



Universidad de las Ciencias Informáticas

Trabajo de Diploma para optar por el título de:

Ingeniero en Ciencias Informáticas

Título:

Solución de configuración y extensión de agentes SNMP para el monitoreo de sistemas

Autor: Marcos Michel Martínez Pérez

Tutor: Ing. Adrián Misael Peña Montero

La Habana, Junio 2013

Año 55 de la Revolución

Declaración de Autoría

Declaro que soy el único autor de este trabajo y autorizo a la Facultad 6 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Marcos Michel Martínez Pérez

Ing. Adrián Misael Peña Montero

Firma de Autor

Firma de Tutor

Datos de contacto

Ing. Adrián Misael Peña Montero

Universidad de las Ciencias Informáticas, Habana, Cuba

e-mail: ampena@uci.cu

Agradecimientos

Agradecer genuinamente es un acto muy poderoso, decir gracias es reconocer que hemos recibido algo, por eso doy mi más profundo agradecimiento a todos aquellos profesores, amigos y familiares que de una forma u otra hicieron posible el cumplimiento de un sueño, hoy hecho realidad.

A mi tutor, Adrián Misael Peña Montero por ser fiel poseedor de la idea que hoy se hace palpable.

A Osman, Suly y a todos los integrantes del grupo 6508, Dalilys, Amarilys, Rosa, Darlys, Lieny, Audrey, Raydel Wilson (Popi), Raúl Alejandro, Raúl Marinas, Reisel, Carlos, Marcos Raúl, Jeovany (el poti), a todos gracias porque a pesar de ser un insoportable en el fondo me aprecian y valoran como soy.

A Yadira Rodríguez Pérez por ser como una hermanita chiquita para mí, tu carisma y tu sencillez cultivaron este cariño que te tengo. Gracias por ser quien eres, sea de tu conocimiento que yo te aprecio más de lo que te imaginas.

A todos mis compañeros del apartamento 97103, René, Luis Angel, Sandy, Osvaldo, Darlon, Alejandro, Tomás, Chávez, Félix, a todos gracias por compartir 3 años de convivencia, y pasar juntos por momentos bien difíciles en la carrera.

A mis amigos iniciadores del Portal Gladiadores, Héctor Alcides, Alejandro Lara, a colaboradores Dieter, Odiel, al profe Yoenis, Eduar, Raudel, Yosvani, y demás integrantes.

A todos mis profesores del departamento PostgreSQL del Centro de Tecnologías de Gestión de Datos.

Agradecer a Yunior, Juan Daniel, Alejandro, Juana, Neni, Brito, por estar siempre cercanos a mí y compartir cada momento que me han dedicado, en los que su comprensión ha sido muy importante, brindándome siempre mucho cariño, afecto y sobre todo una amistad incondicional.

A Tany Fernández Armas, porque te considero una hermana, sea todo el cariño del mundo en un solo beso para ti.

A Daily, Cary, Caridad, Nora, por ser parte de esta historia que un día como hoy abre las puertas al futuro.

A Carlos Pena, y su esposa Beatriz, por brindarme su mano como sostén cuando a las puertas del vacío me vi, su amistad, dedicación e incondicionalidad me fueron suficientes para permitirme una nueva oportunidad.

A Conrado Martínez, por brindarme su amistad y hacer cumplir una de mis necesidades convertidas en sueño, además de soportar mis pesadeces.

Al Tío Leoncio Matos y la Tía Liliana Quintana porque de veras descubrí en ellos un par de amigos incomparables.

A Raúl, al teacher Julián.

A todos los trabajadores y amigos del Joven Club # 3 del municipio Sagua la Grande.

A mis vecinos, Elvira, Oriola, Cary, Elisa (madre e hija), Pedro (el niño), Papo, Yunior, Elvis.

Quiero dar un agradecimiento especial a mis padres, Reinaldo Martínez y Sandra Pérez, en su camino para forjar el futuro, dejaron caer una semilla de amor y gratitud que hoy da sus

frutos, una semilla transformada en un hombre de bien, entregará el honor y el sentir propio para hacer cumplir el sueño de otros que también andan por los caminos del saber.

Y a todo el que no valora mi conocimiento y mi facilidad para aprender lo que me gusta, le digo, que en junio del 2013 ya soy Ingeniero en Ciencias Informáticas, gracias por todo, aún hay mucho camino por recorrer.

A todos gracias, de veras han sido los mejores 5 años de mi vida, 5 años que ya no volverán, 5 años incomparables que una vez llegaron con mucha felicidad y risas y hoy con tristeza y lágrimas se marchan, 5 años en los que llegaron unas amistades y otras se marcharon, 5 años en los que conocí personas con falta de carácter, de honestidad, de sinceridad y sin embargo son de los mejores amigos que he conocido, 5 años de momentos difíciles, tragos amargos, indecisiones, reproches, traición, desconfianza, pero también 5 años en los que la vida me dio una y otra vez una nueva oportunidad para salir adelante, 5 años en los que tuve que tomar grandes decisiones, dejar atrás mis convicciones y renunciar a una vida anterior, regresar nuevamente al presente y cambiar mi forma de pensar, de ser, de hablar y hasta de actuar para sentar las bases de lo que es y será un futuro desconocido.

Dedicatoria

Quiero dedicar este trabajo de diploma especialmente a mi padre Reinaldo Martínez Cuellar, y darle un inmenso agradecimiento por todo lo que ha hecho por mí, por estos 12 años incomparables, magníficos, gracias a estos 12 años hoy soy el hombre que represento. Con tan solo 12 años me diste la posibilidad de elegir lo que era bueno para mí, lo hice y espero no haberme equivocado. Hoy 18 de junio del 2013 una vez más la vida me demuestra que padre como tú no existe, hay que mandarlo a fabricar, y por ser fabricado quedará con imperfecciones en caso de lograrse.

Gracias, todo el agradecimiento del mundo y todas las hojas de este trabajo de diploma no serían suficientes para escribir la inmensidad de agradecimientos que podría otorgarte.

Gracias por ser el hombre que eres.

Gracias por ser la persona que eres.

Gracias por ser el mejor padre del mundo.

Gracias por ser mi padre. . .



*Si avanzo sígueme, si me detengo empújame,
si retrocedo mátame...*

Ernesto Guevara

Resumen

Al realizar tareas de monitoreo es común que se necesite acceder a través de los agentes SNMP, siglas de Simple Network Management Protocol (Protocolo Simple de Administración de Red), a parámetros que no están definidos en las MIB. Para lograr acceder a ellos, es necesario añadir nuevos objetos a la MIB, siglas de Management Information Base (Base de Información de Administración), de los sistemas en cuestión y reconfigurar los agentes SNMP para que puedan acceder a los nuevos parámetros.

Este trabajo de diploma se propone brindar como propuesta un Sistema de Gestión de Extensiones para Agentes SNMP, que facilite a los administradores de sistemas la gestión de extensiones que adicionan nuevas funcionalidades a los agentes. Además estos administradores contarán con un marco de trabajo que le permitirá desarrollar nuevas funcionalidades con el objetivo de ampliar las opciones de monitoreo hacia otros sistemas, para los cuales de forma nativa no se cuenta con el soporte necesario para su control mediante el protocolo SNMP.

La investigación surge de la necesidad que se tenía de realizar actividades de monitoreo sobre servidores de bases de datos como PostgreSQL, para el cual el protocolo SNMP de forma nativa no contiene el soporte necesario para monitorear este servidor. Luego del desarrollo de este trabajo de diploma se diseñó un marco de trabajo para el desarrollo de nuevas extensiones para agentes SNMP, un repositorio para el almacenamiento de las nuevas extensiones o las ya existentes y se implementó una aplicación que permite gestionar el repositorio de extensiones así como los objetos que están siendo monitoreados en un determinado equipo.

Para el desarrollo de esta solución se seleccionó XP, siglas de Extreme Programming (Programación Extrema), como metodología de desarrollo usando el UML, siglas de Unified Modeling Language (Lenguaje de Modelado Unificado), como lenguaje de modelado y como herramienta CASE el Visual Paradigm. Se utilizó C y Java como lenguajes de programación, Liferay como gestor de contenidos y el IDE fue Eclipse.

Palabras claves: Protocolo, Sistema de Gestión de Extensiones, SNMP.

Índice de contenido

<i>Declaración de Autoría</i>	I
<i>Datos de contacto</i>	II
<i>Agradecimientos</i>	IV
<i>Dedicatoria</i>	VI
<i>Resumen</i>	VIII
<i>Introducción</i>	1
<i>Capítulo 1- Fundamentación teórica</i>	5
1.1. Formas de monitoreo	5
1.2. SNMP	6
Surgimiento de SNMP	6
Componentes de SNMP	7
NET-SNMP	7
1.3. MIB.....	9
Introducción a la MIB	9
De MIB-I a MIB-II	9
Estructura de la MIB	10
Estándar para la definición de la MIB.....	12
1.4. Implementación de módulos para NET-SNMP	12
Definición de una MIB.....	13
Creación de objetos MIB.....	14
Implementación del objeto para el agente.....	15
Integración de nuevos módulos al agente SNMP.....	16
1.5. Almacenamiento y gestión de módulos para NET-SNMP	17
Formas de almacenamiento de módulos	17
1.6. Desarrollo de aplicaciones web	18
Análisis de CMS para la creación de portales	18
Desarrollo basado en portlets	19
1.7. Metodología y herramientas propuestas.....	20

Lenguaje de modelado	21
Herramienta de modelado.....	22
Lenguaje de programación	22
Sistema de gestión de bases de datos.....	24
Entorno de desarrollo integrado	24
Conclusiones del capítulo	25
<i>Capítulo 2- Diseño e implementación de la solución</i>	<i>26</i>
2.1. Descripción de la solución	26
Marco de trabajo para el desarrollo de nuevos módulos	26
Repositorio para el almacenamiento de los módulos	26
Aplicación para la gestión de módulos	27
2.2. Diseño de la aplicación.....	30
Historias de Usuario.....	30
Lista de reserva del producto	32
Plan de iteraciones	35
Tarjetas CRC	36
Diagrama de clases	37
Patrones de diseño.....	42
Patrones GRASP	42
2.4. Implementación de la aplicación.....	45
Tareas de ingeniería	45
Estándares de codificación	46
2.5. Conclusiones del capítulo	49
<i>Capítulo 3- Validación de la aplicación.....</i>	<i>51</i>
Tipos de pruebas	51
Técnicas de prueba	52
Método de prueba de caja negra	53
3.1. Flujo de trabajo para el desarrollo de pruebas.....	54
3.2. Análisis de los resultados de las pruebas	57

Conclusiones generales.....58
Recomendaciones59
Referencias Bibliográficas.....60
Bibliografía.....63
Glosario de términos.....65

Índice de Figuras

Figura 1 - Diagrama del árbol MIB	11
Figura 2- Diagrama Entidad-Relación de los metadatos para categorías y extensiones	27
Figura 3- Diagrama de clases	38
Figura 4- Diagrama de clases (Patrón - MVC)	41
Figura 5- Diagrama donde se evidencia el patrón Experto.....	43
Figura 6- Diagrama donde se evidencia el patrón Creador	43
Figura 7- Diagrama donde se evidencia el patrón Bajo Acoplamiento.....	44
Figura 8- Diagrama donde se evidencia el patrón Alta cohesión.....	45
Figura 9- Ejemplo de sentencias package e import.....	47
Figura 10- Ejemplo de sentencia donde se limita la longitud de caracteres por línea.....	47
Figura 11- Ejemplo de un comentario de bloque.....	48
Figura 12- Ejemplo de un comentario de línea.....	48
Figura 13- Ejemplo de un comentario de remolque.....	49
Figura 14- Ejemplo de un comentario de fin de línea	49
Figura 15- Caso de Prueba "Adicionar categoría"	55
Figura 16- Variables para el caso de prueba "Adicionar categoría"	55
Figura 17- Resultado de las pruebas por iteración.....	57

Índice de Tablas

Tabla 1 - Historia de Usuario "Adicionar categoría"	31
Tabla 2- Lista de reserva del producto	32
Tabla 3- Plan de iteraciones	36
Tabla 4- Tarjeta CRC (Clase: paquetes)	37
Tabla 5- Tarea de Ingeniería "Implementar adicionar categorías"	46
Tabla 6- Tabla de ejemplo de no conformidades detectadas	56

Introducción

Con el surgimiento de las Tecnologías de la Información y las Comunicaciones (TIC), la sociedad en general se ha visto beneficiada con los servicios y las facilidades de herramientas y aplicaciones que esta brinda. El uso cada vez mayor de los sistemas de información ha provocado el surgimiento de sistemas informáticos dirigidos íntegramente a la tarea de monitorear las actividades que en estos equipos se realizan. Unido a la necesidad de monitorear los sistemas de información o servicios orientados a las telecomunicaciones y la informática surgen entonces las primeras aplicaciones cuya actividad fundamental está orientada al monitoreo de estos sistemas.

Con el creciente uso de las tecnologías en Cuba surge la necesidad de monitorear los sistemas proveedores de información. Para lograrlo se utilizan y desarrollan nuevas soluciones que permitan cumplir con esta actividad.

Para la Universidad de las Ciencias Informáticas resulta de vital importancia realizar tareas de monitoreo sobre los sistemas con los que cuenta, con el objetivo de controlar parámetros que le permitan obtener información sobre el estado y funcionamiento de los equipos, logrando además determinar fortalezas y/o debilidades que faciliten trazar una estrategia recuperadora o preventiva para los daños que puedan producirse en estos equipos, así como anomalías que pudieran afectar la disponibilidad de sus sistemas. Para el logro de estos objetivos son utilizadas soluciones informáticas que garantizan un control eficiente, además de la implementación de nuevas soluciones dependiendo de las necesidades de monitoreo existentes.

Conocer el estado de cada uno de los sistemas que componen una red es una de las principales tareas que realizan las personas encargadas de realizar la administración de redes. Para realizar esta tarea es muy común que se utilicen algunas de las soluciones de monitoreo que existen en la actualidad. Estas soluciones están compuestas por un grupo de equipos y aplicaciones que de conjunto realizan las actividades necesarias para llevar a cabo el control de la red y los dispositivos que la conforman.

La base de las soluciones de monitoreo es el chequeo de parámetros en los sistemas que son objeto de monitorización. En la bibliografía consultada para el desarrollo de esta investigación fueron encontradas dos formas básicas de realizar el chequeo de parámetros.

La primera de ellas es la simulación de clientes y se utiliza principalmente para monitorear servicios. La segunda es la utilización del protocolo SNMP que se enfoca en monitorear elementos de hardware.

SNMP es un protocolo ampliamente utilizado para el intercambio de información de gestión entre los dispositivos de la red que fue concebido para administrar grandes redes de forma simple y efectiva. Como parte de las tareas de gestión que permite el protocolo, está la obtención de parámetros de estado de los dispositivos que se gestionan. Esta facilidad del protocolo SNMP es lo que ha permitido su amplia utilización para el monitoreo de sistemas.

Para poder utilizar SNMP en las tareas de chequeo de parámetros en los sistemas que son monitoreados, es necesario que en estos esté instalado y correctamente configurado un agente SNMP. A través del agente SNMP instalado en el dispositivo es posible acceder a los valores de los parámetros que son chequeados en estos sistemas.

Para obtener los valores de los parámetros que son chequeados en los dispositivos mediante agentes SNMP, estos utilizan las MIB. Las MIB están organizadas jerárquicamente y en ellas se definen todos los objetos que pueden ser gestionados en un sistema. Debido a esto, a través de los agentes SNMP solo se puede acceder a los parámetros que previamente estén definidos en las MIB instaladas en los dispositivos que son monitoreados.

En el monitoreo de sistemas es común que se necesite acceder, a través de los agentes SNMP a parámetros que no están definidos en las MIB lo que impide que los nuevos sistemas sean monitoreados. Para lograr comprobar las actividades que estos equipos realizan es necesario entonces añadir nuevos objetos a la MIB de los sistemas en cuestión y reconfigurar los agentes SNMP para que puedan acceder a los nuevos parámetros.

Para dar solución a la problemática planteada anteriormente la presente investigación se plantea el siguiente **problema a resolver**: ¿Cómo facilitar la extensión y configuración de agentes SNMP para el monitoreo de equipos de interconexión y servidores?

El **objeto de estudio** está enmarcado en: El monitoreo de sistemas mediante agentes SNMP.

El **campo de estudio** radica en: Configuración y extensión de agentes SNMP.

Para dar solución al problema planteado se define como **objetivo general**: Proveer a los administradores de sistemas de monitoreo una solución que facilite la extensión y configuración de agentes SNMP para el monitoreo de equipos de interconexión y servidores.

Para el cumplimiento del objetivo de la investigación se definen los siguientes **objetivos específicos**:

1. Realizar un estudio bibliográfico sobre la configuración y extensión de agentes SNMP para el monitoreo de sistemas.
2. Definir un marco de trabajo para el desarrollo de módulos que permiten extender los agentes SNMP.
3. Desarrollar una aplicación para facilitar la gestión de las extensiones y la configuración de los agentes SNMP.
4. Validar la solución.

Con vistas a cumplir los objetivos específicos se definen las siguientes **tareas de investigación**:

1. Revisión de la bibliografía existente relacionada con el protocolo y la extensión de los agentes SNMP.
2. Definición del marco de trabajo que permitirá la extensión de los agentes SNMP.
3. Desarrollo del repositorio que permitirá gestionar las extensiones creadas o las nuevas que son creadas.
4. Diseño de la aplicación que gestionará las extensiones para los agentes SNMP.
5. Implementación de la aplicación que gestionará las extensiones para los agentes SNMP.
6. Selección de las técnicas y métodos para la validación de la solución.
7. Realizar la validación de la solución.

Para realizar la investigación serán utilizados los siguientes métodos científicos:

Histórico-Lógico

Se realizará una investigación que permitirá analizar las tendencias actuales sobre el uso del protocolo SNMP para el monitoreo de sistemas, así como la existencia de aplicaciones y soluciones informáticas que hagan uso de esta herramienta con objetivos similares.

Analítico–Sintético

Será utilizado para realizar el análisis de toda la información que se obtenga durante todo el proceso de desarrollo de la investigación y el mismo permitirá obtener una síntesis de los contenidos fundamentales que puedan resultar relevantes para este trabajo de diploma.

La presente investigación se encuentra estructurada en tres capítulos divididos de la siguiente manera:

Capítulo 1- Fundamentación teórica

En este capítulo se realiza toda la fundamentación teórica de la investigación, en la que se abordan los principales conceptos asociados al problema general y se realiza un estudio de las herramientas existentes que permiten seleccionar las que serán utilizadas para realizar la implementación de la solución.

Capítulo 2- Diseño e implementación de la solución

En este capítulo se realiza el diseño del marco de trabajo que permite extender y configurar los agentes SNMP. Se diseña el repositorio para el almacenamiento de las extensiones, se diseña e implementa la aplicación que permite gestionar el repositorio de extensiones y se desarrollan extensiones que extienden las funcionalidades del protocolo SNMP.

Capítulo 3- Validación de la solución

Se diseñan y aplican las pruebas que le serán realizadas a la aplicación que permitirá gestionar el repositorio de extensiones encargadas de extender las funcionalidades del protocolo SNMP.

Capítulo 1- Fundamentación teórica

Luego de mencionar anteriormente todo lo referente al protocolo y la extensión de agentes SNMP es necesario esclarecer algunos conceptos para facilitar el entendimiento del contenido de la presente investigación. A continuación se muestra una descripción de los principales conceptos asociados al contenido y posterior desarrollo.

1.1. Formas de monitoreo

Con el objetivo de controlar el estado y funcionamiento de los sistemas informáticos que permiten la disponibilidad de la información, son realizadas tareas de monitoreo, para la detección temprana de posibles fallas en los dispositivos de interconexión y servidores. Entre las formas existentes de realizar actividades de monitoreo se encuentra el chequeo de parámetros. Este tipo de chequeo se realiza para obtener información específica sobre determinadas variables que miden el funcionamiento del objetivo.

Entre las ventajas de realizar el chequeo de parámetros se encuentran, la posibilidad de obtener información actualizada y precisa sobre áreas específicas de los equipos, y la facilidad de obtención al no agregar tráfico adicional a la red. Existen varias formas para realizar estas actividades de control entre las que se destacan, la simulación de clientes y a través del protocolo SNMP.

La simulación de clientes se basa en la utilización de archivos de código o scripts que realizan la tarea de conectar con el equipo deseado a través del propio servicio instalado en el equipo para consultar la información que se desea, mientras que la utilización del protocolo SNMP está basada en la obtención de datos estadísticos de los dispositivos de interconexión y servidores mediante el intercambio de información de gestión entre ellos. Al mismo tiempo entre sus funcionalidades más importantes se encuentra la capacidad que posee de generar alertas en caso de ocurrir algún proceso inusual en algún equipo.

Para el desarrollo de esta solución será utilizado del protocolo SNMP como vía para la realización de las actividades de monitoreo, dada sus característica que posibilitan agregar nuevas funcionalidades al protocolo sin necesidad de instalar y configurar o recompilar el código fuente del protocolo. Mediante esta vía se puede extender los agentes SNMP que ya se encuentren en funcionamiento sin que se vea interrumpida su actividad.

1.2. SNMP

SNMP es un protocolo de la capa de aplicación de TCP/IP que constituye un estándar para la administración de redes, es muy sencillo pero suficientemente potente para administrar redes heterogéneas y fue diseñado para facilitar el intercambio de información de administración entre los dispositivos de la red. Usando SNMP se puede acceder a la información que permite administrar o manejar la red, permitiendo a los administradores de red comprobar el rendimiento de la misma así como buscar y solucionar problemas.[1]

Surgimiento de SNMP

SNMP se originó inicialmente en la comunidad de Internet con la finalidad de permitir la administración de redes basadas en TCP/IP.

Antes del surgimiento del protocolo SNMP, los productos de administración de estas redes eran fabricados por los propios fabricantes de los sistemas. Esto complicaba grandemente la gestión de las redes heterogéneas ya que al existir variedades de fabricantes en los sistemas que la componen, era preciso entonces obtener los programas de gestión de cada fabricante y en la mayoría de los casos estos sistemas no se podían utilizar libremente. [2]

Para que la gestión de una red tenga éxito, esta debe basarse en un software simple, pequeño y de bajos costos con el objetivo de que pueda ser instalado en todos los dispositivos de la red, surgiendo entonces SNMP como protocolo para estos fines. SNMP fue publicado inicialmente en 1989 y las primeras aplicaciones llegaron al mundo de las redes en 1990.

SNMPv2 hace su aparición en Mayo de 1993 añadiendo nuevas funcionalidades y su última versión SNMPv3 llega finalmente en el año 2002 añadiendo cambios significativos y por primera vez aspectos de seguridad.

SNMP también se ha extendido para cubrir otros sistemas y algunos protocolos propietarios, convirtiéndose en el estándar más ampliamente utilizado en la gestión de las redes de hoy. Mediante el protocolo SNMP pueden ser monitoreados varios parámetros en los equipos de interconexión.[3]

En un escenario de ejemplo, una PC conocida como cliente SNMP, donde se encuentra instalada la aplicación que utilizará el protocolo SNMP para obtener información de administración de cualquier

dispositivo de red ya sea (switch, router o incluso Impresoras, u otra pc), hace una petición de información al agente SNMP del que logra obtener por ejemplo, el uso de CPU, RAM, disco duro, etc.

Componentes de SNMP

SNMP se compone de 3 partes indispensables para su funcionamiento: consola de administración (NMS), agente y MIB.

Agente

Es un programa instalado en un dispositivo de red, ya sea una estación de trabajo, un switch, un router o cualquier dispositivo administrable mediante SNMP, en el que se encuentra instalado el protocolo.

Es el encargado de almacenar la información de administración del dispositivo administrado, ya que el agente reside en él mismo. Posee un conocimiento local de la información de administración, la cual es traducida a un formato compatible con SNMP y organizada en jerarquías.[1]

MIB

Es una base de datos de objetos administrados, que son accesibles por el agente y manipulados vía SNMP para lograr la administración de la red.

Las MIB constituyen una colección de información que se encuentra organizada jerárquicamente y solo se puede acceder a los objetos a través de SNMP.[1]

Consola de administración

Es un programa utilizado en una estación de trabajo que tiene la habilidad de indagar en los agentes SNMP. Es utilizada comúnmente para realizar tareas de administración de forma remota en el dispositivo objetivo. Permite igualmente la obtención de la información de administración del objetivo seleccionado.

NET-SNMP

NET-SNMP es un paquete de aplicaciones que de conjunto son utilizadas para implementar SNMPv1, SNMPv2, SNMPv3 usando IPV4 o IPV6. Este conjunto de aplicaciones incluye:[4]

1. Aplicaciones para línea de comandos que permiten:

- Obtener diferentes informaciones de un dispositivo mediante solicitudes individuales (snmpget, snmpgetnext) o solicitudes múltiples (snmpwalk, snmptable, snmpdelta).[4]
 - Manipular la información de configuración de un determinado dispositivo a través de (snmpset).[4]
 - Obtener un grupo fijo de informaciones de un dispositivo capaz de soportar SNMP mediante (snmpdf, snmpnetstat, snmpstatus).[4]
 - Convertir entre los distintos formatos numéricos y textuales según el OID de las MIB y mostrar el contenido y la estructura de los mismos mediante (snmptranslate).[4]
2. Un navegador gráfico de MIB llamado (tkmib) que utiliza como lenguaje Tk/perl.
 3. Una aplicación de demonio para el recibo de notificaciones SNMP mediante (snmtrapd).[4]
 4. Una agente extensible para responder a las consultas realizadas a través de SNMP que permite gestionar la información (snmpd). Incluye además soporte integrado para una amplia gama de módulos de información a través de MIB's y que pueden ser ampliados mediante módulos de carga dinámica, scripts y comandos externos. Además cuenta con una biblioteca para el desarrollo de nuevas aplicaciones SNMP, en C y en las API de Perl.[4]

Los comandos SNMP son las sentencias mediante las cuales la aplicación cliente hace las peticiones de información al agente SNMP. A continuación solo se describe uno de ellos, el cual será utilizado por la solución para el intercambio de información entre el agente y la aplicación.

snmpwalk: es una pequeña aplicación instalada en el agente que utiliza una búsqueda de forma recursiva por cada uno de los nodos del árbol MIB en busca del parámetro o variable especificado en la consulta por el cliente SNMP. Posee la capacidad de devolver mensajes si ocurre alguna excepción en el proceso.

A pesar de existir más comandos útiles para el trabajo con el protocolo SNMP, solo será utilizado en la solución, snmpwalk, ya que permite recorrer el árbol MIB completamente, mostrando incluso las MIB y los objetos que se encuentran activos en el momento en que se realiza la consulta. Snmpwalk viaja hoja por hoja del árbol obteniendo la información solicitada. El recorrido del árbol mediante este comando puede realizarse desde un equipo externo o desde el propio equipo.

1.3. MIB

Introducción a la MIB

Una MIB constituye una colección de información que se encuentra organizada jerárquicamente. Es parte de la gestión de red definida en el modelo OSI siglas de Open System Interconnection (Modelo de Interconexión de sistemas abiertos), perteneciente a la capa de aplicación. Define las variables usadas por el protocolo SNMP para supervisar y controlar los componentes de una red.

A ellas solo se puede acceder a través del protocolo SNMP. Un objeto MIB posee cualquier número de características específicas de un dispositivo administrado. Los objetos administrados están compuestos por una o más instancias de objetos MIB que en esencia son variables.

Estos objetos administrados se definen bajo dos tipos específicos: escalares y tabulares. Los objetos escalares hacen referencia a un solo objeto mientras que los objetos tabulares definen múltiples instancias de objetos que se relacionan y se agrupan en las tablas MIB.[5]

La MIB está definida en la RFC 1213, siglas de Request For Comments (Petición de comentarios), y se puede afirmar que constituye una base de datos jerárquica organizada en forma de árbol. Cada objeto de una MIB constituye una hoja del árbol. Un objeto puede almacenar números, aunque en ocasiones contienen cadenas de texto o estructuras más complejas como tablas.[5]

De MIB-I a MIB-II

La IAB siglas de Interactive Advertising Bureau (Asociación que representa al sector de la publicidad en España), recomienda una doble estrategia para el desarrollo de estándares de internet y la gestión de redes. SNMP se utiliza en un corto plazo y el uso del modelo OSI para la gestión de la red fue pensado para un largo plazo. Sobre esta base fueron creados dos documentos que definían el SMI siglas de Structure of Management Information (Estructura de Administración de Información), y la MIB. Los documentos fueron redactados pensando en la compatibilidad del SNMP y el modelo OSI. Sin embargo esta estrategia en el corto plazo fue exitosa.

El informe realizado por el grupo de Administración de redes para estos fines, declaró que el SNMP y el modelo OSI para la gestión de las redes no cumplían con todos los requisitos que se necesitaban, por tanto, ambos fueron suspendidos en aquel momento dando lugar a la redacción de una MIB II. La MIB I

fue la primera versión publicada como RFC 1156. La MIB II es un súper conjunto que contiene a la MIB I con algunos objetos adicionales y grupos que están definidos en la RFC 1213.[6]

Estructura de la MIB

Como se puede apreciar en la *Figura 1 - Diagrama del árbol MIB* cada nodo que es representado en el árbol MIB puede ser accedido mediante un número entero único llamado OID, siglas de Object Identifier (Identificador de objeto), que lo define dentro del árbol. Para conformar los OID el Nodo Root conocido como la raíz del árbol es accedido mediante un punto (.). Los objetos de la MIB también pueden llamarse por su nombre seguido de un punto (.) al igual que en la referencia numérica.

Ejemplos de acceso numérico a dos objetos de la MIB:

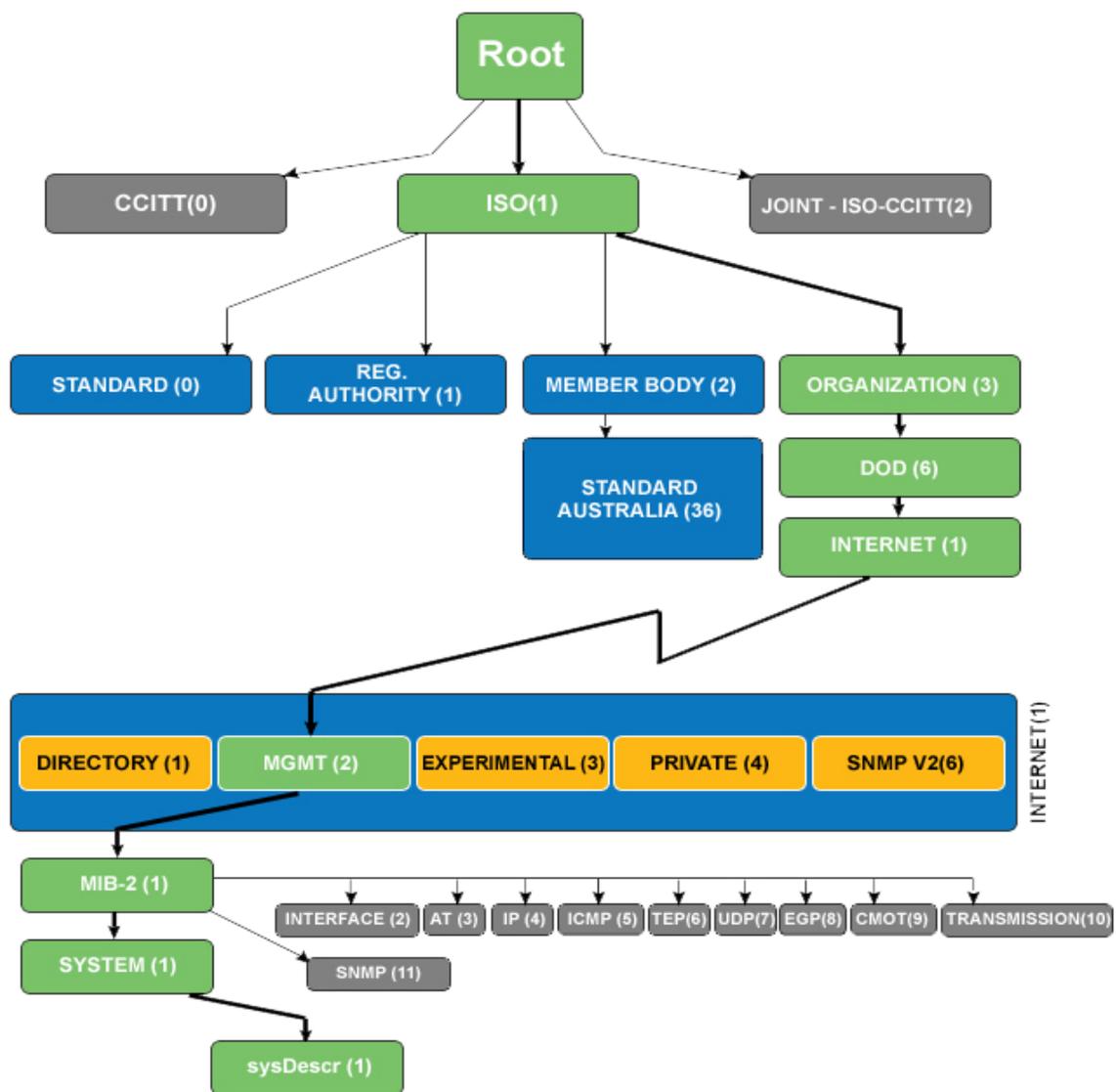
Par acceder a la MIB II el OID sería: .1.3.6.1.2.1

Par acceder a sysDescr el OID sería: .1.3.6.1.2.1.1.1

Ejemplos de acceso por nombres a dos objetos de la MIB:

MIB II: .ISO.ORG.INTERNET.MGMT.MIB

sysDescr sería: .ISO.ORG.INTERNET.MGMT.MIB.SYSTEM.sysDescr



Tomada de: <http://www.manageengine.com>

Figura 1 - Diagrama del árbol MIB

Algunos de los nodos que conforman la MIB II son:

- **System:** de este nodo extienden los objetos que brindan información genérica del sistema gestionado.
- **Interfaces:** en este grupo está la información de las interfaces de red presentes en el sistema. Incorpora estadísticas de los eventos ocurridos en las mismas.

- **At** (address translation o traducción de direcciones). Este nodo es obsoleto, pero se mantiene activo en las versiones actuales para preservar la compatibilidad con la MIB I. En este se almacenan las direcciones de nivel de enlace correspondientes a una dirección IP.
- **Ip**: en este grupo se almacena la información relativa a la capa IP, tanto de configuración como de estadísticas.
- **Icmp**: en este nodo se almacenan contadores de los paquetes ICMP entrantes y salientes.
- **Tcp**: en este grupo está la información relativa a la configuración, estadísticas y estado actual del protocolo TCP.
- **Udp**: en este nodo está la información relativa a la configuración y estadísticas del protocolo.[7]

Estándar para la definición de la MIB

Para definir cada objeto MIB se usa un estándar conocido como SMI el cual establece reglas para nombrar cada objeto, naturaleza de su valor, control de acceso al objeto, tipo de respuesta a una pregunta vertical y descripción. Las MIB y los objetos que éstas contienen son situados en el árbol MIB siguiendo determinadas normas. Estas definiciones se realizan siguiendo un lenguaje de la ISO, siglas de International Organization for Standardization (Organización Internacional de Estandarización), llamado ASN-1 siglas de Abstract Syntax Notation (Sintaxis de notación abstracta). SMI especifica dos maneras de localizar un objeto que se especifican en el epígrafe *Estructura de la MIB*.

1.4. Implementación de módulos para NET-SNMP

Básicamente NET-SNMP consta de tres procesos, formas o técnicas independientes mediante los cuales se pueden implementar y extender las funcionalidades del protocolo, con el objetivo de monitorear otros sistemas más allá de los previstos por los fabricantes. Estas formas o técnicas se nombran y explican a continuación:

Implementación de plugins en PERL

Esta técnica permite incrustar código perl en el agente snmpd. Tiene una función parecida al módulo mod_perl de apache que permite incrustar código perl directamente en el servidor.[8]

Implementación de Shell Scripts

El agente Net-SNMP proporciona una extensión MIB llamada (NET-SNMP-MIB-EXTEND) que se puede utilizar para consultar scripts de shell arbitrarios. Para especificar la secuencia de comandos de shell a ejecutar, son utilizadas las directivas que extienden en el archivo snmpd.conf. Una vez definido esto, el agente deberá proporcionar el código de salida y cualquier salida del comando será a través de SNMP.[8]

Implementación de objetos compartidos que pueden ser cargados dinámicamente

Son archivos binarios que pueden ser cargados dinámicamente por el agente y se ejecutan como parte del propio agente. Esto difiere del concepto introducido en la escritura de un módulo MIB, en que la extensión se encuentra en su propio archivo binario y es cargado por el agente durante la ejecución. Esto aumenta la flexibilidad de añadir nuevas funcionalidades y/o nuevas MIB después de que el agente ha sido compilado e instalado en el dispositivo.[8]

Para el desarrollo de la solución se utilizará la escritura de módulos compartidos que pueden ser cargados dinámicamente, por la facilidad que posee de añadir nuevas funcionalidades al protocolo sin la necesidad de recompilar su código fuente o reinstalarlo en caso de que proceda de otra ubicación. Para habilitar una nueva funcionalidad, la instalación del protocolo por cualquiera de las vías existentes debe realizarse con la opción de objetos compartidos habilitada, la cual permite activar la capacidad del protocolo para trabajar con los módulos de carga compartida.

Seguidamente al proceso descrito anteriormente, el cual debe ser garantizado antes de comenzar a trabajar con los objetos de carga compartida, se debe pasar a la definición de las MIB que se utilizarán para la creación de la nueva funcionalidad. Luego se definen los tipos de datos y los objetos que serán monitoreados, se procede con la implementación de los módulos que permiten la obtención de la información y finalmente se integra dicho módulo al agente SNMP.

Definición de una MIB

El proceso de creación de una MIB se inicia con la definición del nombre que la identificará ante otras en el funcionamiento del protocolo. Son creadas en archivos comunes de texto, escritos en lo que se suele llamar texto plano, no necesitan de una codificación especial ni de alguna aplicación externa que lo traduzca a otra forma de interpretación.

Normalmente se define el mismo nombre que posee el archivo con el nombre interno que contendrá la MIB, dado que el sistema lo interpreta así.

Definición de tipos de objetos

En el RFC 1155 se definen tres tipos de objetos que nos indican el modelo de información que puede ser devuelto por cualquiera de los módulos definidos en la MIB asociada a la operación que se desee realizar.[9]

Los tipos de objetos son:

Objetos universales[9]

- Objetos que representan números enteros.
- Objetos que representan texto.
- Cuando carecen de valor.
- Para nodos estructurales.
- Para objetos de tipo arreglo.

Objetos de tipo aplicación[9]

- Objetos para representar direcciones IP.
- Para representar contadores.
- Objetos utilizados para medición del tiempo.

Objetos de tipo tablas[9]

- Son un tipo de dato estructurado.
- Se definen usando Secuencia , Secuencia de y la cláusula index.

Creación de objetos MIB

La definición de una MIB se inicia con la sección donde se especifica el nombre con el cual se identificará la misma en el sistema, precedido por la sintaxis reservada DEFINITIONS y luego continúa con doble punto, signo de igual y la sintaxis reservada BEGIN.

Luego comienza la sección de los import, donde se definen otras MIB estándares que vienen con el fabricante desde los cuales se pueden importar las definiciones de objetos y tipos de valores que se necesitan para devolver determinado tipo de dato luego cuando sea realizada la consulta.

La próxima sección corresponde a la declaración del módulo identidad de la MIB, ocasionalmente un nombre sugerente, el cual indica el nombre mediante el que se identificará la MIB en el grupo dentro del protocolo.

Y por último la sección final es la conclusión de la declaración de la MIB que debe terminar en la sentencia reservada END, indicando de esta manera que la declaración de todos los módulos representativos de tipos de datos a devolver ha terminado.

Implementación del objeto para el agente

Posteriormente a la creación de los objetos MIB, se define el tipo de datos que debe ser devuelto a través del protocolo SNMP. Es necesario definir e implementar el módulo que se comunicará con el agente a través de la MIB.

Cada módulo solo responde a una petición de SNMP y el mismo puede ser implementado en código Perl o ANSI C. Luego de ser implementado el mismo debe ser compilado y agregado al sistema en la ubicación adecuada de forma que el agente pueda reconocerlo al iniciarse.

El código base del módulo que será interpretado por el agente SNMP puede obtenerse mediante el comando mib2c que genera automáticamente todo lo necesario para la obtención de la información del objeto declarado en la MIB, solo restando las modificaciones pertinentes para ser adaptado a las propias necesidades de administración. La especificación de dicho comando puede ser encontrada en la Guía de extensión y configuración adjunta a este documento.

Este código base contará en su parte inicial con la inclusión de los archivos llamados cabeceras, desde los cuales se redefinen los métodos implementados más adelante en el cuerpo del módulo.

A continuación se prosigue con la rutina de inicialización, la cual será llamada automáticamente por el agente al realizarse una consulta mediante el protocolo, pero para que esto ocurra debe entonces iniciarse con `init_nombre_del_fichero`, es decir que el nombre de la rutina de inicialización debe contener la sintaxis `init_proseguido` del nombre del fichero del módulo. Después continúa una declaración de depuración.

Luego de esto se prosigue con el proceso de registro de las variables que se definieron anteriormente al inicio del módulo en caso de que lo necesitéramos, y donde son declaradas accesibles haciendo que puedan ser modificables más adelante por el agente. Normalmente los valores que son devueltos por el módulo son almacenados en una variable que luego es referenciada por la dirección de memoria que ocupa en el sistema haciendo más preciso el verdadero valor de la misma.

Integración de nuevos módulos al agente SNMP

Posteriormente al proceso descrito y detallado en la Guía de extensión y configuración para la implementación de nuevos módulos para agentes SNMP, se debe proceder con algunos pasos necesarios para poner en funcionamiento las nuevas extensiones que se realicen. Entre estos pasos se encuentran el compilado y la correcta ubicación de los ficheros dentro del agente.

Para que el agente sea capaz de identificar y utilizar las nuevas extensiones, estas deben ser instaladas en el sistema existiendo dos vías para hacerlo: 1) mediante la compilación del código fuente y posterior instalación y 2) añadiendo la extensión al agente luego de que ya se encuentra instalado y correctamente configurado.

En el primero, el archivo de código pre compilado de conjunto con la MIB correspondiente a la definición del tipo de datos que devolverá la extensión, deben ser colocados con anterioridad en los directorios correspondientes dentro de la instalación del agente, y luego realizar el proceso de compilación e instalación con los parámetros definidos anteriormente para la carga dinámica. En la segunda forma el archivo de código pre compilado de conjunto con la MIB son colocados en el directorio correspondiente luego de que el agente ha sido instalado y configurado previamente, siempre contando con la propiedad mencionada, activada, luego con un paso de configuración es suficiente para que el agente reconozca la nueva extensión. Siempre en ambos procesos se recomienda reiniciar el agente al terminar la configuración.

Para el desarrollo de la solución ya se cuenta con agentes instalados y configurados en varios servidores, por lo que se utilizará la segunda vía para la extensión de las funcionalidades del agente. Esta variante permite realizar la extensión de forma rápida y eficiente evitando el proceso de compilado del código fuente.

1.5. Almacenamiento y gestión de módulos para NET-SNMP

Formas de almacenamiento de módulos

Entre la variedad de formas existentes para el almacenamiento de información de forma persistente se encuentran dos que resultan importantes y que tributan a una posibilidad vigente para utilizar en el desarrollo de nuevos módulos para el protocolo SNMP. Estas formas son: 1) Almacenamiento de todo el código correspondiente a la extensión en la base de datos y 2) Almacenamiento físico en el disco duro de los archivos correspondientes a las extensiones.

- 1) Esta opción de almacenamiento, define la existencia en la base de datos de todo el contenido de cada uno de los archivos correspondientes a las extensiones que se han desarrollado, facilitando la búsqueda de información referente a determinada extensión.
- 2) En esta opción de almacenamiento se define la ubicación física de los ficheros de la extensión en una parte específica del disco duro, facilitando la posible modificación en caso de ser necesario, por contar con acceso rápido al código fuente de la extensión.

Dada la necesidad de almacenar los ficheros de las extensiones que se desarrollen y la rápida localización del código fuente ante posibles modificaciones, se decide entonces utilizar las bases de datos como método de almacenamiento para los metadatos que permitirán acceder directamente a la información de la extensión deseada y facilitar las búsquedas en caso de existir varias extensiones, y se utilizará el almacenamiento físico para permitir la modificación del código fuente de la extensión de ser necesario.

Como resultado se obtendrá un mayor rendimiento en la localización y modificación de las extensiones. La búsqueda de extensiones deseadas ahora simplificará el tiempo en que se localizan los ficheros de código fuente, y se contará con información organizada y estructurada acerca de las extensiones que han sido implementadas anteriormente.

Para la gestión de los repositorios comúnmente son utilizadas aplicaciones web que permiten desde cualquier localización realizar variadas operaciones sobre los ficheros almacenados en ese repositorio. Facilitando el acceso a los ficheros desde diversas ubicaciones se decide la creación de una solución web que implemente todas las funcionalidades necesarias para gestionar el repositorio de extensiones. La solución debe garantizar total seguridad sobre la información y el código fuente de las extensiones,

permitiéndole el acceso a dicha información solo a usuarios designados para realizar determinadas operaciones.

Dado que no se cuenta con aplicaciones que gestionen extensiones para los agentes SNMP, resulta necesario desarrollar una solución que contenga una aplicación web para gestionar un repositorio de extensiones, el cual almacenará los ficheros de código fuente de las nuevas funcionalidades.

1.6. Desarrollo de aplicaciones web

Los portales web pueden estar orientados a resolver uno o varios problemas de un usuario específico, grupo de usuarios o comunidades enteras. También se les puede encontrar como interfaz de una empresa dedicada a la venta de cualquier objeto, e incluso en la contante propuesta de anuncios propiciando el intercambio de información de consumo con sus usuarios.

En ocasiones la utilización de portales web implican un desarrollo completo, pero en la actualidad existen los CMS, siglas de Content Management System (Sistemas Gestores de Contenidos), los cuales generalmente son desarrollados por compañías que buscan crear una herramienta que simplifique el proceso de desarrollo de estos portales. Normalmente el ciclo de creación de una aplicación desktop requiere de grandes recursos, tiempo y personal dedicado a la misma, pero con los CMS resulta generalmente rápida la creación de portales web que faciliten la publicación de los servicios que se desean ofrecer.

Análisis de CMS para la creación de portales

Desde el preciso momento en que se decide utilizar herramientas basadas en la web para el desarrollo de proyectos orientados a la prestación de servicios con vistas a dar solución a determinado problema, la cantidad de opciones de herramientas con las que se cuenta resulta ser variada y en la mayoría de los casos la documentación también es abundante. Entre estas opciones se encuentran: Wordpress [10], Joomla [11], y Liferay Portal, siendo este último el seleccionado por ser una plataforma web corporativa que ayuda a desarrollar soluciones empresariales con resultados inmediatos y valor a largo plazo. De código abierto y compatible con la mayoría de los sistemas operativos y gestores de bases de datos. Compatible con tecnologías ligeras de scripting como PHP, Ruby, Python y Groovy, entre otras, con un sólido Framework de desarrollo Java.[12]

Liferay Portal v6.1.1-ce-ga2 tiene un fuerte compromiso en el cumplimiento de estándares abiertos. La adopción de estándares en las tecnologías empleadas para desarrollo de aplicaciones (portlets), almacenamiento y tratamiento de contenidos, servicios web y capa de presentación permiten reducir el coste del desarrollo de portales basados en Liferay.[12]

Liferay Portal incluye una amplia gama de funcionalidades de producto que permite:

- Gestión de contenidos y documentos con integración Microsoft Office®.
- Edición Web y espacios de trabajo compartidos.
- Colaboración a nivel de empresa.
- Redes Sociales o híbridas.
- Portales Corporativos y Gestión de Identidades[12].

Además Liferay Portal facilita el desarrollo basado en portlets. El mismo está orientado a la implementación web de servicios que ofrecen cobertura para la utilización en diversas áreas. Es libre y de código abierto, disponible para todo el que desee desarrollar sobre él y está pensado para el desarrollo de portales y aplicaciones orientadas a la web.

Por tanto, Liferay Portal reúne todas las características necesarias para cumplir con los requisitos propuestos.

Desarrollo basado en portlets

Un portlet es una aplicación de cara al usuario y mostrado por una aplicación web. Se asemejan a aplicaciones con ventanas que pueden estar rodeados de controles y decoración. Diversos portales han adoptado esta tecnología rápidamente, y son en la actualidad los consumidores más notables de portlets. Por lo tanto, una página de un portal puede contener un número de portlets que los usuarios pueden tener la posibilidad de organizar en columnas y filas, y minimizar, maximizar o realizar arreglos a estos, a fin de satisfacer sus necesidades individuales. La reciente entrega de los Servicios Web para Portlets Remotos (WSRP) provee una solución para superar el problema de la compatibilidad en plataformas poco comunes. WSRP utiliza WSDL para la especificación de portlets. El objetivo del uso de los portlet es definir un modelo que permita su integración de forma sencilla a las aplicaciones web.[13]

El desarrollo de aplicaciones web basados en portlets posee marcadas ventajas entre las que se pueden destacar: cuenta con un desarrollo independiente lo que permite desarrollarlos de forma separada y en paralelo con otros desarrollos. Esta característica acelera el tiempo de implementación. La personalización brinda un aspecto final del portal que dependerá de los portlets que contenga y éstos pueden ordenarse y cambiar este orden de una manera sencilla. Posee seguridad frente a fallos donde un fallo producido en un portlet no se transmite al resto de la aplicación por lo que el único componente que deja de funcionar es el portlet que provocó el fallo. Además de poder definir un nivel de privilegio por usuario para cada portlet desarrollado.[14]

1.7. Metodología y herramientas propuestas

La aplicación incluida en la solución de este trabajo de diploma cuenta con un equipo de desarrollo conformado por una sola persona. Se tiene un margen de tiempo relativamente corto para el desarrollo de todas las funcionalidades. Dado que no se cuenta con la experiencia necesaria en la implementación de este tipo de aplicaciones sobre el CMS seleccionado, se precisa entonces de una estrecha relación entre el cliente y el desarrollador. Dada la complejidad de las funcionalidades resulta imprescindible la posibilidad de regresar atrás en el proceso de desarrollo en caso de detectarse alguna anomalía que pudiera afectar la entrega a tiempo de una primera versión funcional.

Actualmente en el mundo existe una gran variedad de metodologías de desarrollo de software, las cuales son utilizadas en distintas áreas, entre las más utilizadas se pueden mencionar: Rational Unified Process (RUP) [15], SCRUM [16] y Extreme Programming (XP).

Dada las condiciones previas existentes en el comienzo del desarrollo de la solución relacionada con el corto tiempo con el que se cuenta para la entrega de la primera versión funcional de la aplicación y la cantidad de integrantes del grupo de trabajo, resulta necesario entonces efectuar una programación rápida de todas las funcionalidades, y debería posibilitarse el regreso atrás en el proceso en caso de detectar alguna anomalía que pudiera afectar el tiempo estimado para la entrega. Por todo lo anteriormente expuesto se decide entonces utilizar la Metodología XP como rectora del proceso de diseño, implementación y prueba de la solución.

XP es una metodología de desarrollo ligera. Está pensada para obtener un software eficiente en un reducido tiempo de trabajo, después de estar en constantes cambios los requisitos del software. A continuación se muestran algunas características que diferencian a XP de las demás metodologías:

Se basa en equipos de desarrollo pequeño; (de 2 a 10) personas trabajando en un mismo sitio y con pocos roles, lo que facilita el intercambio de responsabilidades y el trabajo en conjunto. Consiste en una programación rápida o extrema que está basada en la sencillez, la comunicación y la realimentación o reutilización del código desarrollado. Puesto que puede ser un proyecto para uso interno, aunque también puede utilizarse para otros proyectos. La arquitectura del proyecto se define y mejora a lo largo de su desarrollo.[17]

El cliente forma parte del equipo de desarrollo. Con la utilización de XP la comunicación con el cliente es a través de diálogos cara a cara, el cual es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo y lograr un producto que satisfaga sus necesidades.[17]

Lenguaje de modelado

UML es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG, siglas de (Object Management Group) Grupo de Gestión de Objetos. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.[18]

Es importante resaltar que UML es un "lenguaje" para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo. Se puede aplicar en una gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como el Proceso Unificado Racional), pero no especifica en sí mismo qué metodología o proceso usar.[18]

La decisión de utilizar UML como notación para el desarrollo del software se debe a que se ha convertido en un estándar que tiene las siguientes características:

- Permite modelar sistemas utilizando técnicas orientadas a objetos.

- Permite especificar todas las decisiones de análisis y diseño, construyéndose así modelos precisos, no ambiguos y completos.
- Puede conectarse con lenguajes de programación (Ingeniería directa e inversa).
- Permite documentar todos los artefactos de un proceso de desarrollo[14].

Herramienta de modelado

Visual Paradigm for UML v8.0, es una herramienta CASE, multiplataforma de modelado UML, muy potente y fácil de utilizar. Posee una licencia Freeware. Aporta a los desarrolladores de software una plataforma de desarrollo para construir aplicaciones de gran calidad y rapidez. Permite crear todo tipo de diagramas UML, revertir código fuente a modelos UML y generar código fuente desde dichos diagramas.

Soporta el ciclo de vida completo de desarrollo de un software, desde la fase de análisis hasta el despliegue del mismo. Además de la generación automática de informes en formato PDF, Word o HTML, esta herramienta puede integrarse con diversos IDE's como NetBeans (de Sun Microsystems), Developer (de Oracle), Eclipse (de IBM), JBuilder (de Borland). Es fácil de instalar, utilizar y actualizar.[19]

Se decide la selección de Visual Paradigm como herramienta para realizar el modelado de la aplicación ya que la misma brinda soporte para el lenguaje de modelado seleccionado. Permite la creación e inversión de objetos y código hacia lenguajes de programación de avanzada, prácticos para desarrollar la aplicación.

Lenguaje de programación

Para la implementación de la aplicación contenida en la solución que brindará este trabajo de diploma se utilizarán dos lenguajes de programación: Java y ANSI C. El primero de estos será utilizado para la implementación de la aplicación ya que el CMS seleccionado está desarrollado sobre Java y su desempeño está probado para aplicaciones basadas en portales web y el segundo lenguaje será utilizado para la implementación de las nuevas funcionalidades para el protocolo SNMP dado que el mismo está desarrollado en este lenguaje.

ANSI C

Se trata de un lenguaje débilmente tipificado, de nivel medio pero con muchas características de bajo nivel. La programación en C tiene una gran facilidad para escribir código compacto y sencillo a la misma vez.

Características principales

- Potencia y flexibilidad: se ha usado en contextos tan dispares como el desarrollo de sistemas operativos, procesadores de texto, gráficos, bases de datos, compiladores de otros lenguajes, etc.
- Popularidad: existe una gran variedad de compiladores, librerías, herramientas de apoyo a la programación, etc. Es el lenguaje predominante en el entorno UNIX.
- Portabilidad: el mismo programa escrito en C puede compilarse sin prácticamente ningún cambio en diferentes ordenadores.
- Sencillez: utiliza pocas palabras clave, por lo que puede aprenderse fácilmente.
- Estructura y modularidad: los programas en C pueden escribirse agrupando el código en funciones que a su vez se agrupan en distintos módulos permitiendo su reutilización.[20]

Java

Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.[21]

Java Características

- Totalmente orientado a objetos (Encapsulación, Herencia y Polimorfismo).
- Independencia de la plataforma.
- Dispone de un amplio conjunto de librerías documentadas.
- Facilidad para adicionar nuevas funcionalidades.
- Amplia documentación sobre el lenguaje y como programar sobre él.
- Recolector de basura.

- Multiplataforma.
- Amplio soporte (Variedad de IDE´s que lo soportan).
- Disponibles para aplicaciones de tipo Desktop, Web y Móviles.[22].

Sistema de gestión de bases de datos

Liferay Portal posee soporte para una variedad de gestores de bases de datos, entre los que se encuentran: DB2 [23], Derby [24], Hipersónico [25], Ingres [26], MySQL [27], Oracle [28], PostgreSQL [29], SqlServer [30], Sybase [31].

Para la implementación de la solución se decide la utilización del sistema gestor de bases de datos PostgreSQL en su versión 9.1.1 por las siguientes razones:

Es un gestor de bases de datos de código abierto el cual brinda un control de concurrencia multi-versión (MVCC por sus siglas en inglés) que permite trabajar con grandes volúmenes de datos; soporta gran parte de la sintaxis SQL y cuenta con un extenso grupo de enlaces con lenguajes de programación.

Posee características significativas del motor de datos, entre las que se pueden incluir las subconsultas, los valores por defecto, las restricciones a valores en los campos (constraints) y los disparadores (triggers). Ofrece funcionalidades en línea con el estándar SQL92, incluyendo claves primarias, identificadores entrecomillados, conversión de tipos además de la entrada de enteros binarios y hexadecimales.

Funciona en todos los sistemas operativos Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), y Windows. Debido a la liberación de la licencia, PostgreSQL se puede usar, modificar y distribuir de forma gratuita para cualquier fin, ya sea privado, comercial o académico.[32]

Entorno de desarrollo integrado

Eclipse Juno v4.2 está formado por un conjunto de herramientas de programación de código abierto, multiplataforma utilizado para el desarrollo de aplicaciones sobre el lenguaje Java. Sus funcionalidades pueden extenderse mediante la inclusión de nuevas librerías. Eclipse provee al programador herramientas muy ricas para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de software,

aplicaciones web, etc. Posee soporte para varios lenguajes de programación entre los que se encuentran Java y ANSI C.

El Eclipse es únicamente una de las herramientas que se engloban bajo el denominado *Proyecto Eclipse*. El mismo aúna tanto el desarrollo del IDE Eclipse como de alguno de los plugins más importantes como el JDT, siglas de Java Development Tools (Herramientas para desarrollo en Java) así como CDT, siglas de C/C++ Development Tooling (Herramientas para desarrollo en C/C++). Existen versiones instalables para cualquier plataforma que incluyen el código fuente y los plugins más habituales. [33]

Luego del estudio realizado anteriormente sobre las características del IDE Eclipse y las facilidades que brinda el mismo para el trabajo con el lenguaje Java así como la inclusión de nuevas funcionalidades, se decide entonces optar por este IDE, garantizando el mismo que en una sola plataforma de desarrollo se encuentre disponible la facilidad de programar en los dos lenguajes de programación que se utilizarán en la presente investigación.

Conclusiones del capítulo

Se realizó un estudio bibliográfico sobre la configuración y extensión de agentes SNMP, que permitió seleccionar «La implementación de módulos que pueden ser cargados dinámicamente», como la forma más adecuada de implementar y adicionar nuevas funcionalidades al protocolo.

Se seleccionó XP como metodología de desarrollo usando el UML como lenguaje de modelado y como herramienta CASE el Visual Paradigm. Se utilizó C y Java como lenguajes de programación, Liferay como gestor de contenidos y el IDE fue Eclipse.

Capítulo 2- Diseño e implementación de la solución

Luego de todo el estudio teórico realizado sobre el funcionamiento interno del protocolo SNMP, el análisis de todas las herramientas candidatas para la implementación de la solución, y la definición de la metodología de desarrollo se procede entonces a realizar el Diseño e implementación de la solución con vistas a desarrollar la primera versión funcional en el tiempo acordado.

2.1. Descripción de la solución

Antes de comenzar con el diseño y la implementación de la solución resulta necesario mencionar los componentes que conformarán el resultado final de este trabajo de diploma. Estos componentes son:

- 1) Un marco de trabajo para la configuración y extensión de agentes SNMP.
- 2) Un repositorio de extensiones que contendrá las nuevas funcionalidades que sean implementadas.
- 3) Una aplicación que permitirá gestionar el repositorio de extensiones.

Marco de trabajo para el desarrollo de nuevos módulos

El marco de trabajo para el desarrollo de nuevos módulos estará dividido en dos epígrafes principales: 1) Creación de nuevos módulos para el agente y 2) Proceso de ubicación del módulo en el agente.

En el primero de ellos, el marco de trabajo explicará y expondrá ejemplos en los que se ilustre claramente todo el proceso de creación de nuevos módulos para el agente SNMP, así como la creación de las MIB y la definición de los tipos de objetos que serán devueltos. En el segundo la guía explicará completamente el proceso necesario para la ubicación en el agente de los nuevos módulos y las nuevas MIB que se implementen. Al concluir el proceso descrito por el marco de trabajo, el agente debe ser capaz de trabajar con la nueva extensión que ha sido añadida.

Repositorio para el almacenamiento de los módulos

El repositorio contará con una estructura sencilla, correctamente desarrollada y organizada por categorías. Dicha organización partirá de un directorio raíz el cual contará en el primer nivel con los directorios correspondientes a cada una de las categorías registradas y en un segundo nivel directorios dedicados únicamente a la colocación y organización de los archivos pertenecientes a las extensiones.

Esta estructura jerárquica de directorios destinados a la organización del repositorio lleva además que los metadatos que se encuentran almacenados en la base de datos también contengan una organización que facilite el trabajo con las extensiones. En la *Figura 2- Diagrama Entidad-Relación de los metadatos para categorías y extensiones* se muestra un modelo Entidad-Relación con las tablas que se necesitan para almacenar los metadatos de las categorías y las extensiones asociadas a ellas.

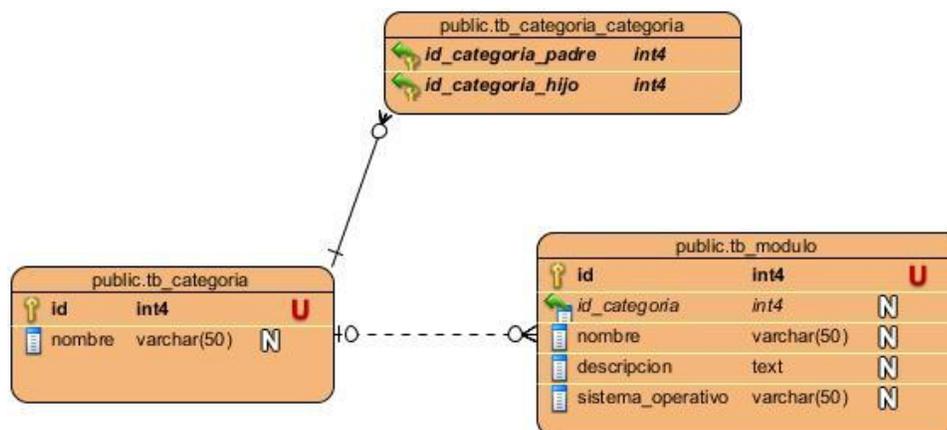


Figura 2- Diagrama Entidad-Relación de los metadatos para categorías y extensiones

En la figura anterior se observa la estructura de la base de datos para el almacenamiento de los metadatos de cada una de las extensiones que sean agregadas al sistema. En él se encuentra representada la tabla `tb_categoria` en la que es almacenado el identificador y el nombre de la categoría. La misma cuenta con una relación con la tabla `tb_categoria_categoria` en la que se almacena el nombre de la categoría a la que pertenece en caso de ser hijo de una ya existente y los id de otras categorías que constituyen hijos.

Además se puede observar otra relación con la tabla `tb_modulo` en la que se almacenan los metadatos para cada una de las extensiones. Estas extensiones entre otros datos almacenan el identificador de la categoría a la que pertenece.

Aplicación para la gestión de módulos

El Sistema de Gestión de Extensiones para Agentes SNMP es una aplicación cuyo objetivo principal está enfocado a gestionar extensiones y conformar paquetes de instalación que permiten extender los agentes

SNMP. Estas extensiones pueden ser creadas por usuarios administradores de sistemas que deseen extender las funcionalidades del protocolo añadiéndole la posibilidad de monitorear otros parámetros de determinado dispositivo. Para garantizar todo esto la aplicación se divide en varios módulos en los cuales estarán agrupadas todas las funcionalidades de esta aplicación.

El módulo denominado *Módulo de Seguridad*, provisto por el propio CMS, será el encargado de controlar la autenticación de los usuarios en el sistema así como la gestión de usuarios, roles y niveles de acceso.

Para la autenticación el sistema debe:

- Permitir la autenticación en el sistema mediante usuario y contraseña.

Para la gestión de los usuarios el sistema debe poder:

- Adicionar nuevos usuarios.
- Actualizar los datos de los usuarios existentes.
- Eliminar usuarios existentes.

Para la gestión de roles el sistema debe poder:

- Adicionar nuevos roles de usuario.
- Actualizar los datos de los roles existentes.
- Eliminar un determinado rol del sistema.

El módulo denominado *Módulo de Gestión del Repositorio* será el encargado de gestionar las extensiones existentes en el sistema. Sobre dichas extensiones el sistema debe permitir:

- Adicionar nuevas extensiones por categorías al sistema.
- Actualizar extensiones.
- Eliminar las extensiones existentes en el sistema.
- Descargar la extensión deseada de la lista de disponibles.
- Adicionar nuevas categorías.
- Actualizar las categorías existentes.
- Eliminar categorías.
- Realizar búsquedas de extensiones.

El módulo denominado *Módulo de Gestión de Sistemas a Monitorear* es el encargado de mantener almacenado en el sistema toda la información referente a los parámetros que pueden ser monitoreados en determinado sistema, así como la oferta de reportes con esta información. Para dar cumplimiento a estas actividades, el sistema debe ser capaz de:

- Permitir el registro de un equipo (servidor o equipo de interconexión) en el sistema.
- Actualizar la información referente a determinado equipo de los que se tiene conocimiento en el sistema.
- Eliminar un equipo de la lista de equipos disponibles.
- Poder obtener la información de los parámetros que pueden ser monitoreados en un sistema dado.
- Almacenar esta información de monitoreo en el sistema permitiendo su disponibilidad para posterior consulta.
- Brindar dicha información de parámetros mediante una interfaz (fichero o servicio) el cual contiene el árbol de información sobre los parámetros que pueden ser monitoreados, de tal forma que pueda ser consultada por aplicaciones externas.

Otro módulo denominado *Módulo de Configuración* será el encargado de gestionar los paquetes que permiten extender los agentes SNMP. Para ello tiene entre sus funcionalidades la Generación de paquetes que facilitan la instalación de los nuevos módulos y la modificación de parámetros orientados al repositorio de extensiones.

Para lograr lo anterior el módulo debe ser capaz de:

- Listar todos los parámetros que pueden ser monitoreados en un equipo dado.
- Listar por categorías todas las extensiones existentes y disponibles en el repositorio de la aplicación que pueden ser instaladas en determinado equipo.
- Presentar al usuario la opción de poder actualizar la información existente en el sistema sobre los parámetros que pueden ser monitoreados en determinado equipo.
- Generar un paquete que contenga la información sobre las extensiones seleccionadas para descargar, así como los ficheros correspondientes a cada extensión.
- Modificar los parámetros de configuración del repositorio.

2.2. Diseño de la aplicación

Historias de Usuario

Las Historias de Usuario son la técnica utilizada por la metodología XP para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las Historias de Usuario es muy dinámico y flexible, en cualquier momento las Historias de Usuario pueden romperse, reemplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas. Cada Historia de Usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarlas en pocas semanas.[34]

No hay que preocuparse si en un principio no se identifican todas las Historias de Usuarios. Al comienzo de cada iteración estarán registrados los cambios en las Historias de Usuario y según eso se planificará la siguiente iteración. Las Historias de Usuario son descompuestas en tareas de programación y asignadas a los programadores para ser implementadas durante una iteración.[34]

- Las Historias de Usuario tienen el mismo propósito que los casos de uso.
- Las escriben los propios clientes tal y como ven ellos las necesidades del sistema.
- Las Historias de Usuario son similares al empleo de escenarios, con la excepción de que no se limitan a la descripción de la interfaz de usuario.
- También conducirán el proceso de creación de los test de aceptación (empleados para verificar que las Historias de Usuario han sido implementadas correctamente).
- Existen diferencias entre estas y la tradicional especificación de requisitos. La principal diferencia es el nivel de detalle. Las Historias de Usuario solamente proporcionarán los detalles sobre la estimación del riesgo y cuánto tiempo conllevará la implementación de dicha Historia de Usuario.[35]

Luego del análisis realizado con el objetivo de determinar los requisitos funcionales que corresponden a la aplicación que se necesita desarrollar, se determinaron 24 Historias de Usuario. En la *Tabla 1 - Historia de Usuario "Adicionar categoría"* se muestra un ejemplo de una Historia de Usuario, las restantes pueden ser

consultadas en la “Planilla de Historias de Usuario”, que se encuentra en el Expediente de Proyecto adjunto a este Trabajo de Diploma.

Tabla 1 - Historia de Usuario "Adicionar categoría"

Historia de Usuario	
Número: 1	Nombre de la Historia de Usuario: Adicionar categorías
Cantidad de modificaciones a la Historia de Usuario: Ninguna	
Usuario: Marcos M. Martínez Pérez	Iteración asignada: 1
Prioridad en negocio: Muy Alta	Puntos estimados: 0.8 semanas
Riesgo en desarrollo: Alto	Puntos reales: 0.6 semanas
Descripción: El usuario accede a la interfaz principal, selecciona la página correspondiente a las categorías, donde elige la opción “Adicionar categorías”, la cual le permite introducir los datos necesarios para realizar la inserción de la nueva categoría en el sistema. Luego ejecuta el botón adicionar y el sistema mostrará un mensaje indicando si fue agregada la nueva categoría.	
Observaciones: En caso de no poderse agregar la nueva categoría el sistema mostrará un mensaje indicando que no se pudo realizar tal operación.	
<p style="text-align: center;">Prototipo de interfaz</p> <p>The image shows a software prototype for a 'Adicionar categoría' dialog box. The dialog has a title bar with the text 'Adicionar categoría' and a close button (red X). Below the title bar, there is a section titled 'Datos de la categoría'. Under this section, there are two labeled text input fields: 'Nombre *' and 'Padre *'. The 'Padre *' field contains the text 'Windows'. At the bottom of the dialog, there are two buttons: 'Adicionar' (with a blue checkmark icon) and 'Cancelar' (with a blue X icon).</p>	

Lista de reserva del producto

La lista de reserva del producto recoge todos los requisitos funcionales y no funcionales con los que debe contar la aplicación que se desea desarrollar. Las funciones se muestran ordenadas por su prioridad definida en cuatro categorías nombradas como (Muy alta, Alta, Media y Baja). A continuación aparecen también los requisitos no funcionales del sistema. En la misma se indica además el tiempo estimado en semanas necesario para la realización de la funcionalidad así como la persona que realiza dicha estimación.

En la *Tabla 2- Lista de reserva del producto* se recogen los requisitos funcionales y los no funcionales con los que debe cumplir la aplicación. Estas funcionalidades son las que permitirán la gestión de las extensiones destinadas a la configuración y extensión de agentes SNMP y las mismas permitirán al usuario interactuar con la aplicación para hacer cumplir su propósito.

Tabla 2- Lista de reserva del producto

No.	Deimulcripción	Estimación	Estimado por
Prioridad: Muy Alta			
1	Adicionar categorías	0.6 semanas	Analista
2	Actualizar categorías	1.2 semanas	Analista
3	Eliminar categorías	0.6 semanas	Analista
4	Listar categorías	1.2 semanas	Analista
5	Adicionar extensión	1.8 semanas	Analista
6	Consultar extensión	0.6 semanas	Analista
7	Eliminar extensión	0.6 semanas	Analista
8	Descargar extensión	1.0 semanas	Analista
9	Listar extensiones	0.4 semanas	Analista
10	Crear instalador	1.2 semanas	Analista
Prioridad: Alta			
11	Adicionar equipos	0.4 semanas	Analista
12	Actualizar equipo	1.2 semanas	Analista
13	Eliminar equipo	0.6 semanas	Analista
14	Listar equipos	0.6 semanas	Analista

Capítulo 2 – Diseño e implementación de la solución

15	Consultar parámetros	2.0 semanas	Analista
Prioridad: Media			
16	Consultar extensiones	0.2 semanas	Analista
17	Adicionar sistema	0.4 semanas	Analista
18	Actualizar sistema	0.8 semanas	Analista
19	Eliminar sistema	0.4 semanas	Analista
20	Listar sistemas	1.0 semanas	Analista
Prioridad: Baja			
21	Almacenar parámetros	0.8 semanas	Analista
22	Listar parámetros	0.8 semanas	Analista
23	Actualizar configuración	1.2 semanas	Analista
24	Brindar información	0.8 semanas	Analista
Requisitos no funcionales			
1	De seguridad: Garantizar que la aplicación solicite usuario y contraseña como datos credenciales necesarios para garantizar la autenticación en el sistema. Cada usuario solo podrá acceder a las opciones a las que tenga permisos.		Analista
2	Requisitos de Rendimiento y Escalabilidad: El sistema debe ser capaz de responder rápidamente a las peticiones hechas por el usuario garantizando que el tiempo de latencia entre una petición y su respuesta sea mínimo.		Analista
3	Requisito de Facilidad de Uso: Para la utilización de la aplicación se		Analista

	debe poseer un conocimiento básico de Informática y sobre el trabajo con aplicaciones web, en las que el usuario interactúa constantemente con el sistema.		
4	Requisitos de Software: Esta aplicación deberá poder utilizarse en todos los sistemas operativos. Debe existir un servidor de bases de datos PostgreSQL v 9.1 o superior y el equipo debe tener instalada la Máquina Virtual de Java JDK. Además debe estar corriendo el servidor Apache Tomcat v7.0.27. En los clientes debe existir El navegador web Mozilla Firefox v 17 o superior.		Analista
5	Requisitos de Hardware: El equipo PC que se utilizará como servidor debe contener como mínimo 1 GB de memoria RAM, 2 GB de espacio libre en Disco Duro, y un microprocesador superior a los 1.5 GHz. Los clientes deberán tener como mínimo 256 MB de RAM, 1 GB de espacio libre en Disco Duro y un microprocesador superior a los 1.6 GHz.		Analista

6	Requisitos de apariencia o interfaz externa: La aplicación debe contener una interfaz que resulte intuitiva, amigable y fácil de aprender y utilizar por parte de los usuarios que hagan uso de la misma. Debe ser compatible con resoluciones 1024x768 o 800x600 píxeles respectivamente.		Analista
---	---	--	----------

Plan de iteraciones

Después de concluido el proceso en el cual se definen las Historias de Usuario que deben ser implementadas con el objetivo de obtener la aplicación, se procede a crear el plan de iteraciones. Este plan resulta necesario puesto que en él se definen las Historias de Usuario que deben ser resueltas en cada versión del programa durante su desarrollo.

Este plan es utilizado por los desarrolladores para planificar el tiempo idóneo para la implementación de cada una de las Historias de Usuario. Este tiempo es acordado en estrecha relación entre el cliente y el desarrollador. También se debe definir la prioridad de cada una de ellas puesto que esta prioridad permitirá posteriormente determinar en qué iteración debe implementarse tal Historia de Usuario.

En la *Tabla 3- Plan de iteraciones* se puede observar el Plan de Iteraciones acordado entre el cliente y el desarrollador, las Historias de Usuario a desarrollar en cada una de las iteraciones y el tiempo propuesto como idóneo para la implementación de las mismas, permitiendo que al concluir ese tiempo se cuente con una primera versión final de la aplicación que gestionará las extensiones para agregar nuevas funcionalidades a los agentes SNMP.

Tabla 3- Plan de iteraciones

No	Descripción de la iteración	Historias a implementar	Duración total
1	Se desarrollan 10 Historias de Usuario de prioridad Muy Alta que resultan de vital importancia para el resto de la aplicación.	1, 2, 3, 4, 5, 6, 7, 8, 9 y 18	9.2 semanas
2	Se desarrollan 5 Historias de Usuario de prioridad Alta, dando paso el registro de equipos y parámetros.	10, 11, 12, 13 y 14	4.8 semanas
3	Se desarrollan 5 Historias de Usuario con una prioridad Media para su desarrollo.	19, 20, 21, 22 y 23	2.8 semanas
4	Se desarrollan 5 Historias de Usuario con una prioridad Baja en su desarrollo que permiten obtener información de un determinado equipo.	15, 16, 17, 24 y 25	3.8 semanas

Tarjetas CRC

Para diseñar la aplicación partiendo del sistema de trabajo en equipo se debe cumplir con tres principios, Cargo o Clase, Responsabilidad y Colaboración (CRC). Las tarjetas CRC permitirán desprendernos del método de trabajo basado en procedimientos y trabajar con una metodología basada en objetos. Las tarjetas CRC permiten que el equipo completo contribuya en la tarea del diseño.

Una tarjeta CRC representa un objeto. El nombre de la clase se coloca a modo de título en la tarjeta, las responsabilidades se colocan a la izquierda, y las clases que se implican en cada responsabilidad a la derecha, en la misma línea que su requerimiento correspondiente.[35]

En el desarrollo de la solución se identificaron un total de 37 tarjetas CRC. En la *Tabla 4- Tarjeta CRC (Clase: paquetes)* se muestra un ejemplo de Tarjeta CRC realizada para la clase Paquetes encargada de la gestión de paquetes de instalación. Las otras tarjetas correspondientes a las demás clases de la aplicación se encuentran en la Planilla de Tarjetas CRC dentro del expediente de proyecto adjunto a este Trabajo de Diploma.

Tabla 4- Tarjeta CRC (Clase: paquetes)

Tarjeta CRC	
Clase: configuración_controladora (Módulo de Gestión de configuración)	
Responsabilidades	Colaboraciones
obtener_configuracion	conexión
actualizar_configuracion	configuración

Diagrama de clases

En la *Figura 3- Diagrama de clases* se muestra el diagrama de clases correspondiente al flujo principal de desarrollo de la aplicación, el cual muestra las relaciones entre las clases que de conjunto conforman el funcionamiento interno de la misma.

Dado que la aplicación cuenta con el soporte para poder utilizar los portlets en otras aplicaciones web, resulta necesario que cada uno de ellos posea su propia clase controladora con los métodos correspondientes que le permiten realizar todas las funcionalidades afines logrando así independencia de otros portlets.

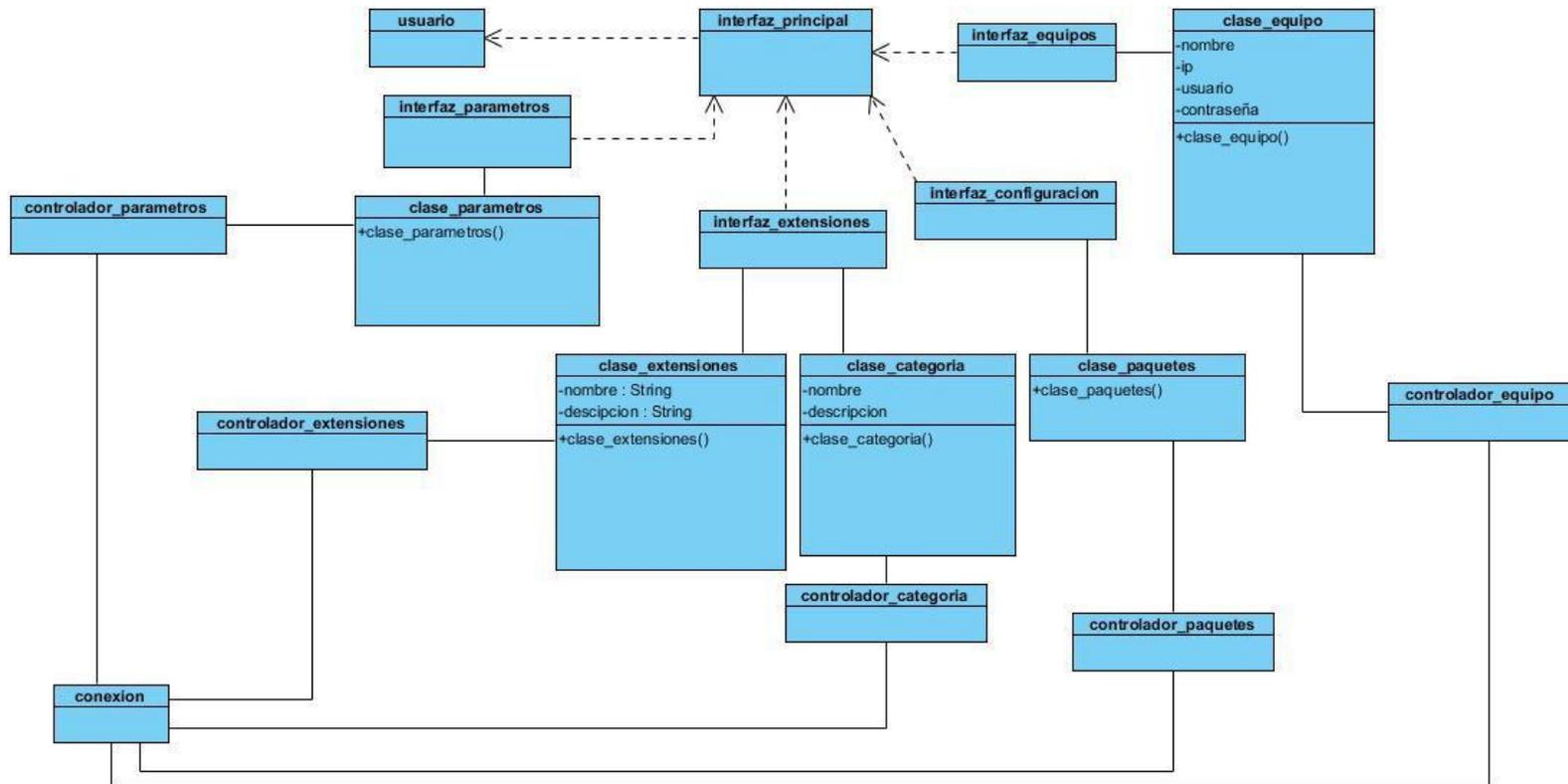


Figura 3- Diagrama de clases

En el diagrama anterior puede observarse las clases que definen el funcionamiento interno de la aplicación que forma parte de esta solución. Este diagrama está dividido en tres partes fundamentales. La primera de ellas es la que corresponde a las clases visuales encargadas de la visualización de la información al cliente, en la que se identificaron un total de 5 clases. La segunda está formada por las clases que manejan el negocio conformada por 5 clases y una tercera parte en la que se identificaron 5 clases controladoras.

2.3. Arquitectura base de la aplicación

Patrones de arquitectura

El concepto de arquitectura de software se refiere a la estructuración del sistema que idealmente se crea en etapas tempranas del desarrollo. Esta estructuración representa un diseño de alto nivel del sistema que tiene dos propósitos primarios: satisfacer los atributos de calidad (desempeño, seguridad), y servir como guía en el desarrollo. Antes de profundizar sobre el tema, es conveniente definir el concepto ya que hoy en día el término de arquitectura se usa para referirse a varios aspectos relacionados con las Tecnologías de la Informática.

De acuerdo con SEI siglas de Software Engineering Institute (Instituto de Ingeniería de Software), la Arquitectura de Software se refiere a las estructura de un sistema, compuestas de elementos con propiedades visibles de forma externa y las relaciones que existen entre ellos.[36]

Liferay Portal no define una arquitectura específica para el desarrollo de portales basados en su plataforma, en sí, está orientado a servicios por lo que admite la integración de mini aplicaciones o portlets a su estructura en los que se puede definir una arquitectura acorde a las necesidades de implementación. Dado esto, se decide la utilización del Modelo–Vista–Controlador [37] como patrón de arquitectura para el desarrollo de la presente solución.

Arquitectura Modelo–Vista–Controlador

Modelo: esta es la representación específica del dominio de la información sobre la cual funciona la aplicación. El modelo es otra forma de llamar a la capa de dominio. La lógica de dominio añade significado a los datos.

Vista: este presenta el modelo en un formato adecuado para interactuar, usualmente un elemento interfaz de usuario.

Controlador: este responde a eventos, usualmente acciones del usuario, e invoca a cambios en el modelo y probablemente en la vista.

En la *Figura 4- Diagrama de clases (Patrón - MVC)* puede observarse la estructura de la aplicación que será desarrollada siguiendo el patrón MVC. La misma ha sido dividida en 3 paquetes donde convergen relacionamente las clases principales con las que debe contar la aplicación luego de implementada.

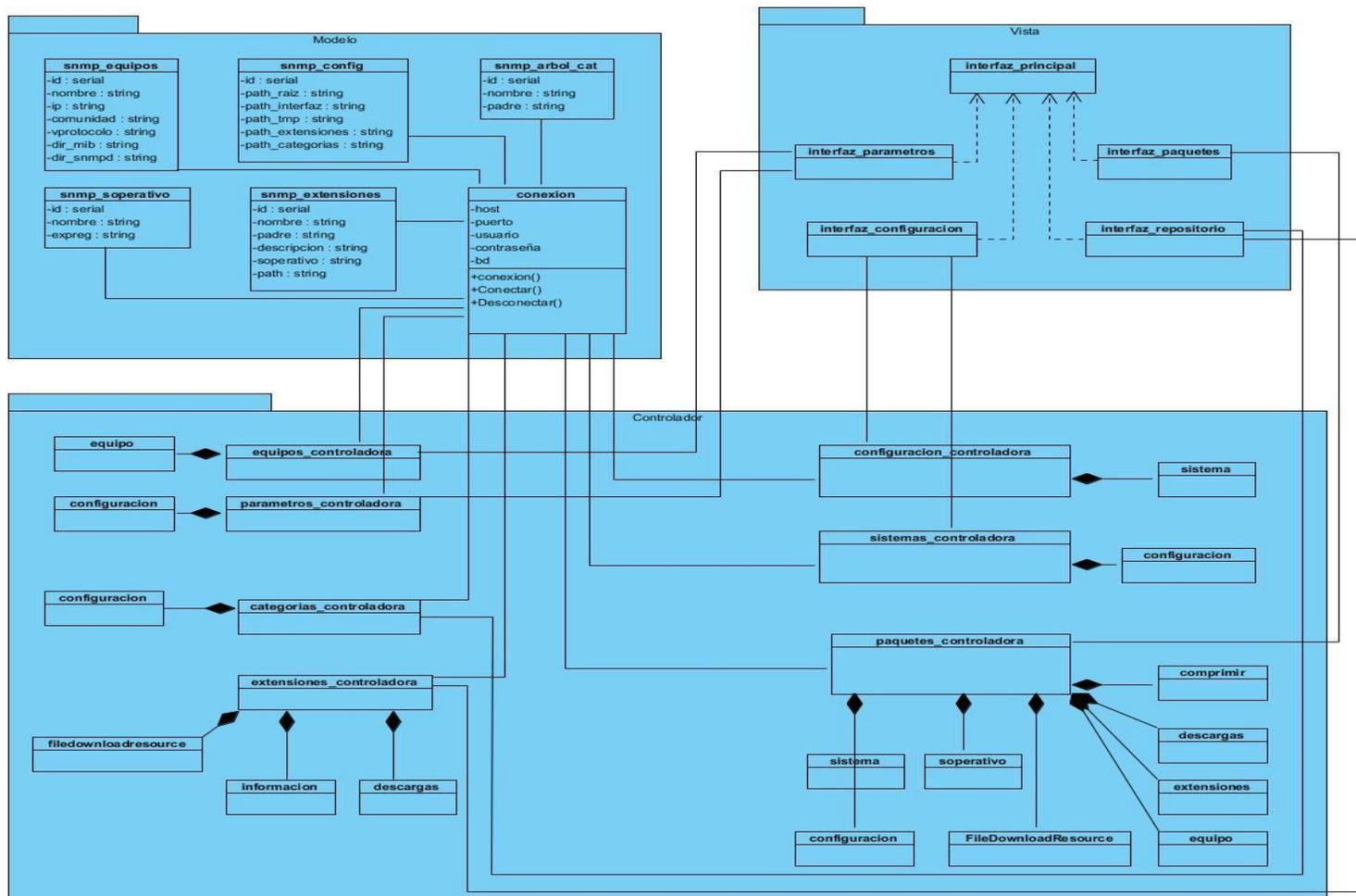


Figura 4- Diagrama de clases (Patrón - MVC)

Patrones de diseño

Según Nicolás Tedeschi, “Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software.” En otras palabras, brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares.[38]

Un Patrón de Diseño define un esquema de refinamiento de los subsistemas o componentes dentro de un sistema o las relaciones entre estos. Este describe una estructura común y recurrente de componentes interrelacionados, que resuelve un problema general de diseño dentro de un contexto particular. Los patrones de diseño trabajan a una escala intermedia. Son menores en escala que un patrón de arquitectura, pero logran ser independientes del lenguaje de programación. La aplicación de un patrón de diseño no tiene efectos sobre la estructura fundamental del sistema (arquitectura), pero puede tener una fuerte influencia sobre la arquitectura de un subsistema.[39]

Patrones GRASP

Los patrones GRASP, siglas de General Responsibility Assignment Software Patterns (Patrones de Asignación de Responsabilidades de Software) describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones, constituyen el fundamento de cómo se diseñará el sistema. Los patrones GRASP son utilizados para asignar responsabilidades.[40]

Patrón Experto

Consiste en asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad.[40] Este patrón se evidencia en la *Figura 5- Diagrama donde se evidencia el patrón Experto* donde la clase `interfaz_configuración` asigna a la clase `configuracion_controladora` la responsabilidad de gestionar los sistemas registrados.

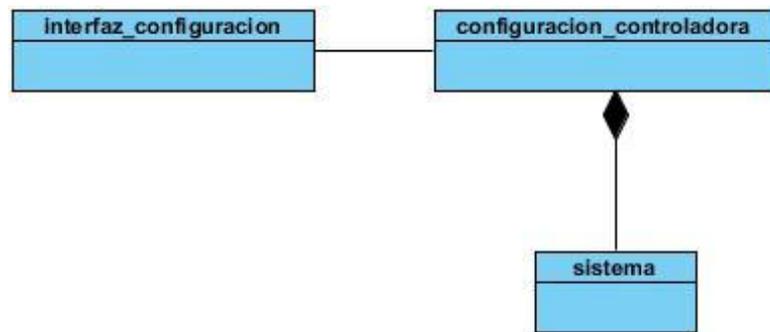


Figura 5- Diagrama donde se evidencia el patrón Experto

Patrón Creador

Consiste en asignar a la clase B la responsabilidad de crear una instancia de clase A en uno de los siguientes casos:[40]

- B agrega los objetos A.
- B contiene los objetos A.
- B registra las instancias de los objetos A.
- B utiliza específicamente los objetos A.
- B tiene los datos de inicialización que serán transmitidos a A cuando este objeto sea creado (así que B es un Experto respecto a la creación de A).[40]

En la *Figura 6- Diagrama donde se evidencia el patrón Creador* se evidencia la utilización de este patrón puesto que controladora, clase B utiliza específicamente objetos de tipo A.

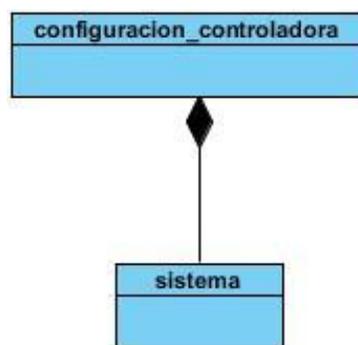


Figura 6- Diagrama donde se evidencia el patrón Creador

Patrón Bajo Acoplamiento

Consiste en asignar una responsabilidad para mantener bajo acoplamiento. Es una medida de la fuerza con que una clase está conectada a otras clases.[40]

El Bajo Acoplamiento estimula asignar una responsabilidad de modo que su colocación no incremente el acoplamiento tanto que produzca los resultados negativos propios de un alto acoplamiento.[40]

En la *Figura 7- Diagrama donde se evidencia el patrón Bajo Acoplamiento* se puede observar que la clase que cuenta con lo necesario para trabajar con extensiones es la clase paquetes_controladora.

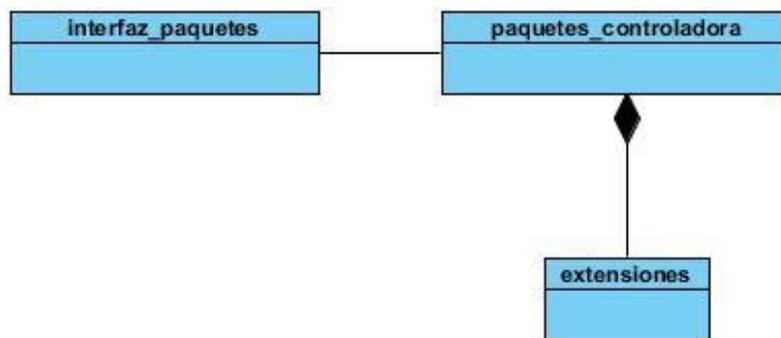


Figura 7- Diagrama donde se evidencia el patrón Bajo Acoplamiento

Patrón Alta cohesión

Consiste en asignar una responsabilidad de modo que la cohesión siga siendo alta. En la perspectiva del diseño orientado a objetos, la cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme.[40]

En la *Figura 8- Diagrama donde se evidencia el patrón Alta cohesión* se puede observar el patrón Alta cohesión ya que entre estas dos clases existe una estrecha cooperación con el objetivo de que todas las funcionalidades que se encuentran definidas en la clase extensiones puedan realizarse dependiendo de la clase paquetes_controladora, pero garantizando siempre la disponibilidad de las funciones.

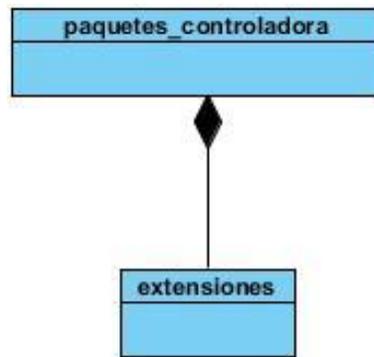


Figura 8- Diagrama donde se evidencia el patrón Alta cohesión

2.4. Implementación de la aplicación

Tareas de ingeniería

El equipo de desarrollo de esta aplicación tiene la responsabilidad de revisar, analizar y evaluar cada una de las funcionalidades o Historias de Usuario correspondientes a la aplicación que se desea elaborar con el objetivo de dividir cada una de estas en tareas más pequeñas, que de conjunto tributen al correcto desempeño de la funcionalidad, representando cada una de ellas una característica específica dentro del sistema.

En la *Tabla 5- Tarea de Ingeniería "Implementar adicionar categorías"* se muestra una tarea de ingeniería la cual está compuesta por el nombre de la tarea, en este caso "Implementar adicionar categorías", el tipo de la tarea que es Desarrollo, y entre otros aspectos se encuentran: el nombre del responsable de realizar la tarea así como el tiempo estimado necesario para darle cumplimiento a la misma. Para la aplicación que se desea desarrollar se determinaron un total de 34 Tareas de Ingeniería. Aunque aquí solo se expone una de ellas, las restantes pueden ser consultadas en la "Planilla de Tareas de Ingeniería", que se encuentra en el Expediente de Proyecto adjunto a este Trabajo de Diploma.

Tabla 5- Tarea de Ingeniería "Implementar adicionar categorías"

Tarea de Ingeniería	
Número Tarea: 1	Número Historia de Usuario: 1
Nombre Tarea: Implementar adicionar categorías	
Tipo de Tarea : Desarrollo	Puntos Estimados: 0.8 semanas
Fecha Inicio: 1/1/2013	Fecha Fin: 4/1/2013
Programador Responsable: Marcos M. Martínez Pérez	
Descripción: El usuario debería tener la posibilidad de adicionar nuevas categorías de extensiones al sistema.	

Estándares de codificación

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Al comenzar un proyecto de software, se debe establecer un estándar de codificación para asegurarse de que todos los programadores del proyecto trabajen de forma coordinada. Cuando el proyecto de software incorpore código fuente previo, o bien cuando realice el mantenimiento de un sistema de software creado anteriormente, el estándar de codificación debería establecer cómo operar con la base de código existente.

La legibilidad del código fuente repercute directamente en lo bien que un programador comprende un sistema de software.[40]

A continuación se reflejarán ejemplos de algunas convenciones de código específicamente para el lenguaje Java, [41], siendo este lenguaje el seleccionado para el desarrollo de la aplicación que gestionará el repositorio de extensiones. Estas convenciones de código resultan importantes para los programadores por varias razones entre las que se encuentran:

- El 80% del coste del código de un programa va a su mantenimiento.
- Casi ningún software lo mantiene toda su vida el autor original.

- Las convenciones de código mejoran la lectura del software, permitiendo entender el código nuevo mucho más rápido y más a fondo.
- Si distribuyes tu código fuente como un producto, necesitas asegurarte de que está bien hecho y presentado como cualquier otro producto.[42]

Para que funcionen las convenciones, cada persona que escribe software debe seguir la convención. Todos sin excepción.

Sentencias package e import

La primera línea no comentario de los ficheros fuente de Java es la sentencia package. Después de esta, pueden seguir varias sentencias import. En la *Figura 9- Ejemplo de sentencias package e import* se puede observar la utilización de esta convención en el desarrollo de la solución.

```
package conexion;  
  
import java.sql.Connection;
```

Figura 9- Ejemplo de sentencias package e import

Longitud de línea

Evitar las líneas de más de 80 caracteres, ya que no son manejadas correctamente por muchas terminales y herramientas. En la *Figura 10- Ejemplo de sentencia donde se limita la longitud de caracteres por línea* se puede observar un ejemplo de restricción impuesta a la línea garantizando que la longitud de la misma esté por encima del estándar.

```
principal_autenticado aut = new principal_autenticado(conn, rol);
```

Figura 10- Ejemplo de sentencia donde se limita la longitud de caracteres por línea

Comentarios

Los programas Java pueden tener dos tipos de comentarios: comentarios de implementación y comentarios de documentación. Los comentarios de implementación son aquellos que también se encuentran en C++, delimitados por `/*...*/`, y `//`. Los comentarios de documentación (conocidos como "doc

comments") existen sólo en Java, y se limitan por `/**...*/`. Los comentarios de documentación se pueden exportar a ficheros HTML con la herramienta Javadoc.

Los comentarios de implementación son para comentar nuestro código o para comentarios acerca de una implementación particular. Los comentarios de documentación son para describir la especificación del código, libre de una perspectiva de implementación, y para ser leídos por desarrolladores que pueden no tener el código fuente a mano.

Se deben usar los comentarios para dar descripciones de código y facilitar información adicional que no es legible en el código mismo. Los comentarios deben contener sólo información que es relevante para la lectura y entendimiento del programa. Por ejemplo, información sobre cómo se construye el paquete correspondiente o en que directorio reside, no debe ser incluida como comentario. Los programas pueden tener cuatro estilos de comentarios de implementación: de bloque, de una línea, de remolque, y de fin de línea.[42]

Comentarios de bloque

```
/*
 * Aquí hay un comentario de bloque.
 */
```

Figura 11- Ejemplo de un comentario de bloque

Comentarios de una línea

```
if (condicion) {
    /* Código de la condicion. */
    ...
}
```

Figura 12- Ejemplo de un comentario de línea

Comentarios de remolque

```
if (a == 2) {
    return TRUE;           /* caso especial */
} else {
    return isPrime(a);
}
```

Figura 13- Ejemplo de un comentario de remolque

Comentarios de fin de línea

```
if (foo > 1) {
    // Hacer algo.
    ...
}
else {
    return false;         // Explicar aqui por que.
}
```

Figura 14- Ejemplo de un comentario de fin de línea

2.5. Conclusiones del capítulo

En el capítulo se realizó un completo análisis de los procesos fundamentales relacionados con el desarrollo de la aplicación que realizará la gestión de extensiones para agentes SNMP.

Se diseñó un marco de trabajo que contiene la descripción necesaria para realizar la extensión de un agente SNMP.

Se diseñó un repositorio que organizado por categorías almacenará las extensiones que sean implementadas.

Se desarrolló una aplicación que facilita la gestión de las extensiones que son implementadas para extender los agentes SNMP.

Para mejorar el entendimiento de la aplicación fueron utilizados algunos elementos de otras metodologías como son el diagrama de dominio y el diagrama de clases.

En el proceso se determinaron un total de 22 Historias de Usuario agrupadas en un total de 36 funcionalidades. Además fue definido el plan de iteraciones de la aplicación y se diseñó e implementó la aplicación interactiva que permitirá gestionar las extensiones.

Con esta aplicación el usuario será capaz con tan solo poseer conocimientos mínimos de informática, gestionar las extensiones que ya se encuentren en la base de datos o las nuevas que se deseen agregar en tan solo varios clics de interacción con la aplicación. La interfaz interactiva, intuitiva y amigable que posee la misma permite al usuario la rápida obtención de conocimientos para el trabajo con la misma.

Capítulo 3- Validación de la aplicación

Uno de los aspectos fundamentales digno de resaltar cuando se hable de Metodología XP lo constituye el proceso de pruebas que se define con el objetivo de medir la calidad del software que se desarrolle. Este proceso permite el aumento de la calidad final de software propiciando una reducción en el número de errores que no hayan sido detectados durante todo el proceso e incluso disminuyendo el tiempo existente entre la detección de un error y la corrección del mismo.

XP divide el proceso de pruebas en dos grandes grupos, 1) Pruebas Unitarias y 2) Pruebas de aceptación.

Tipos de pruebas

Pruebas unitarias

La producción de código está dirigida por las pruebas unitarias. Las pruebas unitarias se establecen antes de escribir el código y son ejecutadas constantemente ante cada modificación del sistema. Los clientes escriben las pruebas funcionales para cada Historia de Usuario que deba validarse. En este contexto de desarrollo evolutivo y de énfasis en pruebas constantes, la automatización para apoyar esta actividad es crucial.[34]

Pruebas de aceptación

Estas pruebas resultan las más importantes dado que las mismas verifican y controlan el grado de aceptación del cliente con el producto que se ha desarrollado, marcando así el final de una iteración y el inicio de la próxima. El cliente final del software es quien debe desarrollar estas pruebas para corroborar que se cumplan correctamente las funcionalidades exigidas por él.[43]

Estas pruebas tienen como objetivo verificar las funcionalidades, y por tal motivo, los requisitos definidos en el sistema son la principal fuente de información a la hora de construir las pruebas de aceptación. Durante una iteración la Historia de Usuario seleccionada en la planificación de iteraciones se convertirá en una prueba de aceptación. El cliente o usuario especifica los aspectos a probar cuando una Historia de Usuario ha sido correctamente implementada.[43]

Una Historia de Usuario puede tener más de una prueba de aceptación, tantas como sean necesarias para garantizar su correcto funcionamiento y no se considera completa hasta que no supera sus pruebas de aceptación. Esto significa que debe desarrollarse una nueva prueba de aceptación para cada iteración o se considerará que el equipo de desarrollo no realiza ningún progreso. Una prueba de aceptación es como una caja negra. Cada una de ellas representa una salida esperada del sistema. Es responsabilidad del cliente verificar la corrección de las pruebas de aceptación y tomar decisiones acerca de las mismas. La garantía de calidad es una parte esencial en la metodología XP. La realización de este tipo de pruebas y la publicación de los resultados deben ser lo más rápido posible, para que los desarrolladores puedan realizar con la mayor rapidez posible los cambios que sean necesarios.[43]

Técnicas de prueba

Hay dos maneras de probar cualquier producto construido, 1) si se conoce la función específica para la que se diseñó el producto, donde se aplican pruebas que demuestren que cada función es plenamente operacional, mientras se buscan los errores de cada función, 2) si se conoce el funcionamiento interno del producto, se aplican pruebas para asegurarse de que “todas las piezas encajan”. Es decir, que las operaciones internas se realizan de acuerdo con las especificaciones, y que se han probado todos los componentes internos de manera adecuada. Al primer enfoque de prueba se le denomina “prueba de caja negra” y al segundo “prueba de caja blanca”. [44]

Pruebas de caja blanca

Las pruebas de caja blanca, en ocasiones llamada prueba de caja de cristal, es un método de diseño que usa la estructura de control descrita como parte del diseño al nivel de componentes para derivar los casos de prueba. Al emplear los métodos de prueba de caja blanca, el ingeniero del software podrá derivar casos de prueba que: 1) garanticen que todas las rutas independientes dentro del módulo se han ejercitado por lo menos 1 vez, 2) ejerciten los lados verdaderos y falsos de todas las decisiones lógicas, 3) ejecuten todos los bucles en sus límites y dentro de sus límites operacionales, y 4) ejerciten estructuras de datos internos para asegurar su validez.[44]

Pruebas de caja negra

Las pruebas de caja negra, también denominadas, “pruebas de comportamiento”, se concentran en los requisitos funcionales del software. Es decir, permiten al ingeniero del software derivar un conjunto de

condiciones de entrada que ejercitarán por completo todos los requisitos funcionales de un programa. Las pruebas de caja negra no es una opción frente a las técnicas de caja blanca, es, en cambio, un enfoque complementario que tiene probabilidades de descubrir una clase diferente de errores que se descubrirían con los métodos de caja blanca.[44]

Las pruebas de caja negra tratan de encontrar errores en las siguientes categorías: 1) funciones incorrectas o faltantes, 2) errores de interfaz, 3) errores en estructuras de datos o en acceso a bases de datos externas, 4) errores de comportamiento o desempeño, y 5) errores de inicialización y término.[44]

Método de prueba de caja negra

Partición equivalente

La partición equivalente es un método de prueba de caja negra que divide el dominio de entrada de un programa en clases de datos a partir de las cuales pueden derivarse casos de prueba. Un caso de prueba ideal de manejo simple descubre una clase de errores, por ejemplo, (procesamiento incorrecto de todos los datos de caracteres) que, de otra manera, requeriría la ejecución de muchos casos antes de que se observe el error general. La partición equivalente se esfuerza por definir un caso de prueba que descubra ciertas clases de errores, reduciendo así el número total de casos de prueba que deben desarrollarse.[44]

Para la realización de las pruebas a la solución desarrollada por este trabajo de diploma, se tuvieron en cuenta algunos aspectos que permitieron definir finalmente cuales pruebas eran totalmente operativas. Entre los aspectos analizados están:

- 1) La aplicación a pesar de ser un portal web desarrollado para permitir peticiones concurrentes y en gran número, no constituye un objetivo de prioridad verificar que ciertamente soporta esta característica, dado que entre sus finalidades está el brindar una aplicación para administradores de red siendo estas normalmente utilizadas por unos pocos usuarios los cuales están destinados a realizar las actividades de control de las redes.
- 2) No resulta de prioridad el analizar cómo fueron desarrolladas cada una de las funcionalidades ni la verificación de cada una de ellas.
- 3) El desarrollo está más bien orientado a la entrega temprana de la primera versión funcional de una solución que permita a los administradores de red gestionar las extensiones que constituyen nuevas funcionalidades para el protocolo SNMP. En la que sean cumplidas correctamente todas

las funcionalidades acordadas al inicio del proceso, en las que no existan errores de interfaz, de estructuras de datos o de acceso a bases de datos.

Dada las condiciones acordadas anteriormente para el desarrollo de la solución propuesta por este trabajo de diploma, se decide entonces utilizar como Tipo de prueba, las pruebas de aceptación, como Técnica de prueba, las pruebas de caja negra y dentro de estas el método de partición de equivalencia.

3.1. Flujo de trabajo para el desarrollo de pruebas

Para el desarrollo de las pruebas que validarán el correcto funcionamiento de la solución, se realizará un flujo completo que comprende un caso de prueba para cada uno de los escenarios generales que dan cumplimiento a cada una de las Historias de Usuarios identificadas. Este flujo describirá el proceso necesario para desarrollar las pruebas con el objetivo de verificar cada escenario por los que está compuesto dicha solución. Para el correcto funcionamiento de cada uno de los escenarios de prueba el usuario debe seguir el flujo central de operaciones a fin de lograr el resultado de la prueba seleccionada.

En la *Figura 15- Caso de Prueba "Adicionar categoría"* se puede observar un ejemplo de caso de prueba correspondiente a la Historia de Usuario "Adicionar categorías". Para probar la solución fueron identificados un total de 27 casos de prueba. Aunque se expone solamente uno de ellos como ejemplo, los demás pueden encontrarse en el expediente de proyecto adjunto a este trabajo de diploma.

Condiciones de ejecución

- Debe encontrarse en correcto funcionamiento el servidor apache tomcat.
- Debe existir con anterioridad una categoría registrada en el sistema.

Figura 15- Caso de Prueba "Adicionar categoría"

Escenario	Descripción	Variable 1	Respuesta del sistema	Flujo central
EC 1.1 Adicionar una nueva categoría al sistema.	La categoría es insertada correctamente.	V (SOS / Todas)	El sistema muestra el mensaje "Categoría agregada correctamente"	1- Seleccionar la pestaña Gestión del repositorio. 2- Seleccionar la categoría superior a la que pertenecerá la nueva categoría. 3- Seleccionar la opción de menú Adicionar categoría. 4- Introducir el nombre de la nueva categoría. 5- Seleccionar el botón adicionar.
EC 1.2 Adicionar una nueva categoría con nombre de categoría existente.	La categoría no es insertada.	l(/cualquier nombre existente en la bd)	El sistema muestra el mensaje "Error al adicionar nueva categoría"	1- Seleccionar la pestaña Gestión del repositorio. 2- Seleccionar la categoría superior a la que pertenecerá la nueva categoría. 3- Seleccionar la opción de menú Adicionar categoría. 4- Introducir el nombre de la nueva categoría. 5- Seleccionar el botón adicionar.
EC 1.3 Adicionar nueva categorías sin especificar nombre nuevo o categorías superior	La categoría no es insertada.	l()	El sistema se queda a la espera de inserción de valore.	1- Seleccionar la pestaña Gestión del repositorio. 2- Seleccionar la categoría superior a la que pertenecerá la nueva categoría. 3- Seleccionar la opción de menú Adicionar categoría. 4- Introducir el nombre de la nueva categoría. 5- Seleccionar el botón adicionar.

Para realizar la prueba de cada uno de los escenarios con los que cuentan los casos de prueba identificados, se tienen variables específicas que permiten validar su funcionamiento. En la *Figura 16- Variables para el caso de prueba "Adicionar categoría"* se muestra un ejemplo de variables para el caso de prueba Adicionar categoría.

Figura 16- Variables para el caso de prueba "Adicionar categoría"

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
Variable 1	nombre	campo de texto	No	Nombre que identificará a la categoría en el sistema.
Variable 2	padre	lista desplegable	no	Nombre de la categoría en la que está contenida

Las pruebas realizadas con el objetivo de verificar el correcto funcionamiento de cada una de las Historias de Usuario, proporcionaron resultados satisfactorios, no así las Historias de Usuarios detectadas con algún tipo de error, las cuales pasaron a ser no conformidades quedando plasmadas en la Planilla de No Conformidades del expediente de proyecto, adjunto a este Trabajo de Diploma.

En la *Tabla 6- Tabla de ejemplo de no conformidades detectadas* se muestran las No Conformidades detectadas en el desarrollo de las pruebas para la Historia de Usuario Adicionar categoría. En dicha tabla solo se muestran 2 No Conformidades, las restantes del proceso para las demás Historias de Usuario incluyendo esta Historia de Usuario de encuentran en el expediente de proyecto.

Tabla 6- Tabla de ejemplo de no conformidades detectadas

Elemento	No	No Conformidad	Aspecto correspondiente	Etapa de detección	Clasificación	Estado NC	Respuesta del equipo de desarrollo
Aplicación	1	El nombre de la categoría acepta el carácter / como caracter válido para el nombre	Adicionar categoría	Prueba	S	02/06/2013 PD 04/06/2013 RA	Se corrigió el error encontrado. El sistema no acepta la entrada del carácter / en los nombres de categoría.
Aplicación	2	El sistema permite agregar nuevas categorías seleccionando un fichero como categoría padre	Adicionar categoría	Prueba	S	03/06/2013 PD 04/06/2013 RA	Se corrigió el error permitiendo adicionar categorías seleccionando como padre otra categoría.

3.2. Análisis de los resultados de las pruebas

Con el objetivo de validar el correcto funcionamiento e implementación de cada una de las Historias de Usuario se aplicaron pruebas específicas basadas en Historias de Usuario. En el gráfico de la *Figura 17- Resultado de las pruebas por iteración* se puede observar el proceso de desarrollo de pruebas en cada iteración así como las No Conformidades detectadas en cada una de ellas.

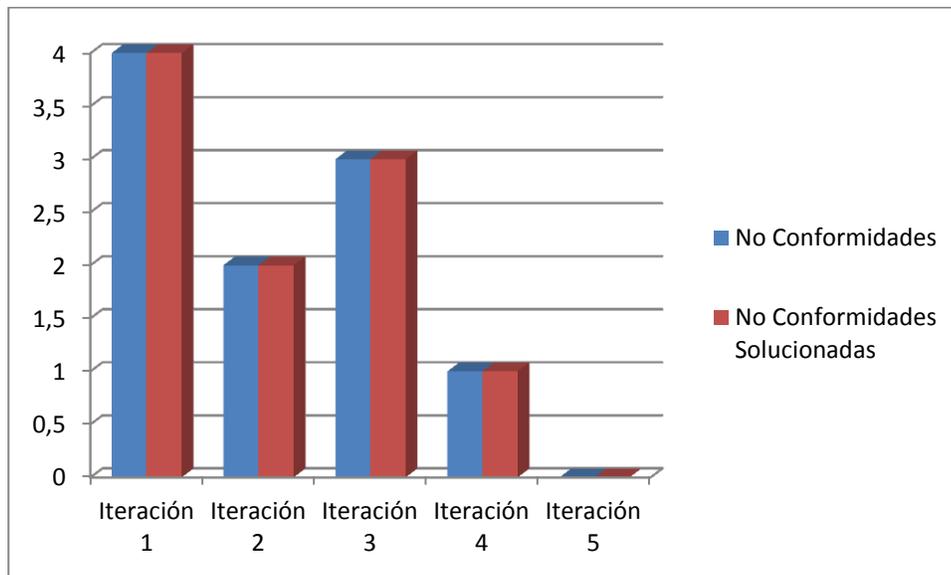


Figura 17- Resultado de las pruebas por iteración

La figura anterior muestra las No Conformidades detectadas en cada etapa de iteración y al mismo tiempo las que fueron corregidas por el equipo de desarrollo, evidenciándose en la última iteración la no existencia de No Conformidades terminándose la fase de pruebas con un total de 10 No Conformidades detectadas de las cuales 10 fueron resueltas por el equipo de desarrollo de forma satisfactoria.

3.3. Conclusiones del capítulo

Se definieron las pruebas que se aplicarían a la aplicación seleccionándose las pruebas de aceptación, la técnica de caja negra y el método de partición de equivalencia como las pruebas hábiles para guiar el proceso.

Se aplicaron las pruebas seleccionadas permitiendo contar luego de 4 iteraciones con una versión final de la aplicación.

Conclusiones generales

Al concluir el desarrollo de este trabajo de diploma y cumplidos cada uno de los capítulos en los que está dividido se arriban a las siguientes conclusiones:

- 1) Se realizó un estudio bibliográfico sobre la configuración y extensión de agentes SNMP, que permitió seleccionar «La implementación de módulos que pueden ser cargados dinámicamente», como la forma más adecuada de implementar y adicionar nuevas funcionalidades al protocolo.
- 2) Se seleccionó XP como metodología de desarrollo usando el UML como lenguaje de modelado y como herramienta CASE el Visual Paradigm. Se utilizó C y Java como lenguajes de programación, Liferay como gestor de contenidos y el IDE fue Eclipse.
- 3) Se diseñó un marco de trabajo que contiene la descripción necesaria para realizar la extensión de un agente SNMP.
- 4) Se diseñó un repositorio que organizado por categorías permite almacenar las extensiones que sean implementadas.
- 5) Se desarrolló una aplicación que permite gestionar las extensiones que son implementadas para extender los agentes SNMP.
- 6) Se realizaron pruebas de aceptación utilizando la técnica caja negra y el método de partición de equivalencia lo que permitió validar el correcto funcionamiento de la solución.

Recomendaciones

Con el objetivo de mejorar y extender el alcance de la Solución de configuración y extensión de agentes SNMP para el monitoreo de sistemas, se recomienda:

- 1) Realizar la implementación de las funciones necesarias que permitan a la aplicación la gestión de extensiones para otros sistemas operativos, así como la consulta de información a otros agentes cuya versión de NET-SNMP sea superior a la 5.0.11.2.
- 2) Rediseñar, extender y probar la Guía de Extensión y Configuración de Agentes SNMP, permitiendo verificar su validez para el desarrollo de extensiones destinados a otros sistemas y equipos.

Referencias Bibliográficas

1. Rodríguez, F.I.R., *Temas sobre SNMP*.2012, La Habana.
2. Araña, N.B., *Modelo de Gestión de Seguridad con soporte a SNMP*.2005, Pontificia Universidad Javeriana.
3. Sevilla, U.E., *Introducción a SNMP*.2005, España.
4. Forge, S. *NET-SNMP*. 2013; Available from: <http://www.net-snmp.org/>.
5. García, A.S., *Gestión de redes con Necromantux*.2006, Coruña.
6. Corp, Z. *Network Management Framework, Element Management, NMS, EMS and Systems Management, WebNMS*. 2009; Available from: http://www.webnms.com/snmputilities/help/quick_tour/snmp_and_mib/snmpmib_miboverview.html.
7. McCloghrie, K. *RFC - 1213 Management Information Base for Network Management*. 1991; Available from: <http://tools.ietf.org/html/rfc1213#page-6>.
8. CHEN, H.-r. and Y.-m. ZHANG, *SNMP and Its Application*. Journal of North China Electric Power University, 2001. **3**: p. 016.
9. M, G.d.T.d.I.R.R. *Petición de Comentarios: 1155 Performance Systems International* Mayo, 1990; Available from: <http://www.normes-internet.com/normes.php?rfc=rfc1155&lang=es>.
10. Sasae, K., *Rethinking Japan-US Relations*1994: Brassey's.
11. Rahmel, D., *Joomla!*2008.
12. EE., L.P. *Sitio web oficial Liferay Portal*. 2013; Available from: <http://www.liferay.com>.
13. Diaz, O. and J.J. Rodriguez, *Portlets as web components: An introduction*. Journal of Universal Computer Science, 2004. **10**(4): p. 454-472.
14. Miguel Ángel Conde González, J.C.-b.G., Rosa María Martín Moren, Inmaculada González Pérez & Francisco José García Peñalvo, *ARQUITECTURA PARA UN LMS BASADA EN PORTLETS: CLAYNET 2.0. s.l.* Vol. Vol. 7. N°2. 2006, Universidad DSalamanca.
15. Sanchez, M.A.M., *Metodologías De Desarrollo De Software*, 2004, Jun.
16. Orjuela, A. and M. Rojas, *Las metodologías de desarrollo ágil como una oportunidad para la ingeniería del software educativo*. Avances en Sistemas e Informática, 2008. **5**(2): p. 159-170.
17. Rodríguez, F.E.R., *Aplicación Web que guíe la implantación del proceso base de Ingeniería de Requisitos en las empresas de producción de Software*.2011, Habana, Cuba.
18. Group, O.M. *UML® Resource Page*. 1997-2013; Available from: <http://www.uml.org/>.

19. Rodríguez, F.E.R., *Aplicación Web que guíe la implantación del proceso base de Ingeniería de Requisitos en las empresas de producción de Software*. Habana2011, Habana, Cuba.
20. Basurto, M.A.P. and J.M.C. Espín, *Introducción a la programación en C*. Vol. 42. 2010: Univ. Politèc. de Catalunya.
21. López, Á., *JAVA la programación del futuro*.1997, Montevideo: Imprenta Rosgal.
22. Moncayo, G., *Curso Fundamentos de Programación en Java*.
23. Schiefer, K.B. and G. Valentin, *DB2 universal database performance tuning*. Data Engineering Bulletin, 1999. **22**(2): p. 12-19.
24. Foundation, A.S. *Apache Derby*. 2004-2012; Available from: <http://db.apache.org/derby/>.
25. Group, T.h.D. *HyperSQL*. 2001-2012; Available from: <http://hsqldb.org/>.
26. Stonebraker, M., et al., *The design and implementation of INGRES*. ACM Transactions on Database Systems (TODS), 1976. **1**(3): p. 189-222.
27. MySQL, A., *Mysql database*, 2000.
28. Loney, K., *Oracle Database 10g: The Complete Reference*2004: McGraw-Hill/Osborne.
29. Stones, R. and N. Matthew, *Beginning databases with PostgreSQL*2001: Wrox Press Chicago.
30. Agrawal, S., et al. *Database tuning advisor for Microsoft SQL Server 2005: demo*. in *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*. 2005. ACM.
31. Maritzen, L.M. and R.D. Dimaandal, *Java-to-Database Connectivity Server*, 1999, Google Patents.
32. Group, T.P.G.D., *PostgreSQL 9.1.0 Documentation*. 1996-2011.
33. Chavarría, R.E., *IDE ECLIPSE, breve guía*.2006, Universidad de Antioquia.
34. Letelier, P., *Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP)*. 2006.
35. Fernandez Escribano, G., *Introducción a Extreme Programming*, 2001, Recuperado el.
36. Bass, L., P. Clements, and R. Kazman, *Software Architecture in Practice, 2nd edn. SEI Series in software engineering*, 2003, Addison-Wesley Pearson Education, Boston.
37. Bascón Pantoja, E., *El patrón de diseño Modelo-Vista-Controlador (MVC) y su implementación en Java Swing*. Revista Acta Nova, 2011. **2**(4).
38. Tedeschi, N. *¿Qué es un Patrón de Diseño?* . 2013; Available from: <http://msdn.microsoft.com/es-es/library/bb972240.aspx>.
39. Ernesto Marquina, J.D.P., *Guía de Patrones, Prácticas y Arquitectura .NET*2008, Contoso.
40. Larman, C., *UML y patrones*.1999: Pearson.
41. Tristán, F.B., *Estándares de Codificación .Net*, Kualtus, S.A. de C.V.

42. Hommel, S., *Code Conventions for the Java Programming Language*, 2003.
43. Maite Rodríguez Corbea, M.O.P., *la Metodología XP aplicable al desarrollo del software educativo en Cuba2007*, Universidad de las Ciencias Informáticas.
44. Pressman, R.S. and J.M. Troya, *Ingeniería del software*1988: McGraw Hill.

Bibliografía

1. Perkins, D. and E. McGinnis, *Understanding SNMP MIBs*1997: Prentice-Hall, Inc.
2. Hiremagalur, D., et al., *A SNMP MIB to manage GMPLS with General Constraints support*. 2012.
3. WU, X.-q., D.-b. XIAO, and Y.-n. CHANG, *SNMP MIB Data Conversion Based on NETCONF*. Computer Engineering, 2010. **2**: p. 024.
4. Dr. Edmund Whelan, C., *SNMP and Potential ASN.1 Vulnerabilities*2002, SANS Institute InfoSec Reading Room.
5. Levi, D.B., B. Stewart, and P. Meyer, *Simple network management protocol (SNMP) applications*. 2002.
6. Vidal, P.E.V., *Simple Network Management Protocol*2009, Departamento de Electrónica - UTFSM.
7. Luo, C., K. Long, and B. Wang. *Research and design of module and symbol table in ASN. 1 compiler*. in *Computer Application and System Modeling (ICCASM), 2010 International Conference on*. 2010. IEEE.
8. Martín Román, A., *Monitorización de tráfico y encaminamiento en redes IP*. 2013.
9. Castro Lechtaler, A., et al. *Monitoreo y optimización a sistemas SCADA convencionales con un enfoque TCP/IP*. in *XVIII Congreso Argentino de Ciencias de la Computación*. 2012.
10. Martínez, P.I.C., *MODELO CONCEPTUAL / MODELO DE DOMINIO*.2012, Universidad Simón Bolívar . Ing . de Software.
11. Krishna, N., *FAST NETWORK DISCOVERY USING SNMP MULTI-CAST*, 2012, US Patent 20,120,226,798.
12. Schonwalder, J. *Characterization of SNMP MIB modules*. in *Integrated Network Management, 2005. IM 2005. 2005 9th IFIP/IEEE International Symposium on*. 2005. IEEE.
13. Harrington, D., R. Presuhn, and B. Wijnen, *An architecture for describing simple network management protocol (SNMP) management frameworks*, 2002, rfc 3411, Dec.
14. INFORMACIÓN, C.R.D., *Administración de Redes Información de administración Protocolos de administración*2012, UNIVERSIDAD TECNOLÓGICA NACIONAL REGIONAL SANTA FE.
15. Rosales Briceño, C., *PROTOCOLO SNMP (PROTOCOLO SENCILLO DE ADMINISTRACIÓN DE REDES)*. TELEMATIQUE, 2010. **3**(1): p. 90-102.
16. Rizo Valero, D. and P.J. Ponce de León Amador, *Seminario: el entorno de desarrollo Eclipse*. Programación 3, 2011.

17. Daniel, A. and D. León. *SLD036 Módulo para el análisis de secuencias de la Plataforma De Servicios Bioinformáticos*. in *Informática Salud 2013*. 2012.
18. Martínez Alfonso, E., *Portal web con tecnología liferar*. 2012.
19. Rodríguez, G. and J. Alberto, *Diseño de una Plataforma Web Empresarial para dar Soporte a los procesos de Administración de Clientes; Cuentas de Ahorro y préstamos en la microoperativa de Ahorro y Crédito del discapacitado de Loja*, 2013.
20. Gil, G.D., et al. *Metodologías ágiles y desarrollo basado en el conocimiento, evaluación cuantitativa de F/OSS para la reutilización, Normas ISO y su aplicación en centros educativos*. in *XIV Workshop de Investigadores en Ciencias de la Computación*. 2012.
21. Bascón Pantoja, E., *El patrón de diseño Modelo-Vista-Controlador (MVC) y su implementación en Java Swing*. *Revista Acta Nova*, 2011. **2**(4).
22. Paradigm, V., *Visual Paradugm*. Online [www. visual-paradigm. com](http://www.visual-paradigm.com)]. Last November, 2011.

Glosario de términos

SNMP: Protocolo Simple de Administración de Red. Es un protocolo de la capa de aplicación que facilita el intercambio de información de administración entre dispositivos de red.

MIB: Base de Información de Administración. Es un tipo de base de datos que contiene información jerárquica, estructurada en forma de árbol, de todos los dispositivos gestionados en una red de comunicaciones. Es parte de la gestión de red definida en el modelo OSI. Define las variables usadas por el protocolo SNMP para supervisar y controlar los componentes de una red.

RFC: Petición de comentarios. Es un documento cuyo contenido es una propuesta oficial para un nuevo protocolo de la red Internet (originalmente de ARPANET), que se explica con todo detalle para que en caso de ser aceptado pueda ser implementado sin ambigüedades.

SMI: Básicamente trata de definir la estructura de la MIB. Se indicarán aquí aspectos como el tipo de datos que podrá albergar la MIB, como representarse y denominarse los recursos en la MIB, etc.

ASN: Notación sintáctica abstracta. Es una norma para representar datos independientemente de la máquina que se esté usando y sus formas de representación internas. Es un protocolo de nivel de presentación en el modelo OSI. El protocolo SNMP usa el ASN.1 para representar sus objetos gestionables.

Scripts: es un programa usualmente simple, que por lo regular se almacena en un archivo de texto plano.

ANSI: ANSI C es un estándar publicado por el Instituto Nacional Estadounidense de Estándares (ANSI), para el lenguaje de programación C

Liferay: Liferay es un portal de gestión de contenidos de código abierto escrito en Java. Se creó en el año 2000 en principio como solución para las organizaciones sin ánimo de lucro.

Agente: Es una aplicación instalada en un dispositivo el cual será administrado mediante el protocolo SNMP.

NMS: Un sistema administrador de red (NMS) ejecuta aplicaciones que supervisan y controlan a los dispositivos administrados.

IPV4, 6: Es un protocolo de Internet, y fue el primero implementado de forma extensiva.

OID: El OID (Object Identifier – Identificador de objetos) describe la dirección del objeto en el árbol MIB.

API: Es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizada por otro software como una capa de abstracción.

CMS: Es un programa que permite crear una estructura de soporte (framework) para la creación y administración de contenidos, principalmente en páginas web.

Diagrama Entidad-Relación: es una herramienta para el modelado de datos que permite representar las entidades relevantes de un sistema de información así como sus interrelaciones y propiedades.