminar
                  su valor real"
: : = {configRecep 2}

configModOperRecep    OBJECT-TYPE
        SYNTAX        INTEGER {
                          monofo(1),
                          estereo(2)
                              }
        ACCESS        read-write
        STATUS        mandatory
        DESCRIPTION
                  " Este objeto permite controlar el modo de operación del
                  receptor, cual puede ser leída, así como también
```

CAPITULO 5: CONCLUSIONES Y RECOMENDACIONES

- Las técnicas orientadas a objetos facilitan la solución de problemas de gestión de redes de Telecomunicaciones, debido fundamentalmente a que los elementos que conforman una red son considerados como "objetos" y además proporcionan un adecuado formato para definir sus características.

- Las técnicas de objetos no solamente pueden ser utilizadas para describir de una manera adecuada a los elementos de una red que se necesitan gestionar, si no que también ayudan a definir los procesos para la comunicación entre esos objetos.

- En la descripción de los objetos gestionados, únicamente se debe tomar en cuenta aquellos que se necesitan para poder realizar una adecuada tarea de gestión, por lo tanto primero se debe analizar qué información del elemento es importante gestionar.

- La MIB para un Repetidor de Radioenlace propuesta en la presente tesis, puede ser mejorada tomando en cuenta criterios de otras personas que están relacionadas con la gestión de sistemas de telecomunicaciones. Las RFC que son presentadas a la comunidad de Internet tienen la posibilidad de ser

mejoradas tomando como base los comentarios de los usuarios de Internet, con el propósito de que éstas se conviertan en un estándar de gestión de red.

No todos lo conceptos que se define en la técnica de objetos son aplicados en todos los procesos de gestión de red, si no que dependen de la arquitectura de gestión de red que se utilice para gestionar.

La técnica de objetos aplicada a la gestión de redes de telecomunicaciones cada vez se vuelve una realidad, a medida que los elementos de una red de telecomunicaciones están en capacidad de contener un agente de gestión que permita comunicarse con SNMP y así poder acceder y administrar la información contenida en una MIB.

Se recomienda profundizar el estudio de la técnica de objetos y determinar los posibles campos de aplicación en el área de las telecomunicaciones, para encontrar un método más eficiente de realizar los procesos de gestión de red.

En la actualidad la mayoría de los elementos que conforman una red de telecomunicaciones incluyen soporte para gestión de red con SNMP, para lo cual es necesario conocer los conceptos básicos de gestión así como también el entendimiento de las bases de información de gestión, para poder realizar la gestión de estos elementos. La presente tesis se puede tomar como una fuente básica de información para el entendimiento de estos conceptos.

Los tópicos tratados son de primordial importancia en la formación de los ingenieros especializados en comunicaciones con énfasis en la gestión de redes, campo que hasta el momento no ha tenido el desarrollo y difusión adecuados desde el punto de vista académico en la Facultad de Ingeniería Eléctrica de la Escuela Politécnica Nacional. Por lo tanto, se recomienda la incorporación de estos conceptos analizados en la elaboración del presente trabajo, en las asignaturas relacionadas con esta temática.

BIBLIOGRAFIA

1. BLACK Uyless, "Network Management Standards", McGraw-Hill Second edition, USA 1995

2. MILLER Mark, "Managing Internetworks with SNMP", M&T Books, USA 1993

3. LEINWAND Allan, FANG Conroy Karen, "Network Management a Practical Perspective", Second edition, Addison Wesley, USA 1996

4. TANENBAUM Andrew, "Redes de Ordenadores", Segunda edición, Prentice Hall, México 1991

5. CASED JD, "SNMP Network Management", Second Edition, McGRaw-Hill, USA 1996

6. Curso de Redes de Computadora, Protocolos de Gestión de Redes, http://roblepntic.mec.es/atei

7. AIDAROUS Salah, "Tecomunication Network Management into the 21ST Century", IEEE 1996

8. CORDOVA Raul, "Técnica Orientada a Objetos", Folleto UASB, Ecuador 1998

9. PRESMAN Roger, "Ingeniería de Software", Segunda edición, McGRaw-Hill, México 1990

10. EGAS Carlos, "Gestión de Redes", CLEI 1998

11. RFC 1155, "Structure and Identification of Management Information for TCP / IP – based Internets

12. RFC 1212, "Concise MIB Definitions"

Network Working Group
Request for Comments: 1155
Obsoletes: RFC 1065

M. Rose
Performance Systems International
K. McCloghrie
Hughes LAN Systems
May 1990

Structure and Identification of Management Information for TCP/IP-based Internets

Table of Contents

1. Status of this Memo

This RFC is a re-release of RFC 1065, with a changed "Status of this Memo", plus a few minor typographical corrections. The technical

content of the document is unchanged from RFC 1065.

This memo provides the common definitions for the structure and identification of management information for TCP/IP-based internets. In particular, together with its companion memos which describe the management information base along with the network management protocol, these documents provide a simple, workable architecture and system for managing TCP/IP-based internets and in particular, the Internet.

This memo specifies a Standard Protocol for the Internet community. Its status is "Recommended". TCP/IP implementations in the Internet which are network manageable are expected to adopt and implement this specification.

The Internet Activities Board recommends that all IP and TCP implementations be network manageable. This implies implementation of the Internet MIB (RFC-1156) and at least one of the two recommended management protocols SNMP (RFC-1157) or CMOT (RFC-1095). It should be noted that, at this time, SNMP is a full Internet standard and CMOT is a draft standard. See also the Host and Gateway Requirements RFCs for more specific information on the applicability of this standard.

Please refer to the latest edition of the "IAB Official Protocol Standards" RFC for current information on the state and status of standard Internet protocols.

Distribution of this memo is unlimited.

2. Introduction

This memo describes the common structures and identification scheme for the definition of management information used in managing TCP/IP-based internets. Included are descriptions of an object information model for network management along with a set of generic types used to describe management information. Formal descriptions of the structure are given using Abstract Syntax Notation One (ASN.1) [1].

This memo is largely concerned with organizational concerns and administrative policy: it neither specifies the objects which are managed, nor the protocols used to manage those objects. These concerns are addressed by two companion memos: one describing the Management Information Base (MIB) [2], and the other describing the Simple Network Management Protocol (SNMP) [3].

This memo is based in part on the work of the Internet Engineering

Task Force, particularly the working note titled "Structure and Identification of Management Information for the Internet" [4]. This memo uses a skeletal structure derived from that note, but differs in one very significant way: that note focuses entirely on the use of OSI-style network management. As such, it is not suitable for use with SNMP.

This memo attempts to achieve two goals: simplicity and extensibility. Both are motivated by a common concern: although the management of TCP/IP-based internets has been a topic of study for some time, the authors do not feel that the depth and breadth of such understanding is complete. More bluntly, we feel that previous experiences, while giving the community insight, are hardly conclusive. By fostering a simple SMI, the minimal number of constraints are imposed on future potential approaches; further, by fostering an extensible SMI, the maximal number of potential approaches are available for experimentation.

It is believed that this memo and its two companions comply with the guidelines set forth in RFC 1052, "IAB Recommendations for the Development of Internet Network Management Standards" [5] and RFC 1109, "Report of the Second Ad Hoc Network Management Review Group" [6]. In particular, we feel that this memo, along with the memo describing the management information base, provide a solid basis for network management of the Internet.

3. Structure and Identification of Management Information

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using Abstract Syntax Notation One (ASN.1) [1].

Each type of object (termed an object type) has a name, a syntax, and an encoding. The name is represented uniquely as an OBJECT IDENTIFIER. An OBJECT IDENTIFIER is an administratively assigned name. The administrative policies used for assigning names are discussed later in this memo.

The syntax for an object type defines the abstract data structure corresponding to that object type. For example, the structure of a given object type might be an INTEGER or OCTET STRING. Although in general, we should permit any ASN.1 construct to be available for use in defining the syntax of an object type, this memo purposely restricts the ASN.1 constructs which may be used. These restrictions are made solely for the sake of simplicity.

The encoding of an object type is simply how instances of that object type are represented using the object's type syntax. Implicitly tied to the notion of an object's syntax and encoding is how the object is represented when being transmitted on the network. This memo specifies the use of the basic encoding rules of ASN.1 [7].

It is beyond the scope of this memo to define either the MIB used for network management or the network management protocol. As mentioned earlier, these tasks are left to companion memos. This memo attempts to minimize the restrictions placed upon its companions so as to maximize generality. However, in some cases, restrictions have been made (e.g., the syntax which may be used when defining object types in the MIB) in order to encourage a particular style of management. Future editions of this memo may remove these restrictions.

3.1. Names

Names are used to identify managed objects. This memo specifies names which are hierarchical in nature. The OBJECT IDENTIFIER concept is used to model this notion. An OBJECT IDENTIFIER can be used for purposes other than naming managed object types; for example, each international standard has an OBJECT IDENTIFIER assigned to it for the purposes of identification. In short, OBJECT IDENTIFIERs are a means for identifying some object, regardless of the semantics associated with the object (e.g., a network object, a standards document, etc.)

An OBJECT IDENTIFIER is a sequence of integers which traverse a

global tree. The tree consists of a root connected to a number of labeled nodes via edges. Each node may, in turn, have children of its own which are labeled. In this case, we may term the node a subtree. This process may continue to an arbitrary level of depth. Central to the notion of the OBJECT IDENTIFIER is the understanding that administrative control of the meanings assigned to the nodes may be delegated as one traverses the tree. A label is a pairing of a brief textual description and an integer.

The root node itself is unlabeled, but has at least three children directly under it: one node is administered by the International Organization for Standardization, with label iso(1); another is administrated by the International Telegraph and Telephone Consultative Committee, with label ccitt(0); and the third is jointly administered by the ISO and the CCITT, joint-iso-ccitt(2).

Under the iso(1) node, the ISO has designated one subtree for use by other (inter)national organizations, org(3). Of the children nodes present, two have been assigned to the U.S. National Institutes of Standards and Technology. One of these subtrees has been transferred by the NIST to the U.S. Department of Defense, dod(6).

As of this writing, the DoD has not indicated how it will manage its subtree of OBJECT IDENTIFIERs. This memo assumes that DoD will allocate a node to the Internet community, to be administered by the Internet Activities Board (IAB) as follows:

```
internet    OBJECT IDENTIFIER ::= { iso org(3) dod(6) 1 }
```

That is, the Internet subtree of OBJECT IDENTIFIERs starts with the prefix:

```
1.3.6.1.
```

This memo, as a standard approved by the IAB, now specifies the policy under which this subtree of OBJECT IDENTIFIERs is administered. Initially, four nodes are present:

```
directory     OBJECT IDENTIFIER ::= { internet 1 }
mgmt          OBJECT IDENTIFIER ::= { internet 2 }
experimental  OBJECT IDENTIFIER ::= { internet 3 }
private       OBJECT IDENTIFIER ::= { internet 4 }
```

3.1.1. Directory

The directory(1) subtree is reserved for use with a future memo that discusses how the OSI Directory may be used in the Internet.

3.1.2. Mgmt

The mgmt(2) subtree is used to identify objects which are defined in IAB-approved documents. Administration of the mgmt(2) subtree is delegated by the IAB to the Internet Assigned Numbers Authority for the Internet. As RFCs which define new versions of the Internet-standard Management Information Base are approved, they are assigned an OBJECT IDENTIFIER by the Internet Assigned Numbers Authority for identifying the objects defined by that memo.

For example, the RFC which defines the initial Internet standard MIB would be assigned management document number 1. This RFC would use the OBJECT IDENTIFIER

```
{ mgmt 1 }
```

or

```
1.3.6.1.2.1
```

in defining the Internet-standard MIB.

The generation of new versions of the Internet-standard MIB is a rigorous process. Section 5 of this memo describes the rules used when a new version is defined.

3.1.3. Experimental

The experimental(3) subtree is used to identify objects used in Internet experiments. Administration of the experimental(3) subtree is delegated by the IAB to the Internet Assigned Numbers Authority of the Internet.

For example, an experimenter might received number 17, and would have available the OBJECT IDENTIFIER

```
{ experimental 17 }
```

or

```
1.3.6.1.3.17
```

for use.

As a part of the assignment process, the Internet Assigned Numbers Authority may make requirements as to how that subtree is used.

3.1.4. Private

The private(4) subtree is used to identify objects defined unilaterally. Administration of the private(4) subtree is delegated by the IAB to the Internet Assigned Numbers Authority for the Internet. Initially, this subtree has at least one child:

```
enterprises   OBJECT IDENTIFIER ::= { private 1 }
```

The enterprises(1) subtree is used, among other things, to permit parties providing networking subsystems to register models of their products.

Upon receiving a subtree, the enterprise may, for example, define new MIB objects in this subtree. In addition, it is strongly recommended that the enterprise will also register its networking subsystems under this subtree, in order to provide an unambiguous identification mechanism for use in management protocols. For example, if the "Flintstones, Inc." enterprise produced networking subsystems, then they could request a node under the enterprises subtree from the Internet Assigned Numbers Authority. Such a node might be numbered:

```
1.3.6.1.4.1.42
```

The "Flintstones, Inc." enterprise might then register their "Fred Router" under the name of:

```
1.3.6.1.4.1.42.1.1
```

3.2. Syntax

Syntax is used to define the structure corresponding to object types. ASN.1 constructs are used to define this structure, although the full generality of ASN.1 is not permitted.

The ASN.1 type ObjectSyntax defines the different syntaxes which may be used in defining an object type.

3.2.1. Primitive Types

Only the ASN.1 primitive types INTEGER, OCTET STRING, OBJECT IDENTIFIER, and NULL are permitted. These are sometimes referred to as non-aggregate types.

3.2.1.1. Guidelines for Enumerated INTEGERs

If an enumerated INTEGER is listed as an object type, then a named-number having the value 0 shall not be present in the list of

enumerations. Use of this value is prohibited.

3.2.2. Constructor Types

The ASN.1 constructor type SEQUENCE is permitted, providing that it is used to generate either lists or tables.

For lists, the syntax takes the form:

```
SEQUENCE { <type1>, ..., <typeN> }
```

where each <type> resolves to one of the ASN.1 primitive types listed above. Further, these ASN.1 types are always present (the DEFAULT and OPTIONAL clauses do not appear in the SEQUENCE definition).

For tables, the syntax takes the form:

```
SEQUENCE OF <entry>
```

where <entry> resolves to a list constructor.

Lists and tables are sometimes referred to as aggregate types.

3.2.3. Defined Types

In addition, new application-wide types may be defined, so long as they resolve into an IMPLICITly defined ASN.1 primitive type, list, table, or some other application-wide type. Initially, few application-wide types are defined. Future memos will no doubt define others once a consensus is reached.

3.2.3.1. NetworkAddress

This CHOICE represents an address from one of possibly several protocol families. Currently, only one protocol family, the Internet family, is present in this CHOICE.

3.2.3.2. IpAddress

This application-wide type represents a 32-bit internet address. It is represented as an OCTET STRING of length 4, in network byte-order.

When this ASN.1 type is encoded using the ASN.1 basic encoding rules, only the primitive encoding form shall be used.

3.2.3.3. Counter

This application-wide type represents a non-negative integer which

monotonically increases until it reaches a maximum value, when it wraps around and starts increasing again from zero. This memo specifies a maximum value of 2^32-1 (4294967295 decimal) for counters.

3.2.3.4. Gauge

This application-wide type represents a non-negative integer, which may increase or decrease, but which latches at a maximum value. This memo specifies a maximum value of 2^32-1 (4294967295 decimal) for gauges.

3.2.3.5. TimeTicks

This application-wide type represents a non-negative integer which counts the time in hundredths of a second since some epoch. When object types are defined in the MIB which use this ASN.1 type, the description of the object type identifies the reference epoch.

3.2.3.6. Opaque

This application-wide type supports the capability to pass arbitrary ASN.1 syntax. A value is encoded using the ASN.1 basic rules into a string of octets. This, in turn, is encoded as an OCTET STRING, in effect "double-wrapping" the original ASN.1 value.

Note that a conforming implementation need only be able to accept and recognize opaquely-encoded data. It need not be able to unwrap the data and then interpret its contents.

Further note that by use of the ASN.1 EXTERNAL type, encodings other than ASN.1 may be used in opaquely-encoded data.

3.3. Encodings

Once an instance of an object type has been identified, its value may be transmitted by applying the basic encoding rules of ASN.1 to the syntax for the object type.

4. Managed Objects

Although it is not the purpose of this memo to define objects in the MIB, this memo specifies a format to be used by other memos which define these objects.

An object type definition consists of five fields:

OBJECT:

A textual name, termed the OBJECT DESCRIPTOR, for the object type, along with its corresponding OBJECT IDENTIFIER.

Syntax:
The abstract syntax for the object type. This must resolve to an instance of the ASN.1 type ObjectSyntax (defined below).

Definition:
A textual description of the semantics of the object type. Implementations should ensure that their instance of the object fulfills this definition since this MIB is intended for use in multi-vendor environments. As such it is vital that objects have consistent meaning across all machines.

Access:
One of read-only, read-write, write-only, or not-accessible.

Status:
One of mandatory, optional, or obsolete.

Future memos may also specify other fields for the objects which they define.

4.1. Guidelines for Object Names

No object type in the Internet-Standard MIB shall use a sub-identifier of 0 in its name. This value is reserved for use with future extensions.

Each OBJECT DESCRIPTOR corresponding to an object type in the internet-standard MIB shall be a unique, but mnemonic, printable string. This promotes a common language for humans to use when discussing the MIB and also facilitates simple table mappings for user interfaces.

4.2. Object Types and Instances

An object type is a definition of a kind of managed object; it is

declarative in nature. In contrast, an object instance is an instantiation of an object type which has been bound to a value. For example, the notion of an entry in a routing table might be defined in the MIB. Such a notion corresponds to an object type; individual entries in a particular routing table which exist at some time are object instances of that object type.

A collection of object types is defined in the MIB. Each such subject type is uniquely named by its OBJECT IDENTIFIER and also has a textual name, which is its OBJECT DESCRIPTOR. The means whereby object instances are referenced is not defined in the MIB. Reference to object instances is achieved by a protocol-specific mechanism: it is the responsibility of each management protocol adhering to the SMI to define this mechanism.

An object type may be defined in the MIB such that an instance of that object type represents an aggregation of information also represented by instances of some number of "subordinate" object types. For example, suppose the following object types are defined in the MIB:

```
OBJECT:
-------
   atIndex { atEntry 1 }

Syntax:
   INTEGER

Definition:
   The interface number for the physical address.

Access:
   read-write.

Status:
   mandatory.


OBJECT:
-------
   atPhysAddress { atEntry 2 }

Syntax:
   OCTET STRING

Definition:
   The media-dependent physical address.
```

Access:
read-write.

Status:
mandatory.

OBJECT:
---
atNetAddress { atEntry 3 }

Syntax:
NetworkAddress

Definition:
The network address corresponding to the media-dependent physical address.

Access:
read-write.

Status:
mandatory.

Then, a fourth object type might also be defined in the MIB:

OBJECT:
---
atEntry { atTable 1 }

Syntax:

```
AtEntry ::= SEQUENCE {
      atIndex
      INTEGER,
      atPhysAddress
      OCTET STRING,
      atNetAddress
      NetworkAddress
      }
```

Definition:
An entry in the address translation table.

Access:
read-write.

Status:
mandatory.

Each instance of this object type comprises information represented by instances of the former three object types. An object type defined in this way is called a list.

Similarly, tables can be formed by aggregations of a list type. For example, a fifth object type might also be defined in the MIB:

OBJECT:
------
atTable { at 1 }

Syntax:
SEQUENCE OF AtEntry

Definition:
The address translation table.

Access:
read-write.

Status:
mandatory.

such that each instance of the atTable object comprises information represented by the set of atEntry object types that collectively constitute a given atTable object instance, that is, a given address translation table.

Consider how one might refer to a simple object within a table. Continuing with the previous example, one might name the object type

{ atPhysAddress }

and specify, using a protocol-specific mechanism, the object instance

{ atNetAddress } = { internet "10.0.0.52" }

This pairing of object type and object instance would refer to all instances of atPhysAddress which are part of any entry in some address translation table for which the associated atNetAddress value is { internet "10.0.0.52" }.

To continue with this example, consider how one might refer to an aggregate object (list) within a table. Naming the object type

```
{ atEntry }
```

and specifying, using a protocol-specific mechanism, the object instance

```
{ atNetAddress } = { internet "10.0.0.52" }
```

refers to all instances of entries in the table for which the associated atNetAddress value is { internet "10.0.0.52" }.

Each management protocol must provide a mechanism for accessing simple (non-aggregate) object types. Each management protocol specifies whether or not it supports access to aggregate object types. Further, the protocol must specify which instances are "returned" when an object type/instance pairing refers to more than one instance of a type.

To afford support for a variety of management protocols, all information by which instances of a given object type may be usefully distinguished, one from another, is represented by instances of object types defined in the MIB.

4.3. Macros for Managed Objects

In order to facilitate the use of tools for processing the definition of the MIB, the OBJECT-TYPE macro may be used. This macro permits the key aspects of an object type to be represented in a formal way.

```
OBJECT-TYPE MACRO ::=
BEGIN
    TYPE NOTATION ::= "SYNTAX" type (TYPE ObjectSyntax)
                      "ACCESS" Access
                      "STATUS" Status
    VALUE NOTATION ::= value (VALUE ObjectName)

    Access ::= "read-only"
                    | "read-write"
                    | "write-only"
                    | "not-accessible"
    Status ::= "mandatory"
                    | "optional"
                    | "obsolete"
    END
```

Given the object types defined earlier, we might imagine the following definitions being present in the MIB:

```
            atIndex OBJECT-TYPE
```

```
                SYNTAX  INTEGER
                ACCESS  read-write
                STATUS  mandatory
                ::= { atEntry 1 }

        atPhysAddress OBJECT-TYPE
                SYNTAX  OCTET STRING
                ACCESS  read-write
                STATUS  mandatory
                ::= { atEntry 2 }

        atNetAddress OBJECT-TYPE
                SYNTAX  NetworkAddress
                ACCESS  read-write
                STATUS  mandatory
                ::= { atEntry 3 }

        atEntry OBJECT-TYPE
                SYNTAX  AtEntry
                ACCESS  read-write
                STATUS  mandatory
                ::= { atTable 1 }

        atTable OBJECT-TYPE
                SYNTAX  SEQUENCE OF AtEntry
                ACCESS  read-write
                STATUS  mandatory
                ::= { at 1 }

        AtEntry ::= SEQUENCE {
            atIndex
                INTEGER,
            atPhysAddress
                OCTET STRING,
            atNetAddress
                NetworkAddress
        }
```

The first five definitions describe object types, relating, for example, the OBJECT DESCRIPTOR atIndex to the OBJECT IDENTIFIER { atEntry 1 }. In addition, the syntax of this object is defined (INTEGER) along with the access permitted (read-write) and status (mandatory). The sixth definition describes an ASN.1 type called AtEntry.

5. Extensions to the MIB

Every Internet-standard MIB document obsoletes all previous such documents. The portion of a name, termed the tail, following the OBJECT IDENTIFIER

```
{ mgmt version-number }
```

used to name objects shall remain unchanged between versions. New versions may:

(1) declare old object types obsolete (if necessary), but not delete their names;

(2) augment the definition of an object type corresponding to a list by appending non-aggregate object types to the object types in the list; or,

(3) define entirely new object types.

New versions may not:

(1) change the semantics of any previously defined object without changing the name of that object.

These rules are important because they admit easier support for multiple versions of the Internet-standard MIB. In particular, the semantics associated with the tail of a name remain constant throughout different versions of the MIB. Because multiple versions of the MIB may thus coincide in "tail-space," implementations supporting multiple versions of the MIB can be vastly simplified.

However, as a consequence, a management agent might return an instance corresponding to a superset of the expected object type. Following the principle of robustness, in this exceptional case, a manager should ignore any additional information beyond the definition of the expected object type. However, the robustness principle requires that one exercise care with respect to control actions: if an instance does not have the same syntax as its expected object type, then those control actions must fail. In both the monitoring and control cases, the name of an object returned by an operation must be identical to the name requested by an operation.

6. Definitions

```
RFC1155-SMI DEFINITIONS ::= BEGIN

EXPORTS -- EVERYTHING
        internet, directory, mgmt,
        experimental, private, enterprises,
        OBJECT-TYPE, ObjectName, ObjectSyntax, SimpleSyntax,
        ApplicationSyntax, NetworkAddress, IpAddress,
        Counter, Gauge, TimeTicks, Opaque;

 -- the path to the root

 internet      OBJECT IDENTIFIER ::= { iso org(3) dod(6) 1 }

 directory     OBJECT IDENTIFIER ::= { internet 1 }

 mgmt          OBJECT IDENTIFIER ::= { internet 2 }

 experimental  OBJECT IDENTIFIER ::= { internet 3 }

 private       OBJECT IDENTIFIER ::= { internet 4 }
 enterprises   OBJECT IDENTIFIER ::= { private 1 }


 -- definition of object types

 OBJECT-TYPE MACRO ::=
 BEGIN
     TYPE NOTATION ::= "SYNTAX" type (TYPE ObjectSyntax)
                       "ACCESS" Access
                       "STATUS" Status
     VALUE NOTATION ::= value (VALUE ObjectName)

     Access ::= "read-only"
                     | "read-write"
                     | "write-only"
                     | "not-accessible"
     Status ::= "mandatory"
                     | "optional"
                     | "obsolete"
 END

    -- names of objects in the MIB

    ObjectName ::=
        OBJECT IDENTIFIER
```

```
-- syntax of objects in the MIB

ObjectSyntax ::=
    CHOICE {
        simple
            SimpleSyntax,

-- note that simple SEQUENCEs are not directly
-- mentioned here to keep things simple (i.e.,
-- prevent mis-use).  However, application-wide
-- types which are IMPLICITly encoded simple
-- SEQUENCEs may appear in the following CHOICE

        application-wide
            ApplicationSyntax
    }

SimpleSyntax ::=
    CHOICE {
        number
            INTEGER,

        string
            OCTET STRING,

        object
            OBJECT IDENTIFIER,

        empty
            NULL
    }

ApplicationSyntax ::=
    CHOICE {
        address
            NetworkAddress,

        counter
            Counter,

        gauge
            Gauge,

        ticks
            TimeTicks,

        arbitrary
            Opaque
```

```
-- other application-wide types, as they are
-- defined, will be added here
    }


-- application-wide types

NetworkAddress ::=
    CHOICE {
        internet
            IpAddress
    }

IpAddress ::=
    [APPLICATION 0]          -- in network-byte order
        IMPLICIT OCTET STRING (SIZE (4))

Counter ::=
    [APPLICATION 1]
        IMPLICIT INTEGER (0..4294967295)

Gauge ::=
    [APPLICATION 2]
        IMPLICIT INTEGER (0..4294967295)

TimeTicks ::=
    [APPLICATION 3]
        IMPLICIT INTEGER (0..4294967295)

Opaque ::=
    [APPLICATION 4]          -- arbitrary ASN.1 value,
        IMPLICIT OCTET STRING   --   "double-wrapped"

END
```

7. Acknowledgements

This memo was influenced by three sets of contributors to earlier drafts:

First, Lee Labarre of the MITRE Corporation, who as author of the NETMAN SMI [4], presented the basic roadmap for the SMI.

Second, several individuals who provided valuable comments on this memo prior to its initial distribution:

James R. Davin, Proteon
Mark S. Fedor, NYSERNet
Craig Partridge, BBN Laboratories
Martin Lee Schoffstall, Rensselaer Polytechnic Institute
Wengyik Yeong, NYSERNet

Third, the IETF MIB working group:

Karl Auerbach, Epilogue Technology
K. Ramesh Babu, Excelan
Lawrence Besaw, Hewlett-Packard
Jeffrey D. Case, University of Tennessee at Knoxville
James R. Davin, Proteon
Mark S. Fedor, NYSERNet
Robb Foster, BBN
Phill Gross, The MITRE Corporation
Bent Torp Jensen, Convergent Technology
Lee Labarre, The MITRE Corporation
Dan Lynch, Advanced Computing Environments
Keith McCloghrie, The Wollongong Group
Dave Mackie, 3Com/Bridge
Craig Partridge, BBN (chair)
Jim Robertson, 3Com/Bridge
Marshall T. Rose, The Wollongong Group
Greg Satz, cisco
Martin Lee Schoffstall, Rensselaer Polytechnic Institute
Lou Steinberg, IBM
Dean Throop, Data General
Unni Warrier, Unisys

6. References

[1] Information processing systems - Open Systems Interconnection, "Specification of Abstract Syntax Notation One (ASN.1)", International Organization for Standardization, International Standard 8824, December 1987.

[2] McCloghrie K., and M. Rose, "Management Information Base for Network Management of TCP/IP-based Internets", RFC 1156, Performance Systems International and Hughes LAN Systems, May 1990.

[3] Case, J., M. Fedor, M. Schoffstall, and J. Davin, The Simple Network Management Protocol", RFC 1157, University of Tennessee at Knoxville, Performance Systems International, Performance Systems International, and the MIT Laboratory for Computer Science, May 1990.

[4] LaBarre, L., "Structure and Identification of Management Information for the Internet", Internet Engineering Task Force working note, Network Information Center, SRI International, Menlo Park, California, April 1988.

[5] Cerf, V., "IAB Recommendations for the Development of Internet Network Management Standards", RFC 1052, IAB, April 1988.

[6] Cerf, V., "Report of the Second Ad Hoc Network Management Review Group", RFC 1109, IAB, August 1989.

[7] Information processing systems - Open Systems Interconnection, "Specification of Basic Encoding Rules for Abstract Notation One (ASN.1)", International Organization for Standardization, International Standard 8825, December 1987.

Security Considerations

Security issues are not discussed in this memo.

Network Working Group
Request for Comments: 1212

M. Rose
Performance Systems International
K. McCloghrie
Hughes LAN Systems
Editors
March 1991

Concise MIB Definitions

Status of this Memo

This memo defines a format for producing MIB modules. This RFC specifies an IAB standards track document for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Table of Contents

1. Abstract

This memo describes a straight-forward approach toward producing concise, yet descriptive, MIB modules. It is intended that all future MIB modules be written in this format.

2. Historical Perspective

As reported in RFC 1052, IAB Recommendations for the Development of Internet Network Management Standards [1], a two-prong strategy for network management of TCP/IP-based internets was undertaken. In the short-term, the Simple Network Management Protocol (SNMP), defined in RFC 1067, was to be used to manage nodes in the Internet community. In the long-term, the use of the OSI network management framework was to be examined. Two documents were produced to define the management information: RFC 1065, which defined the Structure of Management Information (SMI), and RFC 1066, which defined the Management Information Base (MIB). Both of these documents were designed so as to be compatible with both the SNMP and the OSI network management framework.

This strategy was quite successful in the short-term: Internet-based network management technology was fielded, by both the research and commercial communities, within a few months. As a result of this, portions of the Internet community became network manageable in a timely fashion.

As reported in RFC 1109, Report of the Second Ad Hoc Network Management Review Group [2], the requirements of the SNMP and the OSI network management frameworks were more different than anticipated. As such, the requirement for compatibility between the SMI/MIB and both frameworks was suspended. This action permitted the operational network management framework, based on the SNMP, to respond to new operational needs in the Internet community by producing MIB-II.

In May of 1990, the core documents were elevated to "Standard Protocols" with "Recommended" status. As such, the Internet-standard network management framework consists of: Structure and Identification of Management Information for TCP/IP-based internets, RFC 1155 [3], which describes how managed objects contained in the

MIB are defined; Management Information Base for Network Management of TCP/IP-based internets, which describes the managed objects contained in the MIB, RFC 1156 [4]; and, the Simple Network Management Protocol, RFC 1157 [5], which defines the protocol used to manage these objects. Consistent with the IAB directive to produce simple, workable systems in the short-term, the list of managed objects defined in the Internet-standard MIB was derived by taking only those elements which are considered essential. However, the SMI defined three extensibility mechanisms: one, the addition of new standard objects through the definitions of new versions of the MIB; two, the addition of widely-available but non-standard objects through the experimental subtree; and three, the addition of private objects through the enterprises subtree. Such additional objects can not only be used for vendor-specific elements, but also for experimentation as required to further the knowledge of which other objects are essential.

As more objects are defined using the second method, experience has shown that the resulting MIB descriptions contain redundant information. In order to provide for MIB descriptions which are more concise, and yet as informative, an enhancement is suggested. This enhancement allows the author of a MIB to remove the redundant information, while retaining the important descriptive text.

Before presenting the approach, a brief presentation of columnar object handling by the SNMP is necessary. This explains and further motivates the value of the enhancement.

3. Columnar Objects

The SNMP supports operations on MIB objects whose syntax is ObjectSyntax as defined in the SMI. Informally stated, SNMP operations apply exclusively to scalar objects. However, it is convenient for developers of management applications to impose imaginary, tabular structures on the ordered collection of objects that constitute the MIB. Each such conceptual table contains zero or more rows, and each row may contain one or more scalar objects, termed columnar objects. Historically, this conceptualization has been formalized by using the OBJECT-TYPE macro to define both an object which corresponds to a table and an object which corresponds to a row in that table. (The ACCESS clause for such objects is "not-accessible", of course.) However, it must be emphasized that, at the protocol level, relationships among columnar objects in the same row is a matter of convention, not of protocol.

Note that there are good reasons why the tabular structure is not a matter of protocol. Consider the operation of the SNMP Get-Next-PDU acting on the last columnar object of an instance of a conceptual

row; it returns the next column of the first conceptual row or the first object instance occurring after the table. In contrast, if the rows were a matter of protocol, then it would instead return an error. By not returning an error, a single PDU exchange informs the manager that not only has the end of the conceptual row/table been reached, but also provides information on the next object instance, thereby increasing the information density of the PDU exchange.

3.1. Row Deletion

Nonetheless, it is highly useful to provide a means whereby a conceptual row may be removed from a table. In MIB-II, this was achieved by defining, for each conceptual row, an integer-valued columnar object. If a management station sets the value of this object to some value, usually termed "invalid", then the effect is one of invalidating the corresponding row in the table. However, it is an implementation-specific matter as to whether an agent removes an invalidated entry from the table. Accordingly, management stations must be prepared to receive tabular information from agents that corresponds to entries not currently in use. Proper interpretation of such entries requires examination of the columnar object indicating the in-use status.

3.2. Row Addition

It is also highly useful to have a clear understanding of how a conceptual row may be added to a table. In the SNMP, at the protocol level, a management station issues an SNMP set operation containing an arbitrary set of variable bindings. In the case that an agent detects that one or more of those variable bindings refers to an object instance not currently available in that agent, it may, according to the rules of the SNMP, behave according to any of the following paradigms:

(1) It may reject the SNMP set operation as referring to non-existent object instances by returning a response with the error-status field set to "noSuchName" and the error-index field set to refer to the first vacuous reference.

(2) It may accept the SNMP set operation as requesting the creation of new object instances corresponding to each of the object instances named in the variable bindings. The value of each (potentially) newly created object instance is specified by the "value" component of the relevant variable binding. In this case, if the request specifies a value for a newly (or previously) created object that it deems inappropriate by reason of value or

syntax, then it rejects the SNMP set operation by responding with the error-status field set to badValue and the error-index field set to refer to the first offending variable binding.

(3) It may accept the SNMP set operation and create new object instances as described in (2) above and, in addition, at its discretion, create supplemental object instances to complete a row in a conceptual table of which the new object instances specified in the request may be a part.

It should be emphasized that all three of the above behaviors are fully conformant to the SNMP specification and are fully acceptable, subject to any restrictions which may be imposed by access control and/or the definitions of the MIB objects themselves.

4. Defining Objects

The Internet-standard SMI employs a two-level approach towards object definition. A MIB definition consists of two parts: a textual part, in which objects are placed into groups, and a MIB module, in which objects are described solely in terms of the ASN.1 macro OBJECT-TYPE, which is defined by the SMI.

An example of the former definition might be:

```
OBJECT:
-------
     sysLocation { system 6 }

Syntax:
     DisplayString (SIZE (0..255))

Definition:
     The physical location of this node (e.g., "telephone
     closet, 3rd floor").

Access:
     read-only.

Status:
     mandatory.
```

An example of the latter definition might be:

```
     sysLocation OBJECT-TYPE
         SYNTAX  DisplayString (SIZE (0..255))
```

```
            ACCESS  read-only
            STATUS  mandatory
            ::= { system 6 }
```

In the interests of brevity and to reduce the chance of editing errors, it would seem useful to combine the two definitions. This can be accomplished by defining an extension to the OBJECT-TYPE macro:

```
IMPORTS
    ObjectName
        FROM RFC1155-SMI
    DisplayString
        FROM RFC1158-MIB;

OBJECT-TYPE MACRO ::=
BEGIN
    TYPE NOTATION ::=
                                -- must conform to
                                -- RFC1155's ObjectSyntax
                      "SYNTAX" type(ObjectSyntax)
                      "ACCESS" Access
                      "STATUS" Status
                      DescrPart
                      ReferPart
                      IndexPart
                      DefValPart
    VALUE NOTATION ::= value (VALUE ObjectName)

    Access ::= "read-only"
                    | "read-write"
                    | "write-only"
                    | "not-accessible"
    Status ::= "mandatory"
                    | "optional"
                    | "obsolete"
                    | "deprecated"

    DescrPart ::=
                "DESCRIPTION" value (description DisplayString)
                    | empty

    ReferPart ::=
                "REFERENCE" value (reference DisplayString)
                    | empty

    IndexPart ::=
                "INDEX" "{" IndexTypes "}"
```

```
                        | empty
    IndexTypes ::=
                IndexType | IndexTypes "," IndexType
    IndexType ::=
                        -- if indexobject, use the SYNTAX
                        -- value of the correspondent
                        -- OBJECT-TYPE invocation
                value (indexobject ObjectName)
                        -- otherwise use named SMI type
                        -- must conform to IndexSyntax below
                    | type (indextype)

    DefValPart ::=
                "DEFVAL" "{" value (defvalue ObjectSyntax) "}"
                    | empty

END

IndexSyntax ::=
    CHOICE {
        number
            INTEGER (0..MAX),
        string
            OCTET STRING,
        object
            OBJECT IDENTIFIER,
        address
            NetworkAddress,
        ipAddress
            IpAddress
    }
```

4.1. Mapping of the OBJECT-TYPE macro

It should be noted that the expansion of the OBJECT-TYPE macro is something which conceptually happens during implementation and not during run-time.

4.1.1. Mapping of the SYNTAX clause

The SYNTAX clause, which must be present, defines the abstract data structure corresponding to that object type. The ASN.1 language [6] is used for this purpose. However, the SMI purposely restricts the ASN.1 constructs which may be used. These restrictions are made expressly for simplicity.

4.1.2. Mapping of the ACCESS clause

The ACCESS clause, which must be present, defines the minimum level of support required for that object type. As a local matter, implementations may support other access types (e.g., an implementation may elect to permitting writing a variable marked as read-only). Further, protocol-specific "views" (e.g., those indirectly implied by an SNMP community) may make further restrictions on access to a variable.

4.1.3. Mapping of the STATUS clause

The STATUS clause, which must be present, defines the implementation support required for that object type.

4.1.4. Mapping of the DESCRIPTION clause

The DESCRIPTION clause, which need not be present, contains a textual definition of that object type which provides all semantic definitions necessary for implementation, and should embody any information which would otherwise be communicated in any ASN.1 commentary annotations associated with the object. Note that, in order to conform to the ASN.1 syntax, the entire value of this clause must be enclosed in double quotation marks, although the value may be multi-line.

Further, note that if the MIB module does not contain a textual description of the object type elsewhere then the DESCRIPTION clause must be present.

4.1.5. Mapping of the REFERENCE clause

The REFERENCE clause, which need not be present, contains a textual cross-reference to an object defined in some other MIB module. This is useful when de-osifying a MIB produced by some other organization.

4.1.6. Mapping of the INDEX clause

The INDEX clause, which may be present only if that object type corresponds to a conceptual row, defines instance identification information for that object type. (Historically, each MIB definition contained a section entitled "Identification of OBJECT instances for use with the SNMP". By using the INDEX clause, this section need no longer occur as this clause concisely captures the precise semantics needed for instance identification.)

If the INDEX clause is not present, and the object type corresponds to a non-columnar object, then instances of the object are identified

by appending a sub-identifier of zero to the name of that object. Further, note that if the MIB module does not contain a textual description of how instance identification information is derived for columnar objects, then the INDEX clause must be present.

To define the instance identification information, determine which object value(s) will unambiguously distinguish a conceptual row. The syntax of those objects indicate how to form the instance-identifier:

(1) integer-valued: a single sub-identifier taking the integer value (this works only for non-negative integers);

(2) string-valued, fixed-length strings: `n' sub-identifiers, where `n' is the length of the string (each octet of the string is encoded in a separate sub-identifier);

(3) string-valued, variable-length strings: `n+1' sub-identifiers, where `n' is the length of the string (the first sub-identifier is `n' itself, following this, each octet of the string is encoded in a separate sub-identifier);

(4) object identifier-valued: `n+1' sub-identifiers, where `n' is the number of sub-identifiers in the value (the first sub-identifier is `n' itself, following this, each sub-identifier in the value is copied);

(5) NetworkAddress-valued: `n+1' sub-identifiers, where `n' depends on the kind of address being encoded (the first sub-identifier indicates the kind of address, value 1 indicates an IpAddress); or,

(6) IpAddress-valued: 4 sub-identifiers, in the familiar a.b.c.d notation.

Note that if an "indextype" value is present (e.g., INTEGER rather than ifIndex), then a DESCRIPTION clause must be present; the text contained therein indicates the semantics of the "indextype" value.

By way of example, in the context of MIB-II [7], the following INDEX clauses might be present:

```
  objects under          INDEX clause
-----------------        ------------
ifEntry                  { ifIndex }
atEntry                  { atNetIfIndex,
                           atNetAddress }
ipAddrEntry              { ipAdEntAddr }
ipRouteEntry             { ipRouteDest }
ipNetToMediaEntry        { ipNetToMediaIfIndex,
                           ipNetToMediaNetAddress }
tcpConnEntry             { tcpConnLocalAddress,
                           tcpConnLocalPort,
                           tcpConnRemoteAddress,
                           tcpConnRemotePort }
udpEntry                 { udpLocalAddress,
                           udpLocalPort }
egpNeighEntry            { egpNeighAddr }
```

4.1.7. Mapping of the DEFVAL clause

The DEFVAL clause, which need not be present, defines an acceptable default value which may be used when an object instance is created at the discretion of the agent acting in conformance with the third paradigm described in Section 4.2 above.

During conceptual row creation, if an instance of a columnar object is not present as one of the operands in the correspondent SNMP set operation, then the value of the DEFVAL clause, if present, indicates an acceptable default value that the agent might use.

The value of the DEFVAL clause must, of course, correspond to the SYNTAX clause for the object. Note that if an operand to the SNMP set operation is an instance of a read-only object, then the error noSuchName will be returned. As such, the DEFVAL clause can be used to provide an acceptable default value that the agent might use.

It is possible that no acceptable default value may exist for any of the columnar objects in a conceptual row for which the creation of new object instances is allowed. In this case, the objects specified in the INDEX clause must have a corresponding ACCESS clause value of read-write.

By way of example, consider the following possible DEFVAL clauses:

ObjectSyntax	DEFVAL clause
INTEGER	1 -- same for Counter, Gauge, TimeTicks
OCTET STRING	'ffffffffffff'h
DisplayString	"any NVT ASCII string"
OBJECT IDENTIFIER	sysDescr
OBJECT IDENTIFIER	{ system 2 }
NULL	NULL
NetworkAddress	{ internet 'c0210415'h }
IpAddress	'c0210415'h -- 192.33.4.21

4.1.8. Mapping of the OBJECT-TYPE value

The value of an invocation of the OBJECT-TYPE macro is the name of the object, which is an object identifier.

4.2. Usage Example

Consider how the ipNetToMediaTable from MIB-II might be fully described:

```
        -- the IP Address Translation tables

        -- The Address Translation tables contain IpAddress to
        -- "physical" address equivalences.  Some interfaces do not
        -- use translation tables for determining address equivalences
        -- (e.g., DDN-X.25 has an algorithmic method); if all
        -- interfaces are of this type, then the Address Translation
        -- table is empty, i.e., has zero entries.

        ipNetToMediaTable OBJECT-TYPE
            SYNTAX  SEQUENCE OF IpNetToMediaEntry
            ACCESS  not-accessible
            STATUS  mandatory
            DESCRIPTION
                    "The IP Address Translation table used for mapping
                    from IP addresses to physical addresses."
            ::= { ip 22 }

        ipNetToMediaEntry OBJECT-TYPE
            SYNTAX  IpNetToMediaEntry
            ACCESS  not-accessible
            STATUS  mandatory
            DESCRIPTION
                    "Each entry contains one IpAddress to 'physical'
```

```
                address equivalence."
        INDEX   { ipNetToMediaIfIndex,
                  ipNetToMediaNetAddress }
        ::= { ipNetToMediaTable 1 }

    IpNetToMediaEntry ::=
        SEQUENCE {
            ipNetToMediaIfIndex
                INTEGER,
            ipNetToMediaPhysAddress
                OCTET STRING,
            ipNetToMediaNetAddress
                IpAddress,
            ipNetoToMediaType
                INTEGER
        }

    ipNetToMediaIfIndex OBJECT-TYPE
        SYNTAX  INTEGER
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The interface on which this entry's equivalence
                is effective.  The interface identified by a
                particular value of this index is the same
                interface as identified by the same value of
                ifIndex."
        ::= { ipNetToMediaEntry 1 }

    ipNetToMediaPhysAddress OBJECT-TYPE
        SYNTAX  OCTET STRING
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The media-dependent 'physical' address."
        ::= { ipNetToMediaEntry 2 }

    ipNetToMediaNetAddress OBJECT-TYPE
        SYNTAX  IpAddress
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The IpAddress corresponding to the media-
                dependent 'physical' address."
        ::= { ipNetToMediaEntry 3 }

    ipNetToMediaType OBJECT-TYPE
        SYNTAX  INTEGER {
```

```
                other(1),   -- none of the following
                invalid(2), -- an invalidated mapping
                dynamic(3),
                static(4)
            }
        ACCESS  read-write
        STATUS  mandatory
        DESCRIPTION
                "The type of mapping.

                Setting this object to the value invalid(2) has
                the effect of invalidating the corresponding entry
                in the ipNetToMediaTable.  That is, it effectively
                disassociates the interface identified with said
                entry from the mapping identified with said entry.
                It is an implementation-specific matter as to
                whether the agent removes an invalidated entry
                from the table.  Accordingly, management stations
                must be prepared to receive tabular information
                from agents that corresponds to entries not
                currently in use.  Proper interpretation of such
                entries requires examination of the relevant
                ipNetToMediaType object."
            ::= { ipNetToMediaEntry 4 }
```

5. Appendix: DE-osifying MIBs

There has been an increasing amount of work recently on taking MIBs defined by other organizations (e.g., the IEEE) and de-osifying them for use with the Internet-standard network management framework. The steps to achieve this are straight-forward, though tedious. Of course, it is helpful to already be experienced in writing MIB modules for use with the Internet-standard network management framework.

The first step is to construct a skeletal MIB module, e.g.,

```
        RFC1213-MIB DEFINITIONS ::= BEGIN

        IMPORTS
                experimental, OBJECT-TYPE, Counter
                    FROM RFC1155-SMI;

                -- contact IANA for actual number
        root    OBJECT IDENTIFIER ::= { experimental xx }

        END
```

The next step is to categorize the objects into groups. For experimental MIBs, optional objects are permitted. However, when a MIB module is placed in the Internet-standard space, these optional objects are either removed, or placed in a optional group, which, if implemented, all objects in the group must be implemented. For the first pass, it is wisest to simply ignore any optional objects in the original MIB: experience shows it is better to define a core MIB module first, containing only essential objects; later, if experience demands, other objects can be added.

It must be emphasized that groups are "units of conformance" within a MIB: everything in a group is "mandatory" and implementations do either whole groups or none.

5.1. Managed Object Mapping

Next for each managed object class, determine whether there can exist multiple instances of that managed object class. If not, then for each of its attributes, use the OBJECT-TYPE macro to make an equivalent definition.

Otherwise, if multiple instances of the managed object class can exist, then define a conceptual table having conceptual rows each containing a columnar object for each of the managed object class's attributes. If the managed object class is contained within the containment tree of another managed object class, then the assignment of an object type is normally required for each of the "distinguished attributes" of the containing managed object class. If they do not already exist within the MIB module, then they can be added via the definition of additional columnar objects in the conceptual row corresponding to the contained managed object class.

In defining a conceptual row, it is useful to consider the optimization of network management operations which will act upon its columnar objects. In particular, it is wisest to avoid defining more columnar objects within a conceptual row, than can fit in a single PDU. As a rule of thumb, a conceptual row should contain no more than approximately 20 objects. Similarly, or as a way to abide by the "20 object guideline", columnar objects should be grouped into tables according to the expected grouping of network management operations upon them. As such, the content of conceptual rows should reflect typical access scenarios, e.g., they should be organized along functional lines such as one row for statistics and another row for parameters, or along usage lines such as commonly-needed objects versus rarely-needed objects.

On the other hand, the definition of conceptual rows where the number of columnar objects used as indexes outnumbers the number used to

hold information, should also be avoided. In particular, the splitting of a managed object class's attributes into many conceptual tables should not be used as a way to obtain the same degree of flexibility/complexity as is often found in MIB's with a myriad of optionals.

5.1.1. Mapping to the SYNTAX clause

When mapping to the SYNTAX clause of the OBJECT-type macro:

(1) An object with BOOLEAN syntax becomes an INTEGER taking either of values true(1) or false(2).

(2) An object with ENUMERATED syntax becomes an INTEGER, taking any of the values given.

(3) An object with BIT STRING syntax containing no more than 32 bits becomes an INTEGER defined as a sum; otherwise if more than 32 bits are present, the object becomes an OCTET STRING, with the bits numbered from left-to-right, in which the least significant bits of the last octet may be "reserved for future use".

(4) An object with a character string syntax becomes either an OCTET STRING or a DisplayString, depending on the repertoire of the character string.

(5) An non-tabular object with a complex syntax, such as REAL or EXTERNAL, must be decomposed, usually into an OCTET STRING (if sensible). As a rule, any object with a complicated syntax should be avoided.

(6) Tabular objects must be decomposed into rows of columnar objects.

5.1.2. Mapping to the ACCESS clause

This is straight-forward.

5.1.3. Mapping to the STATUS clause

This is usually straight-forward; however, some osified-MIBs use the term "recommended". In this case, a choice must be made between "mandatory" and "optional".

5.1.4. Mapping to the DESCRIPTION clause

This is straight-forward: simply copy the text, making sure that any

embedded double quotation marks are sanitized (i.e., replaced with single-quotes or removed).

5.1.5. Mapping to the REFERENCE clause

This is straight-forward: simply include a textual reference to the object being mapped, the document which defines the object, and perhaps a page number in the document.

5.1.6. Mapping to the INDEX clause

Decide how instance-identifiers for columnar objects are to be formed and define this clause accordingly.

5.1.7. Mapping to the DEFVAL clause

Decide if a meaningful default value can be assigned to the object being mapped, and if so, define the DEFVAL clause accordingly.

5.2. Action Mapping

Actions are modeled as read-write objects, in which writing a particular value results in the action taking place.

5.2.1. Mapping to the SYNTAX clause

Usually an INTEGER syntax is used with a distinguished value provided for each action that the object provides access to. In addition, there is usually one other distinguished value, which is the one returned when the object is read.

5.2.2. Mapping to the ACCESS clause

Always use read-write.

5.2.3. Mapping to the STATUS clause

This is straight-forward.

5.2.4. Mapping to the DESCRIPTION clause

This is straight-forward: simply copy the text, making sure that any embedded double quotation marks are sanitized (i.e., replaced with single-quotes or removed).

5.2.5. Mapping to the REFERENCE clause

This is straight-forward: simply include a textual reference to the

action being mapped, the document which defines the action, and perhaps a page number in the document.

6. Acknowledgements

This document was produced by the SNMP Working Group:

Anne Ambler, Spider
Karl Auerbach, Sun
Fred Baker, ACC
Ken Brinkerhoff
Ron Broersma, NOSC
Jack Brown, US Army
Theodore Brunner, Bellcore
Jeffrey Buffum, HP
John Burress, Wellfleet
Jeffrey D. Case, University of Tennessee at Knoxville
Chris Chiptasso, Spartacus
Paul Ciarfella, DEC
Bob Collet
John Cook, Chipcom
Tracy Cox, Bellcore
James R. Davin, MIT-LCS
Eric Decker, cisco
Kurt Dobbins, Cabletron
Nadya El-Afandi, Network Systems
Gary Ellis, HP
Fred Engle
Mike Erlinger
Mark S. Fedor, PSI
Richard Fox, Synoptics
Karen Frisa, CMU
Chris Gunner, DEC
Fred Harris, University of Tennessee at Knoxville
Ken Hibbard, Xylogics
Ole Jacobsen, Interop
Ken Jones
Satish Joshi, Synoptics
Frank Kastenholz, Racal-Interlan
Shimshon Kaufman, Spartacus
Ken Key, University of Tennessee at Knoxville
Jim Kinder, Fibercom
Alex Koifman, BBN
Christopher Kolb, PSI
Cheryl Krupczak, NCR
Paul Langille, DEC
Peter Lin, Vitalink
John Lunny, TWG

Management Information for TCP/IP-based internets", RFC 1155, Performance Systems International, Hughes LAN Systems, May 1990.

[4] McCloghrie K., and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets", RFC 1156, Hughes LAN Systems, Performance Systems International, May 1990.

[5] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", RFC 1157, SNMP Research, Performance Systems International, Performance Systems International, MIT Laboratory for Computer Science, May 1990.

[6] Information processing systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1), International Organization for Standardization International Standard 8824, December 1987.

[7] Rose M., Editor, "Management Information Base for Network Management of TCP/IP-based internets: MIB-II", RFC 1213, Performance Systems International, March 1991.

8. Security Considerations

Security issues are not discussed in this memo.

9. Authors' Addresses

Marshall T. Rose
Performance Systems International
5201 Great America Parkway
Suite 3106
Santa Clara, CA 95054

Phone: +1 408 562 6222
EMail: mrose@psi.com
X.500: rose, psi, us

Keith McCloghrie
Hughes LAN Systems
1225 Charleston Road
Mountain View, CA 94043
1225 Charleston Road
Mountain View, CA 94043

Phone: (415) 966-7934
EMail: kzm@his.com

ANEXO C

ESPECIFICACIONES TECNICAS DE UN REPETIDOR

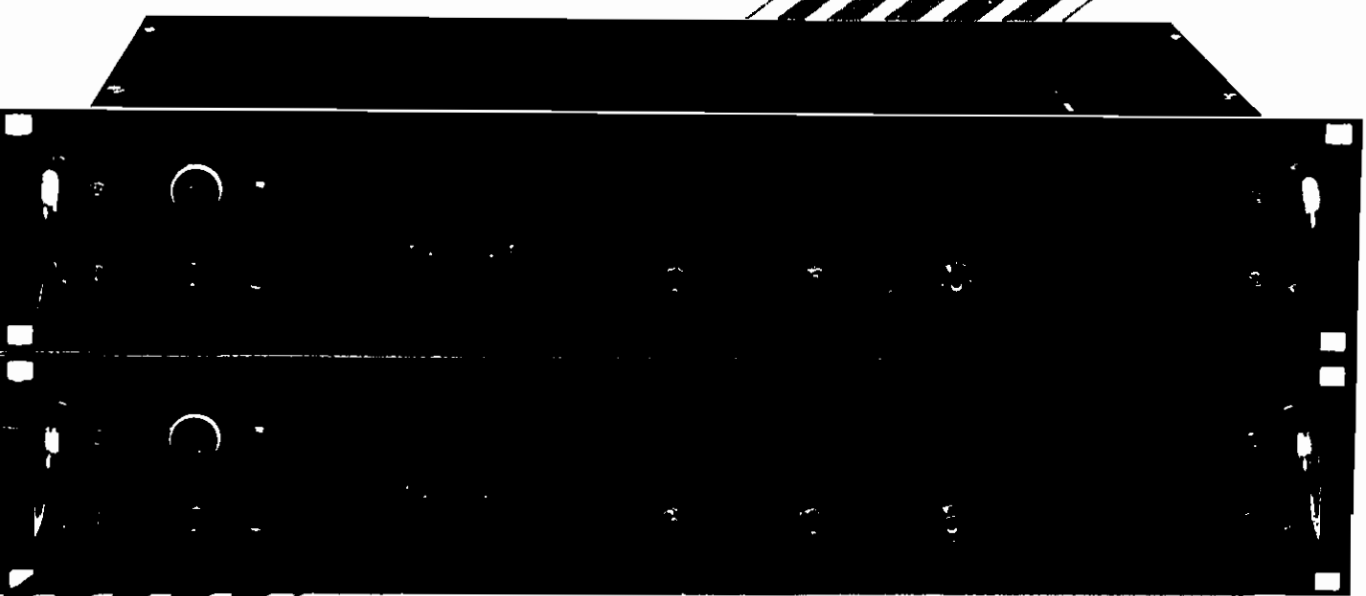
EQUIPO DE RADIOENLACE EMISOR Y RECEPTOR

OMB-PTRL 1-2

OMB

Al Servicio de la Radio y T.V.

La UNIVERSIDAD POLITECNICA DE VALENCIA certifica que este equipo ha pasado satisfactoriamente la prueba de ACEPTACION RADIOFONICA.



Datos técnicos comunes:

Campo de frecuencia: 760 - 1.100 Mhz.
Impedancia de entrada / salida en R.F.: 50 Ohms.
Dimensiones: 425 mm. Ancho x 84 mm. Grueso x 350 mm. Fondo
Adecuado para colocar en Rack estandard de 19"
Frecuencia de la red: 50 - 60 Hz.
Altitud máxima de trabajo: 4.000 m.

Datos técnicos del:

Transmisor

Estabilidad: ± 7 p.p.m.
Supresión de armónicos y espúreas: mejor que FCC y CCIR
Potencia de salida: 3 vatios
Relación señal/ruido: -68 dB. a 400 Hz.
Preénfasis: 50 a 75 uS.
Peso: 8 Kg.

Receptor

Distorsión armónica: < 0,5%
Nivel de Baja Frecuencia: 3,5 V.p.p. max.
Sensibilidad R.F.: 50 µV. x 20 dB. S/R
Deénfasis: 50 ó 75 µS.
Impedancia de salida B.F.: 1 K Ohm.
Peso: 8 Kg.

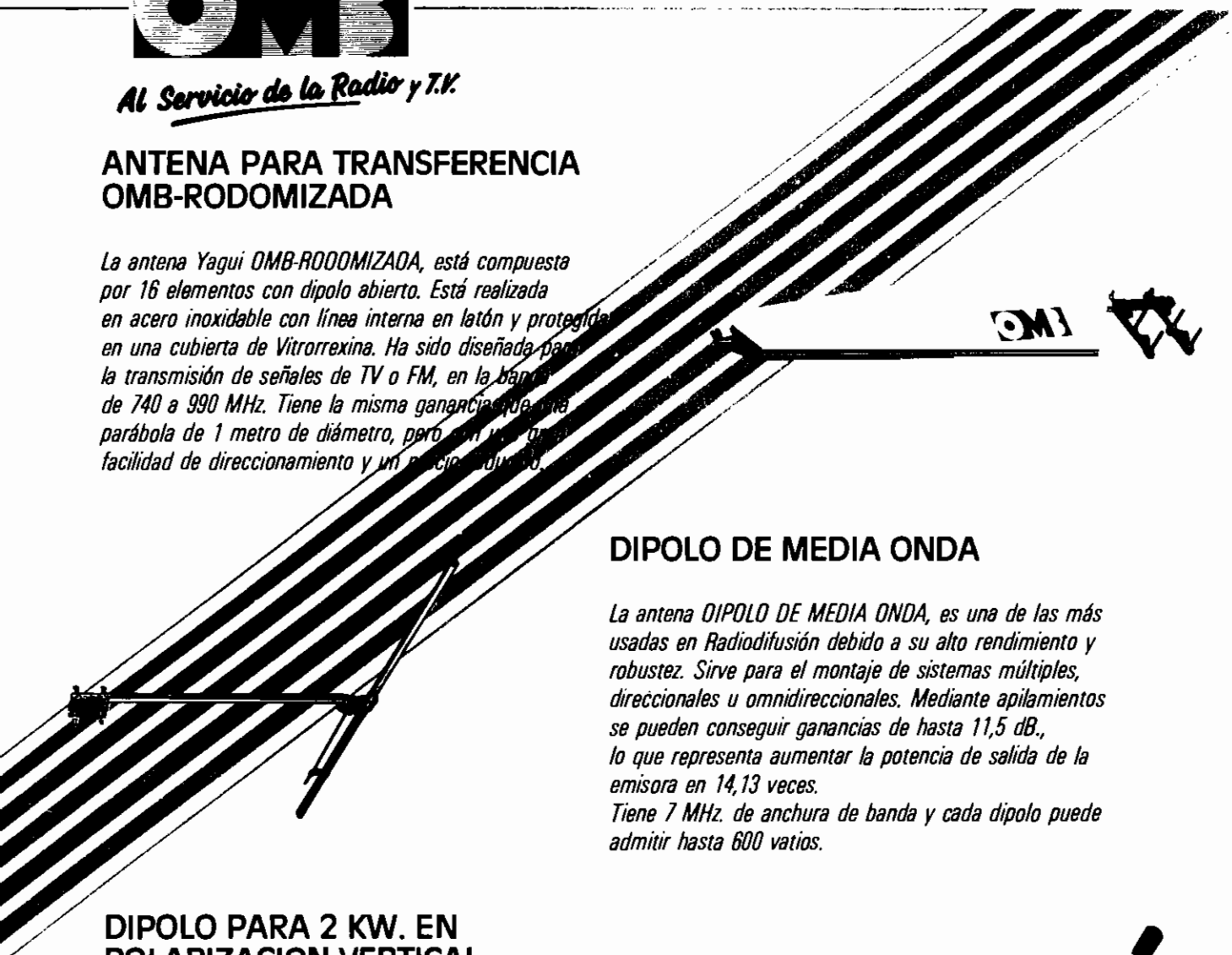
Radioenlace profesional sintetizado para la unión herziana de la baja y la alta frecuencia, totalmente indicado con arreglo a las nuevas disposiciones legales, dentro del campo de frecuencias de 830 a 860 MHz.

ANTENAS PROFESIONALES OMB

Al Servicio de la Radio y T.V.

ANTENA PARA TRANSFERENCIA OMB-RODOMIZADA

La antena Yagui OMB-ROOOMIZAOA, está compuesta por 16 elementos con dipolo abierto. Está realizada en acero inoxidable con línea interna en latón y protegida en una cubierta de Vitrorrexina. Ha sido diseñada para la transmisión de señales de TV o FM, en la banda de 740 a 990 MHz. Tiene la misma ganancia que una parábola de 1 metro de diámetro, pero con mayor facilidad de direccionamiento y un precio reducido.

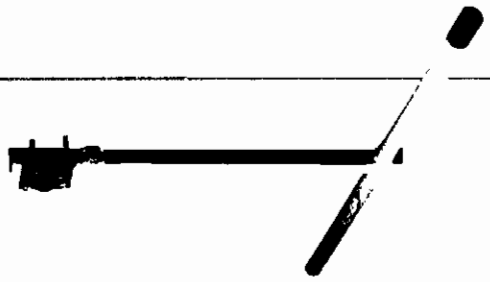


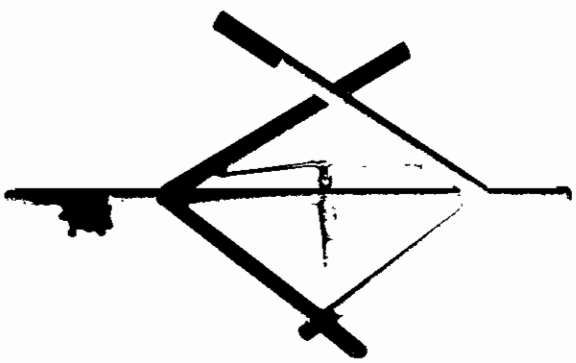
DIPOLO DE MEDIA ONDA

La antena OIPOLO DE MEDIA ONDA, es una de las más usadas en Radiodifusión debido a su alto rendimiento y robustez. Sirve para el montaje de sistemas múltiples, direccionales u omnidireccionales. Mediante apilamientos se pueden conseguir ganancias de hasta 11,5 dB., lo que representa aumentar la potencia de salida de la emisora en 14,13 veces.
Tiene 7 MHz. de anchura de banda y cada dipolo puede admitir hasta 600 vatios.

DIPOLO PARA 2 KW. EN POLARIZACION VERTICAL

Es un dipolo de una gran robustez, omnidireccional y forrado de Vitrorrexina haciéndolo totalmente inmune a los agentes atmosféricos, capacitado para admitir grandes potencias y consiguiendo con apilamientos de 4 ó 6 dipolos, ganancias de 7,5 y 10 dB. Cubre cualquier frecuencia de las comprendidas entre los 87,5 y 108 MHz.





ANTENA DE DIPOLOS CRUZADOS

Esta es la antena ideal cuando se desea radiar, tanto en un plano de polarización vertical como en el horizontal. Se compone de dos dipolos cruzados en un soporte horizontal por donde pasa la línea de alimentación hasta la unión de los dipolos. Todo el conjunto está forrado e impermeabilizado por una gruesa capa de vitrorrexina. La banda en que puede trabajar es de 87'5 a 108 MHz, su admisión de 2 KW. y la radiación del 50% en cada plano.