

INSTITUTO TECNOLÓGICO DE COSTA RICA

Escuela de Ingeniería Electrónica



“Diseño e instalación de un sistema de monitorización en tiempo real de servidores GSM para la red de telefonía móvil del ICE.”

Informe de Proyecto de Graduación para optar por el título de Ingeniero en Electrónica con el grado académico de Licenciatura.

Roiner Juvenal Jiménez Arias

Cartago, Junio 2010

INSTITUTO TECNOLOGICO DE COSTA RICA
ESCUELA DE INGENIERIA ELECTRONICA
PROYECTO DE GRADUACIÓN
TRIBUNAL EVALUADOR

Proyecto de Graduación defendido ante el presente Tribunal Evaluador como requisito para optar por el título de Ingeniero en Electrónica con el grado académico de Licenciatura, del Instituto Tecnológico de Costa Rica.

Miembros del Tribunal



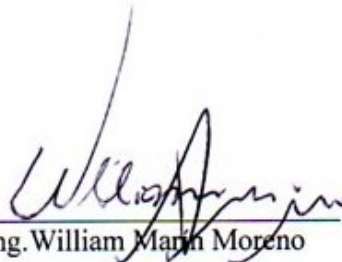
Ing. Anibal Coto Cortés

Profesor lector



Ing. Néstor Hernández Hostaller

Profesor lector



Ing. William Marín Moreno

Profesor asesor

Los miembros de este Tribunal dan fe de que el presente trabajo de graduación ha sido aprobado y cumple con las normas establecidas por la Escuela de Ingeniería Electrónica

Cartago, 24 junio de 2010

Declaración de Autenticidad.

Declaro que el presente Proyecto de Graduación ha sido realizado enteramente por mi persona, utilizando y aplicando literatura referente al tema e introduciendo conocimientos propios.

En los casos en que he utilizado bibliografía, he procedido a indicar las fuentes mediante las respectivas citas bibliográficas.

En consecuencia, asumo la responsabilidad total por el trabajo de graduación realizado y por el contenido del correspondiente informe final.

Cartago, Abril 2010



Roiher Juvenal Jiménez Arias

Céd: 2-0612-0088

Resumen

El presente proyecto describe el proceso de diseño y de incorporación de un sistema de monitorización completo y confiable, el cual permita a ERICSSON y en particular al grupo de Operación y Mantenimiento (O&M) a mantener activa y funcional la red GSM del ICE, para que sea capaz de evitar y por lo tanto resolver a tiempo las fallas sin que el usuario final de la telefonía móvil tenga inconvenientes.

El desarrollo de este proyecto contempla la programación de *scripts* y el funcionamiento correcto de un software de monitoreo, en el cual se utiliza el protocolo SNMP y los MIB propietarios para cada equipo dentro de la red y el MPBN (*Mobile Packet Backbone Network*) de ERICSSON.

Palabras Clave: SNMP, Script, MIB, OID, CACTI, PERL, Linux.

Abstract

This project explains the development of a reliable complete monitoring system, that allows the Ericsson's Operation and Maintenance group to keep up the ICE GSM network, preventing and solving just in time potential failures in the system; avoiding this way any setback or inconvenient to the final user.

The development of this project involves the scripting and the proper functioning of monitoring software, using the SNMP MIB, for each equipment within the MPBN (Mobile Packet Backbone Network) network of ERICSSON.

Keywords: SNMP, Script, MIB, OID, CACTI, PERL, Linux.

Dedicatoria

A mi familia por el apoyo brindado en especial a mis hermanas.

Agradecimiento

A mis profesores por el conocimiento compartido y al equipo de ingenieros de O&M de ERICSSON por aceptarme en su grupo de trabajo.

ÍNDICE GENERAL

Capítulo 1: Introducción.....	14
1.1 Problema existente e importancia de su solución.....	14
1.2 Solución seleccionada.....	17
1.2.1 Requerimientos.....	17
Capítulo 2: Meta y objetivos.....	19
2.1 Meta.....	19
2.2 Objetivo general.....	19
2.3 Objetivos específicos.....	19
Capítulo 3: Marco Teórico.....	21
3.1 Descripción del sistema por mejorar.....	21
3.1.1 Descripción del BACKBONE MPBN ERICSSON.....	21
3.1.2 Descripción de los principales nodos.....	22
3.1.3 Sistema de monitorización MRTG.....	24
3.2 Antecedentes bibliográficos	27
3.2.1 Gestión de red.	27
3.2.2 Protocolo SMI (Structure of management information).....	27
3.2.3 Protocolo SNMP (Simple Network Management Protocol).....	29
3.2.4 Protocolo MIB-II (Management Information Base).....	30
3.2.5 Red informática.....	32

3.2.6 Lenguaje de programación de alto nivel	34
3.2.7 Script	35
3.2.8 Algoritmo de planificación Round Robin	35
3.3 Descripción de los principales principios físicos y/o electrónicos relacionados con la solución del problema.....	36
3.3.1 Medidas de información.....	36
3.2.2 Tráfico en una red informática.....	37
Capítulo 4: Procedimiento metodológico.....	38
4.1 Reconocimiento y definición del problema.....	39
4.2 Obtención y análisis de información.....	41
4.3 Evaluación de las alternativas y síntesis de una solución	43
4.4 Implementación de la solución.....	46
4.5 Reevaluación y rediseño.....	48
Capítulo 5: Descripción detallada de la solución (Explicación del diseño).....	49
5.1 Análisis de soluciones y selección final.....	49
5.2 Descripción del software.....	52
5.2.1 Script para MRTG.....	52
5.2.2 Script para CACTI.....	54
5.2.3 Configuración de sendmail.....	57
5.2.4 Configuración de Apache, PHP y MYSQL.....	58

5.2.5 Configuración de CACTI.....	61
Capítulo 6: Análisis de Resultados.....	62
6.1 Análisis	62
Capítulo 7: Conclusiones y recomendaciones.....	73
7.1 Conclusiones.....	73
7.2 Recomendaciones.....	74
Referencias.....	75
Apéndices y anexos.....	78
A.1 Glosario, abreviaturas y simbología.....	78
A.2 Manual de usuario.....	79
A.3 Scripts	86
A.3.1 Script MRTG.....	86
A.3.1 Script CACTI.....	93
A.4 Información sobre la empresa	100
A.4.1 Descripción de la empresa.....	100
A.4.2 Descripción del departamento.....	100

ÍNDICE DE FIGURAS

Figura 1.1.	Diagrama general de la red GSM del ICE.....	15
Figura 1.2.	Requerimientos y visión general del proyecto.....	18
Figura 3.1.	Arquitectura del Sistema GSM ERICSSON.....	22
Figura 3.2.	Vista de la salida del programa MRTG.....	25
Figura 3.3.	Vista de la página principal de estadísticas de ERICSSON.	26
Figura 3.4.	Diagrama de bloques del software MRTG.....	27
Figura 3.5.	OID privado del agente MIB de NetScreen®.	29
Figura 3.6.	Diagrama general de una NIC enviando una trama Ethernet.....	33
Figura 3.7.	Algoritmo de planificación Round Robin.....	36
Figura 4.1.	Salida del software de monitorización actual.	41
Figura 4.2.	Archivo de configuración para el MRTG ejecutándose en Windows®.	44
Figura 4.3.	Diagrama de flujo del script realizado para almacenar los datos del MRTG en una base de datos MYSQL	45
Figura 4.4.	Vista principal del software CACTI.	47
Figura 4.5.	Salida del programa NET-SNMP.	47
Figura 4.6.	Diagrama de flujo del script de alerta.....	48
Figura 5.1.	Salida del OID propietario de NetScreen gráfico en MRTG.....	49

Figura 5.2.	Diagrama de bloques del programa CACTI.....	50
Figura 5.3.	Inicio del Script para MRTG.....	53
Figura 5.4.	Diagrama de flujo del script para CACTI de las interfaces.....	54
Figura 5.5.	Diagrama de flujo del script para CACTI de los OID gestionados..	55
Figura 5.6.	Subrutina enviar_mail1() escrito en PERL	57
Figura 5.7.	Inicio del programa CACTI..	62
Figura 6.1.	Gráfica generada por CACTI del tráfico por la interfaz Eth 3/1/5.	63
Figura 6.2.	Hoja de Excel con los valores del promedio de carga descargados desde la base de datos.	64
Figura 6.3.	Gráfica generada por CACTI de la última semana del tráfico por la interfaz Eth 3/1/5.	65
Figura 6.4.	Foto del correo enviado por el script de alarma.	65
Figura 6.5.	Uso del CPU de uno de los centros de mensajes del ICE.	68
Figura 6.6.	Equipos monitorizados en el MPBN ERICSSON.	69
Figura 6.7.	Gráfica de salida de la aplicación MRTG, para la interfaz Eth 1/1del Firewall Netscreen®.....	70
Figura 6.8.	Gráfica de salida de la aplicación CACTI, para la interfaz Eth 1/1 del Firewall Netscreen®.....	70
Figura 6.9.	Vista de todos los gráficos de un equipo.	71

ÍNDICE DE TABLAS

Tabla 4.1. Especificaciones y requerimientos del sistema a implementar en el MPBN ERICSSON.....	40
Tabla 4.2. Características de los Software para monitorización de redes.	42
Tabla 6.1. Tiempos de respuesta para reparación de averías antes y después del sistema	66
Tabla 6.2. Multas aplicadas cuando existe afectación de tráfico en la red.	67

Capítulo 1: Introducción.

En este capítulo se presenta y explica el contexto en el que se ubica el problema y el por qué su solución es de importancia para ERICSSON. Se hace referencia a la necesidad que presenta la empresa y como se llegó a su solución.

1.1 Problema existente e importancia de su solución.

La central de telecomunicaciones GSM de telefonía celular de segunda generación (2G), que es un servicio para todo el país y a la cual ERICSSON da soporte y mantenimiento, debería tener una disponibilidad del 100%. Los equipos que conforman la red (servidores, cortafuegos, enrutadores, etc.), deben presentar un rendimiento óptimo; al estar redundantes evita que la telefonía móvil 2G se vea afectada en su totalidad en caso de una falla, sin embargo es importante que si esta fallara sea reestablecida a la mayor brevedad debido a que existe una posibilidad que el equipo redundante pueda tener una deficiencia en su funcionamiento provocando que el servicio sea interrumpido.

Cuando alguno de los nodos de telecomunicaciones de telefonía celular deja de funcionar la multa que cobra el proveedor de servicios a ERICSSON asciende a los \$500 por minuto [21], además provoca molestias en los usuarios porque no pueden realizar sus llamadas y pueden tener dificultades en otros servicios que se prestan vía GSM.

Entre los tipos de fallos que comúnmente se presentan se encuentran sobrecalentamiento, sobrecarga y problemas de software, además de ciertos factores ajenos al equipo de trabajo de mantenimiento como el malware y problemas muy específicos de hardware, tales factores pueden provocar que el servicio se suspenda.

Cuanto más rápido se conozca y se analice la falla menor será el tiempo que se tardará en corregirlo y con ello menor la afectación del servicio brindado.

La monitorización de los equipos se debe realizar 24/7, con el fin de ser eficaces en el tiempo de respuesta para la solución de la falla que afecta la comunicación móvil de usuarios finales y que puede minimizar los ingresos al proveedor de servicios.

En la figura 1.1 se muestra un diagrama general de la red GSM del ICE, en donde se aprecia que la parte fundamental que debe estar siempre en funcionamiento es el backbone MPBN (*Mobile Packet Backbone Network*), el cual debe monitorizarse continuamente.

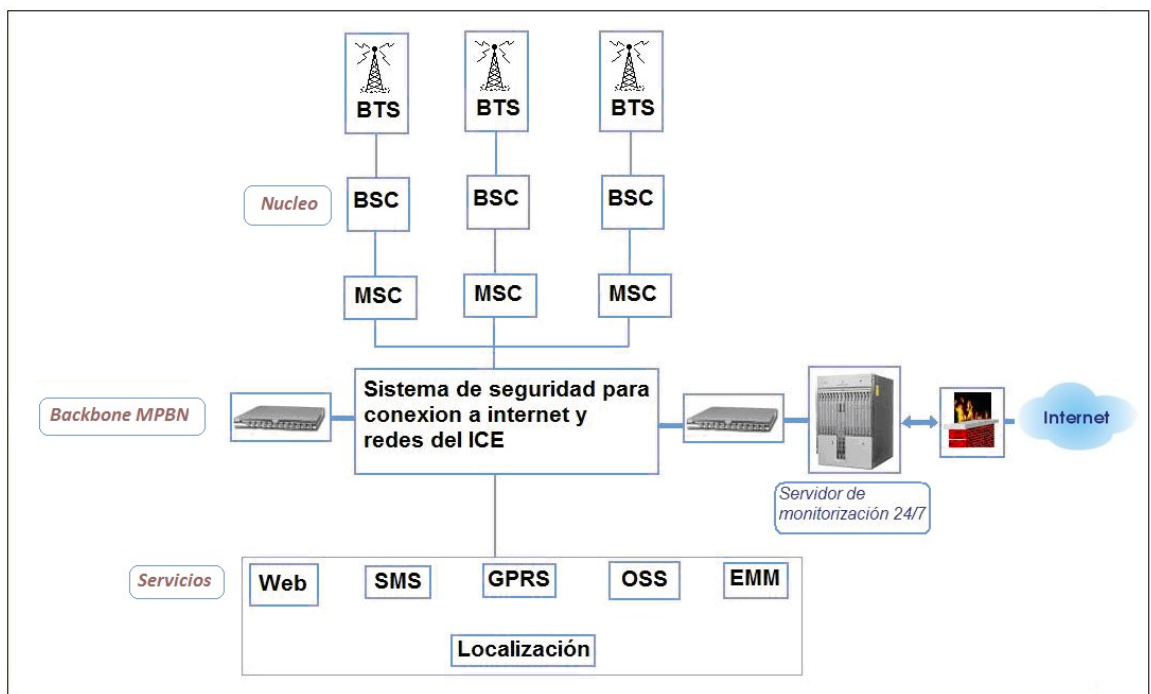


Figura 1.1. Diagrama general de la red GSM del ICE.

Tal supervisión se realiza actualmente por medio de la herramienta MRTG, la cual presenta inconvenientes como: configuración por medio de un archivo de texto,

utilización de un administrador de procesos, carencia de una base de datos y una interfaz gráfica para que el usuario interactúe y pueda agregar, eliminar o modificar parámetros para la supervisión, además no presenta ningún tipo de alarma.

Con base en los problemas anteriormente mencionados, se busca solucionar los inconvenientes que presenta el actual sistema para ello se debe cumplir principalmente el siguiente requerimiento: registrar el historial de fallas en una base de datos en el servidor de monitorización, este registro debe almacenarse semanalmente como mínimo, para la generación de informes e historiales sobre fallas en la red.

Al solucionar el anterior problema el equipo de operación y mantenimiento de ERICSSON, obtendrán los siguientes beneficios:

- Se podrán monitorizar nuevos equipos y agregarse fácilmente a la nueva herramienta.
- Se atenderán rápidamente las fallas en equipos monitorizados.
- Se mantendrá mayor control sobre las fallas de los equipos así como una información accesible en todo momento y actualizada.

1.2 Solución seleccionada.

1.2.1 Requerimientos

Con la finalidad de tener un nuevo sistema de monitorización el cual sea más fácil de usar, confiable y robusto, definiéndose las siguientes características del sistema:

- Monitorización en tiempo real.
- Servidor de base de datos.
- Acceso remoto y seguridad.
- Gráficos de calidad con información detallada, posibilidad de usar identificadores de objeto (OID) SNMP genéricos.
- Utilización del protocolo de administración de red SNMP.
- Alertar en caso de fallo de algún equipo.

Teniendo en cuenta los requerimientos, la solución es una aplicación versátil, que pueda instalarse en un servidor LINUX, fácil de configurar, que permita integrar un módulo de software adyacente el cuál informe confiablemente si ocurre un problema en la red y ser rápido en el tiempo de respuesta para atender la avería. La investigación previa logró aclarar las dudas sobre las ventajas y desventajas que se tenían entre varias opciones de aplicaciones, que existen en este momento según la tabla 4.2, se seleccionó CACTI [14] por las ventajas que posee donde supera los requerimientos propuestos.

Tal y como se muestra en la figura 1.2, se aprecia las disposiciones y requerimientos del sistema, se debe de configurar un servidor *web* (en la figura servidor apache), para acceder desde la red interna e Internet a la herramienta, una base de datos, un software de monitorización que utilice SNMP, como protocolo de gestión, y el desarrollo de *scripts* para comunicarse con los administradores de la red mediante un servidor de correo electrónico.

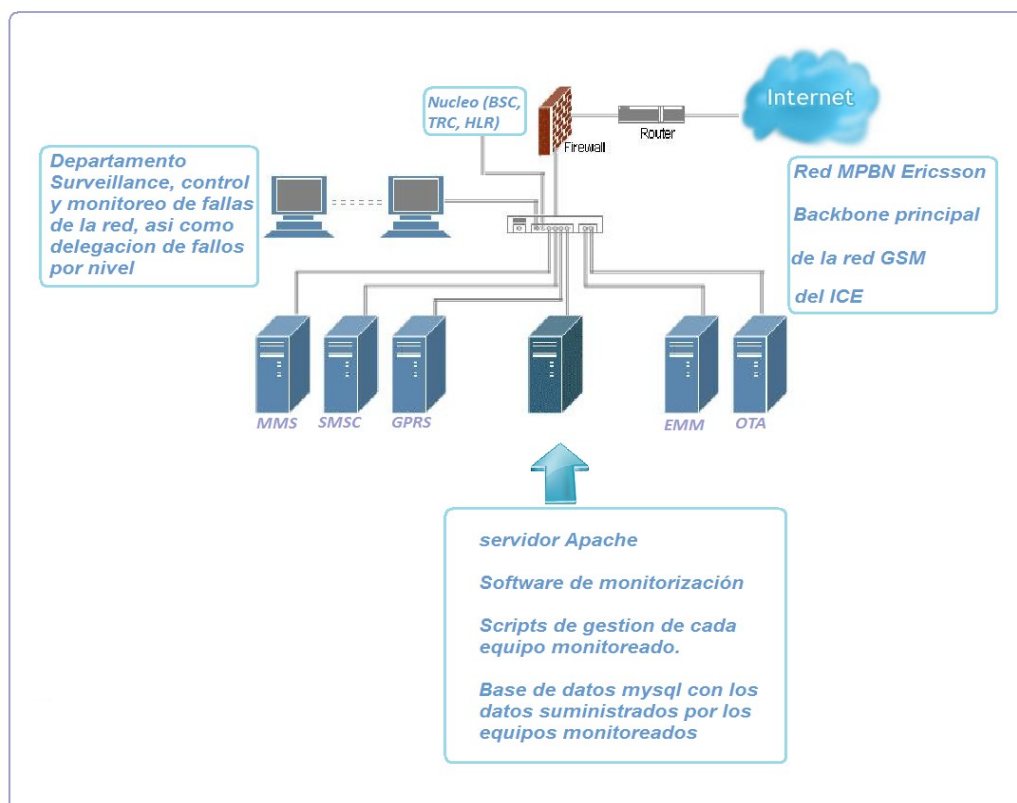


Figura 1.2. Requerimientos y visión general del proyecto.

Capítulo 2: Meta y objetivos.

2.1 Meta.

Diseñar e instalar una herramienta que pueda analizar y detectar de manera eficaz y confiable los distintos problemas asociados con el funcionamiento de la red, como la sobrecarga por tráfico, aumento de la temperatura de los componentes del hardware y problemas en la ejecución del software que en ocasiones se presentan en los nodos en la red de telefonía móvil, para mantener activa y funcionando red GSM.

2.2 Objetivo general

Programar una herramienta de software que pueda usarse en la monitorización y mantenimiento de los equipos que conforman la red de la telefonía móvil para mantenerlos activos y funcionales.

2.3 Objetivos específicos

1. Adaptar la aplicación de monitorización CACTI, para que permita la obtención de tráfico por un servidor GSM y su análisis para la incorporación de nuevos equipos.
2. Adecuar el programa para que muestre el rendimiento del procesador en tiempo real así como la temperatura de la placa base si existe el agente MIB que gestiona esa variable para los cinco equipos nuevos.

3. Adaptar el programa para que mida los máximos, mínimos y promedio, del tráfico obtenido y guardar los datos por semana en una base de datos MySQL.
4. Adecuar el programa para que mida el rendimiento del procesador y número de sesiones obtenido a través del protocolo SNMP.
5. Programar un *script* en PERL que utilice el servidor de correo electrónico con licencia GNU, *sendmail*, para avisar en tiempo real al departamento de SOURVILLANCE, sobre un fallo en la red.
6. Utilizar el algoritmo de planificación Round Robin, en el registro de la información en la base de datos.

Capítulo 3: Marco Teórico

3.1 Descripción del sistema por mejorar.

3.1.1 Descripción del BACKBONE MPBN ERICSSON.

Anteriormente la red celular del ICE, presentaba una arquitectura suministrada por ALCATEL® para la tecnología TDMA (Acceso Múltiple por División de Tiempo, según sus siglas en inglés), pero ésta por percances políticos y además por las desventajas que posee la tecnología analógica respecto a la digital, se migró a la tecnología GSM (Sistema Global de Comunicaciones Móviles, según sus siglas en inglés), en la cual se puede ofrecer una gran variedad de servicios para los usuarios.

El ICE continuó el uso de ALCATEL® como proveedor de servicios de telecomunicaciones, e implementó la primera etapa del proyecto GSM. Para la segunda etapa GSM se cambió de proveedor a ERICSSON®, esta empresa suministró dos proyectos de telecomunicaciones al ICE, en los cuales se implementó el sistema estándar modular AXE que permite las funciones de adicionar, borrar ó modificar sin perturbar la funcionalidad del resto del sistema. En la figura 3.1 se muestra el diagrama de bloques de la arquitectura de la red de datos GSM que utiliza ERICSSON.

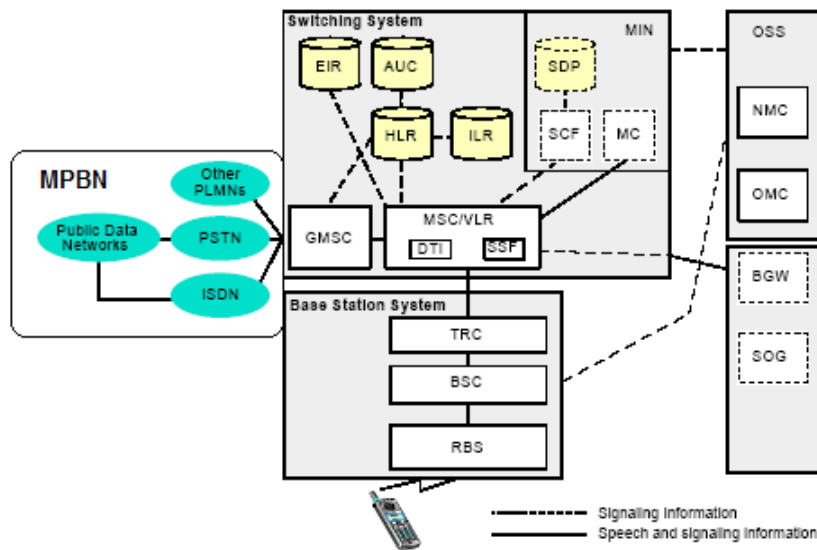


Figura 3.1. Arquitectura del Sistema GSM ERICSSON. (Tomado de Operaciones del BSC GSM[1], pág. 6)

3.1.2 Descripción de los principales nodos.

Base Station System (BSS): Este sistema consta de las siguientes partes principales las cuales son las BSC (Codificadores de Estaciones Base, según sus siglas en ingles), TRC (Controlador de Transcodificador, según sus siglas en ingles), las RBS (Estaciones de Radio Base, según sus siglas en ingles) y las BTS (Transceptores de Estaciones Base, según sus siglas en ingles), que es el equipo de radio que se necesita para dar servicio a una celda. La BTS posee el sistema de antenas, transmisores, receptores, y los equipos de señalización digital.

Las funciones de cada nodo son las siguientes:

- En el caso de la BSC se encarga de controlar y supervisar el número de radio bases (RBS) y conexiones de radio en el sistema.

- El TRC controla y supervisa recursos de los transcodificadores que son usados por las BSC.
- Las BSC's, manipulan la administración de los datos de la celda, el algoritmo de localización, ordena los handovers (cambio de celda de un teléfono en movimiento) y también controla y supervisa recursos de los transcodificadores que son usados por la BSC, todas estas estructuras se pueden apreciar en la figura 3.1.

Switching System (SS): Este sistema consta de las siguientes partes las cuales son MSC (Centro de Conmutación para Servicios Móviles, según sus siglas en inglés) y GMSC (Centro de Conmutación para Servicios Móviles Gateway, según sus siglas en inglés), estos nodos se encargan de establecer, enrutar y supervisar llamadas hacia y desde los usuarios móviles. La GMSC es un MSC usado como una interfaz entre la red móvil y otras redes, tales como la PSTN y ISDN[1].

VLR y HLR, el VLR almacena información temporal de los teléfonos móviles (*Mobile Station*) que actualmente visiten el área de servicio y el HLR es una base de datos para almacenar información de los usuarios que pertenecen a la red del proveedor. La información del usuario consiste de información de localización e información de los servicios asignados como es el caso de Internet o GPS.

EIR (Registro de Identidad de Equipos, según sus siglas en inglés), AUC (Centro de Autenticación). El AUC genera tripletas que se usan en la identificación de la tarjeta módulo de identidad de suscriptor (SIM), se usan en el cifrado de la conversación, datos, y señalización que viaja por la red [1], el EIR es la base de datos responsable de la validación del equipo móvil (teléfonos).

Servidores: Es un computador que forma parte de una red, provee servicios a otros computadores clientes. Existen diferentes tipos de servidores en el caso de estudio, un servidor de telefonía, el cual realiza funciones en el campo de automatización

como lo es el almacenar mensajes de voz, sistemas interactivos y enrutamiento de llamadas.

3.1.3 Sistema de monitorización MRTG.

El sistema de monitorización MRTG (*Multi Router Traffic Grapher*), ver figura 3.2, es un programa multiplataforma, que permite graficar en tiempo real el tráfico por un dispositivo activo de un equipo de una red (enrutadores, cortafuegos, etc). Es un programa hecho por Tobi Oetiker, quien diseñó y confeccionó la mayoría de módulos que presenta la aplicación, el programa es de licencia GNU, por lo que diferentes programadores a nivel mundial han hecho aportes para mejorar la manera en como el programa funciona, entre las mejoras que se han hecho están la de no usar archivos *logs*, para almacenar los eventos SNMP, sino hacer uso de una base de datos *round robin*, como lo es RRDTOOL, que también ha sido desarrollada en licencia GNU, bajo la tutela de Tobi Oetiker. Un ejemplo claro de la aplicación del MRTG y el RRDTOOL es un programa que se llama ROUTERS2, desarrollado en su totalidad con *perl* y HTML.

Traffic Analysis for 6 -- NS500PS01

System: NS500PS01 in
Maintainer:
Description: ethernet3/2
ifType: ethernetCsmacd (6)
ifName:
Max Speed: 12.5 MBytes/s
Ip: 172.27.8.129 ()

The statistics were last updated **Wednesday, 25 November 2009 at 16:35**,
at which time 'NS500PS01' had been up for **10 days, 20:46:14**.

'Daily' Graph (5 Minute Average)

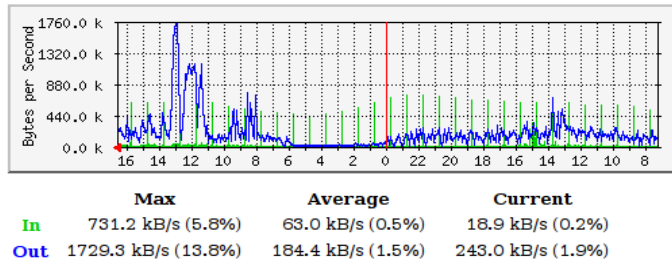


Figura 3.2. Vista de la salida del programa MRTG.

El programa MRTG, es una herramienta eficaz y confiable pero complicada de implementar y agregar equipos para poder observar el tráfico de una red, de manera visual. Este software es el que se tiene monitorizando el MPBN (*Mobile Packet Backbone Network*), como se observa en la figura 3.4, donde cada pestaña contiene los gráficos de tráfico para los nodos del MPBN. Se observa en la figura 3.4, que para acceder a la información del programa MRTG para un firewall de pruebas, se tiene un enlace diferente. Cada enlace mostrado (por ejemplo *SN ethernet 2/1*), muestra un gráfico como el de la figura 3.2.



Figura 3.3. Vista de la página de estadísticas para el Firewall Netscreen 500® con MRTG.

En la figura 3.4 se muestra el diagrama de bloques del programa MRTG que se tiene monitorizando el MPBN de ERICSSON, el programa en PERL necesita ser ejecutado cada cinco minutos, por lo que se usa el planificador de tareas de Windows sino el *crontab* de Linux, el programa lee los MIB genéricos definidos en los memorándums RFC (*Request for Comments*), de la IETF (*Internet Engineering Task Force*), como ejemplos de estos MIB genéricos están los definidos en el memorándum RFC-1213 o el RFC-2790. También el software puede leer MIB's propietarios de las empresas que fabrican los equipos monitorizados, esto se debe programar en el archivo de configuración que se explica en la página oficial [2].

Tal como se muestra en el diagrama de bloques general (figura 3.4), cada interfaz o cada OID (*Object Identifier*) que se quiera monitorizar crea una página *web* que lo describe, se debe de generar un archivo *index.html*, que sería una página principal que se pueda enlazar con los demás, como se representa en la figura 3.3.

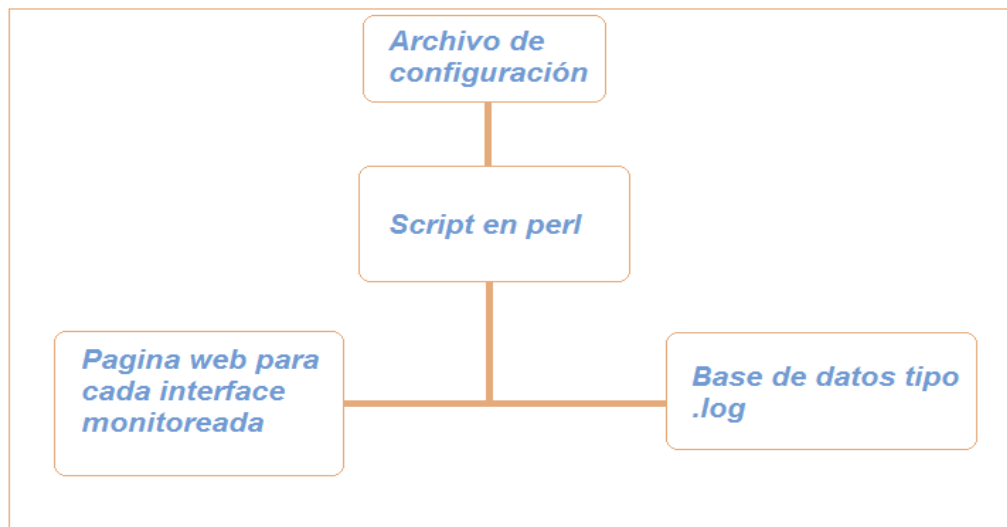


Figura 3.4. Diagrama de bloques del software MRTG.

3.2 Antecedentes bibliográficos

3.2.1 Gestión de red.

Para la gestión de red se hace uso de los protocolos SMI, MIB-II y SNMP, estos protocolos se definen en los RCF 1115, RCF 1213 y RCF 1157 respectivamente.

3.2.2 Protocolo SMI (Structure of management information).

Es un protocolo que almacena en una base de datos las variables que utiliza el SNMP, en el RCF 1115 se explica el uso adecuado de los MIB y los OID (*Object Identifier*). Además define las pautas a seguir para escribir nuevos OID's, ya sean *private*, *experimental*, *mgmt* ó *directory*, se explica la sintaxis utilizada y la descripción de las estructuras de cada objeto MIB, además de la salida del OID correspondiente, o sea el tipo de objeto que llevará información sobre un equipo. Existen los siguientes tipos de OID's para aplicaciones: *NetworkAddress*, *IpAddress*,

Counter, Gauge, TimeTicks y Opaque, también están los universales que son: *Integer, Octet, Null, Object Identifier y Sequence*.

Un ejemplo de esto sería como el caso de obtener datos de la memoria o de los dispositivos de almacenamiento, podría utilizarse *gauge* en el caso que exista un sensor que mida la memoria, pero por lo general es un *counter* el cual contiene el valor de la memoria disponible y otro *counter* que mantiene la memoria ocupada. En la figura 3.6 se muestra un OID privado de NetScreen ®.

La descripción de los identificadores de objeto (OID) se hace utilizando un subconjunto de ASN.1 (*Abstract Syntax Notation 1*, estándar-ISO 8824), un lenguaje de descripción de datos. La definición del tipo de objeto consta de cinco campos [3]:

- *Objet*: nombre textual, llamado descriptor del objeto, para el tipo del objeto, junto con su correspondiente identificador de objeto.
- *Sintaxis*: la sintaxis abstracta para el tipo el objeto. Las opciones son *SimpleSyntax (Integer, Octet, Null, Object Identifier y Sequence)*, *ApplicationSyntax (NetworkAddress, IpAddress, Counter, Gauge, TimeTicks y Opaque)* u otro tipo de sintaxis de aplicación (ver el RFC 1155).
- *Definition*: descripción textual del significado del tipo.
- *Acces*: sólo lectura, sólo escritura, lectura - escritura o no accesible.
- *Status*: obligatorio, opcional u obsoleto.

En la figura 3.5 se observa claramente la sintaxis descrita en el RCF 1115, y la estructura de un OID privado realizado por los ingenieros del NetScreen.

```

NETSCREEN-RESOURCE-MIB DEFINITIONS ::= BEGIN
IMPORTS
-----
    DisplayString          FROM RFC1213-MIB
    netscreenResource     FROM NETSCREEN-SMI;

nsResCPU OBJECT IDENTIFIER ::= { netscreenResource 1 }
nsResMem  OBJECT IDENTIFIER ::= { netscreenResource 2 }
nsResSession OBJECT IDENTIFIER ::= { netscreenResource 3 }

    nsResCpuAvg OBJECT-TYPE
        SYNTAX      INTEGER
        ACCESS      read-only
        STATUS      mandatory
        DESCRIPTION
            "Average System CPU utilization in percentage."
        ::= { nsResCPU 1 }

```

Figura 3.5. OID privado del agente MIB de NetScreen®.

3.2.3 Protocolo SNMP (Simple Network Management Protocol)

SNMP es un protocolo de la capa de aplicación que sirve para consultar los diferentes equipos que conforman una red, (enrutadores, switches, módems, servidores, etc). El SNMP usa UDP (Protocolo de datagrama de usuario, por sus siglas en ingles), protocolo no orientado a la conexión, y usa el puerto 161 y 162.

Los dispositivos que conforman la red presentan procesos que se llaman agentes SNMP, estos agentes son programas que residen en los equipos y tienen conocimiento de las variables locales y convierten esa información en formato ANS.1 para enviarlo por la red cada cierto tiempo. Los dispositivos de la red recompilan y guardan información de administración y la ponen a disposición de los NMS (*Network Management System*), por medio de los agentes SNMP. Existen varias versiones de este protocolo las cuales por razones de seguridad han ido apareciendo a medida que las redes informáticas e Internet crecen, las versiones actuales son SNMPv1, SNMPv2 y SNMPv3, las diferencias principales de estas versiones están en que el SNMPv2 utiliza más funciones y una validación de seguridad como lo es la

comunidad a la que pertenece el equipo gestionado, la versión SNMPv3 utiliza encriptación y validación por contraseña.

El SNMP permite crear herramientas de gestión de red que [4]:

- Informen del funcionamiento de la red.
- Detecten averías y funcionamiento incorrecto de la red.
- Permitan actuar sobre los nodos de la red (modificando configuración o apagando el equipo).
- El SNMP define dos acciones, lectura de un valor del registro del agente MIB o modificación de alguno de estos valores.

3.2.4 Protocolo MIB-II (Management Information Base)

Es una base de datos que mantiene información en forma jerárquica de todos los equipos gestionados en una red informática, en el MIB se definen las variables usadas por el protocolo SNMP. Existen muchas modificaciones durante el desarrollo del Internet, para definir la gestión de red en el modelo OSI, por lo que se han creado diferentes versiones como la MIB y la MIB-II, definidas en los RCF 1156 y RCF 1213 respectivamente, el MIB está clasificado ahora como protocolo histórico con *status* no recomendado. Conforme las redes crecen se agregan nuevos OID's que puede ser monitorizados, esto hace que se realicen una gran cantidad de RCF's, con las modificaciones hechas.

Los siguientes son ejemplos de objetos de cada grupo. La lista completa está definida en el RFC 1213 [3].

- Grupo de sistema
 - sysDescr - Descripción completa del sistema (versión, Hardware, SO)
 - sysObjectID - Identificación que da el distribuidor al objeto

- sysUpTime - Tiempo activo
- sysContact - Nombre de la persona que hace de contacto
- sysServices - Servicios que ofrece el dispositivo.
- Grupo de interfaces
 - ifIndex - Número de interfaz
 - ifDescr - Descripción de la interfaz
 - ifType - Tipo de la interfaz
 - ifAdminisStatus - Estado de la interfaz.
- Grupo de traducción de direcciones
 - atTable - Tabla de traducción de direcciones
 - atPhysAddress - La dirección física dependiente del medio
 - atNetAddress - La dirección de red correspondiente a la dirección física.
- Grupo IP
 - ipForwarding - Indicación de si la entidad es una IP de gateway
 - ipRouteMask - Máscara de subred para el enrutamiento.
- Grupo ICMP
 - icmpInMsgs - Número de mensajes ICMP recibidos
 - icmpInDestUnreachs - Número de mensajes ICMP destino inalcanzable (*destination unreachable*) recibidos.
 - icmpInTimeExcds - Número de mensajes ICMP *time exceeded* (tiempo excedido) recibidos.
- Grupo TCP
 - tcpRtoAlgorithm - Algoritmo que determina el *timeout* para retransmitir octetos para los que no se ha recibido reconocimiento

- tcpMaxConn - Límite en el número de conexiones TCP que puede soportar la entidad
- tcpInSegs - Número de segmentos recibidos, incluyendo aquellos con error
- tcpConnRemAddress - La dirección IP remota para esta conexión TCP
- tcpInErrs - Número de segmentos desechados debido a errores de formato.
- Grupo UDP
 - udpInDatagrams - Número de datagramas UDP entregados a usuarios UDP
 - udpNoPorts - Número de datagramas UDP recibidos para los que no existía aplicación en el puerto de destino
 - udpOutDatagrams - Número de datagramas UDP enviados por la entidad.
- Grupo EGP
 - egpInMsgs - Número de mensajes EGP recibidos sin error
 - egpInErrors - Número de mensajes EGP con error
 - egpOutMsgs - Número de mensajes EGP generados localmente.

3.2.5 Red informática.

Una red informática es un número de equipos conectados por algún medio que transporte de datos (cables, ondas de radio, luz, etc), que comparten información, recursos y servicios [5].

Para lograr un enlace entre dos computadores se necesita un medio y una tarjeta de red (NIC) que es la encargada de transmitir y codificar según un protocolo (conjunto de normas y estándares), la información.

En la figura 3.6 se muestra el diagrama para el caso de una NIC, enviando una trama.

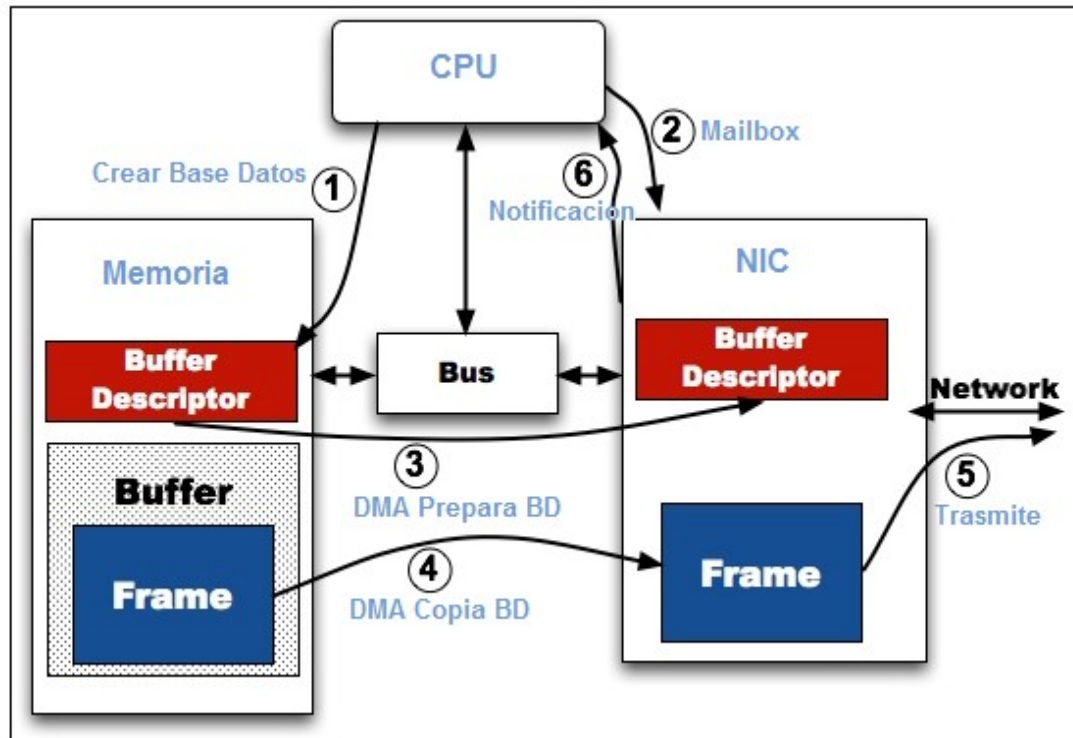


Figura 3.6. Diagrama general de una NIC enviando una trama Ethernet [6].

Generalmente, el SO (sistema operativo) del dispositivo donde se encuentra la NIC, tiene una serie de *buffers*, los cuales mantienen los encabezados y contenidos de las tramas. El SO, tiene también una pila con la descripción de los buffers, que le informa al SO acerca de información relevante como en que parte de la memoria se encuentra el *buffer* y el tamaño. Esa pila de descripción, es la unidad de comunicación entre el SO y la NIC. Los siguientes son los pasos que tiene que hacer el computador para poder enviar una trama de Ethernet [6]:

- El SO, es informado que la trama en memoria esta lista para ser enviada y se crea un descriptor de buffer en la pila.
- El SO, notifica a la NIC que hay un descriptor de buffer, que está listo para ser procesado (*mailbox*).
- La NIC, inicia la lectura con el acceso directo a memoria (DMA), lee el descriptor de buffer para saber cuánto y hasta donde leer, una vez que todos los segmentos de la trama son cargados en el buffer de la NIC, la NIC trasmite la trama a la red.
- Cuando la NIC ha transmitido toda la trama, ella manda una interrupción al procesador para avisarle que ya se terminó.

3.2.6 Lenguaje de programación de alto nivel

Un lenguaje de programación es un idioma artificial, diseñado para expresar mandatos o ejecuciones que pueden ser llevadas a cabo por las máquinas como es el caso de los computadores. Pueden ser utilizados para crear programas o algoritmos, que controlen el comportamiento físico y lógico de una máquina. Está formado por un conjunto de símbolos y reglas sintácticas, semánticas que definen su estructura y el significado de sus elementos y expresiones. Un lenguaje de alto nivel, se caracteriza por expresar los algoritmos y su semántica lo más cercano al entendimiento humano, en lugar de la capacidad ejecutora de las máquinas. Entre los más populares lenguajes de programación de alto nivel están [7]:

- C++
- Java
- Python
- PERL

- MATLAB
- PHP
- Fortran

3.2.7 Script

Es un conjunto de instrucciones, que permiten la automatización de tareas, creando pequeñas aplicaciones. Es muy utilizado en la administración de sistemas UNIX. Son ejecutados por un intérprete de línea de comandos como lo es el BASH de Linux [8].

3.2.8 Algoritmo de planificación Round Robin

Es un método para seleccionar todos los elementos en un grupo de manera equitativa y en un orden, normalmente se empieza por el primer elemento de la lista hasta llegar al último y comenzando de nuevo desde el primer elemento (método circular).

Es un método para ejecutar diferentes procesos de manera concurrente, para la utilización equitativa de los recursos del equipo, es limitando cada proceso a un pequeño período (*quantum*), y luego suspendiendo éste proceso para dar oportunidad a otro proceso y así sucesivamente [16], como vemos en la figura 3.7, el algoritmo define un periodo de tiempo al cual asigna recursos, una vez transcurrido este tiempo los procesos o los elementos de una lista rotan para continuar con el siguiente así ningún elemento o proceso se queda sin asignación de recursos.

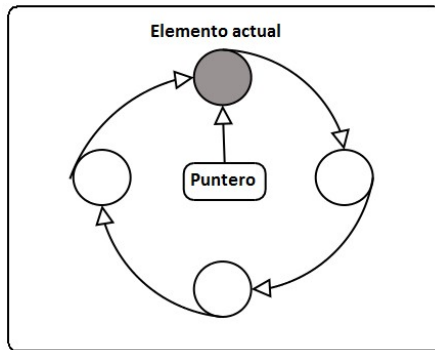


Figura 3.7. Algoritmo de planificación Round Robin.

3.3 Descripción de los principales principios físicos y/o electrónicos relacionados con la solución del problema

3.3.1 Medidas de información.

Una alta probabilidad de ocurrencia de un hecho representa una poca información, y una baja probabilidad de ocurrencia de un hecho representa mucha información[9].

$$I = \log_p \left(\frac{1}{p} \right) [bits] \quad (3.1)$$

I: Información

P: Probabilidad de ocurrencia.

3.2.2 Tráfico en una red informática.

El tráfico en una red de computadores, es la capacidad de transferencia de información en un enlace de red (dos computadores conectados entre sí) y se expresa en unidades de datos por unidades de tiempo ejemplo: bits por segundo (bps), kilo bits por segundo (kbps), mega bits por segundo (mbps), kilo bytes por segundo (KBps), etc[10].

Para medir el tráfico en un enlace se puede hacer de dos maneras, la primera: Transferir un archivo grande de tamaño conocido por el enlace y medir el tiempo que tarda en llegar al computador destino. Este método es eficiente para hacer pruebas de eficiencia en redes, uno de los problemas más grandes que presenta es que satura el enlace, por lo que la red tiene que estar única y exclusivamente activa para transferir el archivo, porque sino la medición seria inexacta al haber más tráfico del que queremos medir en ese enlace.

La otra manera y mucho más sencilla es obtener la información del tráfico que pasa por los elementos activos del enlace. La NIC (tarjeta de red), de cada computador del enlace, tiene conocimiento de cuanta cantidad de información envía y cuanta información recibe del enlace habilitado, un agente SNMP pone estos contadores a disposición de los administradores de red mediante el protocolo SNMP.

Capítulo 4: Procedimiento metodológico

Inicialmente se investigó sobre el protocolo SNMP, los diferentes MIB's y como se puede acceder a la información de ellos. Para este fin se leyó minuciosamente información relacionada con aplicaciones de monitorización, además del funcionamiento de la estructura de la red del ICE, se instaló el software MRTG en una computadora personal y haciendose las pruebas con este programa.

Para ello se investigó sobre los identificadores de objeto así como el desarrollo de scripts, para administración de equipos y ordenadores. Esto se realizó con ayuda de la investigación exhaustiva y con el conocimiento sobre la estructura y sintaxis del lenguaje para la realización de scripts PERL, existe gran cantidad de información y páginas dedicadas a la enseñanza de este lenguaje de programación.

Se trató de mejorar el software MRTG, pero no fueron aceptadas las mejoras hechas, para tal caso se investigó sobre bases de datos existentes, seguridad y confiabilidad, se requería un software más amigable con el usuario que pudiera manejarse cómodamente sin necesidad de editar archivos de texto, además la información mostrada debía ser confiable. Así se investigó sobre versiones comerciales como el software PRTG y versiones de licencia libre como lo es Routers2, CACTI y NAGIOS. Se investigó también sobre los requerimientos mínimos que debía tener el sistema para poder hacer uso de la aplicación, además del sistema operativo en el que se puede instalar.

Por requerimientos técnicos además de las ventajas que ofrece el software libre y las distintas versiones de LINUX, se debía instalar la aplicación seleccionada en un servidor LINUX UBUNTU, el cual presentaba el problema de ser una versión muy antigua, por lo que el soporte para software estaba desactualizado, se tuvo que utilizar un repositorio de la distribución DEBIAN de LINUX, para poder instalar los programas que eran necesarios.

4.1 Reconocimiento y definición del problema

Para el reconocimiento del problema se entrevistó al jefe del departamento de *Surveillance*, que es el departamento de operación y mantenimiento que se encarga de reportar y encontrar las averías y fallos en la red, además de gestionar y filtrar las quejas del servicio de telefonía móvil.

En este departamento se encuentra una persona las veinticuatro horas del día, los trescientos sesenta y cinco días del año, debido a que el servicio no puede estar fuera de línea, existen ingenieros de guardia que son llamados cuando ocurre una falla grave y tienen que tener disponibilidad inmediata.

En la figura 1.2 se detalla un diagrama general de la red MPBN y la importancia del departamento *Surveillance*.

En la tabla 4.1 se especifica las principales características que debe tener el sistema, en los cuales los puntos más importantes son: las alarmas que se implementen para tratar el problema lo más rápido posible y el acceso a las bases de datos para la presentación de estadísticas al proveedor de servicios.

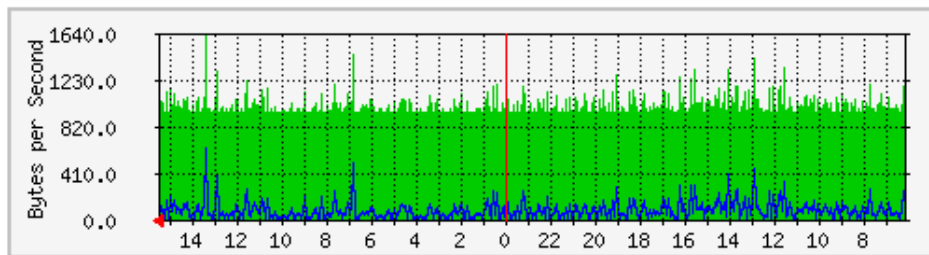
Tabla 4.1. Especificaciones y requerimientos del sistema a implementar en el MPBN
ERICSSON.

	Aplicación Actual	Aplicación Requerida	Aplicación Deseable
Zoom	X	✓	✓
Base de datos mysql	X	✓	✓
Exportación de los datos	X	✓	✓
Alarmas	X	✓	✓
Medición de tráfico y OID's genéricos	✓	✓	✓
Control de acceso	X	✓	✓
Amigable con el usuario	X	✓	✓
Tiempo real	✓	✓	✓
Presentación de gráficas diario, mensual y anual.	✓	✓	✓
Comunicación de dos vías con el equipo gestionado	X	X	✓
Generación de traps de pruebas	X	X	✓

4.2 Obtención y análisis de información

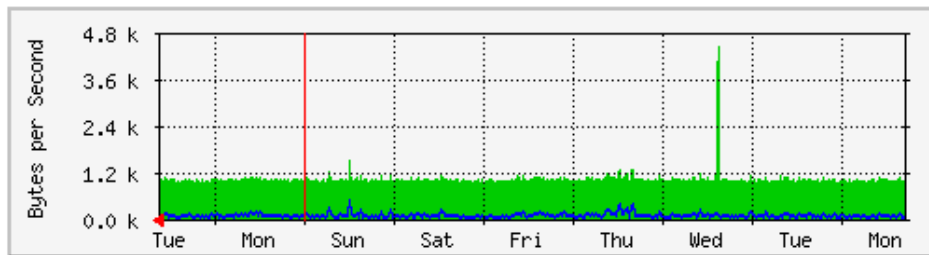
La figura 4.1 es una salida del software actual de monitorización SNMP. Analizando la figura 4.1, existe información relevante pero no puede ser extraída de la gráfica en forma automática, solo manual anotando en un archivo el valor mostrado, además el programa es no es amigable con el usuario.

'Daily' Graph (5 Minute Average)



	Max	Average	Current
In	1627.0 B/s (0.0%)	1014.0 B/s (0.0%)	946.0 B/s (0.0%)
Out	607.0 B/s (0.0%)	72.0 B/s (0.0%)	9.0 B/s (0.0%)

'Weekly' Graph (30 Minute Average)



	Max	Average	Current
In	4450.0 B/s (0.0%)	1034.0 B/s (0.0%)	1051.0 B/s (0.0%)
Out	453.0 B/s (0.0%)	74.0 B/s (0.0%)	104.0 B/s (0.0%)

Figura 4.1. Salida del software de monitorización actual.

Investigando sobre programas de monitorización en internet se encontró gran cantidad y de diferentes precios, también se investigó sobre el sistema operativo en que pueden ser instaladas las aplicaciones, el servidor de monitorización es un servidor Ubuntu 6.10, la tabla 4.2 resume los más populares que se encontraron:

Tabla 4.2. Características de los Software para monitorización de redes.

Programa	Plataforma	Licencia	Ventajas
Routers2.cgi	multiplataforma	GNU	Provee una base de datos RRDtool, para el MRTG, además usa autenticación de usuario. [17]
PRTG	Microsoft Windows	Comercial/freeware	Provee una base de datos para generar reportes, puede monitorizar el uso del CPU y temperatura, versión freeware solo acepta 10 dispositivos. [18]
Bigbrother	Multiplataforma	Comercial	Alertas, mapa geográfico, amigable con el usuario, interfaz flash, descubrimiento de dispositivos automático y 3rd party software para encontrar scripts[19].

CACTI	Multiplataforma	GNU	Monitorización de servicios de red y de recursos, base de datos para hacer reportes, autenticación de usuario, amigable con el usuario y 3rd party software. [14]
Nagios	Plataformas UNIX	GNU	Monitorización de servicios de red (SNMP, POP3, Http, etc), Monitorización de recursos (CPU, procesos, usuarios, etc), Monitorización en tiempo real, presenta archivos logs rotacionales. [20]

4.3 Evaluación de las alternativas y síntesis de una solución

Para la monitorización de una red de computadores existe una variedad de aplicaciones que hacen uso de los protocolos ya antes mencionados, aunque algunos presentan mayores beneficios que otros, se marcó las características que tenía que tener el nuevo sistema a implementar en el MPBN. Desarrollar una aplicación de este tipo desde cero es un trabajo de varios años, y el proceso de depurado sería otro par de años es por eso que se decidió usar un software bastante popular y agregarle más funciones.

Para el monitorización en tiempo real se comenzó utilizando el software MRTG que es el que actualmente se tiene para la monitorización, se instaló en una computadora personal y se hicieron pruebas para monitorizar OID`s privados de un *firewall Netscreen*, en la figura 4.2 se muestra el archivo de configuración que carga el MRTG para monitorizar una red.

```
# Created by
# cfigmaker iceapbnericsson@172.27.8.140 --global "workdir: c:\www\artg" --output artg.cfg

### Global Config Options
# for UNIX
# WorkDir: /home/http/artg

# or for NT
# WorkDir: c:\artgdata

### Global Defaults
# to get bits instead of bytes and graphs growing to the right
# Options[_]: growright, bits

EnableIPv6: no
WorkDir: c:\www\artg
RunAsDaemon: yes
LoadMIBs: C:\www\artg\NS-RES.mib
LoadMIBs: C:\www\artg\NS-SMI.mib
Language: spanish

#####
# System: NSS00PS01
# Description: NetScreen-500 version 5.1.0r3.4 (SN: 0100052004000174, GPRS Firewall+VPN)
# Contact:
# Location:
#####

### Interface 1 >> Descr: 'ethernet1/1' | Name: '' | Ip: '172.27.8.226' | Eth: '00-10-db-ff-20-a0' ###
Target[172.27.8.140_1]: 1:iceapbnericsson@172.27.8.140:
SetEnv[172.27.8.140_1]: MRTG_INT_IP="172.27.8.226" MRTG_INT_DESCR="ethernet1/1"
MaxBytes[172.27.8.140_1]: 12500000
Title[172.27.8.140_1]: Traffic Analysis for 1 -- NSS00PS01
PageTop[172.27.8.140_1]: <h1>Análisis de Trafico Eth 1 (NSS00PS01)</h1>
                        <div id="sysdetails">
                          <table>
```

Figura 4.2. Archivo de configuración para el MRTG ejecutándose en Windows®.

Se trabajó con este software y se hizo una base de datos en MYSQL, la cual era cargada por un script hecho en PERL, y se implementó la medición de OID`s propietarios de NetScreen®. Para el firewall NetScreen-500, se realizó un script para interactuar con el archivo LOG y la base de datos.

Este script fue de las primeras pruebas realizadas y por ser un poco complicado modificar el código fuente del MRTG, además de presentar un solo gráfico por página, el asesor de ERICSSON lo revocó.

En la figura 4.3 se muestra el diagrama de flujo del script utilizado para interactuar con la base de datos.

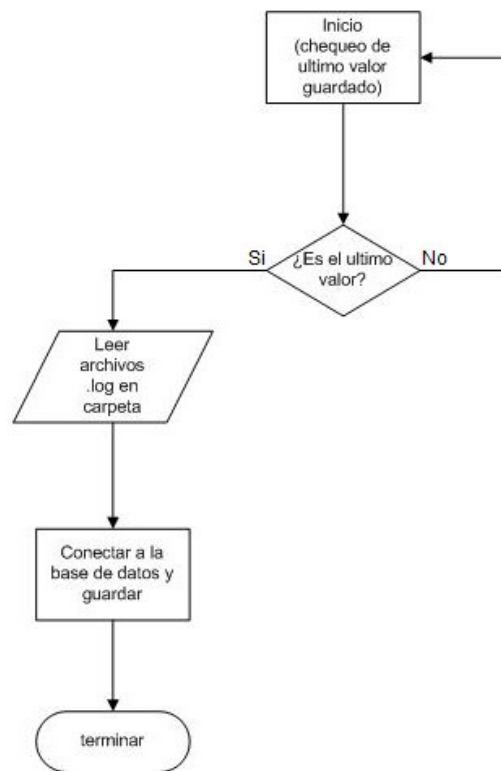


Figura 4.3. Diagrama de flujo del script realizado para almacenar los datos del MRTG en una base de datos MYSQL.

Se decidió entonces usar RRDTOOL una herramienta para hacer gráficos con bases de datos que utiliza el algoritmo de planificación Round Robin. Pero con la investigación se encontró las facilidades que poseía el software CACTI que también

hacia uso de RRDTOOL, por lo que se procedió a instalarse en una maquina con SO Linux Ubuntu 9.04, para pruebas.

El proceso de elección del software se basó principalmente en el tamaño de la red, la falta de presupuesto para la compra de un software, los requerimientos de los administradores de la red y el sistema operativo donde ejecutaría el software.

Las aplicaciones descritas en la tabla 4.2, eran las más completas y populares, las opciones más fuertes eran CACTI y NAGIOS, software que presenta muchas ventajas económicas y de depuración con respecto a los demás.

Investigando en Internet se observó las opiniones de usuarios, así como los *screenshots* de las diferentes aplicaciones, se hicieron pruebas en una computadora personal para poder observar el desempeño y programar los scripts que son propios del presente proyecto.

4.4 Implementación de la solución

Para la implementación de solución se ha propuesto un programa fácil de configurar y usar, como CACTI, el cuál investigó previamente en la tabla 4.1.

En la figura 4.4 se muestra la pantalla de inicio de la aplicación cuando un usuario ha ingresado correctamente.

El programa es escrito en PHP, y es versátil al configurar ya que hace uso de configuración por plantillas, lo cual significa que si algún equipo tiene características similares con otro, se puede hacer una plantilla que coincida en las características que tienen en común, eso también se aplica al tipo de gráfico y a la forma de obtener los datos, la adquisición de datos puede ser vía SNMP ó script.

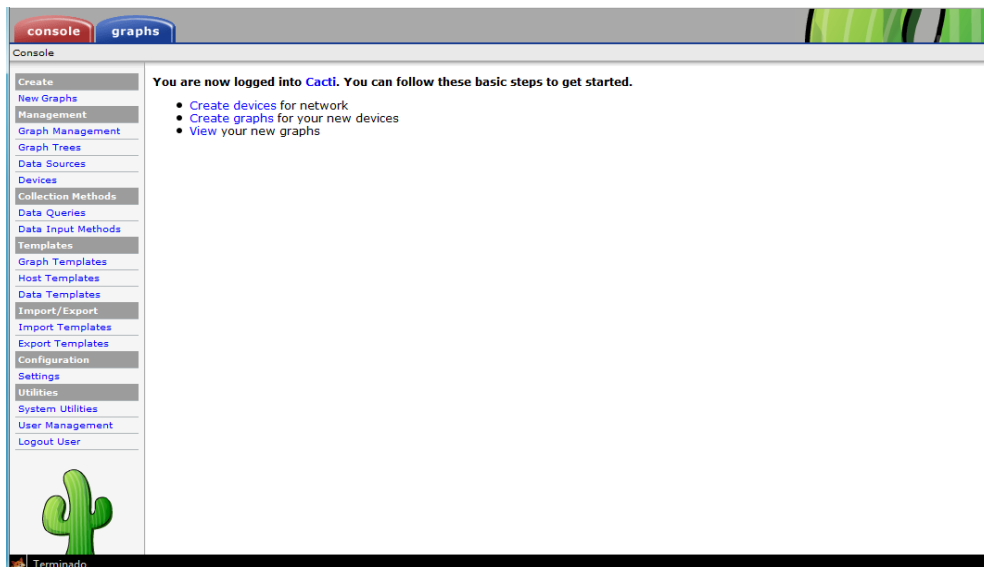


Figura 4.4. Vista principal del software CACTI.

El programa hace uso del paquete NET-SNMP (ver figura 4.5), de Linux para capturar los datos y depuración.

```

matica@mipecera:~$ snmpwalk -v2c -c public 172.27.19.133 | more
SNMPv2-MIB::sysDescr.0 = STRING: SMQE32.CMG.COM HP rx3600 (1.67GHz/9.0MB) IA64 OpenVMS V8.3-1H1 HP TCP/IP Services for OpenVMS
SNMPv2-MIB::sysObjectID.0 = OID: SNMPv2-SMI::enterprises.36.2.15.22.1
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (2219394462) 256 days, 20:59:04.62
SNMPv2-MIB::sysContact.0 = STRING: CMG-WDS
SNMPv2-MIB::sysName.0 = STRING: SMQE32.CMG.COM
SNMPv2-MIB::sysLocation.0 = STRING: SMQE31
SNMPv2-MIB::sysServices.0 = INTEGER: 72
SNMPv2-MIB::sysORLastChange.0 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORID.1 = OID: SNMPv2-SMI::enterprises.36.2.15.22.1.1
SNMPv2-MIB::sysORID.2 = OID: SNMPv2-SMI::enterprises.36.2.15.22.1.1.2
SNMPv2-MIB::sysORDescr.1 = STRING: Base o/s agent (OS_MIBS) capabilities
SNMPv2-MIB::sysORDescr.2 = STRING: Base o/s agent (HR_MIB) capabilities
SNMPv2-MIB::sysORUpTime.1 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.2 = Timeticks: (0) 0:00:00.00
IF-MIB::ifNumber.0 = INTEGER: 7
IF-MIB::ifIndex.1 = INTEGER: 1
IF-MIB::ifIndex.4 = INTEGER: 4
IF-MIB::ifIndex.5 = INTEGER: 5
IF-MIB::ifIndex.6 = INTEGER: 6
IF-MIB::ifIndex.7 = INTEGER: 7
IF-MIB::ifIndex.8 = INTEGER: 8
IF-MIB::ifIndex.9 = INTEGER: 9

```

Figura 4.5. Salida del programa NET-SNMP.

Además se hizo un script para reportar las averías y falta de tráfico, en la figura 4.6 se ve el diagrama de flujo, en donde se explica los parámetros y la dependencia del tiempo que presenta el programa.

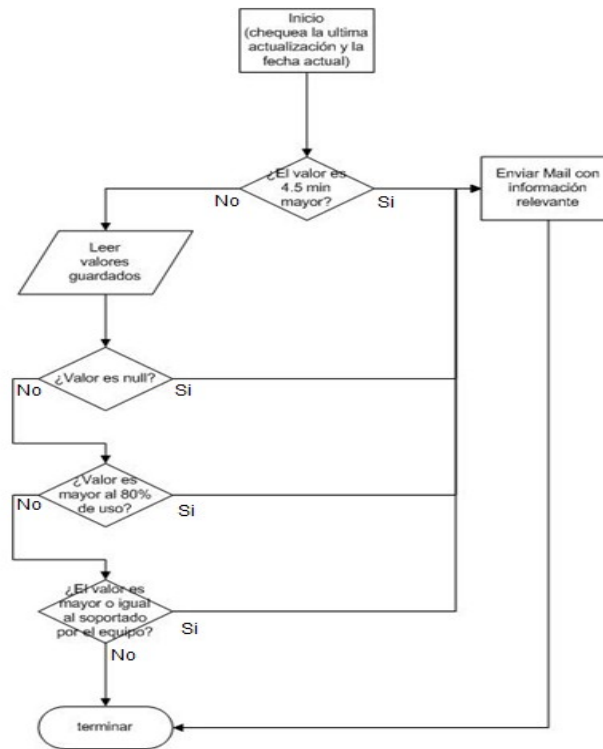


Figura 4.6. Diagrama de flujo del script de alerta.

4.5 Reevaluación y rediseño

Existen modificaciones importantes que se podrían hacer para hacer más eficiente la monitorización una de ellas sería visualizar la fecha de adquisición de un dato, como un cuadro de dialogo que muestre el valor fuera del gráfico, también en el script implementado usar SNMPSET para poder modificar algún valor o aplicar un reset por software vía SNMP. Además se puede implementar un modulo de comunicación externa para que se encienda una luz indicadora cuando ocurre algo fuera de lo normal con los equipos monitorizados.

Capítulo 5: Descripción detallada de la solución (Explicación del diseño)

5.1 Análisis de soluciones y selección final

Al implementar, la primera opción de solución era la de modificar y agregar los equipos nuevos, los scripts de alarmas y la base de datos al software que se estaba utilizando el MRTG. Para agregar los nuevos equipos se tenía que modificar el archivo de configuración e investigar sobre los OID's propietarios del equipo para que fuese monitorizado, una vez que se investigó sobre los OID's privados, para los cuales las empresas hacen sus diferentes agentes y MIB's, se procedió a hacer pruebas con el MRTG, para eso se uso una computadora personal.

Los OID's se graficaron después de investigar sobre las funcionalidades y la salida que tenían, en la figura 5.1 se observa la salida de esas pruebas:

Carga del CPU

The statistics were last updated **Monday, 14 December 2009 at 15:50**, at which time 'NS500PS01' had been up for **477 days, 5:26:43**.

'Daily' Graph (5 Minute Average)

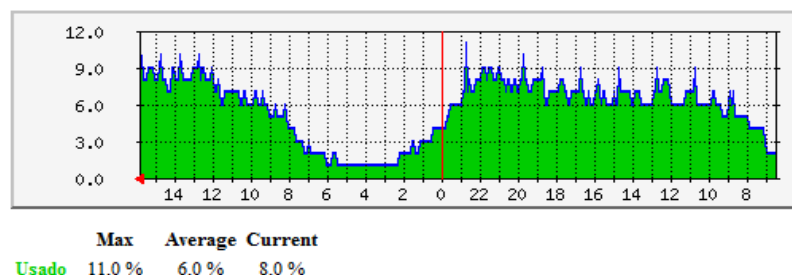


Figura 5.1. Salida del OID propietario de NetScreen gráficoado en MRTG.

Realizado esto, se implementó el script descrito en el diagrama de flujo de la figura 4.3, en el apéndice A.3 se encuentran los scripts realizados.

Esta solución fue revocada por lo poco amigable que es el MRTG con el usuario y las pocas ventajas que posee con respecto a aplicaciones más modernas. Pero una de las ventajas que posee este software es su robustez, ya que al ser lo más básico posible, las fallas que tiene son mínimas, está estructurado (ver figura 3.4) de manera que un módulo lee los datos del buffer lo copia en un archivo `.log` y lo imprime en una imagen `.png` que es cargada en una página `web .html`, la cual se muestra en la figura 5.1.

Para la solución final se procedió a instalar el software CACTI y a investigar sobre la configuración. Dicha configuración resultó ser más sencilla que la del MRTG. Esta configuración se puede revisar la página oficial de CACTI [14]. En la siguiente sección se explicara brevemente la mayoría de funciones.

En la figura 5.2 se muestra el diagrama de bloques de la aplicación.

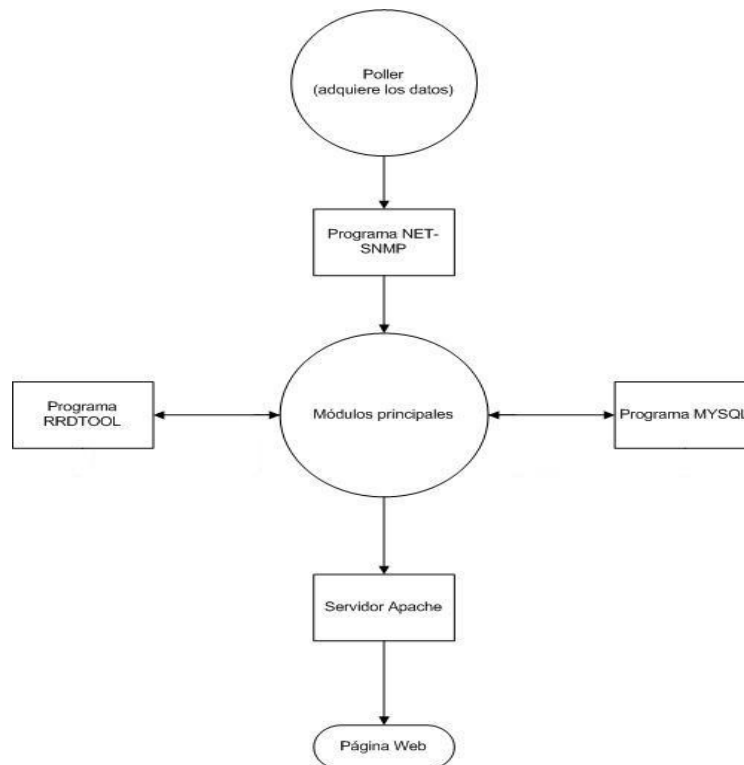


Figura 5.2. Diagrama de bloques del programa CACTI.

Como se observa en el diagrama de bloques el programa hace uso de tres programas de licencia libre GNU, como lo son MYSQL, RRDTOOL y NET-SNMP.

MYSQL es un programa para hacer bases de datos de diferentes tamaños y con múltiples usuarios, es desarrollado por SUN Microsystems®. En el caso de CACTI se utiliza MYSQL para guardar los usuarios y sus permisos (súper usuario ó usuario). Guarda también la configuración de cada HOST (equipo gestionado), los datos por el tiempo definido por el usuario (una semana, un día, un mes, un año) y las diferentes plantillas y *plugins* que el usuario haya instalado o creado.

El programa NET-SNMP es un paquete de software multiplataforma desarrollado para utilizar el protocolo SNMP de manera eficaz. Está integrado por los siguientes módulos[11]:

- Aplicaciones de línea de comandos como SNMPWALK ó SNMPGET.
- Navegador de MIB
- Una aplicación “daemon”, que recibe las notificaciones SNMP.
- Un agente SNMP
- Una librería para desarrollar nuevas aplicaciones SNMP.

RRDTOOL, es una aplicación multiplataforma desarrollada por [Tobias Oetiker](#), para crear bases de datos de alto rendimiento (*Round-Robin*), y realizar la representación gráfica de los datos almacenados. Este programa lo utiliza CACTI para hacer las gráficas y guardar los datos temporalmente, después de obtenerlos con SNMP[12].

Apache es un servidor HTTP multiplataforma y de licencia libre, el cual fue configurado adecuadamente para poder incluir MYSQL, PHP y CACTI.

Entre las ventajas más importantes que tiene el proyecto es el control de usuarios, las bases de datos, la monitorización con alarmas, la monitorización de nuevos OIDS y el soporte que presenta la aplicación instalada.

5.2 Descripción del software

Para el desarrollo del proyecto se crearon dos scripts, uno para el software MRTG y otro para la aplicación CACTI, que se comentan a continuación. Además se explicará más en detalle el software CACTI que se utiliza para representar los gráficos.

5.2.1 Script para MRTG

Como se observa en el diagrama de flujo de la figura 4.5, el programa pregunta si el valor es el último que ha sido ingresado en la pila donde se guardan los datos adquiridos vía SNMP. Esta pila es un archivo .log para cada interfaz o cada OID que se esté graficando en el MRTG. Para preguntar cuál es el valor se utilizó la siguiente lógica:

En la figura 5.3 se muestra un extracto del código del *script*, el cual lee un archivo (intermedio.txt) en el que se mantiene el valor en bytes del archivo cpu.1.log. Se lee el tamaño del archivo cpu.1.log y si los dos valores, el anterior y el leído con la función stat() de PERL, son iguales, el programa es terminado. Si el valor actual en el script \$size es mayor, significa que tiene nuevos valores almacenados, por lo que se llama a las funciones primera línea() y query().

```

sub main {
    # inicio de programa principal

    open(IN2,"intermedio.txt");#poner un 0 en intermedio para correrlo por primera vez.
    my $sizeb= readline(*IN2);
    close(IN2);
    open(IN,"C:/mrtg/cpu.1.log");
    open(OUT,">intermedio.txt")|| die "lo siento, no puedo crear intermedio.txt\n";
    my $size =(stat('C:/mrtg/cpu.1.log'))[7];#tamaño del archivo en bytes
    print OUT "$size\n";
    close(IN);
    close(OUT);

    if ($size == $sizeb){
        die "Programa terminado con exito!!\n";
    }
    if ($size > $sizeb && $sizeb !=0 ){
        primeralinea();
        query();
    }
    else {
        die "Programa terminado con exito!!\n";
    }
}

```

Figura 5.3. Inicio del Script para MRTG.

La función `primeralinea()` lee desde un archivo `.log` generado por el MRTG, y le da formato. Se abre el archivo `cpu.1.log`, se leen los primeros diez caracteres que contienen el tiempo *Unix* o tiempo POSIX (cantidad de segundos transcurridos desde la medianoche del 1 de enero de 1970), se pasa a un formato `dd/mm/aaaa`, con la función `localtime()` y así se tiene la fecha más entendible para el ser humano. Una vez que se tiene el tiempo listo se leen los datos almacenados en los archivos `.log`.

Posteriormente se leen los datos y se van guardando en variables. Una vez terminado de leer todos los datos validos, se imprime de manera que que la función `query()` lo pueda cargar en la base de datos MySQL.

Para poder cargar una tabla en una base de datos MySQL, los datos deben estar separados por un tabulador. Debido a esto, al imprimir en OUT, se define como `%s\t`, lo que significa una cadena seguido de un tabulador.

La función `query()` define una variable para poder hacer la consulta a la base de datos. En este caso lo que hace es definir la ruta donde se encuentra el archivo de texto `forMysql`, e ir acomodándolo en la tabla `ns500`. Después se llama a la

función db_connect(), la cual usa el modulo DBI instalado de CPAN. CPAN es una organización sin fines de lucro que desarrollan software para el lenguaje PERL. Para instalar algún módulo se debe escribir lo siguiente en la línea de comandos de Linux:

- \$sudo perl -MCPAN -e shell
- install DBI

En Windows para instalar módulos de PERL se usa el ppm (perl package manager), que viene con la distribución de PERL para Windows.

5.2.2 Script para CACTI

Para el *script* de CACTI se utilizó la lógica de la figura 5.4:

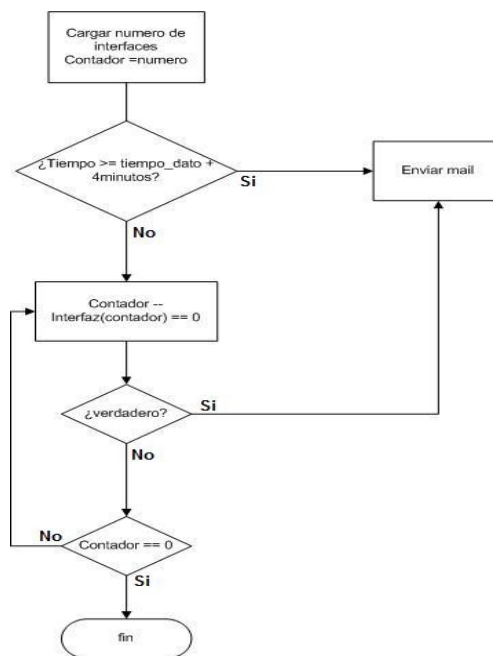


Figura 5.4. Diagrama de flujo del script para CACTI de las interfaces.

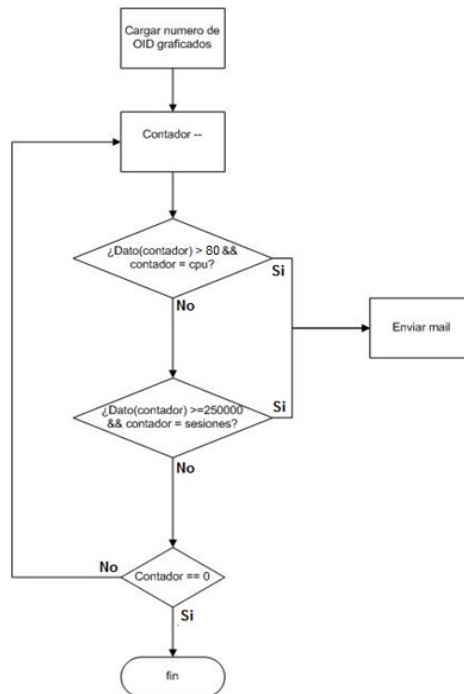


Figura 5.5. Diagrama de flujo del script para CACTI de los OID gestionados.

Se procedió a programar los algoritmos anteriores. El *script* final se encuentra en el anexo A.3.

A continuación se explicará los principales módulos del script:

La subrutina `leerrrd()` (ver anexo A.3) primero pregunta a la base de datos Round Robin, que es donde se almacena temporalmente después de ser capturada con SNMP la última actualización. Esto se hace con la función `system()` de PERL y el programa `RRDTOOL` lo manda a un archivo de texto llamado `base_leer.txt`.

Para saber cuál fue la última actualización, se utiliza la opción *last* en el programa `RRDTOOL` y la base de datos Round Robin con la que se está trabajando. Este tipo de base de datos tiene terminación `.rrd`. Si pasan 7.5 minutos sin tener un dato nuevo, el programa llama a la función `enviar_mail`, la cual adjunta la fecha y

hora en que el último valor fue adquirido y le adiciona una leyenda al equipo de *Surveillance*, para saber que está pasando.

Si el programa está funcionando correctamente, el script llamará de nuevo con la función *system()* a la base de datos, esta vez con la opción *fetch* para capturar el ultimo valor insertado. Esto se hace según la documentación del RRDTOOL, utilizando las opciones *-r* (resolución), en este caso 900 porque así lo guarda CACTI, *-s* (inicio) y *-e* (final), y se pasa como parámetro el último valor que se consiguió con el *last* y otro valor menos 5 minutos, para hacer un rango de un solo valor guardado.

Esta función tiene un *for* para recorrer todas las bases de datos Round Robin y así probar si alguna de las interfaces esta sin tráfico, como se ve en el diagrama de flujo de la figura 5.4, para los OID`s que se están monitoreando.

Como se ve en la figura 5.5, se utiliza la misma lógica explicada anteriormente, con la diferencia de que se pregunta si el valor leído es mayor que 80% del uso, debido a que normalmente el uso del CPU para este equipo es del 2% de uso y el tráfico pico no aumenta más allá del 10%. Por eso un 80% de uso significaría un problema en el equipo gestionado. En el caso de el número de sesiones tramitadas por el equipo, el máximo según la hoja de datos para este equipo es 250000 sesiones.

En la figura 5.6 se observa la rutina para enviar un mail. Esta rutina hace uso del modulo MIME (*Multipurpose Internet Mail Extensions*) de PERL y del servidor de correo Sendmail.


```

172 #+++++
173 sub enviar_mail1 {
174
175 -----
176 my $from_address = 'Infocacti_ns500@ericsson.com';
177 my $to_address = 'roiner@gmail.com,rafaelmata357@gmail.com';
178 my $mail_host = '172.27.10.194';
179
180 $msg = MIME::Lite->new (
181     From => $from_address,
182     To => $to_address,
183     Subject => $subject,
184     Type => 'multipart/mixed'
185 ) or die "Error creating multipart container: $!\n";
186
187
188 $msg->attach (
189     Type => 'TEXT',
190     Data => $message_body
191 ) or die "Error adding the text message part: $!\n";
192
193 $msg->send();
194 #MIME::Lite->send('smtp', $mail_host, Timeout=>60);
195
196 MIME::Lite->send('smtp', $mail_host, Timeout=>60);
197 $msg->send;
198 print "Mensaje enviado!\n";
199
200
201 #
202 };
203 #+++++

```

Figura 5.6. Subrutina enviar_mail1() escrito en PERL .

La subrutina deco() recibe un dato como parámetro, en este caso el contador, y devuelve lo que significa para el administrador, por ejemplo una interfaz o el CPU, para adjuntar esa información al email enviado y darle un mejor seguimiento.

5.2.3 Configuración de sendmail

Para configurar el servidor *Sendmail*, en una maquina Linux, en este caso Ubuntu 6.10, lo primero es instalar *sendmail* con la siguiente instrucción del *Shell*:

- `$sudo apt-get install sendmail`

Una vez instalado nos referimos a la carpeta /etc y se busca la carpeta mail. Si la carpeta existe se instaló correctamente el servidor, sino hay que descargarlo y ejecutarlo manualmente con las dependencias que presente la aplicación.

Se ingresa a la carpeta y se buscan los archivos de configuración. Estos archivos son: *sendmail.cf*, *local-host-names*, donde se indica a que dominios pertenece el

servidor de correo, *access*, que contiene la lista de direcciones IP y dominios que tienen permiso para utilizar el servidor de correo, *mailertable*, donde se pueden aplicar enrutamientos para dominios y *sendmail.mc* que es un fichero de macros, donde se pueden configurar distintas opciones de *sendmail*[13].

- Primero se añade en el archivo *local-host-names* el nombre del dominio de correo electrónico, en este caso *ericsson.com*.

- Se abre el archivo *access* y se agregan las IP de confianza de la red a la que pertenece:

```
192.168.10 RELAY
```

```
213.198.23.24 RELAY
```

```
172.27.8.161 RELAY
```

- Se asegura de que *sendmail* “escucha” al puerto 25. Para eso se escribe el siguiente comando: `netstat -ln` si la salida incluye algo similar a `tcp 0 0 0.0.0.0:25 0.0.0.0:* LISTEN`, entonces está correcto sino, habría que modificar el archivo *sendmail.mc* y comentar la opción que menciona al `localhost`, cuya dirección ip es `127.0.0.1`; esto porque sino solo sería accesible desde el propio servidor.

- Para comentar se coloca delante de ella “`dnl`” y se reinicia el proceso. Para hacer eso hay que ejecutar el programa para macros `m4` e introducirle los parámetros indicados: `m4 /etc/mail/sendmail.mc > /etc/sendmail.cf`, luego dentro de la carpeta ejecutar *make* y reiniciar el servidor de *sendmail*.

5.2.4 Configuración de Apache, PHP y MYSQL

Para la configuración del servidor apache, PHP y MYSQL en un servidor Ubuntu 6.10, lo primero es instalar los respectivos programas en Ubuntu de la siguiente manera:

```
$ sudo apt-get install apache php5 mysql-server snmp rrdtool
```

Una vez que se instalan los programas anteriores, hay que instalar otros paquetes para que las aplicaciones se comuniquen lo cual se hace de la siguiente forma:

```
$sudo apt-get install php5-cli libapache-mod-php5 php5-mysql
```

Después de haber instalado estos paquetes se tiene que modificar los archivos de configuración de la siguiente manera:

1. Se busca el archivo de configuración (por defecto se encuentra en la carpeta /etc/) php.ini y se agrega la siguiente línea:

```
extension_dir = /etc/php.d
```

2. Se busca el archivo de configuración de MYSQL mysql.ini y se agrega la siguiente línea:

```
extension=mysql.so
```

3. Se busca el archivo de configuración de SNMP snmp.ini y se agrega la siguiente línea:

```
extension=snmp.so
```

4. Para que corra en Apache se busca el archivo de configuración de PHP (php.conf), en la carpeta /etc/httpd y se agregan las siguientes líneas:

```
# PHP is an HTML-embedded scripting language which attempts to make it
# easy for developers to write dynamically generated webpages.
LoadModule php5_module modules/libphp5.so
#
# Cause the PHP interpreter to handle files with a .php extension.
AddHandler php5-script .php
AddType text/html .php
#
# Add index.php to the list of files that will be served as directory
# indexes.
DirectoryIndex index.php
```

5. Una vez hechos esos cambios, se reinician los procesos para eso. Se ingresa en la carpeta `/etc/init.d` y se ejecuta lo siguiente:

```
$sudo ./apache2 restart
$sudo ./snmpd restart
$sudo ./mysql restart
```

6. Para probar si los cambios que se realizaron surtieron efecto, se hace un pequeño programa de PHP que contenga la siguiente línea y se copia en la carpeta `/var/www`

```
<?php phpinfo() ?>
```

Se abre el navegador *web* y en la barra de URL se escribe lo siguiente:

7. `http://localhost/programa.php`

Si el programa se ejecuta con éxito nos dará la información de los módulos soportador por el servidor. Se tiene que buscar el modulo MYSQL para que todo esté listo para instalar CACTI.

8. Se abre el CLI (Instrucciones por línea de comandos, según sus siglas en ingles) de MYSQL con la siguiente instrucción:

```
$mysql
```

Se le asigna una contraseña al usuario root:

```
mysql> set password for root@localhost=password('contraseña');
```

Se crea una nueva base de datos para CACTI:

```
mysql> create database cactidb;
```

Se hace un usuario nuevo y se le brinda derechos de administrador sobre la nueva base de datos hecha y se sale de MYSQL:

```
mysql> grant all on cactidb.* to root;
```

```
mysql> grant all on cactidb.* to root@localhost;
mysql> grant all on cactidb.* to cactiuser;
mysql> grant all on cactidb.* to cactiuser@localhost;
mysql> set password for cactiuser@localhost=password('yyyyy');
mysql> exit
```

Habiendo hecho estos pasos, y si todo está correcto, CACTI podría ponerse en funcionamiento sin ningún problema como se muestra en la siguiente sección.

5.2.5 Configuración de CACTI

Para la configuración de CACTI primero se descarga la carpeta comprimida de la dirección <http://www.cacti.net/downloads/cacti-0.8.7e.tar.gz>, luego se descomprime y se guarda en la carpeta en /var/www (servidor Apache).

Posteriormente se abre la carpeta /var/www/cacti. Una vez ahí se carga la estructura de base de datos que trae CACTI llamada cacti.sql a la base de datos que se hizo anteriormente, de la siguiente manera:

```
$mysql -user=root -password=xxxxxx cactidb < cacti.sql
```

Ahora se modifica el propietario a los siguientes directorios que vienen dentro de CACTI:

```
$sudo chown -R usuario rra/ log/
```

Una vez hecho esto se debe modificar el archivo de configuración para que CACTI pueda conectarse con MYSQL para eso se abre el archivo config.php que viene dentro de la carpeta *include* y cambiamos las líneas siguientes:

```
$database_default = "cactidb";
$database_hostname = "localhost";
$database_username = "cactiuser";
$database_password = "yyyyy";
```

Una vez hecho esto, estamos listos para poner el *poller* de datos en el *crontab* de Linux para eso agregamos la siguiente línea[14]:

```
# nano /etc/cron.d/cacti
```

```
*/* * * * * usuario /usr/bin/php /var/www/cacti/poller.php > /dev/null 2>&1
```

Esto significa que el poller se ejecutara cada cinco minutos todos los días y todos los años y la salida se enviara a /dev/null que es un tipo de agujero negro en Linux.

Se abre el navegador y se digita la siguiente dirección, lo cual nos mostrara una pantalla de bienvenida como la mostrada en la figura 5.7:

```
http://localhost/cacti
```

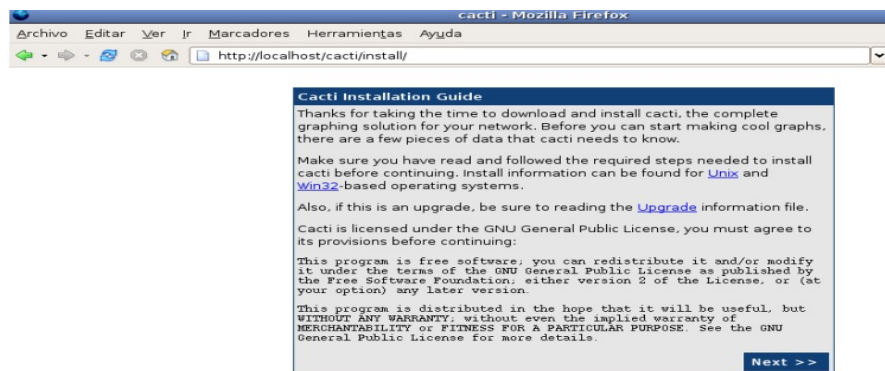


Figura 5.7. Inicio del programa CACTI.

Capítulo 6: Análisis de Resultados

6.1 Análisis

A continuación se realiza el análisis de los resultados obtenidos.

En la figura 6.1 se muestra la salida del programa CACTI. Se observa claramente el tiempo y la cantidad de bits por segundo que pasan por la interfaz Eth 3/1/5 del NetScreen 500, esta imagen es con una resolución máxima de la última media hora de tráfico. El software presenta varios tipos de resolución, aunque la que más interesa es la que está pasando en el momento que se observa la imagen, por lo que ver el tráfico en resolución de los últimos treinta minutos es más informativo que en el anterior *software*.

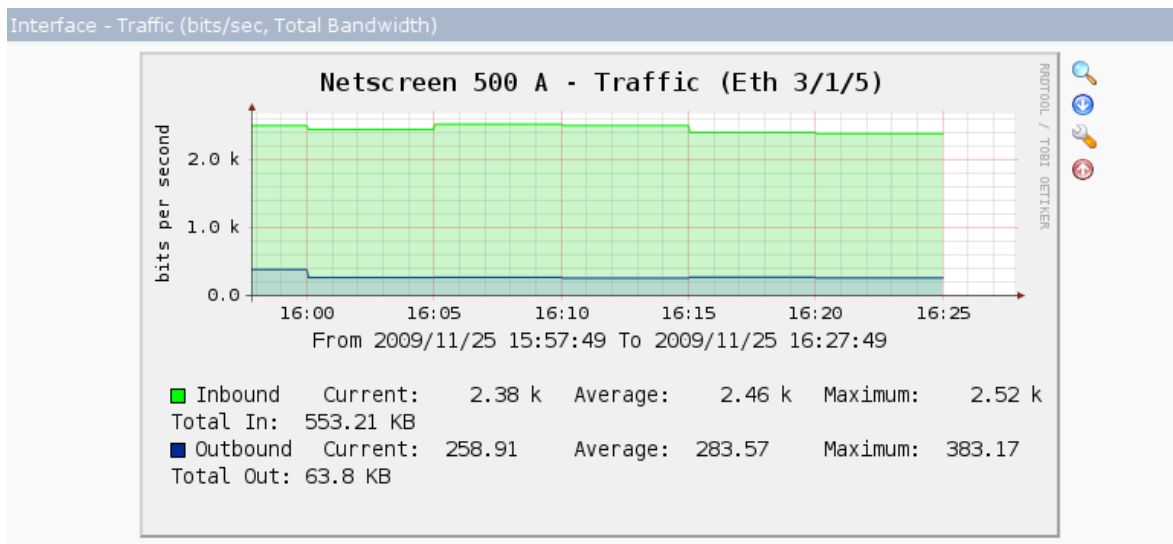


Figura 6.1. Gráfica generada por CACTI del tráfico por la interfaz Eth 3/1/5.

En la figura 6.2 se presenta la información almacenada en la base de datos MYSQL en una hoja de cálculo de Excel, lo cual le ayuda en gran manera a los administradores de la red a llevar una estadística del tráfico. Una vez que se tiene en la hoja de cálculo es fácil aplicar las funciones con las celdas ya llenas, sin necesidad

de ir llenándola manualmente. El servidor de monitorización lleva el nombre de “mipecera”.

	A	B	C	D	E	F	G	H
1	Title; "mipecera - Load Average"							
2	Vertical Label; "processes in the run queue"							
3	Start Date; "2009-12-13 10:25:00"							
4	End Date; "2009-12-15 10:25:00"							
5	Step; "300"							
6	Total Rows; "577"							
7	Graph ID; "906"							
8	Host ID; "1"							
9								
10	Date, "1 Minute Average", "5 Minute Average", "15 Minute Average", "col4-cdefg"							
11	2009-12-13 10:25:00,"2.4820666667e+00","2.1618666667e+00","2.7612666667e+00","7.4052000000e+00"							
12	2009-12-13 10:30:00,"2.6985333333e+00","2.0904666667e+00","2.6308666667e+00","7.4198666667e+00"							
13	2009-12-13 10:35:00,"2.3931000000e+00","1.9613000000e+00","2.4914000000e+00","6.8458000000e+00"							
14	2009-12-13 10:40:00,"3.4032000000e+00","2.0990666667e+00","2.4304000000e+00","7.9326666667e+00"							
15	2009-12-13 10:45:00,"7.7660000000e-01","1.5753000000e+00","2.2023000000e+00","4.5542000000e+00"							
16	2009-12-13 10:50:00,"3.2815333333e+00","4.8889000000e+00","3.8744000000e+00","1.2044833333e+01"							
17	2009-12-13 10:55:00,"2.3165333333e+00","3.0126666667e+00","3.4131333333e+00","8.7423333333e+00"							
18	2009-12-13 11:00:00,"2.1411333333e+00","2.2748666667e+00","3.0623333333e+00","7.4783333333e+00"							
19	2009-12-13 11:05:00,"2.7956000000e+00","2.1706666667e+00","2.8613333333e+00","7.8276000000e+00"							
20	2009-12-13 11:10:00,"2.4026666667e+00","2.0408666667e+00","2.6812000000e+00","7.1247333333e+00"							
21	2009-12-13 11:15:00,"2.6682000000e+00","2.0400000000e+00","2.5707333333e+00","7.2789333333e+00"							
22	2009-12-13 11:20:00,"8.5220000000e-01","2.0400000000e+00","2.5799333333e+00","5.4721333333e+00"							
23	2009-12-13 11:25:00,"2.4492000000e+00","2.1194666667e+00","2.5601333333e+00","7.1288000000e+00"							
24	2009-12-13 11:30:00,"3.1851333333e+00","2.3882000000e+00","2.6096666667e+00","8.1830000000e+00"							
25	2009-12-13 11:35:00,"2.3683000000e+00","2.1524000000e+00","2.5110000000e+00","7.0317000000e+00"							
26	2009-12-13 11:40:00,"2.6679333333e+00","2.1500000000e+00","2.4901333333e+00","7.3080666667e+00"							
27	2009-12-13 11:45:00,"7.3300000000e-01","1.7030000000e+00","2.3012666667e+00","4.7372666667e+00"							

Figura 6.2. Hoja de Excel con los valores del promedio de carga descargados desde la base de datos.

Como se muestra en la figura 6.3 el tráfico por la interfaz se detuvo el sábado. El script de alarma alertó correctamente a los administradores informándoles vía correo electrónico como se puede ver en la figura 6.4.

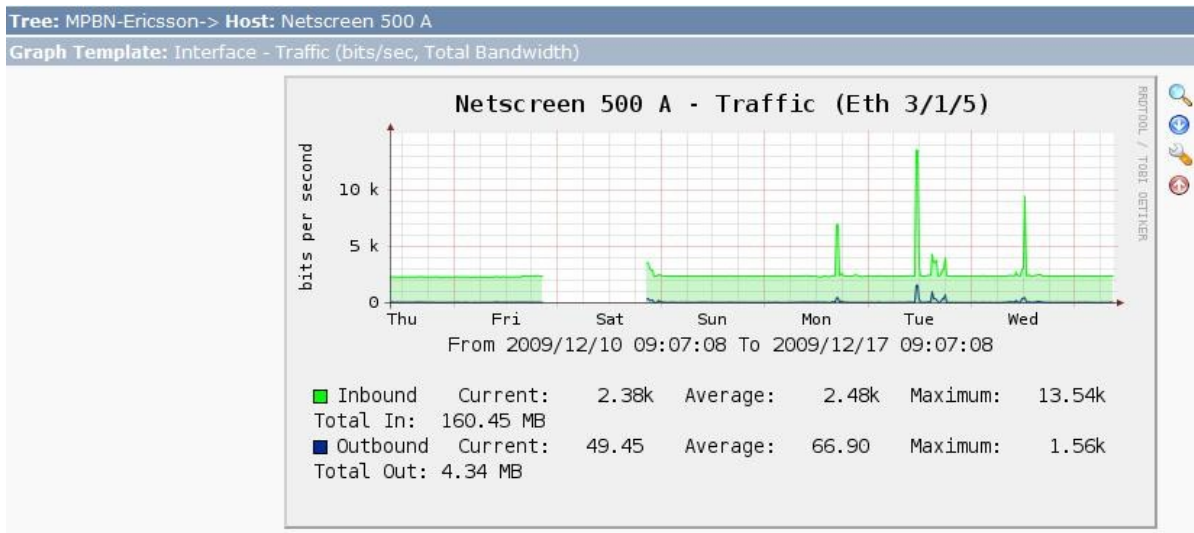


Figura 6.3. Gráfica generada por CACTI de la última semana del tráfico por la interfaz Eth 3/1/5.

En la figura 6.4 se observa la alarma que envía el *script*, informando de que la interfaz se encuentra sin tráfico. Como la falla se dió un sábado, la reparación se realizo hasta el domingo temprano, como el equipo es redundante, el tráfico se desvió hacia el servidor de respaldo.



Figura 6.4. Foto del correo enviado por el *script* de alarma.

En la tabla 6.1 se resume los diferentes escenarios que puede tener la red con respecto a una avería; sin sistema de alarma, con el sistema que se tenía antes del presente proyecto (MRTG) y con el sistema implementado (CACTI). Como se puede observar antes de la aplicación CACTI el reporte de fallas era hasta que los usuarios

avisaran de algún problema con los servicios que querían acceder, ya que a nivel de centro de gestión no se puede observar una avería o un aumento de sesiones en un equipo.

Con la monitorización, utilizando aplicación MRTG, solo era posible observar la afectación de tráfico y los problemas de red. Con el nuevo sistema, es posible estar monitorizando las sesiones, la sobrecarga, la red y la afectación de tráfico en tiempo real.

Tabla 6.1. Tiempos de respuesta para reparación de averías antes y después del sistema [22].

Avería	Tiempo de respuesta sin monitorización	Tiempo de respuesta monitorización con MRTG	Tiempo de respuesta monitorización CACTI más scripts de alarma.
Aumento de sesiones	2 días	2 días	5 minutos
Sobrecarga de equipos	1 día	1 día	5 minutos
Problemas de red	4 horas	1 hora	5 minutos
Afectación de tráfico	2 horas	[15-30] minutos	5 minutos

Como se observa en la tabla 6.2 al gestionar lo mas pronto posible una avería, el tiempo total de reparación va a ser mucho menor, por lo que el gasto de la empresa por atención de averías disminuye en mas de un 50% como claramente se puede observar en la tabla 6.2.

Tabla 6.2. Multas aplicadas cuando existe afectación de tráfico en la red.

	Gastos por multas sin monitorización	Gastos por multas monitorización MRTG	Gastos por multas monitorización CACTI
Costo de una avería sin tiempo de reparación	\$60,000.00	\$15,000.00	\$2,500.00
Costo de una avería con tiempo de reparación de una hora.	\$90,000.00	\$45,000.00	\$17,500.00

Uno de los equipos que más problemas y averías genera es el centro de mensajes de texto del ICE, por lo que ahora está siendo monitorizado de manera que si los niveles de utilización de CPU son mayores al 80%, como en el caso de la figura 6.5, el administrador del centro de mensajes será notificado para reparar la falla o habilitar el segundo centro de mensajes. En estos momentos el centro de mensajes tiene mas de un mes de no presentar afectación de tráfico.

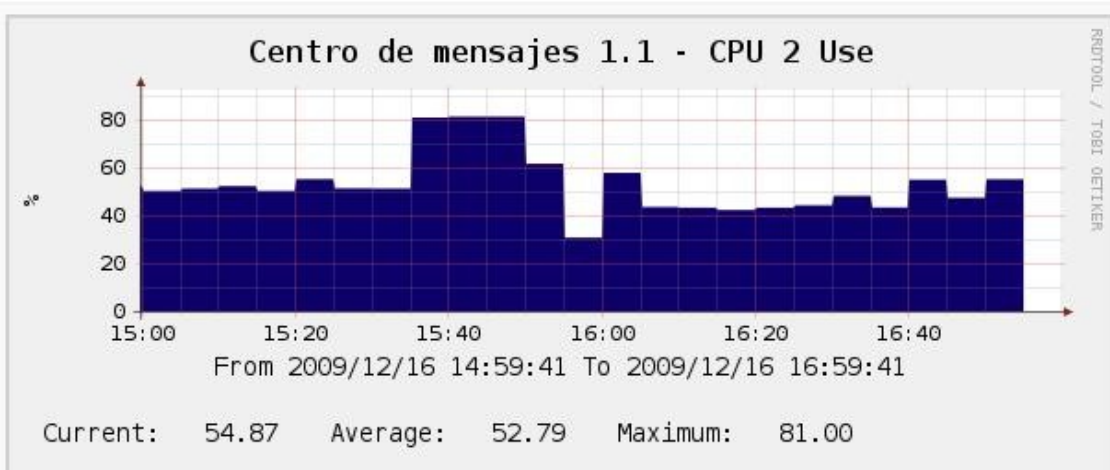
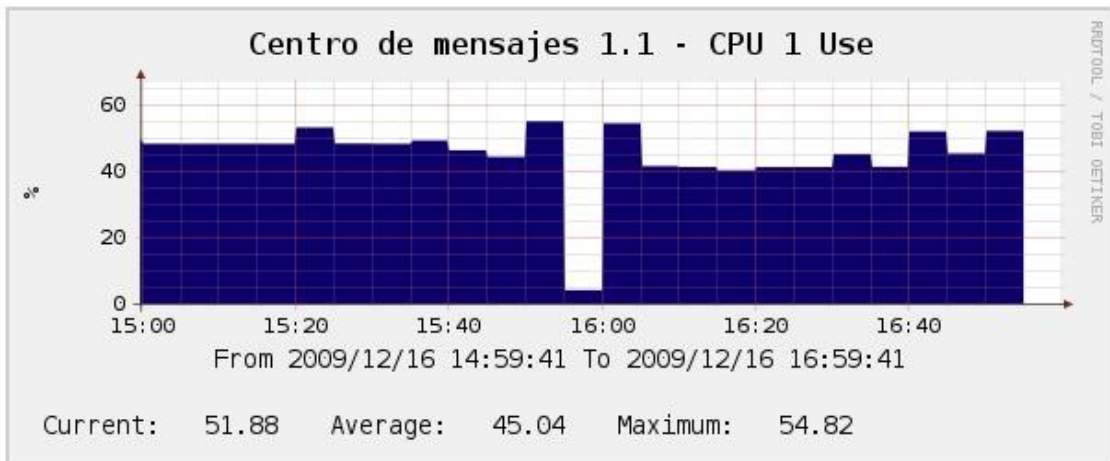


Figura 6.5. Uso del CPU de uno de los centros de mensajes del ICE.

En la figura 6.6 se observan los equipos monitorizados en el MPBN de ERICSSON. La Información referente a las direcciones IP no puede mostrarse por seguridad. En el sistema implementado se hayan siendo monitorizados los equipos nuevos y los que anteriormente estaban siendo monitorizados. Esto se puede ver en la figura 3.3.

<< Previous		Showing Rows 1 to 26 of 26 [1]								Next >>	
Description**	ID	Graphs	Data Sources	Status	Event Count	Hostname	Current (ms)	Average (ms)	Availability		
Centro de mensajes 1.1	25	9	9	Up	0	172.27.10.100	1746.07	1933.83	100		
Centro de mensajes 1.2	26	9	9	Up	0	172.27.10.100	1278.55	864.92	100		
Centro de mensajes 2.1	27	9	9	Up	0	172.27.10.100	2171.67	2060.36	100		
Centro de mensajes 2.2	28	9	9	Up	0	172.27.10.100	39.11	857.98	100		
Firewall Gi A	12	3	3	Up	0	172.27.10.100	2092.66	1572.96	100		
Firewall Gi B	13	20	20	Up	0	172.27.10.100	1422.22	639.08	100		
Firewall GO A	14	10	10	Up	0	172.27.10.100	3408.4	1341.72	100		
Firewall GO B	15	9	9	Up	0	172.27.10.100	1435.99	808.7	100		
Firewall Gp A	16	9	9	Up	0	172.27.10.100	1907.16	1260.61	100		
Firewall Gp B	17	9	9	Up	0	172.27.10.100	51.35	691.92	100		
Juniper M7 A	3	39	39	Up	0	172.27.10.100	1387.75	979.91	99.99		
Juniper M7 B	4	14	14	Up	0	172.27.10.100	4582.14	1125.59	99.99		
mipecera	1	6	7	Up	0	172.27.10.100	0.11	0.15	100		
Netscreen 500 A	2	11	11	Up	0	172.27.10.100	1053.98	989.76	99.99		
Netscreen 500 B	5	11	11	Up	0	172.27.10.100	1228.54	1104.66	99.99		
OSS	22	2	2	Up	0	172.27.10.100	1278.83	1588.64	100		
Router Redback 1	24	64	64	Up	0	172.27.10.100	796.23	743.97	100		
Router Tibas Centro	29	5	5	Up	0	192.168.10.10	2086.9	2213.08	100		
Switch Core A	10	13	13	Up	0	172.27.10.100	1974.09	1275.73	100		
Switch Core B	11	12	12	Up	0	172.27.10.100	2786.81	1450.48	100		
Switch PS A	6	29	29	Up	0	172.27.10.100	1035.17	1012.45	99.99		
Switch PS B	7	37	37	Up	0	172.27.10.100	1761.63	1216.49	99.99		
Switch SN A	8	37	37	Up	0	172.27.10.100	1621.89	1126.43	100		
Switch SN B	9	28	28	Up	0	172.27.10.100	2059.63	1316.57	100		
Switch Tib A	18	33	34	Up	0	172.27.10.100	1362.23	1131.18	100		
Switch Tib B	19	60	60	Up	0	172.27.10.100	1826.78	1279.37	100		

Figura 6.6.Equipos monitorizados en el MPBN ERICSSON.

En la figura 6.7 y 6.8, se observa la comparación entre el sistema anterior y el sistema implementado, donde en el sistema nuevo el tráfico está dado en bits por segundo y se puede exportar todos los datos en un archivo excel como el de la figura 6.2, además se puede ver todas las interfaces en una misma página web, por lo que los operadores de *Surveillance* puedan estar chequeando un equipo observando una mayor cantidad de figuras como se puede ver en la figura 6.9.

The statistics were last updated **Monday, 14 June 2010 at 9:00**,
 at which time 'NS500PS01' had been up for **486 days, 6:11:15**.

'Daily' Graph (5 Minute Average)

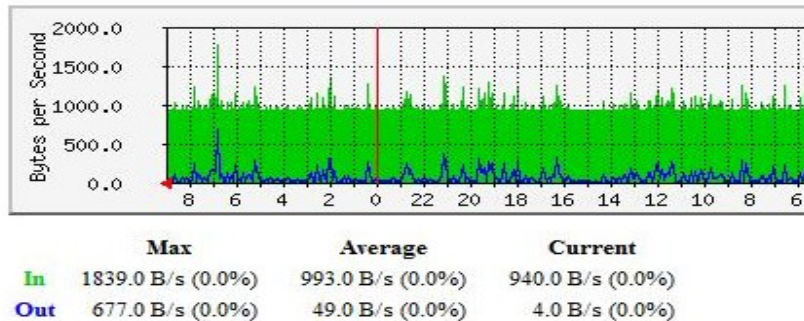


Figura 6.7. Gráfica de salida de la aplicación MRTG, para la interfaz Eth 1/1 del Firewall Netscreen®.

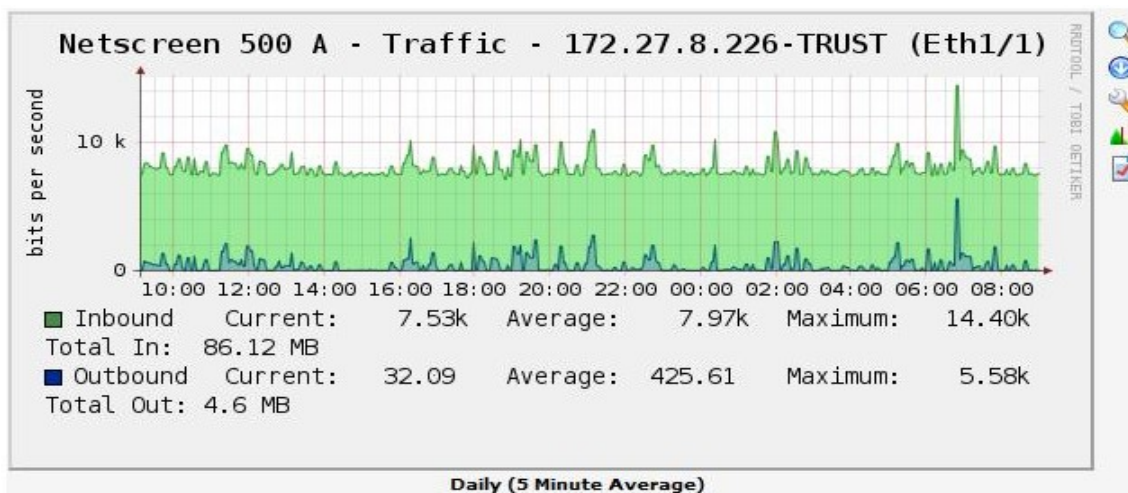


Figura 6.8. Gráfica de salida de la aplicación CACTI, para la interfaz Eth 1/1 del Firewall Netscreen®.

En la figura 6.9 se muestran los gráficos con máxima resolución de los OID privados, y el tráfico por cada interfaz en una sola página web.

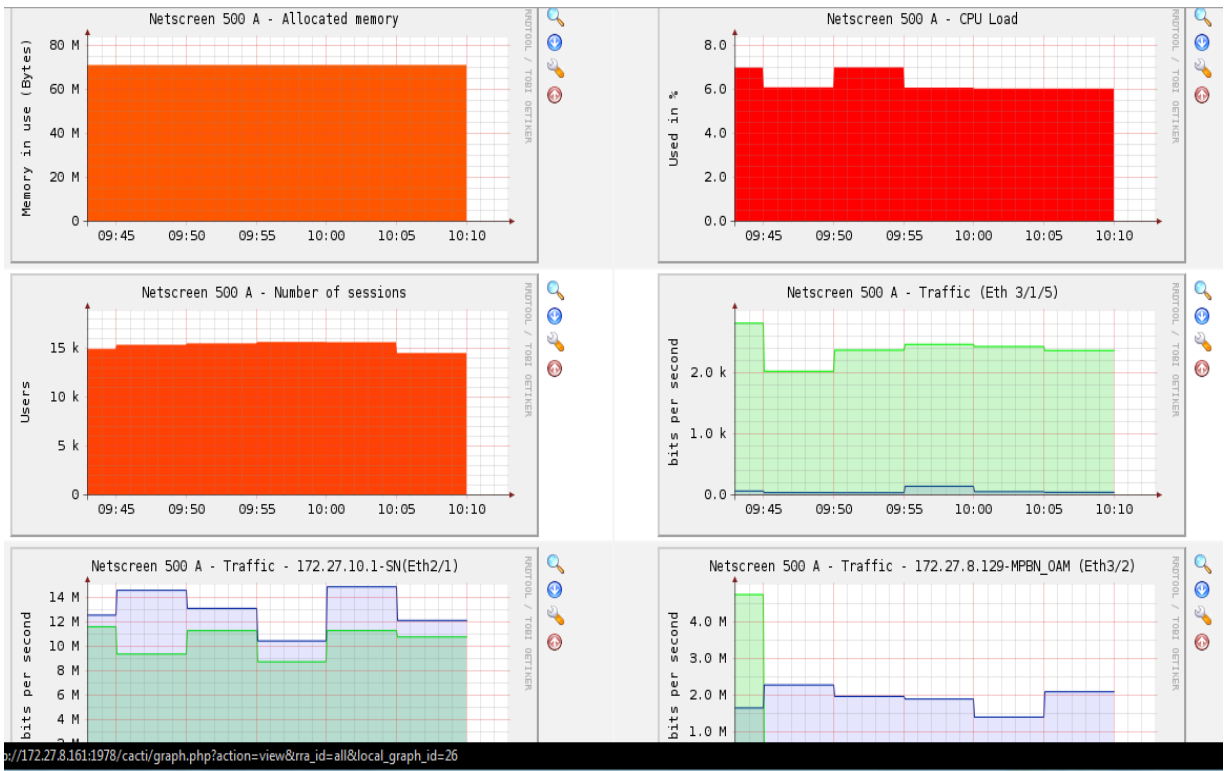


Figura 6.9. Vista de todos los gráficos de un equipo.

Capítulo 7: Conclusiones y recomendaciones

7.1 Conclusiones.

- Las aplicaciones de monitorización utilizando SNMP, son las herramientas de monitorización modernas para mantener las redes informáticas siempre activas y funcionales.
- El uso de scripts y software de alto nivel facilita el desarrollo de aplicaciones funcionales de administración de equipos computacionales.
- Se logró monitorizar el rendimiento del CPU en tiempo real y con eso fiscalizar la sobrecarga de los equipos.
- Se logró almacenar en una base de datos MYSQL el historial de tráfico y sesiones en un equipo por mas de un mes.
- El tiempo de respuesta de una avería en la red MPBN disminuyo a cinco minutos después de presentarse la falla.
- Se logró graficar los OID privados para empresas y alertar a los administradores de la red de sobrecargas en los equipos.
- El gasto por efecto de pago de multas disminuyó en mas de un 50% con el nuevo sistema de monitorización y alarmas.

7.2 Recomendaciones

- Utilizar las funciones SNMPSET si el identificador de objeto presenta la opción sólo escritura ó escritura – lectura.
- Programar un tipo de alarma gráfica e integrarla al sistema desarrollado.
- Desarrollar agentes SNMP para instalarlos en los equipos y poder monitorear la temperatura.
- Actualizar el sistema operativo así como las aplicaciones en el servidor de monitorización.
- Integrar un módulo de generación de mensajes tipo trap así como un visor de mensajes *trap* en la aplicación instalada.

Referencias

- [1] ERICSSON AB [Documento digital]. Operaciones del BSC GSM, Descripción del sistema. ERICSSON AB 2003, Documento LZT 123-3801 R5A. Disponible en <<http://www.ERICSSON.com/globalservices>>.
- [2] Oetiker, Tobias tobi@oetiker.ch, MTRTG Reference [en línea]. Última Actualización 09/17/09. Disponible en <<http://oss.oetiker.ch/mrtg/doc/index.en.html>> [Consulta: 30 agosto 2009].
- [3] García C., José Manuel, Marín L., Rafael, Gestión de Red [En línea]. Última Actualización 17/09/03. Disponible en <<http://ditec.um.es/laso/docs/tut-tcpip/3376fm.html>> [Consulta: 27 noviembre 2009].
- [4] Tobal, Javi. SNMP [En línea]. Última actualización 30/09/99. Disponible en: <<http://www.arrakis.es/~tobal/snmp.htm>> [Consulta: 1 diciembre 2009].
- [5] Enciclopedia wikipedia. Red de computadoras [En línea]. Disponible en: <http://es.wikipedia.org/wiki/Red_de_computadoras> [Consulta: 1 diciembre 2009].
- [6] George R. Brown School of Engineering. A 10 Gigabit Programmable Network Interface Card - How NICs Work [En línea]. Disponible en: <http://www.ece.rice.edu/~willmann/teng_nics_hownicswork.html#hownicswork> [Consulta: 23 octubre 2009].
- [7] Enciclopedia wikipedia. Lenguaje de programación [En línea]. Disponible en: <http://es.wikipedia.org/wiki/Lenguaje_de_programacion> [Consulta: 1 diciembre 2009].
- [8] Enciclopedia wikipedia. Script [En línea]. Disponible en: <<http://es.wikipedia.org/wiki/Script>> [Consulta: 1 diciembre 2009].

- [9] Coto C, Anibal. Apuntes Curso Comunicaciones Eléctricas, Medidas de Información. ITCR, Cartago 2008.
- [10] Medir tráfico – bandwidth [En línea]. Lista de correo del Grupo de Usuarios de Software Libre en Córdoba. Fecha: 2006-09-25 12:17, Grulic lista de correo <<http://proposicion.org.ar/lurker/list/grulic.es.html>> [Consulta: 12 septiembre 2009]
- [11] NET-SNMP [En línea]. Última actualización 02/03/07 <<http://www.net-snmp.org/>> [Consulta: 12 septiembre 2009]
- [12] Oetiker, Tobias tobi@oetiker.ch, About RRDTOOL [en línea]. Última Actualización 12/09/09. Disponible en <<http://oss.oetiker.ch/rrdtool/>> [Consulta: 17 septiembre 2009].
- [13] Como instalar y configurar Sendmail en Ubuntu Dapper [en línea]. Última Actualización 16/07/06. Disponible en <<http://vpuch.livejournal.com/26941.html>> [Consulta: 17 octubre 2009].
- [14] The Cacti Group. Configure PHP [en línea]. Última Actualización 21/11/09. Disponible en <http://www.cacti.net/downloads/docs/html/install_unix.html> [Consulta: 1 octubre 2009].
- [15] ERICSSON - the leading telecom supplier [en línea]. Última Actualización 21/11/09. Disponible en <<http://www.ERICSSON.com/ERICSSON/corpinfo/index.shtml>> [Consulta: 1 agosto 2009].
- [16] Enciclopedia wikipedia. Script [En línea]. Disponible en: <<http://es.wikipedia.org/wiki/Script>> [Consulta: 1 diciembre 2009].
- [17] Shipway, Steve s.shipway@auckland.ac.nz, Routers2.cgi [en línea] Disponible en: <<http://www.steveshipway.org/software/index2.html>> Última Actualización 16/07/09. [Consulta: 1 diciembre 2009].

- [18] PAESSLER [en línea] Disponible en: <http://www.paessler.com/prtg/?source=adwords&campaign=prtg_search_campaign&adgroup=prtg_traffic_grapher&adnum=017&gclid=CP2EIMWxmaECFQaU7Qoduj1aMw> Última Actualización 25/09/09. [Consulta: 1 diciembre 2009].
- [19] Quest software, Big Brother [en línea] Disponible en: <<http://www.bb4.org/>> [Consulta: 1 diciembre 2009].
- [20] Nagios enterprises, Nagios [en línea] Disponible en: <<http://www.nagios.org/>> Última Actualización 15/11/09. [Consulta: 1 diciembre 2009].
- [21] Cartel de licitación [Documento digital], INSTITUTO COSTARRICENSE DE ELECTRICIDAD, CONTRATACION DIRECTA No: 2009CD-004554-PROV. [Consulta: 25 Marzo 2010].
- [22] Ing. Rafael Mata Moya, Multas y averías en el MPBN. ERICSSON CR, 2010.

Apéndices y anexos

A.1 Glosario, abreviaturas y simbología

- SNMP (*Simple Network Management Protocol*), Protocolo simple de gestión de red
- MIB (*Management Information Base*), Base de información gestionada
- Protocolo: Es un conjunto de reglas usadas por computadoras para comunicarse unas con otras a través de una red.
- GSM (Sistema Global de Comunicaciones Móviles)
- SO(Sistema operativo)
- NIC (*Network Interface Card*), Tarjeta de red
- Buffer: Es una ubicación de la memoria en una computadora o en un equipo, reservada para el almacenamiento temporal de información digital, mientras que está esperando ser procesada.
- Programa: Secuencia de instrucciones que una computadora puede interpretar y ejecutar.
- Ancho de banda digital: Es la cantidad de datos que se pueden transmitir en una unidad de tiempo.
- bps: Es el número de impulsos elementales (1 ó 0) transmitidos en cada segundo.
- *Shell*: Una familia particular de intérpretes de órdenes.
- Capa de aplicación:

- Tráfico: Es la cantidad de datos enviados y recibidos

A.2 Manual de usuario

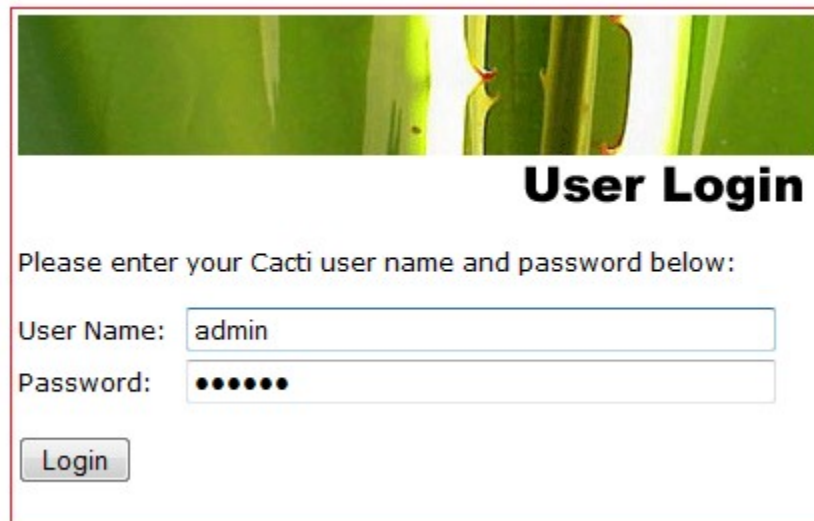
Manual de usuario para el programa de monitorización CACTI.

1. Resumen de la aplicación.

Cacti es una aplicación desarrollada para visualizar y manejar equipos mediante el protocolo SNMP. Cacti es un software desarrollado por diferentes programadores a nivel mundial mediante licencia libre GNU (*General Public Licence*).

2. Agregar un nuevo equipo.

Para agregar un nuevo equipo, tenemos que ingresar a la aplicación, como vemos en la figura A.2.1.



User Login

Please enter your Cacti user name and password below:

User Name:

Password:

Figura A.2.1 Figura que muestra la pantalla de ingreso a la aplicación.

Una vez que estamos dentro de la aplicación con los derechos de administrador, le damos *clic* en la pestaña *console* de la figura A.2.2.



Figura A.2.2 Figura que muestra la pantalla de bienvenida para la configuración (*console*) de la aplicación.

Ahora tenemos que crear un nuevo objeto tipo *device*, para lo cual le damos *clic* en los círculos rojos de la figura A.2.2, y nos aparece la siguiente pantalla:

Para crear un nuevo dispositivo llenamos los diferentes campos que se muestran en la figura A.2.3, con la información que se describe en la tabla A.2.1. Una vez definidos los campos según las especificaciones definidas en cada campo ejemplo *Hostname*, la descripción de este campo es *fully qualified hostname or IP address for this device*.

The screenshot shows the 'Devices [new]' configuration page in Cacti. The page is divided into several sections with the following fields and values:

- Description:** New Host
- Hostname:** hostname.domain.com
- Host Template:** ucd/net SNMP Host
- Notes:** (Empty text area)
- Disable Host:** Disable Host
- Downed Device Detection:** Ping
- Ping Method:** UDP Ping
- Ping Port:** 23
- Ping Timeout Value:** 400
- Ping Retry Count:** 1
- SNMP Options:**
 - SNMP Version:** Version 3
 - SNMP Username (v3):** snmpuser
 - SNMP Password (v3):** (masked with asterisks)
 - SNMP Auth Protocol (v3):** MD5 (default)
 - SNMP Privacy Passphrase (v3):** myprivpass
 - SNMP Privacy Protocol (v3):** DES (default)
 - SNMP Context:** (Empty)
 - SNMP Port:** 161
 - SNMP Timeout:** 500
 - Maximum OIDs Per Get Request:** 10

At the bottom right, there are 'cancel' and 'create' buttons.

Figura A.2.3 Figura que muestra la pantalla para la creación de un nuevo objeto tipo *device*.

Tabla A.2.1 Definición de campos para la creación de un nuevo objeto tipo *device*.

Campo	Descripción
Description	Nombre que aparece en el título de los gráficos especificando al equipo.
Hostname	Se define la dirección IP o se especifica el hostname con DNS activado.
Host Template	Es la plantilla que se utiliza para un equipo específico ej: equipo con snmp activo.
Notes	Cualquier tipo de texto o comentario para el equipo.
Downed Device Detection	Las opciones son: NONE: Sin detección PING and SNMP: Se usan ambas pruebas SNMP: verifica SNMP ICMP: Ejecuta un PING con opciones.
Ping Method	ICMP: Prueba ICMP. UDP: Prueba UDP TCP: Prueba TCP

Ping Timeout Value	Unidades en milisegundos
Ping Retry Count	El numero de intentos para conectar al equipo remoto antes de fallar.
SNMP Version	Versiones del protocolo SNMP (1,2 ó 3)
Ping Port	Valido solo para PING UDP/TCP.

3. Agregar nuevos gráficos.

Una vez realizado lo anterior para los equipos que queremos gestionar creamos los gráficos para cada nodo. Para eso hacemos el siguiente:

En el *device* que se creo anteriormente le damos clic en la opción **Create Graphs for this Host** y nos aparece la siguiente pantalla:

my own routing device (router) ucd/net SNMP Host

Create new graphs for the following host: [Edit this Host](#)
[Create New Host](#)

my own routing device (router)

Graph Templates

Graph Template Name	
Create: ucd/net - CPU Usage	<input type="checkbox"/>
Create: ucd/net - Load Average	<input type="checkbox"/>
Create: ucd/net - Memory Usage	<input type="checkbox"/>
Create: (Select a graph type to create)	<input type="checkbox"/>

Data Query [SNMP - Interface Statistics]

Index	Status	Description	Type	Speed	Hardware Address	
1	Up	Ethernet0	ethernetCsmacd(8)	100000000	00:30:30:2E:35:30:2E:37:46:2E:30:43:2E:30:30:2E:44:16:00:00:00:01	<input checked="" type="checkbox"/>
2	Up		0	0		<input type="checkbox"/>
3	Up		0	0		<input type="checkbox"/>
4	Up	Ethernet1	ethernetCsmacd(8)	100000000		<input type="checkbox"/>

Select a graph type: **In/Out Bits with Total Bandwidth**

Figura A.2.4 Figura que muestra la pantalla para la crear gráficos en la aplicación.

En la pantalla anterior seleccionamos los gráficos que queremos monitorizar. En la pestaña *graphs* (ver figura A.2.2) se ven los gráficos que se han creado. En la figura A.2.4, se puede seleccionar que tipo de medición y gráfico que queremos las opciones son: In/out bytes, In/out bits, In/out packets para gráficos de trafico. Para gráficos de almacenamiento existen otras opciones. Una vez seleccionado el tipo de grafico le damos el check y luego *create*, como en la siguiente imagen:

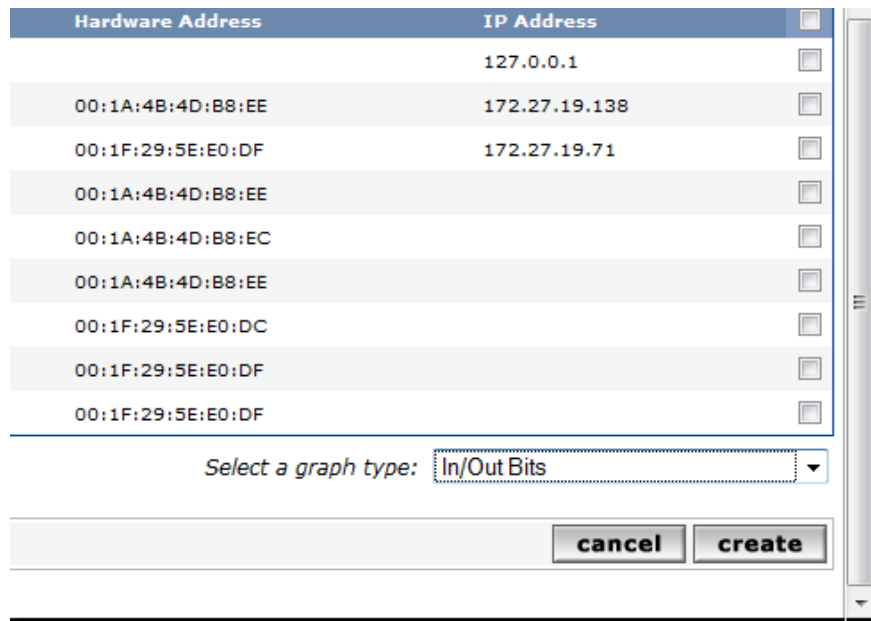


Figura A.2.5 Figura que muestra la comprobación para crear el grafico.

Una vez creado el dispositivo y el gráfico esperamos a que el script *poller* (unos 10 minutos) adquiera los datos, seleccionamos la pestaña *graphs* y notamos a ver como empieza a graficar como se muestra en la figura A.2.6

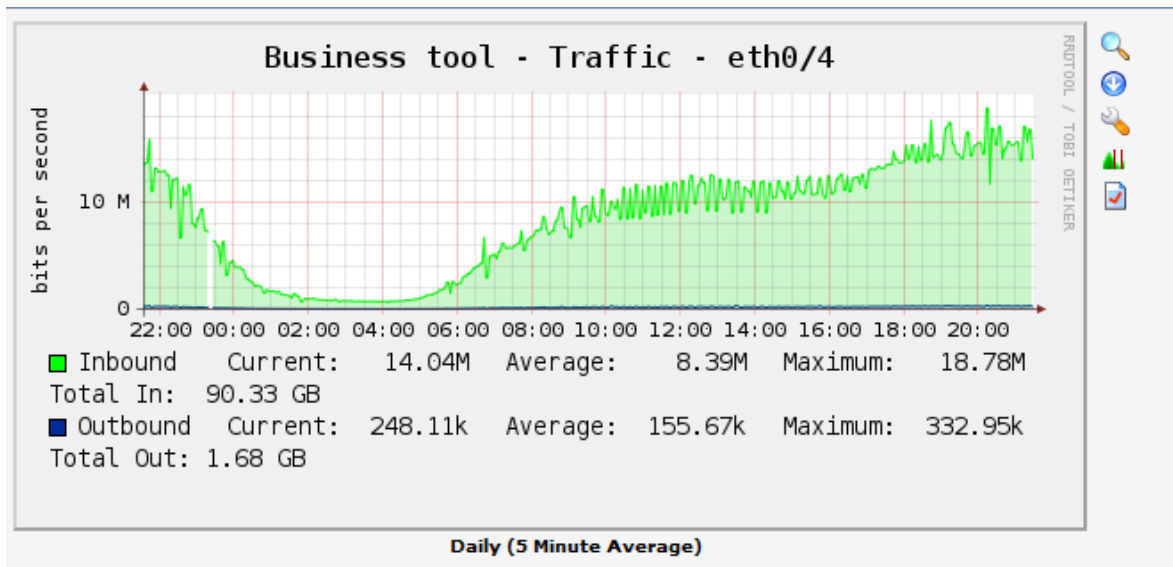


Figura A.2.6 Figura que muestra el tráfico por la interfaz Eth 0/4.

Para una mayor información sobre configuraciones especiales, además de trucos y software 3PP se puede visitar la pagina oficial de la aplicación en el siguiente link: http://docs.cacti.net/manual:087:2_basics.0_principles_of_operation#basics.

A.3 Scripts

A.3.1 Script MRTG

```
#!/usr/bin/perl -w

require 5.005;

use IO::Handle;

use strict;

use FindBin;

use POSIX;

use DBI;

#variables

my($fecha,$rengin,$rengmed,$linea,$var1,$var2,$var3,$dia,$hora,$year,$min,$mes,$re);

my (@inlog,@inmed);

$rengin=0;

$rengmed=0;

$var1 =0;

$var2 =0;

$var3 =0;

main();

exit 0;

sub main {

    # inicio de programa principal

        open(IN2,"intermedio.txt");#poner un 0 en intermedio para correrlo por
        primera vez.

        my $sizeb= readline(*IN2);
```

```

close(IN2);

open(IN, "C:/mrtg/cpu.1.log");

    open(OUT, ">intermedio.txt") || die "lo siento, no puedo crear
intermedio.txt\n";

my $size =(stat('C:/mrtg/cpu.1.log'))[7];#tamaño del archivo en bytes

    print OUT "$size\n";

close(IN);

close(OUT);

if ($size == $sizeb){

die "Programa terminado con exito!!\n";

}

if ($size > $sizeb && $sizeb !=0 ){

    primeralinea();

    query();

}

else {

die "Programa terminado con exito!!\n";

}

# fin de programa principal

}

#++++++

sub db_connect{#inicia connect

    # variables a editar

    my $db_user = "roi";

    my $db_pass = "micontra";

    my $host_name = "localhost";

    my $db_name = "prueba";

```

```

        my $q_string = "DBI:mysql:host=$host_name;database=$db_name";
        return (DBI->connect ($q_string, $db_user, $db_pass,
            {PrintError => 1, RaiseError => 1})) ;

    }#termina connect

#+++++

#+++++

sub query {
    my $query= "LOAD DATA LOCAL INFILE 'C:/Prueba/forMysql.txt' INTO TABLE
ns5000 lines terminated by '\n'";
    my($dbh,$sth);
    $dbh = db_connect() or die("Couldn't execute statement: " . $sth-
>errstr);
    $sth = $dbh->prepare($query) or die("Couldn't execute statement: " .
$sth->errstr);
    $sth->execute();
    $sth->finish;
    $dbh->disconnect
}

#+++++

#+++++

sub primeralinea(){
    my $posi;
    my $posi2;

    #archivo 1
    open(IN,"C:/mrtg/cpu.1.log");
    #my $algo= readline(*IN);#lee la primera linea de algo.log

```

```

read(IN,$var2,16);#tiene 16 caracteres
#print "$var2\n";
$var1 = $var2;

$var1 = substr($var2,0,10);
$var2 = substr($var2,10,16);#(cadena, desplaz, longitud)
$posi = index ($var2," ");#detecta un espacio en blanco en la cadena
leida copia solo el dato
$posi2 =$posi + 1;
$posi2 = index ($var2," ",$posi2);
$var2 =substr($var2,$posi + 1,$posi2 - 1);

$dia = (localtime($var1))[3];
$hora =(localtime($var1))[2];
$year =(localtime($var1))[5];
$year +=1900;
$min = (localtime($var1))[1];
$mes = (localtime($var1))[4];
$mes ++;
$fecha = "$year/$mes/$dia-$hora:$min";
close(IN);

#archivo 2
open(IN,"C:/mrtg/memasignada.1.log");
read(IN,$var3,25);#tiene 20 primeros caracteres
#print "$var3\n";
$var3 = substr($var3,10,25);#(cadena, desplaz, longitud)
close(IN);
$posi = index ($var3," ");#busca en la cadena el espacio en blanco
$posi2 =$posi + 1;

```



```

$posi2 = index ($var3," ",$posi2);
$var3 =substr($var3,$posi + 1,$posi2 - 1);

#archivo 3
open(IN,"C:/mrtg/sesion.1.log");
my $var4;
read(IN,$var4,25);#tiene 15 primeros caracteres
$var4 = substr($var4,10,25);#(cadena, desplaz, longitud)
close(IN);
#$re="$var1/$var2/$var2/$var3/$var4";
$posi = index ($var4," ");
$posi2 =$posi + 1;
$posi2 = index ($var4," ",$posi2);
$var4 =substr($var4,$posi + 1,$posi2 - 1);

#archivo 4
open(IN,"C:/mrtg/172.27.8.140_1.log");
my $var5;
read(IN,$var5,25);#tiene 20 primeros caracteres
$var5 = substr($var5,10,25);#(cadena, desplaz, longitud)
close(IN);
$posi = index ($var5," ");
$posi2 =$posi + 1;
$posi2 = index ($var5," ",$posi2);
$var5 =substr($var5,$posi + 1,$posi2 - 1);

#archivo 5
open(IN,"C:/mrtg/172.27.8.140_2.log");
my $var6;

```

```

read(IN,$var6,25);#tiene 20 primeros caracteres
$var6 = substr($var6,10,25);#(cadena, desplaz, longitud)

close(IN);

$posi = index ($var6," ");

$posi2 =$posi + 1;

$posi2 = index ($var6," ",$posi2);

$var6 =substr($var6,$posi + 1,$posi2 - 1);

#archivo 6

open(IN,"C:/mrtg/172.27.8.140_3.log");

my $var7;

read(IN,$var7,25);#tiene 20 primeros caracteres
$var7 = substr($var7,10,25);#(cadena, desplaz, longitud)

close(IN);

$posi = index ($var7," ");

$posi2 =$posi + 1;

$posi2 = index ($var7," ",$posi2);

$var7 =substr($var7,$posi + 1,$posi2 - 1);

#archivo 7

open(IN,"C:/mrtg/172.27.8.140_4.log");

my $var8;

read(IN,$var8,25);#tiene 20 primeros caracteres
$var8 = substr($var8,10,25);#(cadena, desplaz, longitud)

close(IN);

$posi = index ($var8," ");

$posi2 =$posi + 1;

$posi2 = index ($var8," ",$posi2);

$var8 =substr($var8,$posi + 1,$posi2 - 1);

```

```

#archivo 8
open(IN,"C:/mrtg/172.27.8.140_5.log");
my $var9;
read(IN,$var9,25);#tiene 20 primeros caracteres
$var9 = substr($var9,10,25);#(cadena, desplaz, longitud)
close(IN);
$posi = index ($var9," ");
$posi2 =$posi + 1;
$posi2 = index ($var9," ",$posi2);
$var9 =substr($var9,$posi + 1,$posi2 - 1);

#archivo 9
open(IN,"C:/mrtg/172.27.8.140_6.log");
my $var10;
read(IN,$var10,25);#tiene 20 primeros caracteres
$var10 = substr($var10,10,25);#(cadena, desplaz, longitud)
close(IN);
$posi = index ($var10," ");
$posi2 =$posi + 1;
$posi2 = index ($var10," ",$posi2);
$var10 =substr($var10,$posi + 1,$posi2 - 1);

#archivo 10
open(IN,"C:/mrtg/172.27.8.140_8.log");
my $var11;
read(IN,$var11,25);#tiene 20 primeros caracteres
$var11 = substr($var11,10,25);#(cadena, desplaz, longitud)
close(IN);

```

```

        $posi = index ($var11, " ");
$posi2 =$posi + 1;
$posi2 = index ($var11, " ", $posi2);
        $var11 =substr($var11,$posi + 1,$posi2 - 1);

#archivo 10
open(IN, "C:/mrtg/172.27.8.140_9.log");
my $var12;
read(IN,$var12,25);#tiene 20 primeros caracteres
$var12 = substr($var12,10,25);#(cadena, desplaz, longitud)
        close(IN);
        $posi = index ($var12, " ");
$posi2 =$posi + 1;
$posi2 = index ($var12, " ", $posi2);
        $var12 =substr($var12,$posi + 1,$posi2 - 1);

        open(OUT, ">forMysql.txt")|| die "lo siento, no puedo crear
forMysql.txt\n";
        printf (OUT"%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\n", $fecha,
$var2, $var3, $var4, $var5, $var6, $var7, $var8, $var9, $var10, $var11, $var12);
        close(OUT);

}

#+++++

```

A.3.1 Script CACTI

```

#! /usr/bin/perl -w

use strict;

use MIME::Lite;

use Net::SMTP;

```

```

#variables
my($fortraf,$foroids,$i);

my($rengin,$rengmed,$var_time,$var_trafic,$var3,$posi,$posi2,$varAux,$min,$mes,
$bandera,$fecha);

my (@inlog,@inmed);

my ($subject,$message_body,$msg);

$rengin=0;

$rengmed=0;

$bandera=0;

#inicializacion

$fortraf = 29; #agregar el numero de interfaces q tiene cada equipo esto solo para
ns500

$foroids = 33; #oids mips del ns 500

main();

exit 0;

#++++++

#++++++

sub main {# inicio de programa principal

    leer_rrd();

}

#++++++

#++++++

sub leer_rrd()

    #variables

    my ($name,$time);

    for ($i= 15; $i <= $fortraf; $i++ ) {#for para el trafico fijarse en la secuencia
en la carpeta rra de cacti

        #print "Hola mundo $i\n";

```

```

        if( $i == 516 | $i == 528 ){#meter los id de cada interfaz q no
corresponden!!

                system "/usr/bin/rrdtool last /var/www/cacti/rra/3/$i.rrd >
/var/www/cacti/rra/3/base_leer.txt";

                open(IN, '/var/www/cacti/rra/3/base_leer.txt') or die $!;

                read(IN,$var_time,10); #tiene 10 caracteres

                close(IN);

                #print "$var_time\n";

                $time = time;#tiempo linux de mi pc

                $var3=$var_time;

                $rengin=$var_time;

                $var_time=localtime($var_time);#lo paso mm/dd/yyyy

        if ($time >= ($var3 + 450)){#deside si hay 10 min sin tener refrescamiento

        print "[info] enviando email, Problemas con cacti al adquirir datos!!\n";

        $subject = "Cacti no puede leer los datos\n";

        $message_body = "$var_time ultima actualizacion de cacti, revisar las conexiones de
la pecera posible desconexion de la red";

                &enviar_maill;

                exit 0;

        }

        $varAux=$rengin-300;#para hacer el fetch a la ultima actualización

        system "/usr/bin/rrdtool fetch /var/www/cacti/rra/3/$i.rrd MAX -r 900 -s $varAux -e
$var3 > /var/www/cacti/rra/3/base_leer.txt";

                open(IN, '/var/www/cacti/rra/3/base_leer.txt') or die $!;

                @inlog = <IN>;#todos los renglones de IN estan en inlog

                close(IN);

                chomp($inlog[2]);#elimina el fin de linea "\n"

                $bandera = $inlog[2].' ';#concatena un espacio

                $posi = index ( $bandera," ");#detecta un espacio en balnco en la cadena
leida copia solo el dato

```

```

        $posi2 = $posi + 1;

        $posi2 = index ( $bandera, " ", $posi2);

        $posi2 = $posi2 - $posi;

$var_trafic = substr( $bandera, $posi + 1, $posi2 - 1); #(cadena, desplaz, longitud)
tiene el trafico

if ($var_trafic eq "nan"){

$varAux = &deco("$i"); #llamamos la funcion enviando un elemento

print "[info] enviando email, La interfaz $varAux del Ns-500 esta fuera de linea!\n";

$subject = "Interfaz sin trafico\n";

$message_body = "La interfaz $varAux se encuentra inactiva desde: $var_time\n";

&enviar_maill;}

#print "$var_trafic\n";

$var_trafic = int($var_trafic);

#print "$var_trafic\n";

$var_time = substr( $bandera, 0, $posi - 1); #tiene el tiempo linux en q sucedio
$var_time=localtime($var_time);

if ($var_trafic==0){

    $varAux = &deco("$i"); #llamamos la funcion enviando un elemento

    print "[info] enviando email, La interfaz $varAux del Ns-500 esta fuera de linea!\n";

    $subject = "Interfaz sin trafico\n";

    $message_body = "La interfaz $varAux se encuentra inactiva desde: $var_time\n";

    &enviar_maill;

}

}#del if

}

        for ($i= 31; $i <= $foroids; $i++ ) {#for para los oids genericos tambien en
la misma carpeta

                system "/usr/bin/rrdtool fetch /var/www/cacti/rra/3/$i.rrd MAX -r 900 -s
$varAux -e $var3 > /var/www/cacti/rra/3/base_leer.txt";

                open(IN, '/var/www/cacti/rra/3/base_leer.txt') or die $!;

```

```

@inlog = <IN>;#todos los renglones de IN estan en inlog

close(IN);

chomp($inlog[2]);#elimina el fin de linea "\n"

#print $inlog[2];

#print "\n";

$bandera = $inlog[2].' '#concatena un espacio

$posi = index ( $bandera," ");#detecta un espacio en blanco en la cadena leida
copia solo el dato

    $posi2 =$posi + 1;

    $posi2 = index ( $bandera," ",$posi2);

    $posi2 = $posi2 -$posi;

$var_traffic = substr( $bandera,$posi + 1,$posi2 - 1);#(cadena, desplaz, longitud)
tiene el trafico

    # print "$var_traffic\n";

    $var_traffic = int($var_traffic);

    #print "$var_traffic\n";

#$var_time = substr( $bandera,0,$posi - 1);#tiene el tiempo linux en q sucedio

    #$var_time=localtime($var_time);

    #print "$var_time\n";

    if ( $var_traffic > 80 && $i == 31){

        $varAux = &deco("$i"); #llamamos la funcion enviando un elemento

print "[info] enviando email, El CPU del Firewall esta sobrecargado valor:
$var_traffic!\n";

$subject = "Cpu sobrecargado! valor normal 5% valor actual: $var_traffic\n";

$message_body = "Aviso, el procesador del Firewall esta sobrecargado valor de
lectura: $var_traffic ocurrido"

&enviar_mail1;

    }

if ( $var_traffic >= 250000 && $i == 32){

$varAux = &deco("$i"); #llamamos la funcion enviando un elemento

print "[info] enviando email, El Firewall tiene excesivo numero de sesiones valor:
$var_traffic!\n";

```



```

$subject = "Exceso de sesiones: $var_trafic\n";

$message_body = "Aviso, exceso de sesiones valor:$var_trafic, revisar las interfaces
posible ataque de seguridad"

&enviar_maill;
    }
}

#####

sub enviar_maill {
my $from_address = 'Infocacti_ns500@ERICSSON.com';
my $to_address = 'roiner@gmail.com,rafaelmata357@gmail.com';
my $mail_host = '172.27.10.194';
$msg = MIME::Lite->new (
    From => $from_address,
    To => $to_address,
    Subject => $subject,
    Type =>'multipart/mixed'
) or die "Error creating multipart container: $!\n";

$msg->attach (
    Type => 'TEXT',
    Data => $message_body
) or die "Error adding the text message part: $!\n";

MIME::Lite->send('smtp', $mail_host, Timeout=>60);
$msg->send;
print "Mensaje enviado!\n";
};

```

```

#+++++

#deco para C/u interfaz

#+++++

sub deco() {
    use Switch;
    my $salida;
    my $dato= shift;

    switch ($dato) {
        case 23 { $salida="Traffic blackberry-> fe-0/2/3.6";next}
        case 24 { $salida="SN_OAM_Eth1/2";next}
        case 25 { $salida="ha 2/9";next}
        case 26 { $salida="UNTRUST_Eth2/2";next}
        case 27 { $salida="Eth 3/1/5";next}
        case 28 { $salida="MPBN_OAM_Eth3/2";next}
        case 29 { $salida="ha 1/8";next}
        case 30 { $salida="Eth3/2";next}
        case 31 { $salida="Cpu";next}
        case 32 { $salida="Sesiones";next}
        case 33 { $salida="Traffic Wap-> fe-1/3/1.9";}
    }

    return ($salida);
}

#+++++

```

A.4 Información sobre la empresa

A.4.1 Descripción de la empresa

ERICSSON es una empresa líder en telecomunicaciones la cual tiene como visión ser número uno en una comunicación global^[15]. ERICSSON trabaja en la implementación de nuevas tecnologías y en mejorar el rendimiento de las tecnologías existentes, en proveedores de servicio de telefonía móvil. Una empresa transnacional donde su división en Costa Rica, se encarga principalmente de prestar servicios y vender tecnología de primer nivel.

Trabaja con operadores en los diferentes países, en el caso de Costa Rica es el ICE; Los operadores de telecomunicaciones, buscan nuevas plataformas de servicios para ofrecerlos a sus clientes, además para mejorar las redes de comunicaciones disponibles, ofreciendo un mejor servicio a los usuarios de telefonía móvil.

En esta empresa es de suma importancia que los usuarios de telefonía móvil, puedan comunicarse con cualquier persona que deseen, sin limitaciones de tiempo y horarios.

El servicio que le brinda ERICSSON al ICE es el de operación y mantenimiento de la red de telefonía móvil GSM, esta red debe de tener una disponibilidad del 100% para los servicios de voz y datos que así los usuarios puedan comunicarse sin interrupción del servicio.

A.4.2 Descripción del departamento

El departamento de OYM, cuyo jefe es Freddy Gómez Aparicio, es el departamento encargado de resolver problemas de la red, en lo referente a mantenimiento y desempeño, mientras exista un contrato con el proveedor de servicios para dicho fin.