UNIVERSIDAD DE ZARAGOZA

CENTRO POLITÉCNICO SUPERIOR

PROYECTO FIN DE CARRERA

Ingeniería de Telecomunicación

DESARROLLO DE UN GESTOR SNMPv3 CON INTERFAZ WEB PARA LA GESTIÓN DE INFORMACIÓN TÉCNICA EN ESCENARIOS DE TELEMONITORIZACIÓN DOMICILIARIA

Verónica García Latorre

Director:

Nelia Lasierra Beamonte

Ponente:

Álvaro Alesanco Iglesias

Dpto. de Ingeniería Electrónica y Comunicaciones

Marzo de 2012

Agradecimientos

En primer lugar, quiero agradecer a ***Álvaro*** *la oportunidad que me ha brindado para realizar este proyecto fin de carrera. Sin duda, lo más interesante que he aprendido en la carrera, lo he aprendido en sus clases. También quiero agradecer a* ***Nelia*** *por su apoyo incondicional, su tiempo, su esfuerzo, su paciencia, su eficiencia, por haberme ayudado cuando estaba atascada, por haberme motivado cuando más lo necesitaba... Sin su ayuda, no lo habría conseguido. Siempre estaré muy agradecida.*

A mis padres y hermanos, les dedico este proyecto fin de carrera como agradecimiento al apoyo recibido durante estos años de estudio, al sacrificio que han tenido que realizar para hacer posible que yo haya llegado hasta aquí, por vuestro cariño y por todo lo que habéis hecho por mi... Gracias.

A mis compañeros a lo largo de la carrera, por todos los buenos momentos que hemos pasado juntos. No los quiero nombrar para evitar olvidarme de alguno pero, en especial, me gustaría agradecer a ***Nacho*** *por haber sido mi gran compañero de prácticas y por saber cómo ponerme las pilas. A* ***Andreas****, por todo el tiempo que has pasado ayudándome. A* ***Pedro****,* ***Yas*** *y* ***Rubén****, porque no todos los días se conoce gente como vosotros.*

A mis compañeros de trabajo, ***José*** *y* ***Rubén****, por todas las horas que os he estado contando mis problemas con el proyecto y me habéis escuchado. Gracias también a los que me amenizan la hora de la comida.*

A ***Adam****, porque tú me has aguantado más que nadie.*

A todos los que me prestaron su tiempo para poder realizar las evaluaciones. En especial a ***María*** *y* ***Cris****, porque no tenéis mucho y aún así, vinisteis tan pronto como os avisé.*

Y, por último, a ***Laura****, mi sobrina, que aunque todavía es muy joven para darse cuenta, basta un minuto con ella para sacarme una sonrisa.*

DESARROLLO DE UN GESTOR SNMPv3 CON INTERFAZ WEB PARA LA GESTIÓN DE INFORMACIÓN TÉCNICA EN ESCENARIOS DE TELEMONITORIZACIÓN DOMICILIARIA

RESUMEN

En este proyecto fin de carrera se ha desarrollado un gestor SNMPv3 (*Simple Network Management Protocol*) con interfaz web para la gestión técnica de dispositivos en escenarios de telemonitorización domiciliaria. Para ello, este gestor se adapta al agente desarrollado en (1), el cual implementa una MIB (*Management Information Base*) que permite, además de gestionar datos de la información técnica de los dispositivos médicos que están conectados, gestionar los datos del dispositivo concentrador que se encarga de recogerlos, así como de definir y configurar alarmas y eventos.

Para facilitar la configuración de las tablas SNMP y el acceso a la visualización de resultados, este proyecto desarrolla un interfaz gráfico amigable e intuitivo, proporcionando una gestión transparente a la utilización de la arquitectura SNMP implementado en Java, con Servlets y JSP con el uso de algunos frameworks como Struts2 o Tiles y realizando conexiones seguras utilizando el protocolo TLSv1 sobre HTTP.

El agente desarrollado en (1) permite tanto la gestión de los datos como la configuración de alarmas y eventos que, una vez configuradas, enviarán mensajes asíncronos al gestor. Para la visualización de esos datos, la aplicación cuenta con herramientas adicionales que los representan gráficamente. En cuanto a los mensajes asíncronos, la aplicación incorpora una herramienta para la captura y visualización de dichos mensajes.

Por último, se han realizado evaluaciones sobre usuarios de diferentes perfiles para testear si se han alcanzado los objetivos previstos para este proyecto y conocer los puntos en los que se puede mejorar. Los resultados obtenidos han sido muy satisfactorios.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción y Objetivos

1.1 Introducción

Actualmente vivimos en una sociedad con un progresivo aumento de la esperanza de vida de su población. De manera que, al aumentar el número de personas con edad superior a 60 años, también lo hace el número de pacientes crónicos que requieren atención sanitaria periódica. Estas revisiones periódicas repercuten en los gastos en materiales médicos y en los desbordamientos en los centros de salud impulsando la necesidad de diseñar y desarrollar nuevos servicios de telemedicina que disminuyan los gastos y mejoren la calidad de vida de estos pacientes.

Dentro de la telemedicina, el desarrollo de sistemas de telemonitorización es, a dia de hoy, un ambito de gran interés ya que permite al personal sanitario realizar el seguimiento de los pacientes sin necesidad de que éstos vayan a los centros de salud. Desde el punto de vista sanitario, se evita, en cierta medida, los desbordamientos que los pacientes crónicos ocasionan: se reducen listas de espera, se disminuye el gasto sanitario que conllevan y se facilita la accesibilidad al histórico de datos. Desde el punto de vista del paciente, no requiere desplazamientos ya que el control se realiza desde el hogar de los pacientes e implica un mayor control de su enfermedad debido a que las mediciones se realizan con mayor frecuencia, lo que se traduce en un aumento en su calidad de vida.

En la figura 1.1 se muestra un esquema de un sistema general de telemonitorización de pacientes. Como se puede observar, este se compone de diferentes entidades:

1. **El Hogar del Paciente**, donde se encuentra el paciente crónico, los dispositivos médicos y el CE *(Compute Engine)*. Un CE es un dispositivo concentrador de datos utilizado en casa de los pacientes para recoger la información de los MDs *(Medical Devices)* y enviarlos posteriormente al centro correspondiente.

2. **El Centro Remoto**, lugar donde se almacenaran los datos enviados por el CE y donde se encuentra el personal médico (gestor clínico) encargado de controlar la evolución de los datos clínicos, es decir, las medidas que realiza el paciente desde su hogar, para detectar posibles cambios anómalos en los mismos.

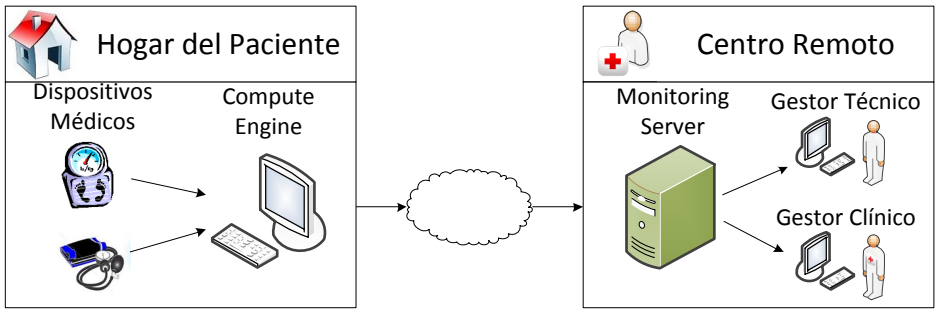


Figura 1.1: Esquema de un Sistema General de Telemonitorización de pacientes.

Por otro lado, tan importante es recoger y transferir los datos clínicos como asegurarse de que los datos se adquieran y envíen correctamente. La gestión técnica es, por tanto, un ámbito de gran interés en este tipo de escenarios. Una solución interesante para abordar esta problemática la encontramos en (1) (2). Esta solución se basa en la implementación de un agente SNMP *(Simple Network Management Protocol)* en casa del paciente el cual envía a un centro remoto los datos recogidos acerca del funcionamiento del CE y los MDs. Además de ser una de las pocas soluciones que a día de hoy abordan esta problemática, la solución propuesta

permite proporcionar seguridad y eficiencia en la transferencia de información al hacer uso de la arquitectura de gestión de redes SNMP.

A pesar de que SNMP es un protocolo simple, las herramientas de gestión que existen en la actualidad no son sencillas de manejar y precisan que el gestor externo tenga conocimientos del funcionamiento de esta arquitectura. Por ello, y puesto que la gestión de estos escenarios podría ser de interés del personal médico, surge la necesidad de implementar una interfaz amigable que permita el control de los datos recibidos sin necesidad de conocer el protocolo. Como solución a este problema, este proyecto fin de carrera implementa una extensión a la solución presentada en (2) proponiendo el diseño y desarrollo de una aplicación Web con un gestor SNMP versión 3 integrado que permita la gestión técnica de los datos enviados por el agente.

De esta forma, el gestor SNMP integrado en la aplicación establecerá comunicaciones SNMP con el agente instalado en casa del paciente permitiendo que tanto la visualización de estos datos como las actividades de gestión que éste permite se realicen a través de un interfaz amigable y facil de utilizar.

1.2 Estado del arte

SNMP es una arquitectura de gestión de redes, para la gestión de redes TCP/IP (implementada por la mayoria de fabricantes) que garantiza seguridad y eficiencia en las comunicaciones. Por ello, existen actualmente numerosas herramientas (gestores SNMP) que permiten la comunicación con agentes de la arquitectura y por tanto la gestión de dispositivos y redes mediante la misma.

De manera general, estas herramientas que permiten el acceso a diferentes agentes y por tanto visualización de las MIBs *(Management Information Base)* que implementan se denominan MIB-Browser. Uno de los más populares es el de MG-Soft cuyo interfaz se muestra en la figura 1.2. Está aplicación de MG-Soft requiere instalación y permite visualizar la jerarquía de diferentes MIBs en forma de

árbol con información adicional sobre cada nodo. El primer paso para utilizar esta herramienta es conectarse al agente al que se quiera preguntar la información. Para ello, es necesario conocer la dirección IP del dispositivo y el tipo de autenticación que requiere. Una vez conectado al dispositivo, se podrá pedir o manipular toda la información que se desee al mismo tiempo que se recibirán mensajes asíncronos que sean enviados desde el dispositivo. Para pedir o manipular esa información hay que conocer los mensajes que se pueden enviar y cómo está llegando la información pedida para poder interpretarla correctamente.

En la parte izquierda de la figura 1.2, se observa una MIB en forma de árbol, tal y como la muestra MG-Soft. En la parte derecha, se observa el formato en que MG-Soft devuelve la información pedida.

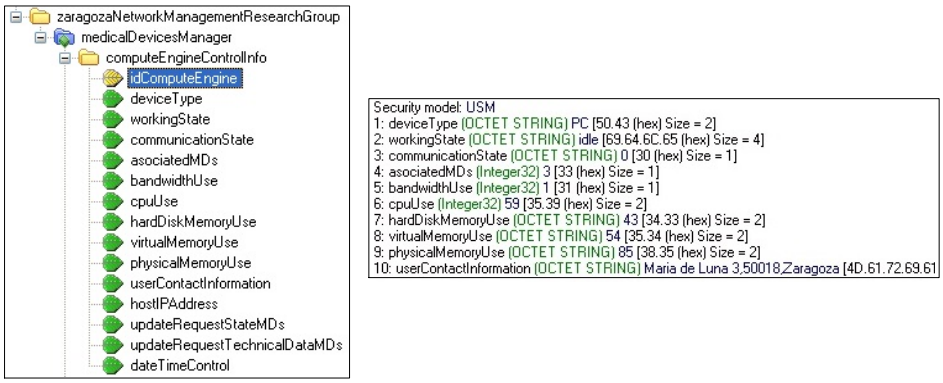


Figura 1.2: Vistas del programa MG-Soft.

Existen otras herramientas como CACTI que permiten de forma más dinámica y sencilla la gestión de redes mediante SNMP. CACTI hace uso de herramientas RRDTools *(Round Robin Database Tool)* para recoger y almacenar los datos gestionados de manera periódica y así ofrecer un histórico de la información gestionada que puede mostrarse de forma gráfica. Estos gráficos pueden ser configurados por el usuario. Los dispositivos a telemonitorizar también necesitan ser añadidos así como el gestor tiene que autenticarse al pedir información sobre los mismos.

Estas herramientas son útiles para el manejo de la información recibida mediante SNMP pero requieren conocimientos sobre el mismo. Para el caso de

telemonitorización domiciliaria, se necesita crear una interfaz que se pueda manejar de manera sencilla por cualquier persona que no tenga un perfil técnico. Esto facilita que, en un momento dado, un médico pueda acceder a conocer datos técnicos de los dispositivos sin conocer cómo se organizan en una estructura de MIB o qué mensajes deben enviarse para visualizar los datos. Por este motivo, para dar mayor flexibilidad a la aplicación, como se demuestra en (3), en este proyecto fin de carrera se ha desarrollado una aplicación web que ayude a la visualización y gestion de los datos disponibles en el agente de telemonitorización propuesto en (2). Con esta aplicación, cualquier gestor técnico será capaz de interpretar los datos y gestionar las alarmas que considere oportunas desde cualquier dispositivo con conexión a Internet y sin necesidad de instalar ningún software ni adquirir conocimientos adicionales.

1.3 Propuesta

En este proyecto se propone el desarrollo de una aplicación Web integrada con un gestor SNMPv3 para la gestión de información técnica en escenarios de telemonitorización domiciliaria. De esta forma se permite el acceso a la gestión remota desde cualquier dispositivo con acceso a Internet y se evita la instalación de cualquier herramienta software adicional.

Un esquema de la aplicación se muestra en la figura 1.3, donde se puede observar como el gestor SNMP Web se puede conectar simultáneamente a varios agentes para obtener información sobre los mismos o los dispositivos médicos asociados.

Como se observa en la figura, esta aplicación tiene dos partes distinguidas. Por un lado, la aplicación integra un gestor que se encarga de la comunicación con el agente por medio del protocolo SNMP versión 3. De esta forma, recibe los datos recogidos por los dispositivos de manera segura y eficiente. Además, esta parte también será la encargada de enviar los mensajes necesarios para configurar nuevas alarmas o recoger los mensajes asíncronos enviados por el agente.

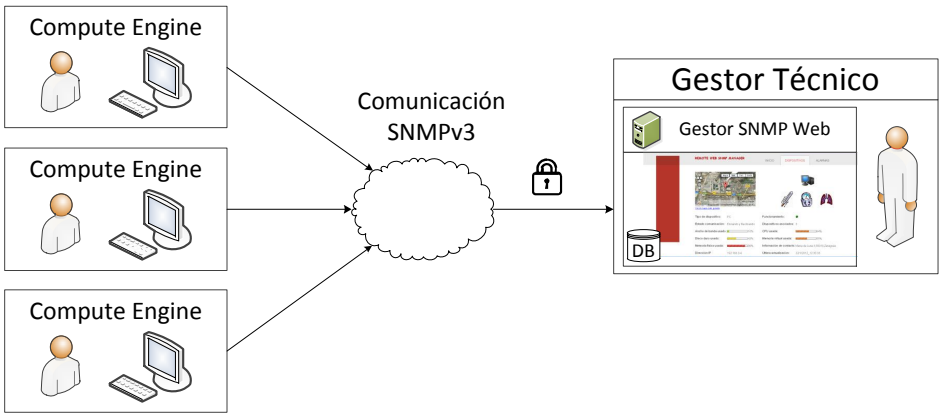


Figura 1.3: Esquema General de la Arquitectura de la Aplicación de Gestión Propuesta.

Concretamente, el gestor propuesto va a adaptarse al agente desarrollado en (1). Este agente implementa la MIB MD *(MIB Medical Devices)*. Esta MIB permite, además de gestionar información técnica de los dispositivos médicos que están conectados a él, gestionar datos del CE, así como definir y configurar alarmas y eventos.

Por otro lado, se propone desarrollar una aplicación Web que proporcione un interfaz gráfico amigable e intuitivo para el gestor y que facilite, por lo tanto, la configuración de tablas SNMP y el acceso a la visualización de resultados. Así, a través de la aplicación Web se proporciona una gestión transparente a la utilización de la arquitectura SNMP con los beneficios en la configuración que ello conlleva.

A través del gestor Web se podrá acceder de forma sencilla a la información relativa al funcionamiento de un dispositivo médico que un paciente tiene en su casa, así como a la información propia del dispositivo concentrador de datos. Puesto que estos agentes desarrollados van a permitir la configuración de alarmas (relativas al funcionamiento de los dispositivos que éste controla) y posterior envío de mensajes asíncronos enviados por un agente al gestor, la aplicación propuesta incorpora además herramientas para la captura y visualización de dichos mensajes así como para recoger y registrar las incidencias técnicas comunicadas por cada uno de los dispositivos.

Para proporcionar un entorno completo de gestión, el gestor Web propuesto incorpora herramientas adicionales como, por ejemplo, herramientas para la representación gráfica de los datos que permitan evaluar de manera más precisa los resultados obtenidos. Además, se han desarrollado diferentes vistas del sistema en función de los permisos de los que disponga el usuario.

Otro aspecto importante a tener en cuenta a la hora de diseñar las comunicaciones es la seguridad. Este gestor tiene dos enlaces de comunicación. Por un lado, la comunicación SNMP con el agente que se realiza sobre la versión 3 de la arquitectura. Por otro lado, se utiliza el protocolo TLSv1 sobre HTTP. Además, se incorpora autenticación de usuarios para acceso al sistema.

Por último, destacar que para el desarrollo de la aplicación Web se ha seleccionado el Framework Struts2 (4) el cual basa su funcionamiento en el paradigma de programación MVC (modelo, vista, controlador) que ofrece una división de las diferentes partes que conforman una aplicación: datos de la aplicación, interfaz de usuario y lógica de la aplicación, facilitando así su desarrollo y posterior modificación si fuera necesario.

1.4 Objetivos

El objetivo de este proyecto es el desarrollo de un gestor SNMP Web para la gestión técnica en la telemonitorización de pacientes en el hogar. Para llevar a cabo el proyecto, se han completado los siguientes objetivos:

1. Desarrollo de un gestor SNMP.
   (a) Documentación y estudio de la arquitectura SNMP
   (b) Estudio Agente y MIB MedicalDevices (AgenteMD) del PFC (1)
   (c) Diseño de las comunicaciones entre el agente y el gestor utilizando la versión 3 del protocolo SNMP que proporciona seguridad.
   (d) Desarrollo de una herramienta para la gestión de mensajes asíncronos.

2. Desarrollo de la aplicación Web.

   (a) Documentación y estudio de applicaciones Web y Frameworks asociados.

   (b) Diseño y desarrollo del software que da soporte al gestor Web.

   (c) Implementación de técnicas de manejabilidad y usabilidad prestando especial atención a un diseño sencillo e intuitivo.

   (d) Diseño de una base de datos para el almacenamiento de los usuarios con acceso al sistema así como sus roles, los agentes SNMP asociados a cada uno de los usuarios y los mensajes asíncronos recibidos.

   (e) Implementación de herramientas que permitan la visualización de la información en forma de gráficos.

3. Integración de ambas herramientas para poder acceder a visualizar datos y gestionar alarmas y eventos.

   (a) Evaluación del sistema final.

1.5 Organización de la memoria

En el capítulo 1 se ha desarrollado una breve introducción al PFC así como sus objetivos principales.

En el capítulo 2 se describen los diferentes módulos que integran el sistema (Gestor SNMP y soporte Web) describiendo los principales materiales utilizados para el desarrollo así como sus ventajas.

En el capítulo 3 se describe la primera parte del desarrollo tecnológico del gestor. En concreto se describen las funcionalidades SNMP y de la aplicación Web que intervienen en los procesos de visualización de datos del agente.

En el capítulo 4 se describe la segunda parte del desarrollo tecnológico del gestor. En concreto se describen las funcionalidades SNMP y de la aplicación

Web que intervienen en los procesos de gestión de alarmas y captura de mensajes asíncronos del agente.

En el capítulo 5 se describe la evaluación del sistema que se ha realizado sobre los usuarios y se comenta sus resultados.

Finalmente, en el último capítulo se exponen las conclusiones y lineas futuras del proyecto.

Capítulo 2

Arquitectura del Sistema

2.1 Descripción General

El gestor Web SNMP se encarga de establecer la comunicación con los agentes desarrollados en (1) que recogen la información de los dispositivos médicos (agentesMD a partir de ahora) y de permitir la visualización de la misma así como la configuración de determinados parámetros asociados a esos dispositivos o al propio agenteMD. Todo ello se ha incluido en la misma aplicación, por lo que solo será necesario instalar el software en un servidor al que se podrá acceder desde cualquier dispositivo con conexión a Internet.

La arquitectura general del sistema se compone de dos bloques:

1. **El gestor SNMP**. Es el encargado de establecer la comunicación con el agenteMD para pedir los datos que se deseen mediante el protocolo SNMPv3. También es el encargado de capturar los mensajes asíncronos (llamados Traps) que envía el agente cuando se ha activado una de las alarmas que nuestro gestor ha configurado previamente.

2. **Aplicación Web**. Se encarga de facilitar la visualización de los datos recogidos por el gestor SNMP además de permitir la creación y configuración de distintas alarmas y eventos asociados a los dispositivos gestionados por el agenteMD. Está creada siguiendo el patrón de arquitectura MVC *(Modelo Vista Controlador)*.

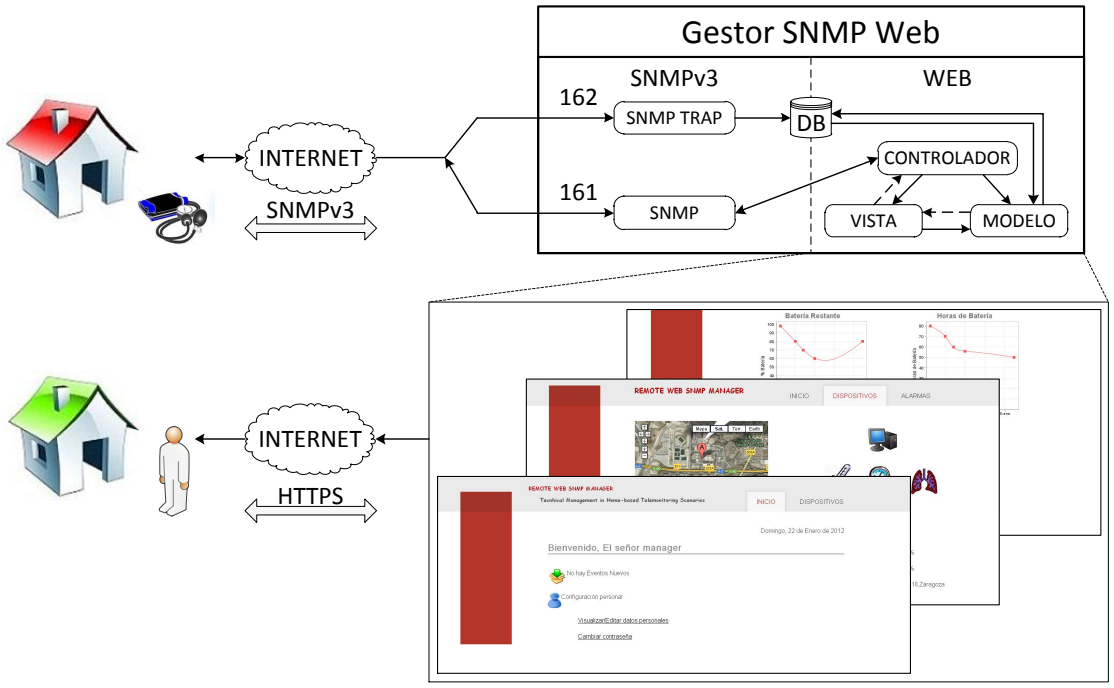


Figura 2.1: Esquema Global del Sistema.

El esquema global del sistema se puede ver en la figura 2.1. El gestor técnico se conecta a la aplicación Web vía Internet. Para ello se necesita autenticación mediante usuario y contraseña. Una vez verificada su identidad, la aplicación Web mostrará los traps que se han registrado mientras ha estado desconectado y los agentesMD que ese usuario tiene asociados. Cuando éste desee visualizar cualquier información sobre algún dispositivo o sobre el propio CE, el gestor SNMP se comunica con el agenteMD enviando los mensajes SNMP necesarios preguntando esa información. Una vez recibida la respuesta del agenteMD, la aplicación Web la muestra de manera sencilla y transparente permitiendo ver los últimos datos recibidos, un histórico de ellos o visualizándolos gráficamente.

Lo mismo ocurre cuando el usuario desea configurar una alarma. La aplicación Web muestra los datos a introducir con un formulario intuitivo y el gestor SNMP es el encargado de enviar los mensajes para crear esa alarma.

Además de establecer comunicaciones con el agente, el gestor SNMP se encarga de capturar los Traps que éste le envíe e informar de ellos al usuario instantáneamente si éste está conectado en ese momento o registrarlos e informar cuando realice la próxima conexión.

Por último, en la figura 2.2, se muestran las comunicaciones que establecen los dos bloques del sistema. El bloque del gestor SNMP se comunica con el agente mediante mensajes SNMP que se explicarán con más detalle en los próximos apartados mientras que el de la aplicación Web mantiene la comunicación con el navegador Web mediante peticiones HTTP.

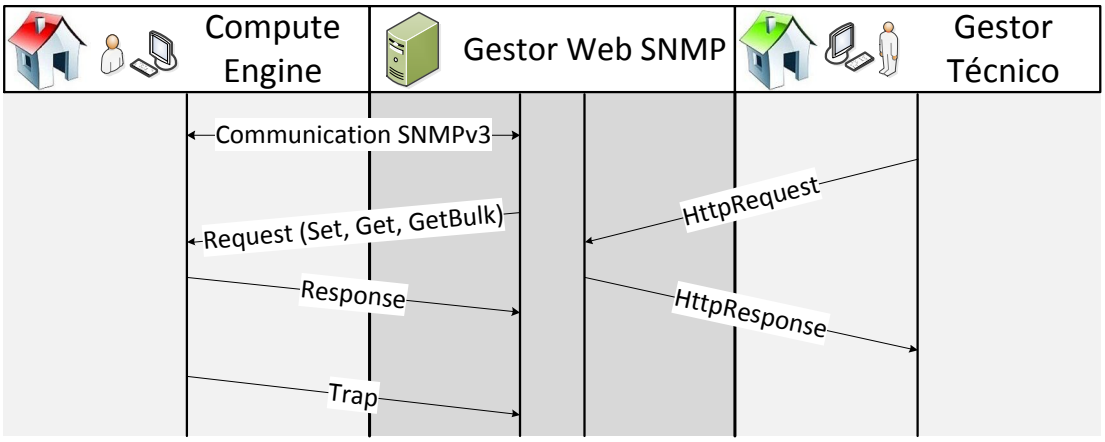


Figura 2.2: Comunicaciones establecidas en los distintos bloques.

2.2 Estructura Aplicación Web

Una aplicación Web es una aplicación a la que se puede acceder a través de Internet mediante un navegador. Esta aplicación puede ser utilizada por miles de usuarios pero sólo está en un servidor, lo que facilita el mantenimiento y su actualización, permitiendo ver los cambios a todos los usuarios instantáneamente al mismo tiempo que no requiere instalar ningún programa para su visualización.

Para el desarrollo de la aplicación Web, se ha seleccionado un entorno Java con la utilización de Servlets y páginas JSP *(Java Server Pages)* y basado en la arquitectura MVC *(Model View Controler)*. Se ha escogido Java por ser una tecnología Web versátil y eficiente, con multitud de herramientas OpenSource para satisfacer todas las necesidades y compatibilidad con JavaScript y AJAX. Además, para su implementación se han seleccionado los Frameworks Struts2, Tiles e Hibernate y se utiliza el servidor Apache Tomcat 6.0 y una base de datos MySQL 5.0.

2.2.1 Tecnología Web y Framework Struts2

El patrón Modelo Vista Controlador está diseñado para separar la lógica de negocio, los datos y la interfaz del usuario en tres partes. Esta separación permite una gran interactividad con los usuarios, por lo que su uso es recomendable en aplicaciones Web.

En la siguiente figura, se muestra el patrón Modelo Vista Controlador implementado para una aplicación Web.

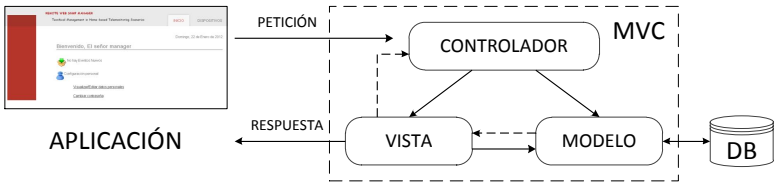


Figura 2.3: Arquitectura Modelo Vista Controlador.

Las tres partes en las que se divide este patrón son:

1. **Modelo:** El Modelo es el responsable de acceder a la capa de almacenamiento de datos, manipularlos y definir la lógica de negocio. No tiene conocimiento específico del Controlador o de la Vista, ni referencias a ellos.

2. **Vista:** La Vista representa la interfaz de usuario, obtiene los datos del Modelo y se los muestra al usuario. Se encarga de enviar la respuesta generada a partir de la petición que recibe el Controlador. Tiene que ser fácil para interactuar ya que es la capa de la aplicación que ve el usuario. La Vista tiene una referencia al Modelo y un registro de su Controlador asociado.

3. **Controlador:** El Controlador es el encargado de traducir las interacciones que el usuario realiza a través de la Vista en peticiones que serán tratadas por el Modelo o por la Vista.

Struts2 (4) es un Framework para el desarrollo de aplicaciones Web. Está basado en la arquitectura MVC obteniendo como resultado un Framework con

todas las características necesarias para facilitar el desarrollo de aplicaciones Web en Java con JSP *(Java Server Page)*, Servlet y código básico Java. Su configuración se realiza en un fichero llamado *struts.xml*. En este fichero, se indica la clase Action y los Interceptores que se tienen que ejecutar además de la vista que se mostrará en función del resultado obtenido. Por último, para poder usarlo en nuestra aplicación, hay que indicarlo en el fichero de configuración de la aplicación: *web.xml*.

Algunas de las características del Framework que llevaron a la elección del mismo son las siguientes: uso de buenas prácticas, simplicidad de diseño, fácil extensibilidad e integración con otros componentes. Estas y otras ventajas se detallan en el Anexo A. Struts2 es la versión mejorada del Framework Struts1 cuyas mejoras, entre otras, son (se detallan en A.3): uso de cualquier clase Java como clase Action, no necesidad de seguridad en los Threads, elimina la dependencia con el Servlet o permite el testeo de los Actions.

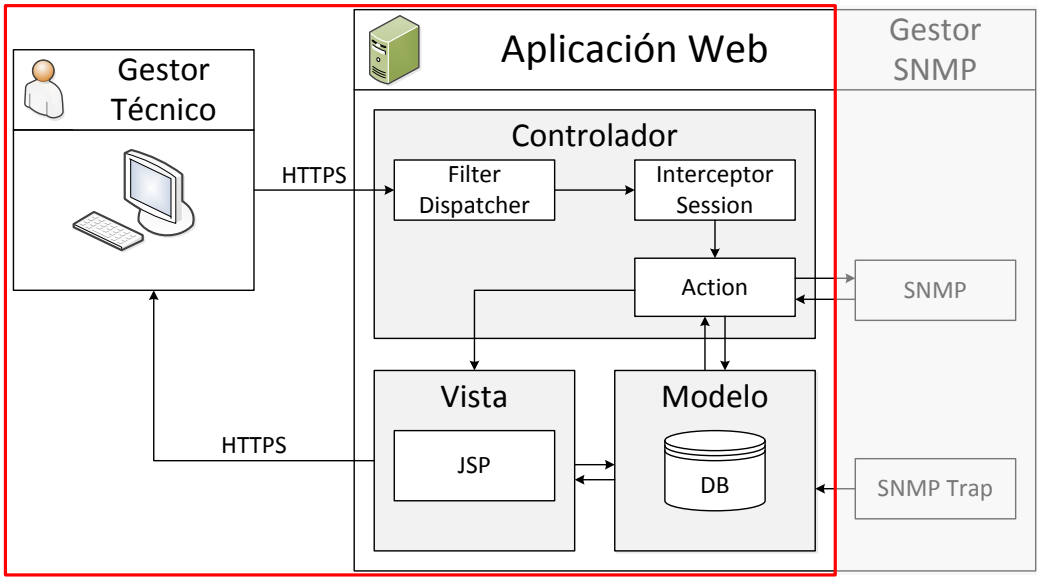


Figura 2.4: Funcionamiento General Aplicación Web.

El esquema general de la aplicación Web se muestra en la figura 2.4. El controlador está compuesto por el FilterDispatcher, los Interceptores y los Actions. El FilterDispatcher es un ServletFilter cuyo principal objetivo es interpretar todas las peticiones entrantes y determinar qué Action y qué Interceptores deberían

ejecutarse. Los Interceptores son clases Java que se ejecutan siempre antes y después del Action invocado. En nuestra aplicación, las peticiones tienen que pasar por un Interceptor que comprueba si el usuario tiene una sesión abierta. Una vez ejecutados los Interceptores, la petición llega al Action correspondiente. La clase Action es una clase Java encargada de ejecutar la lógica de negocio y proporcionar un resultado según los datos obtenidos. Durante la ejecución del Action, éste puede recurrir al modelo que contiene la base de datos. El resultado del Action indica la página JSP (perteneciente a la vista) que se mostrará. Una página JSP es una página HTML con hojas de estilo CSS *(Cascading Style Sheets)* que permite integrar Java en ella para poder hacer dinámico su contenido. En el Anexo A se detalla el Framework.

Además de todas estas ventajas, Struts2 proporciona una librería de etiquetas, *taglib*, que facilita las tareas de validación e internacionalización. Esta librería permite definir las variables en ficheros *.properties* para la implementación de la aplicación en varios idiomas.

Para el acceso a la base de datos MySQL se ha seleccionado utilizar el Framework Hibernate (8). Hibernate se utiliza para el mapeo de los objetos relacionales de Java. Permite guardar/extraer los datos que utiliza una aplicación directamente sobre la base de datos. Proporciona un Framework que mapea tablas de una base de datos en clases Java, copiando el contenido directamente sobre la clase. También copia el contenido de una clase Java en la base de datos. En este caso, dependiendo de la definición de la clase, podría guardarse la información en una o varias tablas.

2.2.2 Diseño de las páginas

En el diseño de páginas Web, entra un concepto muy importante llamado Usabilidad (5) (6). La usabilidad es la disciplina que estudia la forma de diseñar páginas Web para que los usuarios puedan interactuar con ellas de la forma más fácil, cómoda e intuitiva posible.

Algunas de las principales reglas que se han tenido en cuenta para hacer esta aplicación Web son:

1. **Simple.** Las vistas de la aplicación Web están separadas por dos tipos de acciones que se pueden realizar en el gestor: la visualización de datos y la gestión de alarmas y eventos. La navegación entre las distintas páginas se realiza de manera muy intuitiva. Cada hipervínculo expresa con claridad a dónde va dirigido. Además, se incorpora Ayuda en todas las vistas.

2. **Lenguaje común.** Los textos están escritos en un lenguaje cercano al usuario, sin usar vocabulario técnico. Son sencillos y claros de entender.

3. **Legible.** El texto tiene un tamaño y un color adecuado. Además está espaciado para que no resulte cargante para el usuario.

4. **No scroll.** Estudios han demostrado que muchos usuarios no se molestan en hacer scroll para descubrir el resto de la página. Por este motivo, se han ajustado las páginas, en la medida de lo posible, para mostrar todo el contenido sin tener que utilizar la barra de desplazamiento horizontal.

5. **Consistente.** Las distintas zonas de navegación y contenidos están aproximadamente en el mismo lugar en todas las vistas.

En cuanto a herramientas que facilitan el diseño de las páginas, se ha utilizado Tiles (7). Tiles es un Framework que hace mucho más fácil la creación de las diferentes vistas de una aplicación Web ya que separa las distintas partes de las vistas para que puedan ser reutilizadas tantas veces como se necesiten. La idea del Framework para el ejemplo concreto de nuestra aplicación se muestra en la siguiente figura. Podemos observar que cada una de las vistas de nuestra página se compone de Header, Menu, Body y Footer. Esta estructura se mantiene para todas las vista, variando únicamente el Body entre unas y otras.

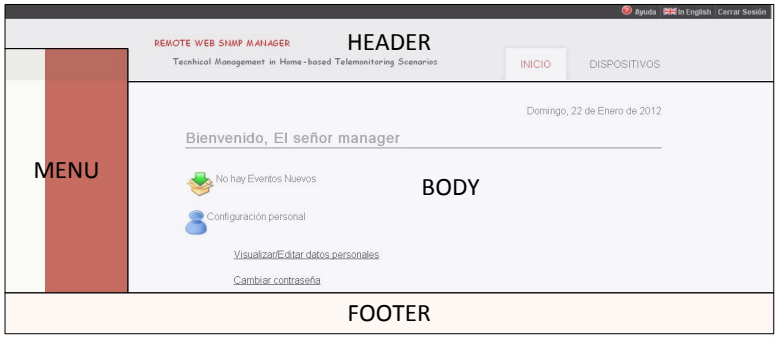


Figura 2.5: Ejemplo separación por Tiles en una de las vistas de la Aplicación Web.

2.3 Comunicación SNMPv3

SNMP es la arquitectura más popular para la gestión de redes TCP/IP y los dispositivos conectados a las mismas. Permite a los administradores un alto mantenimiento de la red, encontrar los problemas que puedan surgir en la misma y resolverlos.

La versión 3 del protocolo es la más reciente e incorpora mejoras en cuanto a seguridad con respecto a las versiones anteriores. Con esta versión, se asegura la integridad del mensaje mediante el cifrado del mismo. Para conseguirlo, SNMPv3 define dos bloques: USM (10) *(User-based Security Model)* y VACM (11) *(View-based Access Control Model)*. USM define la estrutura de usuarios e información necesaria para el cifrado y autenticación. Esto se realiza proporcionando claves secretas, para la autenticación mediante los protocolos MD5 o SHA y para el cifrado mediante los algoritmos DES-CBC o AES-128. VACM describe los privilegios que tiene cada usuario USM y se encarga de comprobar si el usuario tiene permitido el acceso a la lectura/escritura de determinados objetos.

2.3.1 Arquitectura de gestión de redes SNMPv3

Para la utilización del protocolo SNMP se necesitan dos componentes: el agente y el gestor. El agente es un programa instalado en un dispositivo que va a recoger información útil para permitir su gestión. El gestor es una aplicación encargada de

interrogar a un agente para pedir información del mismo así como de modificarla según sus necesidades.

La información que contiene el agente se almacena en unas bases de datos denominadas MIBs. Una MIB está organizada en forma de árbol y presenta de manera jerárquica los objetos y valores que pueden ser gestionados por el agente. Está definida a través del lenguaje SMI (12) *(Structure of management Information)*. Cada uno de los objetos está identificado por una secuencia de números separados por puntos, cada uno de los cuales se corresponde a un salto de nivel en el árbol que compone la MIB. Esta secuencia de números es denominada OID *(Object Identifier)*.

Se distinguen dos tipos de MIBs: públicas y privadas. Las públicas son aquellas que están definidas mediante estándares y se caracterizan por proporcionar información general de cualquier sistema de comunicaciones. La MIB pública más conocida es la MIB-II (originalmente definida en RFC1213 (9)). Esta MIB esta soportada por todos los agentes SNMP y contiene información sobre el dispositivo telemonitorizado como puede ser su tráfico o el uso de su CPU pero no contiene información a alto nivel como puede ser su sistema operativo. Las MIB privadas son aquellas definidas por los fabricantes para gestionar sus dispositivos.

Con respecto a la comunicación entre el agente y el gestor, los tipos de mensajes SNMP son los siguientes:

- **GET:** petición por el valor específico de un objeto en la MIB del agente.
- **GETNEXT:** petición por el valor del siguiente objeto en la MIB del agente.
- **GETBULK:** petición por una serie de valores de objetos que son consecutivos en la MIB. El número de valores que se pide viene determinado por *max-repetitions*. También se pueden pedir variables sin repetir mediante el campo *non-repeaters*. Esta petición disminuye considerablemente el número de mensajes que se tienen que intercambiar un gestor y un agente para la obtención de los mismos datos.

- **SET:** utilizado para cambiar un valor de un objeto en la MIB del agente, en el caso de que el objeto tenga habilitada la lectura y escritura de su valor.

Para todos estos mensajes, el agente responderá con un mensaje **RESPONSE** indicando el valor por el que se ha preguntado o si ha habido algún tipo de error.

Por otro lado, un agente SNMP también puede mandar un mensaje sin ser por petición del gestor si se ha producido algún suceso excepcional. En ese caso, el agente mandara un mensaje de tipo **TRAP** con las variables que se hayan definido para ese tipo de evento.

2.3.2 Comunicación con el AgenteMD

La MIB situada en el AgenteMD está compuesta por 5 grupos separados según la información que contienen (la información está en Anexo B):

1. *ComputeEngineControlInfo.* Contiene información relacionada con los recursos propios del CE.

2. *MedicalDeviceInfo.* Este grupo está compuesto por cuatro tablas:

   (a) *MedicalDeviceControlTable.* Registra la información acerca de características técnicas de los MDs: fabricante, modelo, etc.

   (b) *MedicalDeviceDataTable.* Contiene un histórico de la información de los MDs que varía en el tiempo: nivel de batería, horas de batería, etc.

   (c) *MedicalDeviceStateTable.* Almacena un histórico de la información correspondiente a los cambios de estado de los MDs: operativo, desconectado, etc.

   (d) *SpecificErrosTable.* Registra un histórico de la información correspondiente a los errores recibidos en el funcionamiento de los MDs.

3. *AlarmTable.* Esta tabla permite configurar alarmas relativas al funcionamiento de los dispositivos MDs y del CE.

4. *EventTable.* Está compuesto por dos tablas:

   (a) *ConfigEventTable.* En esta tabla se configuran los eventos que se realizarán al activarse una alarma.

   (b) *LogTable.* En esta tabla encontramos los Logs registrados al activarse alguna alarma.

5. *ManagerTable.* Contiene la información correspondiente a los gestores asociados al agente.

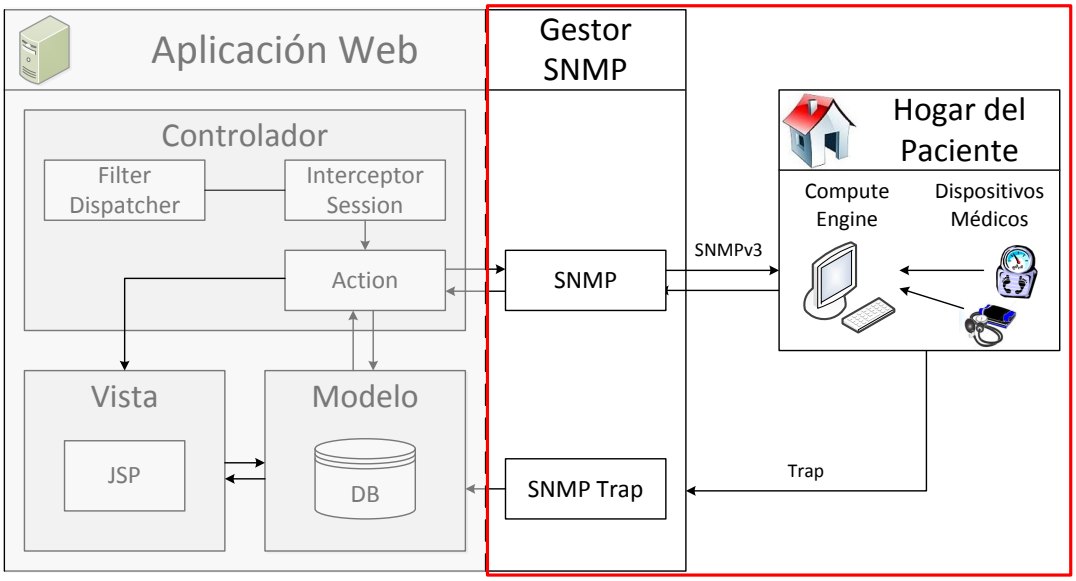


Figura 2.6: Funcionamiento General Gestor SNMP.

En la figura 2.6, podemos observar el esquema general de las comunicaciones de nuestro gestor con el AgenteMD. En concreto, el gestor SNMP Web se comunica con el agente MD mediante los mensajes GET cuando son pocos los datos a pedir o se requieren objetos aislados de diferentes tablas o mensajes GETBULK cuando se desean pedir varios objetos consecutivos.

Capítulo 3

Desarrollo Tecnológico I: Visualización

3.1 Acceso al Sistema

El gestor SNMP Web cuenta con un sistema de autenticación de usuarios para el acceso al sistema. En el servidor donde se coloca la aplicación, hay una base de datos que guarda todos los usuarios con sus respectivas contraseñas para dicha autenticación. Esta base de datos también guarda la relación de los usuarios con sus CEs asociados almacenando los parámetros necesarios para la conexión a cada uno de ellos. Una vez introducido un nombre de usuario y contraseña válidos, se inicia la sesión. La vista de acceso al sistema, se muestra en la figura 3.1.

Figura 3.1: Vista del Acceso al Sistema.

Una vez el usuario esté identificado, la aplicación cuenta con dos vistas diferentes del sistema según los roles que posea el usuario: administrador o gestor técnico. Gracias al uso del Framework Tiles y al diseño de la aplicación, se podrán implementar nuevas vistas en un futuro de manera sencilla.

La página de inicio para un gestor técnico se muestra en la figura 3.2. En esta vista, el gestor técnico puede ver los últimos Traps registrados que no haya visto anteriormente o puede cambiar sus datos personales.

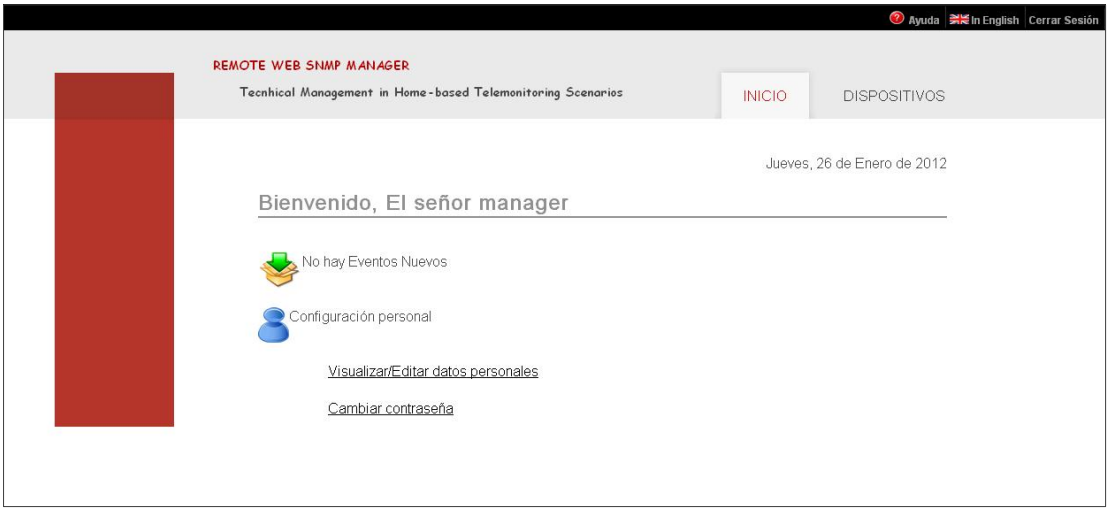


Figura 3.2: Página de Inicio del Sistema.

En la figura 3.3, se muestra la cabecera detallada. En la parte superior derecha de todas las vistas, sobre un fondo negro, hay tres opciones. La primera opción es **Ayuda**. Al pulsar sobre este botón se abre un popup con información detallada sobre lo que se puede ver en la vista actual y las opciones que hay en ella. La segunda opción es **In English**. Esta opción muestra las páginas en Inglés. Para ello, la aplicación hace uso de la librería *taglibs* mostrando los textos que contiene el fichero *.properties* del idioma deseado. La última opción es **Cerrar Sesión**. Al pulsar este botón, la sesión queda finalizada.

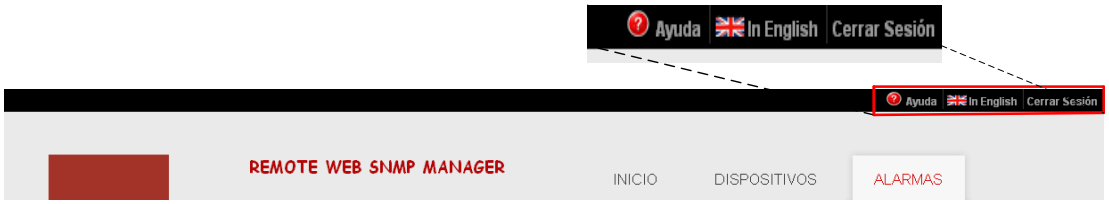


Figura 3.3: Detalle Cabecera Gestor Técnico.

En esta imagen, también se pueden apreciar las tres pestañas que tiene el gestor técnico. Estas tres pestañas son: *Inicio*, *Dispositivos* y *Alarmas*. La pestaña de

Dispositivos permite la visualización de datos de los CEs y MDs asociados al gestor técnico que tienen activada la sesión y la de *Alarmas* permite la configuración de eventos y alarmas de los mismos. Un usuario de tipo administrador tiene las mismas opciones que un gestor técnico y, además, puede gestionar los usuarios. Por lo tanto, la cabecera de un administrador incluye una opción más, que es la pestaña *Usuarios*. Esto se aprecia en la figura 3.4.

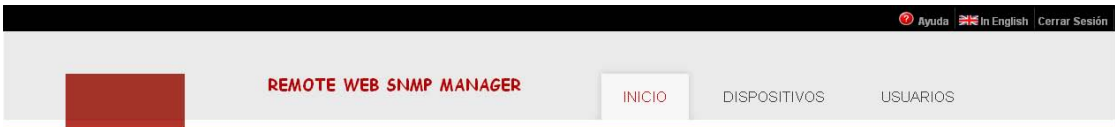


Figura 3.4: Detalle Cabecera Administrador.

3.2 Gestor SNMP

Para mostrar los datos acerca del CE y los MDs, el gestor SNMP integrado en la aplicación envía peticiones SNMP al agenteMD. Esto lo realiza la parte de la aplicación correspondiente al gestor SNMP. Como se ha explicado anteriormente, el gestor SNMP puede enviar tres tipos de mensajes para pedir datos al agente. Estos mensajes son: GET, GETNEXT o GETBULK. Para la optimización de los recursos utilizados, se ha realizado un estudio del tiempo y el tráfico requerido por cada tipo de petición.

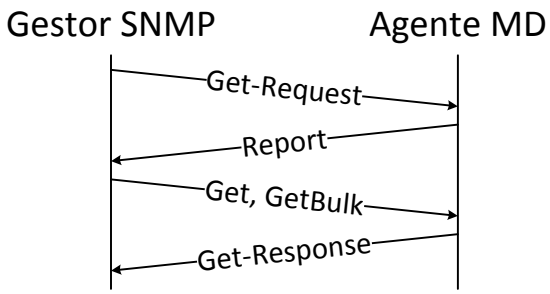


Figura 3.5: Comunicación Gestor-Agente.

Dada la configuración seleccionada en el agenteMD (MaxRepetitions es igual a 10), cada vez que se envía un mensaje GETBULK el número de valores recibidos es

igual a 10. En la imagen 3.5, se puede ver que cada vez que se realiza una petición al agente, se tiene que enviar un GET-REQUEST para la autenticación. Por este motivo, cada petición se convierte en 4 paquetes.

En la siguiente tabla, podemos ver la comparativa entre los mensajes GET y GETBULK mostrando el número de valores que recogen, los bytes que se transmiten y el tiempo transcurrido. Estas valores incluyen todos los paquetes que se transmiten en la petición, es decir, 4 paquetes.

Número de Objetos Pedidos a la MIB MD	Get		GetBulk	
	KBytes	Tiempo(Seg.)	KBytes	Tiempo(Seg.)
1	0.5102	0.186	0.6973	1.731
2	1.0205	0.372		
3	1.5308	0.558		
4	2.0410	0.744		
5	2.5513	0.96		
6	3.0615	1.116		
7	3.5718	1.302		
8	4.0820	1.488		
9	4.5923	1.674		
10	5.1025	1.86		

Tabla 3.1: Comparación del tiempo y el tráfico enviados según el número de objetos pedidos

En la tabla anterior, se puede ver el número de KBytes intercambiados y el tiempo transcurrido durante el intercambio según el número de valores que el gestor ha pedido al agenteMD. Para un mensaje de tipo GETBULK, estos parámetros no varían según el número de valores que se pidan, son valores constantes. Para un mensaje de tipo GET, estos parámetros son proporcionales al número de valores pedidos. Estos mensajes requieren autenticación con cada valor pedido.

Comparando los valores que se visualizan en la tabla, la decisión tomada consiste en utilizar GETBULK cuando la cantidad de datos a pedir es desconocida o cuando es superior a 6 si todos estos valores pertenecen a la misma tabla. Los mensajes de tipo GET se utilizan cuando se quieran pedir datos aislados o se conozca con certeza que el número de valores a pedir es menor que 6. Estos mensajes se envían gracias a una API llamada SNMP4j (13) sobre la que se ha desarrollado el gestor SNMP.

Cuando el gestor SNMP haya obtenido los datos que necesita, los almacenará en objetos Java iguales a las tablas que tiene la MIB para su posterior visualización mediante la interfaz.

3.3 Visualización

3.3.1 Visualización datos CE

Una vez dentro del sistema, hay dos tipos de datos que se pueden visualizar: los datos asociados al CE y los datos asociados a los MDs. Para visualizarlos, hay que entrar en la pestaña *Dispositivos*. La primera vista de esta pestaña muestra los CEs asociados al gestor como se ve en la figura 3.6. En esta figura, cada casa representa a un CE asociado al gestor que tiene iniciada la sesión. Además, pulsando sobre el enlace *Ver Localización de los agentes* es posible visualizar sobre un mapa la ubicación de todos los CEs gracias a la utilización de la API de Google Maps (14) que ha sido incorporada en esta aplicación. Esta información acerca de la ubicación del CE se encuentra en el campo *UserContactInformation* del grupo *ComputeEngineControlInfo* de la MIB MD explicada en el Anexo B.

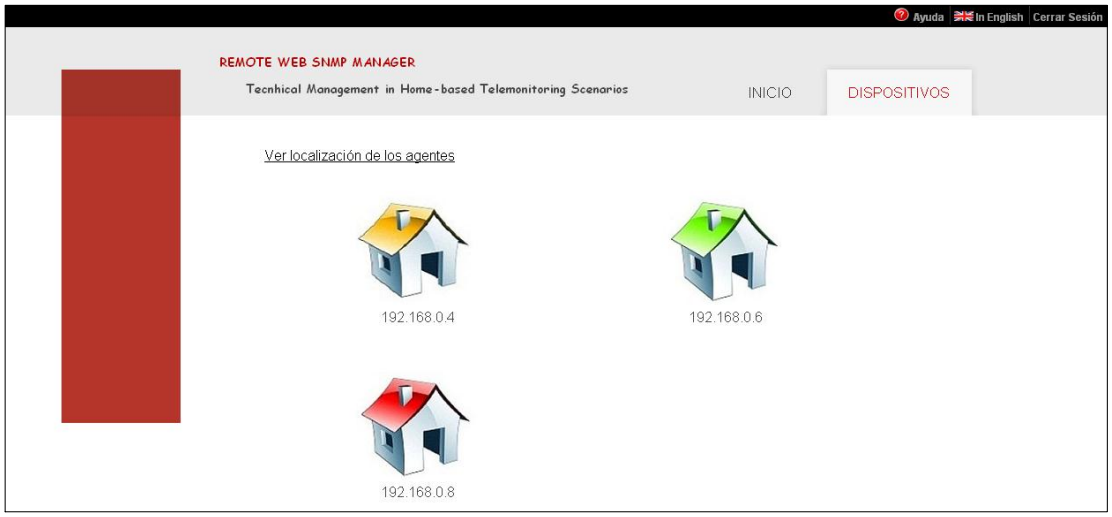


Figura 3.6: Vista CEs asociados al Gestor Técnico.

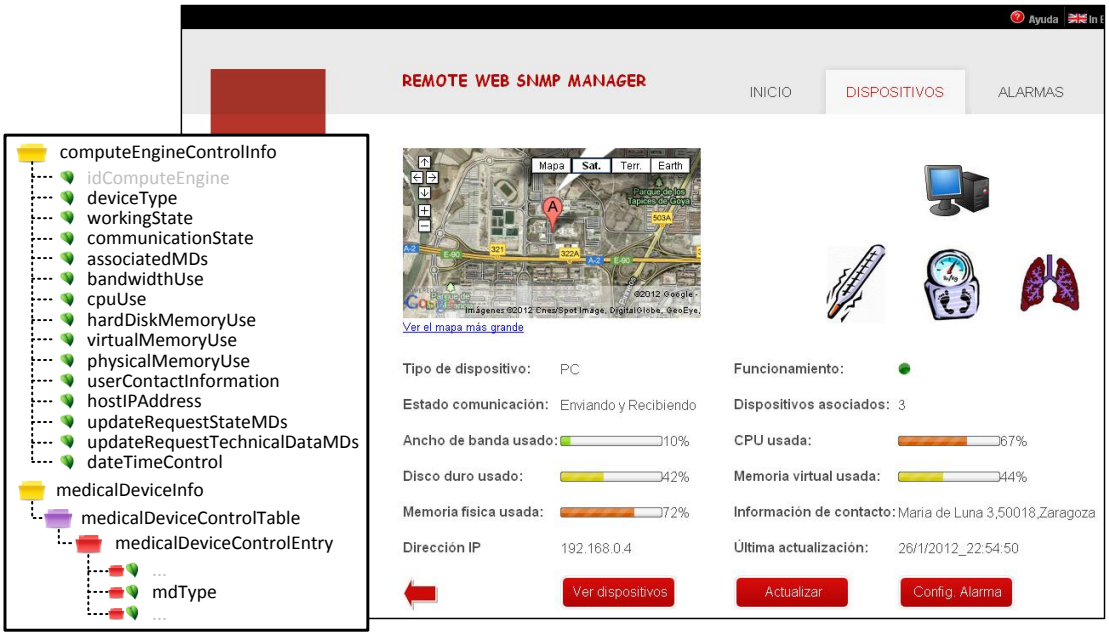


Figura 3.7: Visualización datos asociados al CE.

Al pulsar sobre cualquiera de las casas, se visualiza la información asociada al CE. Esta vista, junto con las tablas de la MIB MD que se pueden ver en ella, corresponde a la figura 3.7. En esta figura, y en las sucesivas con páginas que contienen información de la MIB MD, se ha incluído, en la parte de la izquierda, el dibujo de la MIB MD con los campos exactos que se representan en la vista para facilitar la visualización del mapeo de la aplicación con la MIB MD. Dentro del body, en la parte superior izquierda se observa el mapa que muestra la ubicación exacta del CE seleccionado. En la parte superior derecha del body se aprecia el tipo de dispositivo que es el CE (este dato corresponde al campo *deviceType* del grupo *ComputeEngineControlInfo*), y los tipos de MDs asociados. Los tipos de MDs asociados al CE, están en el campo *mdType* de la tabla *medicalDeviceControlTable* perteneciente al grupo *medicalDeviceInfo*. Para pedir estos datos, el gestor SNMP envía tantos mensajes GET como dispositivos asociados haya al CE (este dato corresponde al campo *asociatedMDs*). Una vez recogidos los valores, se muestran sus imágenes correspondientes en lugar del texto para mostrar más intuitivamente de qué tipo de dispositivos se trata ya que siempre es más fácil recordar una imagen que un código. Pulsando sobre cada una de las figuras que representan a cada tipo de MD se accede a la información técnica asociada a ese dispositivo

(ver figura 3.9). El resto de información que se visualiza en esta vista son todos los datos contenidos en el grupo *ComputeEngineControlInfo*. En este grupo, se almacena información de los recursos propios del CE, como son la CPU o el ancho de banda. Los elementos de la tabla que contienen estos recursos, se muestran mediante barras de colores en función de su saturación para alertar visualmente cuando un recurso podría estar más saturado de lo deseado. Los colores van desde el verde, que representa una carga mínima, pasando por el amarillo y el naranja hasta llegar al rojo que representa una carga crítica. En la parte inferior de la página, se observan cuatro botones. El primero de ellos es para volver a la última página visitada. Este botón se mantiene en todas las vistas aproximadamente en el mismo lugar. El botón *Actualizar* sirve para pedir una actualización tanto de los datos técnicos como de los posibles cambios de estados de todos los MDs asociados. *Config.Alarma* permite configurar una alarma asociada al CE. Este punto se explicará en el próximo capítulo. Y, por último, el botón *Ver Dispositivos* conduce a la página que se ve en la figura 3.8.

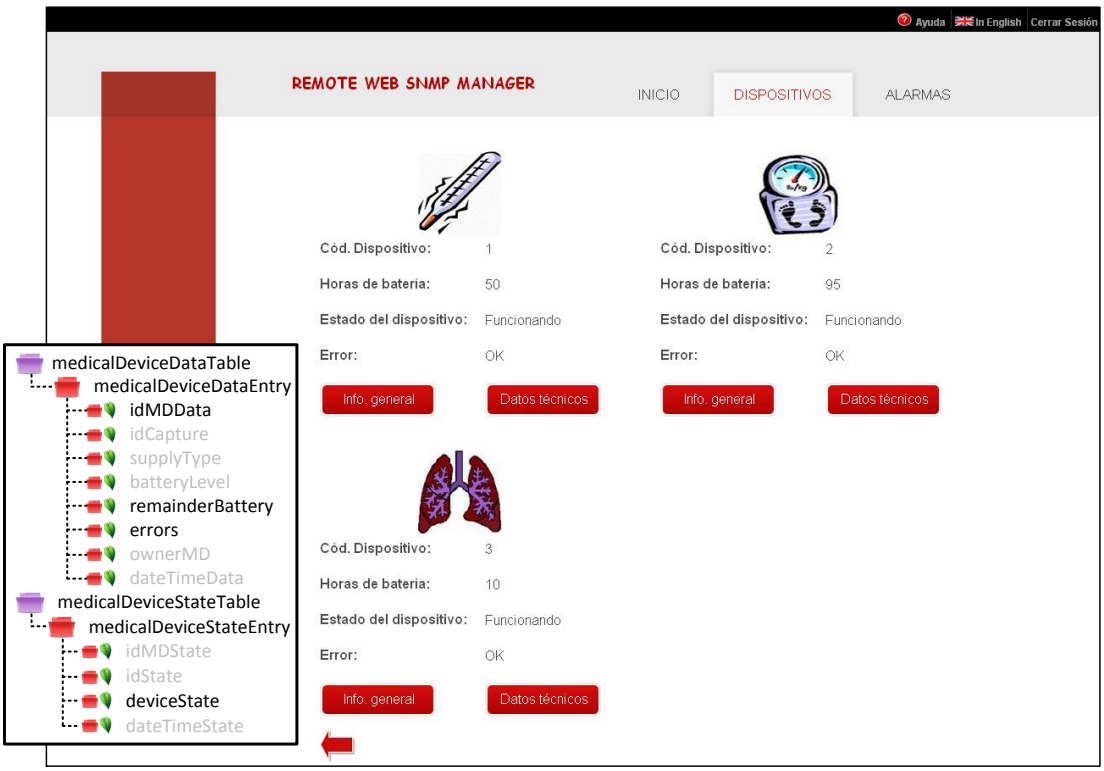


Figura 3.8: Vista del estado actual de los MDs asociados al CE.

3.3.2 Visualización datos MDs

En la vista de la figura 3.8, se puede observar todos los MDs asociados al CE con la última información básica recibida sobre ellos. Esta información contiene las horas de batería que le quedan, el estado en el que se encuentra y si ha habido errores en la última captura. El objetivo de esta vista es mostrar la información más relevante de todos los dispositivos de manera mucho más efectiva ya que solo se piden las últimas entradas de los objetos de la MIB MD marcados en la figura en lugar de tener que recorrer toda la tabla. Una de las ventajas que aporta esta vista es poder comparar gráficamente los dispostivos entre sí y ver si alguno de ellos requiere más prioridad. Además, esta vista incluye dos botones por cada MD. El primero de ellos es *Info. General* que muestra la información estática asociada al dispositivo contenida en la tabla *medicalDevicesControlTable*. Al pulsar sobre el botón *Datos técnicos* se accede a visualizar los datos técnicos de la última captura asociada al MD seleccionado tal y como se muestra en la figura 3.9.

Figura 3.9: Visualización datos asociados al MD.

En la figura 3.9, vemos la información correspondiente a la última captura de datos técnicos recibida y el estado actual del MD seleccionado. En la parte de la izquierda se muestra el tipo de MD que estamos visualizando. En la parte central, se muestran los datos que contiene la entrada más actual en las tablas *medicalDevicesDataTable*, *medicalDevicesStateTable* y el objeto de la tabla *SpecificErrorsTable* que ha producido el error en caso de haber. En la parte de la derecha, hay un conjunto de checkboxs para realizar una de las siguientes acciones: visualizar el histórico de datos en forma de tabla, visualizar el histórico de datos de forma gráfica o configurar una alarma asociada a ese MD y un elemento concreto. Una vez seleccionados los checkboxs correspondientes a la acción que se desea realizar, se pulsa el botón *Go!* situado en la parte inferior del body. Esta vista también cuenta con otro botón, *Info. General*, que muestra la información estática asociada al MD que también es accesible desde otras páginas.

Para acceder a la información que se está mostrando en la vista, utilizando otro software, habría que recorrer tres tablas completas y se obtendría la información de una forma complicada de interpretar. Al utilizar esta vista para poder ver la información, se muestra sólo la información correspondiente a la última captura de los datos técnicos, con el motivo del error correspondiente registrado en la tabla de error en caso de haber y el estado actual del MD y se tiene libertad para visualizar el histórico de datos como se desee o de realizar otras acciones sobre los datos.

3.3.3 Visualización Gráfica Histórico de datos

En la figura 3.10, se muestra un ejemplo de la visualización del histórico de datos técnicos y estados del MD en forma de tabla. La estructura del body se divide en dos tablas, la tabla situada arriba corresponde a las capturas de datos técnicos y la situada abajo corresponde a los cambios de estado del MD. El resto de columnas, depende del número de elementos que se hayan seleccionado en la vista anterior. Esta vista permite ver la evolución de los datos a través del tiempo de manera visual y fácil de entender.

Figura 3.10: Visualización del histórico de datos técnicos y estados del MD en forma de tabla.

En la figura 3.11, se muestra un ejemplo de la visualización del histórico de datos técnicos y estados del MD de forma gráfica. Para ello se ha utilizado la librería JFreeChart (15). Esta librería permite mostrar los datos mediante diversos gráficos. Según los elementos seleccionados, los gráficos mostrados variarán. Si sólo se selecciona un único elemento, se mostrará un gráfico total con todos los datos y gráficos diarios. Si se seleccionan dos o más elementos, todos los gráficos serán totales. En concreto, se han seleccionado tres tipos de gráficos. Para los elementos cuyos valores son numéricos (horas de batería y porcentaje de batería), se ha seleccionado un gráfico XYPlot que muestra la evolución de los valores en función del tiempo. Para los datos cuyos valores son conjuntos de caracteres, se utiliza un gráfico de tipo Pie para los datos totales que permite visualizar rápidamente el porcentaje de tiempo que el objeto ha mantenido cada valor y un diagrama de barras para la visualización de datos diarios que permite visualizar el porcentaje de tiempo que el objeto ha mantenido cada valor diariamente. Se ha añadido la opción de ver los datos gráficamente porque los gráficos son más amenos para seguir que las tablas de datos. Además, el uso de gráficos para representar los datos facilita la interpretación de la información de manera más rápida y eficiente. Por ejemplo, mirando el gráfico de *Errores*, se puede ver instantaneamente si se están

produciendo muchos errores y los motivos de los mismos; o, mirando el gráfico *Horas de batería* se puede ver cuando el dispositivo está cerca de quedarse sin batería.

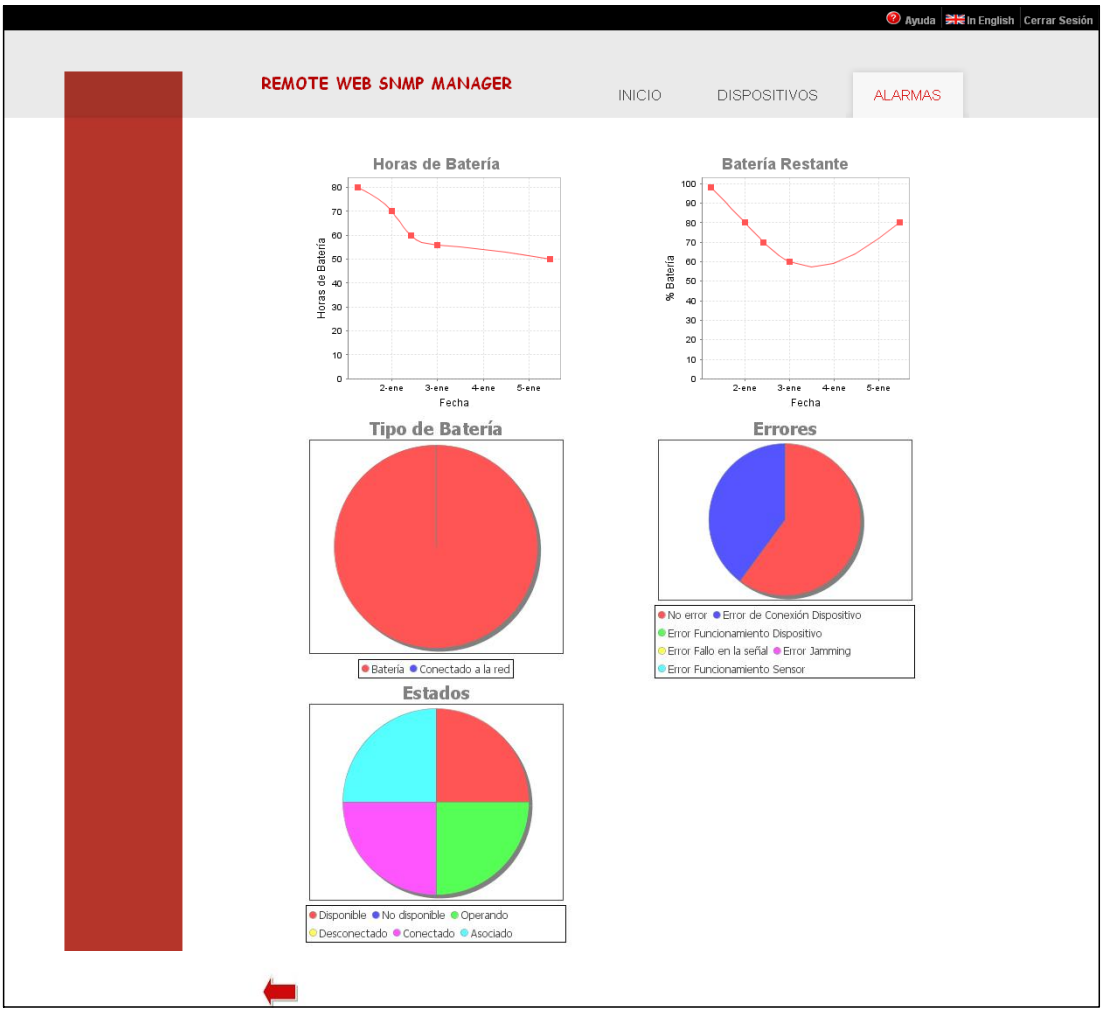


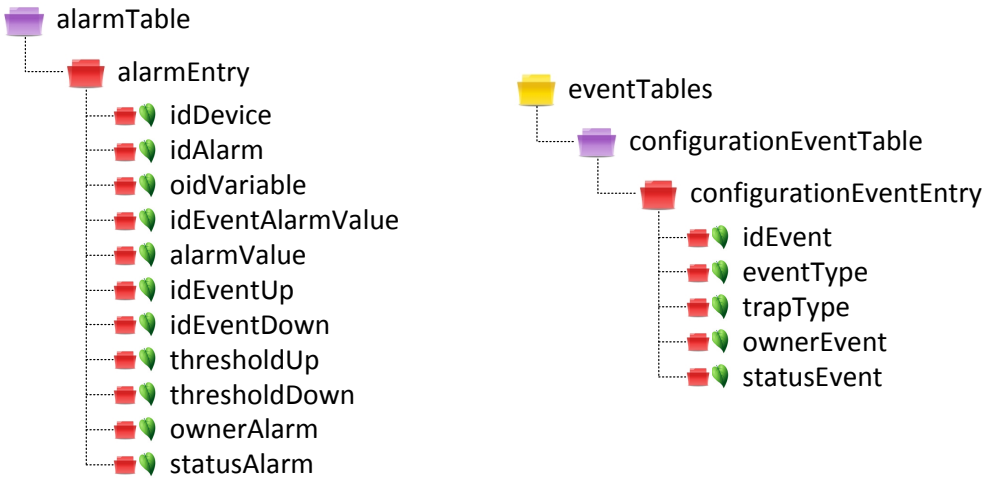
Figura 3.11: Visualización gráfica del histórico de datos técnicos y estados del MD.

Capítulo 4

Desarrollo Tecnológico II: Gestión de Alarmas

4.1 Gestor de Alarmas con SNMP

El agente MD permite la configuración de alarmas y eventos sobre los objetos de la MIB relativos al funcionamiento del CE y los MDs a través de la tabla *alarmTable* y *configurationEventTable*. Para realizar la configuración de alarmas, se necesita añadir entradas en la tabla de alarmas (y en la tabla de eventos, si fuese necesario) con los parámetros que se quieren controlar.



(a) Tabla de Alarmas de la MIB MD. (b) Tabla de Eventos de la MIB MD.

Figura 4.1: Tablas de configuración de alarmas y eventos.

La tabla de alarmas se muestra en la figura 4.1(a). Para crear una entrada en dicha tabla, hay que seguir la filosofía RMON, es decir, hay que enviar una petición al agenteMD con la creación de la entrada. Este procedimiento se puede ver en la figura 4.2. Una vez creada la entrada, el resto de objetos, deben ser rellenados por el gestor mediante mensajes de tipo SET. Al terminar,la alarma se activa.

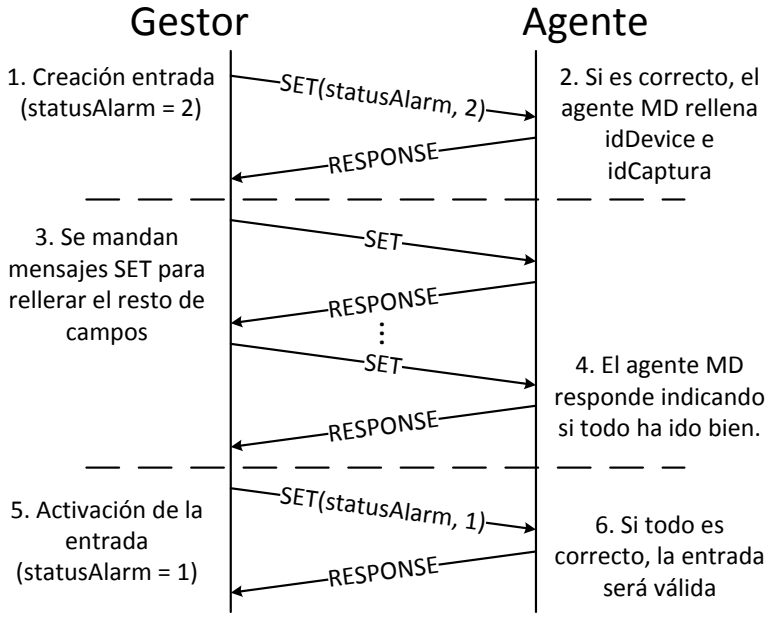


Figura 4.2: Comunicación Gestor-Agente para añadir Alarma.

La tabla de eventos se muestra en la figura 4.1(b). Para crear una entrada en esta tabla, hay que seguir el mismo procedimiento que para la tabla de alarmas. El primero paso a realizar es enviar una petición para la creación de la entrada, poniendo el valor de *statusEvent* a 2(create). El objeto *eventType* indica el tipo de evento que se lanzará cuando alguna de las alarmas con este evento asociado sea activada. Los valores que puede tomar son: mandar un trap, crear una entrada en la tabla de logs o ambas. El objeto *trapType* contiene el tipo de Trap que se enviará en caso de que el tipo de evento elegido sea enviar un trap. Los posibles valores son: SystemOvercharged, WrongDeviceWorking, SpecificError, NewMD y Warning. Una vez rellenados todos los campos de la entrada en la tabla, hay que activarla.

Estas dos son las tablas que hay que configurar para poder gestionar las alarmas. Al igual que en el capítulo anterior, el gestor SNMP enviará un conjunto de mensajes GET/GETBULK para visualizar la tabla de logs, alarmas, etc.

Uno de los tipos de eventos que se puede configurar es el envío de traps. Los traps son mensajes asíncronos que el agenteMD va a enviar al gestor cuando se active alguna alarma cuyo evento sea este o el agenteMD tenga programado el envío del mismo. Estos mensajes se envían al gestor por el puerto 162. Para capturarlos, el gestor SNMP se mantiene escuchando en ese puerto. Cuando llega un trap, el gestor SNMP extrae la información necesaria y la almacena en una base de datos. La tabla que hay en esta base de datos registra la dirección IP del agenteMD que ha enviado el trap, el identificador del dispositivo asociado, el identificador de la captura que ha hecho activar la alarma, la fecha en la que se ha activado y el tipo de trap que se ha enviado.

4.2 Gestión de Alarmas

Para gestionar y visualizar las alarmas y eventos de la MIB, hay que acceder a la pestaña *Alarmas*. Esta pestaña da lugar a la vista de la figura 4.3. En ella, aparecen cuatro figuras: *Últimos Eventos*, *Visualizar tabla de logs*, *Visualizar Alarmas Configuradas* y *Configurar Alarmas*. Con esta organización, quedan claramente separados los sucesos recibidos (traps recogidos por un lado y logs registrados por el otro) de la visualización y configuración de las alarmas y eventos que los lanzan. Pulsando sobre el botón de *Configurar Alarmas*, se llega a la figura 4.4.

Desde esta página se configuran las alarmas. Esta página es una parte muy importante de la aplicación ya que permite al gestor la configuración de las alarmas que quiera de manera transparente e intuitiva, separando los distintos eventos por grupos y utilizando herramientas para deshabilitar las opciones que no serían necesarias. Es accesible desde la página de la figura anterior, desde los datos del CE o desde los datos de alguno de los MDs asociados ya que una alarma puede estar asociada a cualquiera de los MDs o al propio CE. En la parte superior del cuerpo

Figura 4.3: Vista de la pestaña Alarmas.

de la página, hay dos desplegables, *Dispositivo* y *Recurso*. *Dispositivo* contiene todos los tipos de MDs asociados y el CE y *Recurso* contiene todos los OIDs a los que se les puede asociar una alarma, como por ejemplo, uso del CPU, estado del dispositivo, etc. El desplegable *Dispositivo* vendrá con el valor preseleccionado si se accede a la vista desde los datos del CE o del MD mientras que *Recurso* solo vendrá preseleccionado si se accede desde los datos del MD (el recurso se selecciona mediante los checkbox que se han visto en la figura 3.9 del capítulo anterior).

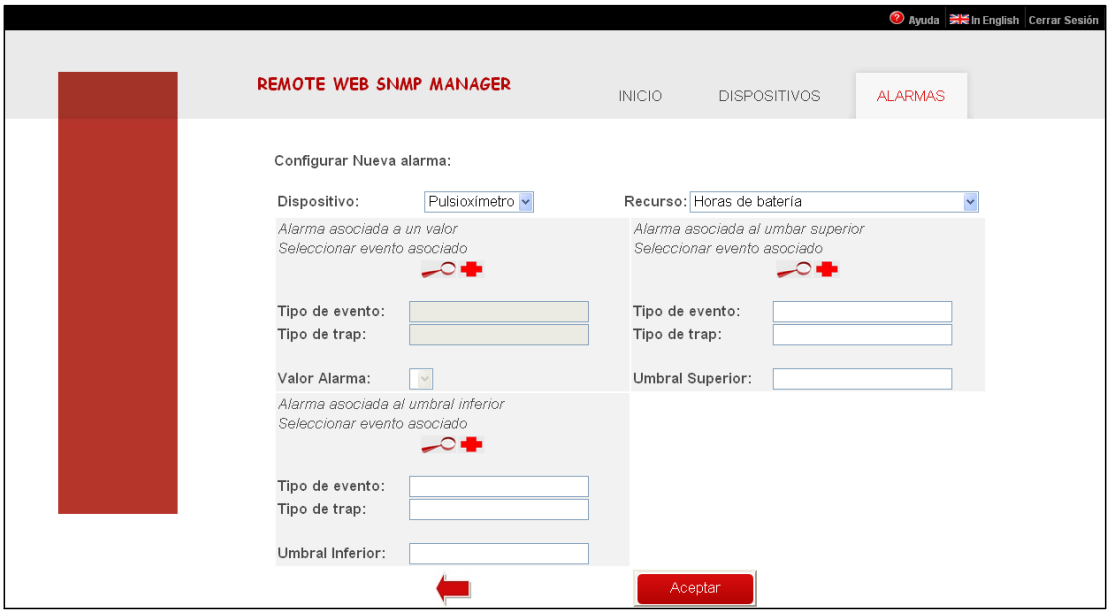


Figura 4.4: Vista Configurar Alarmas.

Una alarma permite asociar hasta tres eventos que serán: activados por valor, por umbral superior o por umbral inferior. Estos tres eventos se pueden distinguir en la página porque están separados en tres bloques diferentes. Dependiendo del recurso que se esté controlando, no tiene sentido definir alguno de ellos. Por ejemplo, si el recurso controlado es el error de conexión del dispositivo, no tiene sentido definir un umbral superior o un umbral inferior ya que este campo solo puede tomar los valores "OK" y "Error". Para no crear eventos que nunca serán utilizados, cada vez que se seleccione un nuevo valor en el desplegable *Recurso*, se ejecuta una función Javascript que deshabilita los eventos innecesarios. En la figura del ejemplo, se ha seleccionado el recurso "Horas de batería". Este objeto toma valores numéricos, por lo que, no tendrá sentido una alarma por valor. Se puede observar como el bloque de alarma por valor esta deshabilitado y los bloques de alarma por umbral superior y alarma por umbral inferior están habilitados. Dentro de cada bloque, en la parte superior, se lee a qué alarma corresponde. Justo debajo de ese texto, se encuentran dos pequeñas imágenes que sirven para indicar el evento asociado. La primera es una lupa que, al ser presionada, muestra un popup con los eventos activos que existen en la MIB MD (figura 4.5) y la segunda es el signo más que, al ser presionado, muestra un popup que permite crear un nuevo evento diferente a los que ya existe.

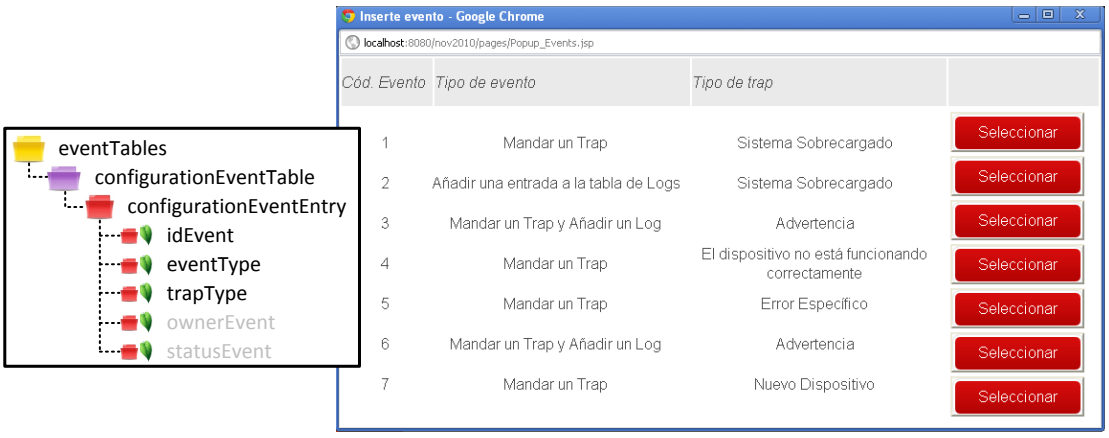


Figura 4.5: Popup Eventos Existentes.

Una vez seleccionado el evento deseado, solo queda definir el umbral o el valor que activará la alarma. Este valor se escribe en el campo: *Valor Alarma*, *Umbral*

Superior o *Umbral Inferior*, dependiendo del bloque que se esté configurando. Para las alarmas activadas por umbral, basta con escribir el número deseado. Para las alarmas asociadas por valor y para evitar errores, al seleccionar el recurso, se rellenará un desplegable con todos los valores que puede tomar ese recurso. Cuando se hayan introducido todos los datos necesarios, se pulsará el botón *Aceptar* para que el gestor SNMP procesa a enviar los mensajes correspondientes. Si todo ha funcionado correctamente, se mostrará una pantalla indicándolo.

Figura 4.6: Vista Tipos de Alarmas.

Para visualizar las alarmas que hay configuradas en el agente MD, hay que ir a la pestaña *Alarmas*, y pulsar la imagen correspondiente a *Visualizar Alarmas Configuradas*. La visualización de alarmas se ha separado por grupos, como se puede ver en la figura 4.6. El primer icono mostrará las alarmas que activarán algún evento cuando el recurso asociado tome un determinado valor. El segundo icono mostrará las alarmas que se activan por umbral superior y el tercero por umbral inferior. Al pulsar sobre cualquiera de estos iconos, se verá una página como la de la figura 4.7 donde se muestran todas las alarmas que tienen eventos del tipo seleccionado. En esta página se muestran las entradas activas de la tabla *AlarmTable* incluyendo solo el evento seleccionado. Dentro de esta página, se puede acceder a la información de control del MD/CE (*medicalDeviceControlTable, computeEngineControlTable*) pulsando sobre la imagen del dispositivo; se puede

ver la vista detallada de la alarma pulsando sobre el código de la alarma en la figura 4.8 (entrada correspondiente en la tabla *AlarmTable*); y, por último, pulsando sobre el código del evento se muestra la entrada correspondiente en la tabla *configurationEventTable*.

Figura 4.7: Vista Visualizar Alarmas configuradas.

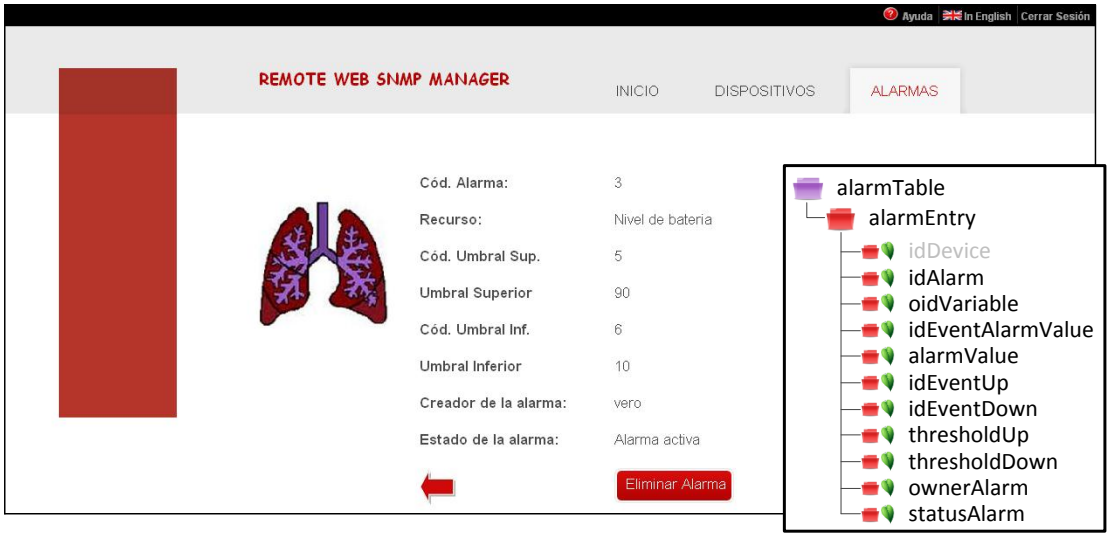


Figura 4.8: Visualizar Datos Alarma.

En la figura 4.8 se puede ver en detalle todos los campos cuyo valor sea distinto de *NULL* de la entrada de la tabla *AlarmTable* que se ha seleccionado. Por ejemplo, la alarma que se ve en la figura esta asociada al dispositivo de la imagen, sobre el recurso *Nivel de Batería*. Si este recurso supera el valor 90, la alarma se

activará enviando el evento asociado al umbral superior (cuyo código es el 5) y, si este recurso es inferior a 10, la alarma se activará enviando el evento asociado al umbral inferior (cuyo código es el 6).

Para la visualización de la tabla de logs, hay que ir a la pestaña *Alarmas* y pulsar sobre la imagen *Visualizar tabla de logs*. De este modo, se accede a una página con todos los logs que tiene registrados el agenteMD en la tabla *logTable*. Además de visualizar la tabla *LogTable*, permite visualizar el evento y la alarma asociados pulsando sobre el código del evento/alarma que se quiera ver en detalle.

La vista de *Últimos Eventos* se muestra en la figura 4.9. En ella se pueden visualizar los últimos traps recibidos. Los traps que se hayan recibido pero todavía no se hayan visualizado, se muestran sobre un fondo amarillento y con un "New" en la primera columna. El resto de datos que se muestran son los que se guardan en la base de datos que se ha explicado en el punto anterior. También incluye la posibilidad de visualizar la infomación de control del dispositivo (*medicalDeviceControlTable*) pulsando sobre la imagen del dispositivo o los datos técnicos correspondientes a la captura que ha hecho activar la alarma (*medicalDeviceDataTable*).

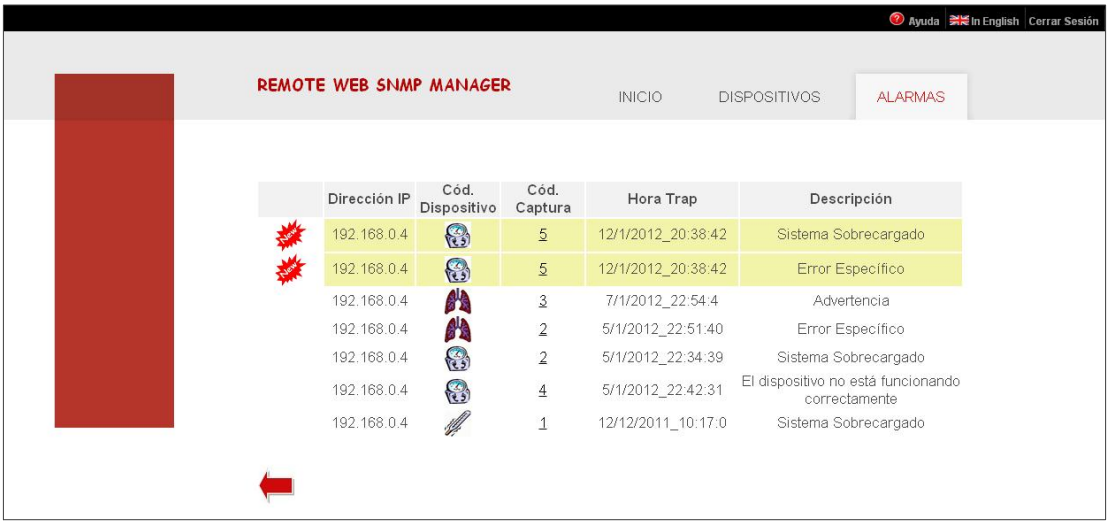


Figura 4.9: Vista Visualizar Últimos Eventos.

A esta vista también se puede acceder desde la pantalla de inicio que indica los traps nuevos que se han recibido desde la última conexión. Como se ha explicado anteriormente, el gestor SNMP implementa una aplicación que está constantemente esperando la llegada de nuevos traps. Cada vez que se reciben estos nuevos eventos se envía un mensaje al usuario gracias a la implementación de una función en AJAX (16). AJAX es una herramienta que permite dar respuesta inmediata a los eventos que ocurren ya que se ejecutan en el lado del cliente (navegador del usuario), es decir, los elementos visuales de la página pueden variar sin necesidad de refrescar la página. Con esto se gana mucho tiempo ya que no se tiene que enviar la petición al servidor y no se pierden datos de contexto al no refrescar la página. Por este motivo, se ha implementado una función en AJAX que avisará de la llegada de nuevos traps con el icono parpadeante de la 4.10. Esta imagen aparecerá para alertar al usuario que se acaba de recibir, al menos, un nuevo trap y se convertirá en un enlace a la página que permite visualizar los últimos eventos recibidos.

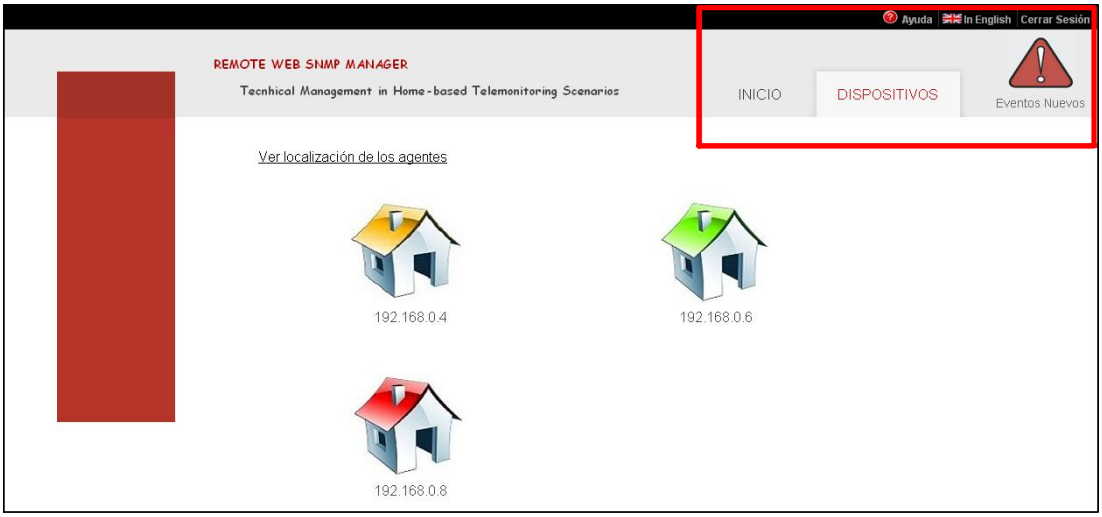


Figura 4.10: Detalle Cabecera con Icono Alarma.

Capítulo 5

Evaluación del Sistema

5.1 Objetivo de la evaluación

Uno de los objetivos de este proyecto es facilitar la visualización de los datos técnicos de los dispositivos que permitirán la telemonitorización de los pacientes. Como se ha visto anteriormente, la recolección de esos datos se hace mediante un agente SNMP instalado en un CE que los guarda en una MIB. Esta MIB es difícil de interpretar con los programas existentes en la actualidad sin tener conocimientos sobre el tema.

Es importante asegurarse de que cualquier usuario sea capaz de utilizar la herramienta y conseguir visualizar/configurar lo que necesite sin tener ningún conocimiento de la tecnología que está implementando. También es importante saber el grado de satisfacción del personal con el sistema, si lo encuentra útil o si tienen sugerencias. Por estos motivos, y para recoger pruebas que ayuden a mejorar su desarrollo, se hace la evaluación con pruebas reales.

5.2 Metodología

Esta evaluación se ha dividido en dos fases para las que se han utilizado diferentes cuestionarios (Anexo F) que recogen las respuestas de los usuarios que han participado en este proceso de evaluación. Estas fases son:

- **Fase I: Evaluación Guiada.** Este cuestionario se compone de 10 preguntas que piden al usuario realizar una serie de acciones, desde algo sencillo como ver los CEs asociados hasta algo más complicado como configurar una alarma. Viendo lo fácil o difícil que distintos tipos de usuarios encuentran la aplicación, nos da una idea de cómo podría ser aceptada al ser implementada. También nos permite conocer si la estructura de la aplicación es la adecuada, es decir, si los distintos tipos de acciones que se pueden realizar sobre ella están separadas e indicadas correctamente o cuáles son los puntos que no parecen estar tan claros. Para la realización de esta encuesta, el usuario está conectado a la aplicación intentando realizar las consultas o gestiones que se le van pidiendo mientras se cronometra el tiempo empleado para cada una de ellas.

- **Fase II: Evaluación General.** Este cuestionario contiene 15 preguntas que se realizarán después de haber utilizado la aplicación. Su finalidad es obtener más información acerca de la opinión del usuario, especialmente qué les ha parecido la experiencia y qué podría ser mejorado. Es una encuesta muy importante tanto para valorar el estado actual de la página como para ver fallos en su funcionamiento o recibir sugerencias e ideas sobre las partes de la aplicación que podrían ser cambiadas mejorando el uso por el usuario.

Para la realización de la evaluación, se han escogido personas de tres perfiles diferentes:

- **Perfil Técnico.** La aplicación está orientada al control de los datos técnicos de los dispositivos médicos, por lo que, este perfil es obligatorio en la evaluación. Los usuarios que gestionen está información deberán tener este perfil. Esta evaluación la han realizado dos hombres y una mujer de edades comprendidas entre 24 y 28 años.

- **Perfil Clínico.** En este momento, el proyecto está implementado para controlar los datos técnicos de los dispositivos médicos pero se puede ampliar en un futuro para la gestión de los datos clínicos. Por este motivo, es

importante que personas con perfil clínico puedan entender la aplicación sin problemas y aportar ideas o mejoras que ayudarán al diseño de la parte clínica. Este grupo está compuesto por dos mujeres y un hombre de edades comprendidas entre 24 y 25 años.

- **Otros.** Se ha realizado también la evaluación sobre un grupo formado por personas pertenecientes a perfiles diferentes al técnico o al clínico para asegurar que cualquier tipo de usuario sin conocimientos ni técnicos ni clínicos pueda usar la aplicación para el fin de la misma. Para este perfil, se han seleccionado tres personas, un hombre y dos mujeres con edades comprendidas entre los 45 y los 57 años.

5.3 Resultados

En la primera fase, la evaluación guiada, se ha ido pidiendo al usuario que realice una serie de acciones controlando el tiempo que ha tardado en conseguirlo. En la figura 5.1, se puede ver el tiempo medio que ha tardado cada uno de los perfiles en realizar cada una de las acciones que son las siguientes:

1. Acceder a la gestión de dispositivos.
2. Visualizar la información de un CE.
3. Decir el número de dispositivos asociados y sus tipos.
4. Ver los últimos datos técnicos registrados.
5. Visualizar el histórico de los datos de forma gráfica.
6. Configurar una alarma.
7. Acceder al registro de las alarmas configuradas.
8. Ver en detalle los datos de un evento asociado a una alarma.
9. Visualizar la tabla de logs.
10. Acceder a los últimos eventos recibidos.

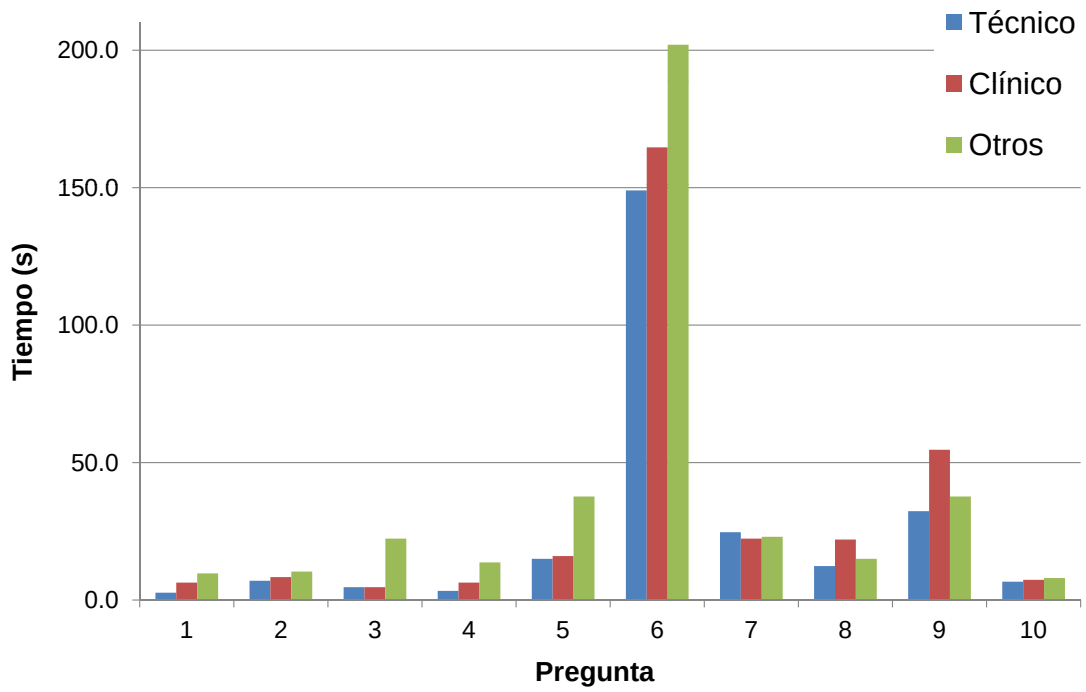


Figura 5.1: Tiempo medio empleado por perfiles en cada una de las preguntas de la evaluación guiada.

Cabe destacar que estos tiempos han sido controlados en la primera vez que los usuarios han utilizado el sistema. Como es de esperar, estos tiempo se verán reducidos notablemente conforme los usuarios aprendan dónde mirar y cómo utilizarlo debidamente. Las primeras cinco acciones se corresponden a visualizar los datos del CE o del MD (capítulo 3) y las siguientes a configurar alarmas o visualizar los datos provenientes de ellas. Es más que evidente que la acción más difícil para todos los perfiles ha sido la número 6. En ella se pedía que el usuario configurase una alarma mientras que en el resto sólo se pedía visualizar datos. En esta acción, sólo el 44,4 % de los usuarios ha consultado la ayuda a pesar de no entender exactamente los datos que tenían que introducir. El 22,2 % ha intentando configurar una alarma con menos datos de los necesarios mientras que el 33,3 % ha intentando rellenar todos los campos que aparecían en la pantalla.

La segunda acción que ha requerido más tiempo en ser realizada ha sido la 9 que corresponde a visualizar la tabla de logs. Dos de los tres miembros del perfil clínico han comentado que desconocían el significado de la palabra *log* y, por lo tanto, no sabían con qué asociarlo.

Como conclusiones sobre este cuestionario, se ha observado que los enlaces realizados mediante imágenes resultan más intuitivos que los que llevan algún texto o código y que la información contenida en *Ayuda* apenas es consultada.

En la segunda fase, la evaluación general, se ha dejado a los usuarios un formulario para rellenar con sus opiniones y sugerencias del sistema. Algunas de estas opiniones, se muestran en la siguiente tabla, indicando el porcentaje de usuarios que han contestado SÍ y el que ha contestado NO. El 100 % de los usuarios opina que la aplicación es intuitiva y fácil de manejar además de no haber sufrido ningún error durante la prueba anterior. El 11,11 % de los usuarios considera que el aviso que indica la llegada de nuevos eventos no es suficiente. Sin embargo, la mayoría de los usuarios ha aportado ideas para mejorarlo: cambiar el tamaño, color, añadir efectos de sonido, etc. Un usuario ha comentado la posibilidad de incluir diferentes tipos de avisos según la importancia del evento recibido.

Pregunta	**SÍ**	**NO**
¿El sistema es fácil de usar?	100 %	0 %
¿Es intuitivo?	100 %	0 %
¿Se están produciendo fallos en el funcionamiento del sistema?	0 %	100 %
¿La información proporcionada es suficiente para el seguimiento del correcto funcionamiento de los dispositivos?	100 %	0 %
¿Cree que el aviso de llegada de nuevos eventos es suficiente?	88,89 %	11,11 %
¿Aconsejaría la adopción de un sistema de estas características en un entorno de salud de forma permanente?	100 %	0 %

Tabla 5.1: Respuesta a algunas preguntas de la evaluación general

Respecto a las dificultades encontradas, el 44,4 % de los usuarios no ha comentado ninguna mientras que el 66,6 % de los usuarios del perfil clínico y el 33,3 % de otro perfil han encontrado dificultades para entender algunas palabras técnicas. El 11,11 % de los usuarios ha confesado haber tenido dificultades a la hora de configurar alarmas.

Capítulo 6

Conclusiones y Líneas Futuras

6.1 Conclusiones

En este proyecto se ha implementado un gestor SNMPv3 con un interfaz Web para la gestión de dispositivos médicos en entornos de atención domiciliaria. Para centralizar la información obtenida por los dispositivos, se utiliza el agente SNMP desarrollado en (1). El software disponible para pedir la información recogida mediante esta tecnología requiere conocimientos sobre la misma siendo complicado de utilizar e interpretar. Por este motivo, se ha desarrollado un gestor Web propio que se encargue tanto de pedir los datos deseados, mostrándolos de manera que sea fácil de interpretar para el usuario, como de facilitar la configuración de alarmas y eventos en la medida de lo posible y avisar de manera llamativa la llegada de nuevos eventos. De esta manera, la gestión de dispositivos y sus CEs se realiza de manera transparente e intuitiva por cualquier usuario sin necesidad de tener conocimientos específicos desde cualquier ordenador con conexión a Internet haciendo accesible la telemonitorización de los datos técnicos de estos dispositivos sin requerir personal cualificado para ello.

La tecnología Web utilizada para realizarlo se basa en Servlets, Java y JSP. En concreto, se utiliza el Framework Struts2 que ofrece un potente marco de programación basado en el modelo MVC. Este patrón de arquitectura separa las partes de la aplicación en lógica de negocio, interfaz gráfica y datos, lo que hace

más fácil el manejo de aplicaciones grandes y sus diseños más claros. Lo que esto significa es que se ha creado una aplicación que va a ser fácil de ampliar en un futuro, ya que modificar cualquiera de las partes no va a afectar las otras dos. Además, se utiliza el Framework Tiles que separa las vistas de la aplicación por partes para poder reutilizar las partes comunes, lo que se traduce en poder crear nuevas vistas de manera sencilla.

Los datos de los usuarios que tienen acceso al sistema, de sus agentes asociados y de los datos de conexión necesaria de los mismos así como el registro de los traps recibidos se almacenan en una base de datos MySQL colocada en el mismo servidor donde se encuentra la aplicación Web. Para la conexión a esta base de datos desde la aplicación se utiliza el Framework Hibernate que permite mapear las tablas de la base de datos con clases Java.

Para mostrar los datos técnicos se han incluido herramientas como JFreeChart que permiten visualizarlos de forma gráfica. La visualización gráfica de los datos permitirá interpretar la información de una manera más rápida y eficiente. Por ejemplo, gráficamente se puede apreciar muy bien si un dispositivo está teniendo muchos errores ya que su gráfico tendrá mucho el color correspondiente o se podrá apreciar instantáneamente cuando se está quedando sin batería, etc. También se han creado funciones JavaScript para facilitar la elección de opciones en las distintas páginas y asegurar que el usuario que no va a intentar seleccionar acciones que no tienen sentido.

La comunicación que establece el gestor con el agente MD se realiza mediante el protocolo SNMP que se ha implementado en Java, en la parte correspondiente a la lógica de negocio de la aplicación web usando la librería SNMP4j. Dependiendo de las acciones que se quieran realizar, los mensajes enviados variarán.

Para la captura de los traps recibidos, se ha desarrollado una aplicación que se mantiene escuchando en el puerto 162 y registra todos los eventos recibidos. Si el usuario está conectado al sistema en el momento en el que llega un evento, aparece un icono parpadeante en la esquina superior derecha que avisa su llegada. Esto es

posible con una función AJAX que comprueba la existencia de nuevos eventos periódicamente. AJAX no refresca la página para mostrar datos nuevos sobre la vista, por lo que el usuario no percibirá cuando la función está comprobando los eventos hasta que llegue alguno y la señal de aviso aparezca. Al no refrescar la página, se reduce el tráfico al servidor y se evitan las molestías que podría ocasionar un refresco de página que perdería los datos introducidos.

Por último, se ha realizado una evaluación mediante dos cuestionarios para comprobar si los usuarios son capaces de utilizar todas las funciones del sistema sin ningún problema y recoger opiniones y sugerencias que permitan mejorar el sistema en un futuro.

6.2 Líneas futuras

Algunas posibles líneas futuras para este proyecto serían:

- Diseñar un gestor para la gestión clínica de los dispositivos, para posteriormente, añadirlo a esta aplicación.
- Implementarse en fase de pruebas mientras se sigue evaluando para poder mejorar los fallos que pueda surgir.
- Modificar el agente para permitir la gestión de más tipos de dispositivos y, posteriormente, ampliar la aplicación.

Bibliografía

[1] A.Muñoz Zuara. "Desarrollo de un Agente SNMPv3 para la Gestión Técnica de Dispostivos en Escenarios de Atención Domiciliaria", 2011.

[2] N.Lasierra, A.Muñoz, A.Alesanco, J.Escayola, J.García. "Una solución SNMP para la gestión técnica de dispositivos médicos ISO/IEEE 11073", 2011.

[3] J. García and A. Alesanco. "Web-Based System for Managing a Telematics Laboratory Network" *IEEE TRANSACTIONS ON EDUCATION*, 2004.

[4] "Framework Struts2", http://struts.apache.org/2.x/.

[5] "Normas Usabilidad", http://www.posicionamiento.cl/faq_normas_de_usabilidad.php.

[6] "¿Qué es usabilidad?", http://www.desarrolloweb.com/articulos/221.php.

[7] "Framework Tiles", http://tiles.apache.org/.

[8] "Framework Hibernate", http://www.hibernate.org/.

[9] K. McCloghrie and M. Rose. "RFC 1213 - Management Information Base for Network Management of TCP/IP-based internets: MIB-II",*Internet Engineering Task Force (IETF)*,1990. http://tools.ietf.org/html/rfc1213.

[10] U. Blumenthal and B. Wijnen. "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)". *RFC 2574, Internet Engineering Task Force (IETF)*, 1999.

[11] B. Wijnen, R. Presuhn, and K. McCloghrie. "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)". *RFC 2575, Internet Engineering Task Force (IETF)*, 1999.

[12] K. McCloghrie, D. Perkins, and J. Schoenwaelder. "Structure of Management Information Version 2 (SMIv2)". *RFC 2578, Internet Engineering Task Force (IETF)*, 1999.

[13] "API SNMP4j". http://www.snmp4j.org/

[14] "API Google Maps ". http://code.google.com/intl/es/apis/maps/.

[15] "Librería OpenSource JFreeChart". http://www.jfree.org/jfreechart/

[16] "Tutorial AJAX". http://www.ajaxya.com.ar/

[17] "Página de Oracle". http://www.oracle.com/index.html

[18] "Página XAMPP". http://www.apachefriends.org/es/xampp.html

[19] "Servidor Apache Tomcat 6.0". http://tomcat.apache.org/download-60.cgi

[20] "Programa OpenSSL". http://openssl.org

[21] "Keytool". http://docs.oracle.com/javase/1.3/docs/tooldocs/win32/keytool.html

Anexo A

Información Framework Struts2

A.1 Arquitectura Struts2

El framework Struts2 está compuesto compuesto por los siguiente elementos:

1. *FilterDispatcher.* Es el controlador en Struts2 y es el primer componente en actuar en el ciclo de vida de la petición. Básicamente, es un ServletFilter cuyo principal objetivo es interpretar todas las peticiones entrantes y determinar qué Action y qué interceptores deberían ejecutarse.

2. *ActionMapper.* Esta clase es usada por el FilterDispatcher para determinar si la action debería ser invocada o no. Es la clase que guarda toda la información de mapeo necesaría para invocar un Action.

3. *ActionProxy.* Si la petición debe invocar un Action, el FilterDipatcher delega el control al ActionProxy que consulta el ConfigurationManager y luego crea un ActionInvocation.

4. *ConfigurationManager.* Es un objeto java que representa el fichero struts.xml. Es creado al iniciar la aplicación y contiene toda la información de configuración.

5. *Struts.xml.* Este fichero es el núcleo de la configuración de Struts2. Contiene los mapeos definidos por el usuario para cada Action, Interceptors y resultados.

6. *ActionInvocation.* Es el responsable de invocar cualquier interceptor antes de invocar al propio Action. Una vez que el Action ha terminado, es el responsable de buscar el resultado correspondiente asociado a la salida del Action y ejecutarlo.

7. *Interceptor.* Los interceptores son invocados tanto antes como después de que el Action es ejecutado. No tienen que realizar acciones en ambas ocasiones necesariamente pero la petición pasará por ellos igualmente.

8. *Action.* Struts2 está basado en la arquitectura MVC, por lo que para cada tarea específica, debe haber un Acion concreto que la maneje.

9. *Result.* El resultado traduce el estado de la aplicación en una presentación visual con la que el usuario puede interactuar. El Action es responsable de decidir qué respuesta mandará para determinada petición.

En la figura A.1, se observa la arquitectura de Struts2, con los elementos descritos arriba.

Una petición en Struts2 sigue los siguientes pasos:

1. El usuario envía la petición.

2. Esa petición pasa por el FilterDispatcher que es el filtro que se encarga de determinar la acción que tiene que realizar.

3. Antes de llegar a la acción, pasará por los interceptores, si es necesario.

4. Se ejecuta la acción.

5. Se devuelve el resultado de la acción pasando por los interceptores en orden inverso, si los hubiera.

6. El resultado es mostrado al usuario.

A.2 Características Struts2

Las principales ventajas que ofrece Struts2 son las siguientes:

1. *Buenas prácticas.* Basado en la arquitectura MVC.

2. *Simplicidad de diseño.* La mayoría de las clases Action de Struts2 están basadas en interfaces aunque no es necesario y son independientes del código HTTP. Están simplificadas para parecer simples POJOs (simples clases java). Cualquier clase java con el método execute() puede ser utilizada con una clase Action.

3. *Extensibilidad.* Fácilmente extensible debido al ligero acoplamiento de sus componentes.

4. *Resultados flexibles.* Struts2 proporciona flexibilidad a la hora de crear múltiples tipos de salida y esto ayuda a preparar la respuesta.

5. *Uso de anotaciones.* Las aplicaciones pueden usar Annotations como alternativa a los XML y los ficheros properties de configuración.

6. *Integración con otros componentes.* permite el uso de Plugins de componentes e integración con otros Framework.

A.3 Struts2 vs Struts1

Struts2 es la versión mejorada de Struts1 ya que complementa las mejores características de ésta con las del framework WebWorks. A continuación, se detallan las principales diferencias.

1. **Clases Action**

   (a) *Struts1.-* Las clases Action extienden de una clase base abstracta en vez de implementar una interfaz.

(b) *Struts2.-* Las clases Action pueden implementar una interface Action, además de otras interfaces. Struts2 proporciona una clase base ActionSupport que implementa las interfaces más comunes. Sin embargo, la interfaz Action no es requerida, ya que se puede usar cualquier objeto POJO *(Plain Old Java Object)* con un método execute() como clase Action.

2. **Modelo Threading**

(a) *Struts1.-* Los objetos solo son instanciados una vez para manejar todas las peticiones de ese Action. Esta estrategia impone restricciones en lo que se puede hacer.

(b) *Struts2.-* Los objetos del Action son instanciados en cada petición, por lo que no hay que aplicar temas de seguridad en los Threads.

3. **Dependencia Servlet**

(a) *Struts1.-* HttpServletRequest y HttpServletResponse son pasados al método execute cuando el Action es invocado.

(b) *Struts2.-* Los Actions son simples POJOs así que la mayoría de los contextos del Servlet son representados como Maps, eliminando esa dependencia. A pesar de ello, se puede acceder a la petición y respuesta originales, si es necesario.

4. **Testeabilidad**

(a) *Struts1.-* El método execute del Action es dependiente del Servlet, por lo que para testearlo, se necesita una extensión, Struts TestCase, que ofrece un set de objetos simulados para ellos.

(b) *Struts1.-* En este caso, el Action puede ser testeado solo con ser instanciado, iniciar las variables e invocar los métodos.

5. **Recolección parámetros de entrada**

   (a) *Struts1.-* Utiliza ActionForms para captura los parámetros de entrada. Al igual que los Actions, todos los ActionForms tienen que extender de una clase base. JavaBeans no pueden ser usados como ActionForms, por lo que los desarrolladores crean clases redundantes para capturar los parámetros.

   (b) *Struts2.-* Utiliza las propiedades del Action como los parámetros de entrada, eliminando la necesidad de un segundo objeto de entrada. En este caso, las propiedades también pueden ser clases.

6. **Expresiones de Lenguaje**

   (a) *Struts1.-* Está integrado con JSTL que tiene los objetos básicos.

   (b) *Struts2.-* También puede utilizar JSTL pero el Framework tiene integrado un sistema más flexible, el OGNL.

7. **Unión valores dentro de las vistas**

   (a) *Struts1.-* Utiliza los mecanismos tradicionales de JSP para unir los objetos dentro de la página de contexto.

   (b) *Struts2.-*Usa la tecnología "ValueStack"para que la que vista pueda acceder a los valores usando taglibs.

8. **Tipo de conversación**

   (a) *Struts1.-* Todas las propiedades del ActionForm son, normalmente, cadenas. Para la conversión, se utiliza Commons-Beanutils.

   (b) *Struts2.-*Utiliza OGNL para la conversión. Además incluye conversores a tipos primitivos o para algunos tipos de objetos más comunes.

9. **Validación**

   (a) *Struts1.-* Soporta validación manual por medio del método validate() del ActionForm o extendiendo de Commons Validator.

(b) *Struts2.-* También soporta la validación por medio del método validate() y por medio del Framework Xwork Validator. Este framework permite diferentes tipos de validaciones según el contexto en el que se encuentren.

10. **Control de la ejecución del Action**

(a) *Struts1.-* Permite distintas peticiones para cada módulo pero todas los Actions del mismo módulo comparten el mismo ciclo de vida.

(b) *Struts2.-* Permite crear diferentes ciclos de vida por Action.

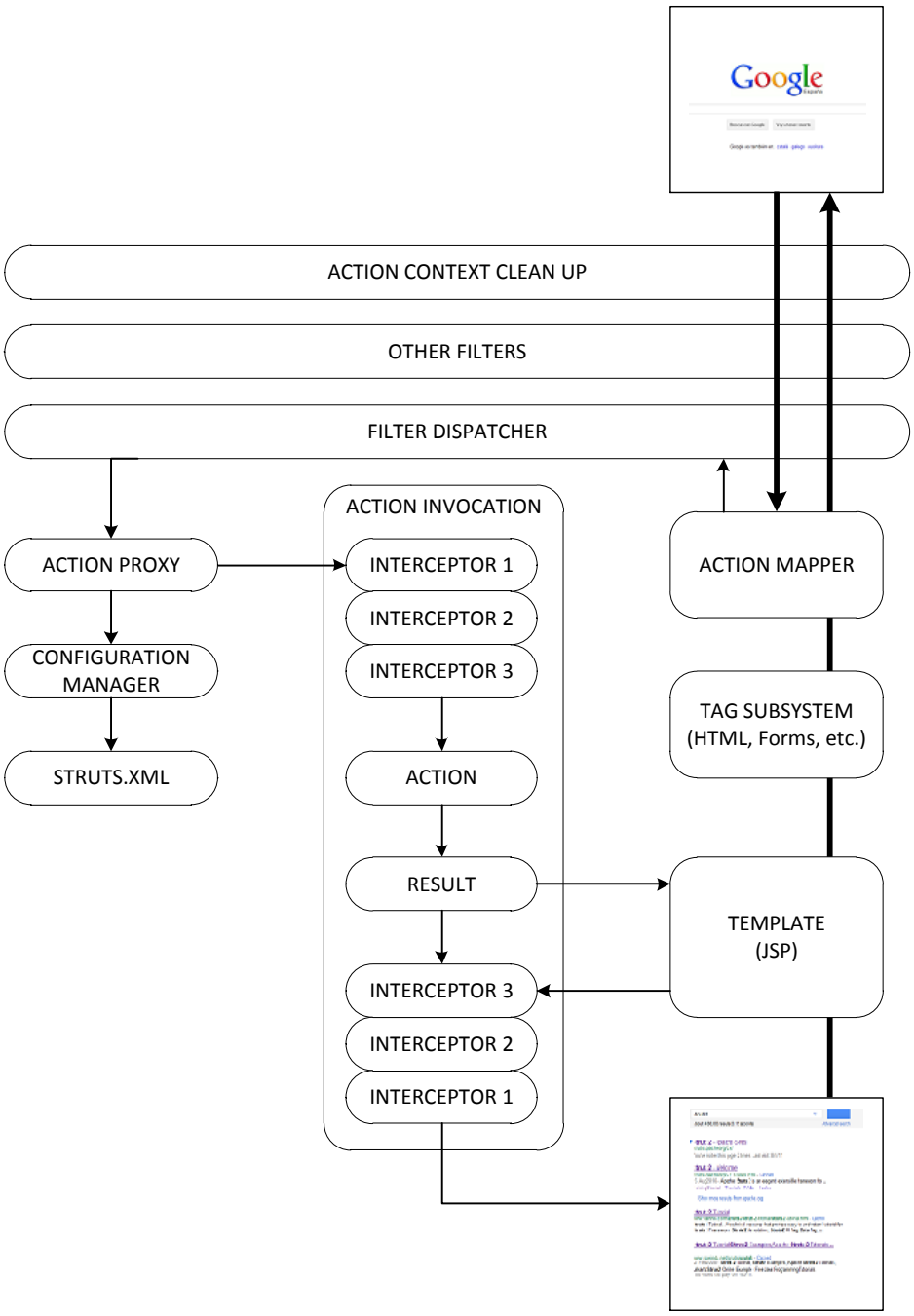


Figura A.1: Arquitectura Framework Struts2.

Anexo B

Descripción detallada objetos MIB MD

La MIB diseñada en (1) es una MIB privada diseñada dentro del grupo zaragozaNetworkManagementResearchGroup, creado en la Universidad de Zaragoza para contener MIBs creadas por sus grupos de investigación. La MIB MD, llamada MedicalDevicesManager, tiene el índice 4 dentro de este grupo, por lo que el OID para acceder a sus datos es: 1.(iso). 3(org). 6(dod). 1(internet). 4(private). 1(enterprises). 28308(zaragozaNetWorkManagementResearchGroup). 4(medicalDevicesManager).

En la figura B.1 se observa la estructura jerárquica de la MIB. En la figura B.1 se observa la estructura de la MIB. Está compuesta por cinco grupos que se detallan a continuación.

B.1 Grupo ComputeEngineControlInfo

En la figura B.2, se muestra este grupo. En este grupo se almacena desde la información estática asociada al CE como los recursos utilizados por el mismo. También se incluye en este grupo la dirección dónde se encuentra el CE y dos variables que permiten al gestor pedir actualizaciones de la información técnica y estados de los dispositivos. Los parámetros de este grupo se muestran a continuación:

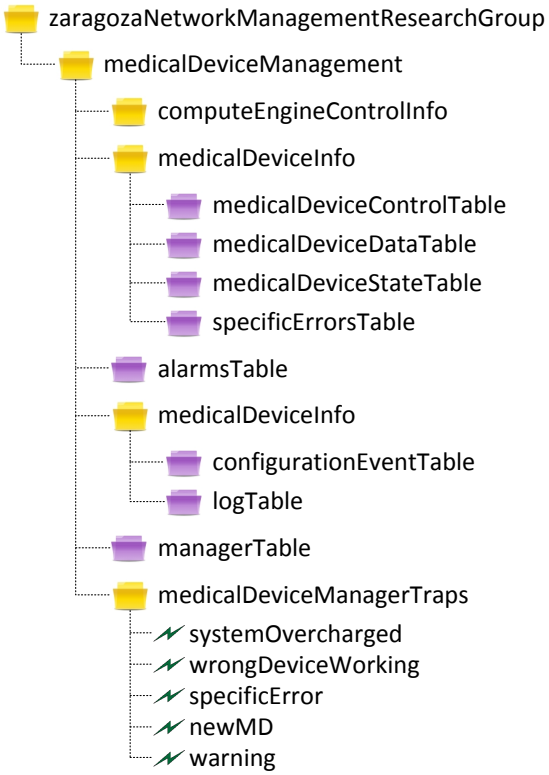


Figura B.1: Estructura MIB MD.

- **IdComputeEngine.** Identificador del CE.

- **DeviceType.** Tipo de dispositivo que es el CE.

- **WorkingState.** Indica el estado de funcionamiento del CE. Este estado puede ser operativo o dañado.

- **CommunicationState.** Este parámetro indica si el CE está mandando información, recibiendo información o en estado *idle*.

- **AsociatedMDs.** Número de MDs registrados en este CE.

- **BandwidthUse.** Porcentaje del ancho de banda utilizado por el CE sobre el total disponible. Este parámetro permite evitar problemas de congestión.

- **CpuUse.** Porcentaje del uso del procesador por parte del CE. Su control permite evitar sobrecargar de procesos el CE y por tanto ralentizarlo.

- **HardDiskMemoryUse.** Porcentaje de disco duro utilizado por el CE. El

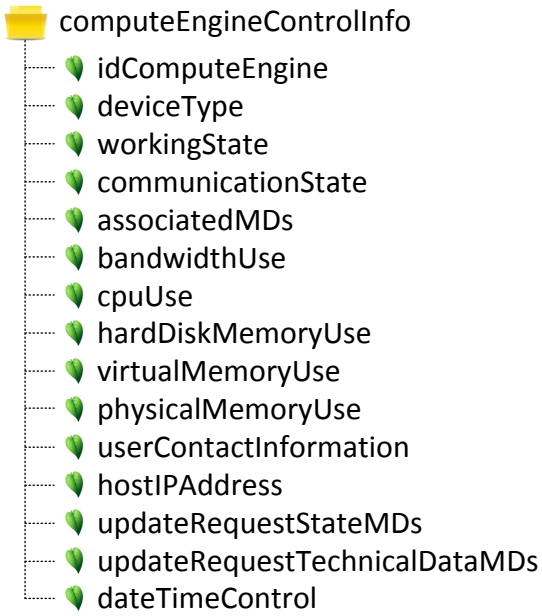


Figura B.2: Estructura Grupo ComputeEngineControlInfo MIB MD.

control de esta variable nos permite conocer la cantidad de espacio libre en el disco duro y evitar quedarnos sin espacio para almacenamiento.

- **VirtualMemoryUse.** Porcentaje de memoria virtual utilizada por el CE. Su control permite evitar la falta de memoria virtual.

- **PhysicalMemoryUse.** Porcentaje de memoria física, o memoria RAM, utilizada por el CE. Su control permite evitar la falta de memoria física y por tanto el ralentizamiento del procesador.

- **UserContactInformation.** Especifica la dirección donde se encuentra el CE.

- **HostIPAddress.** Dirección IP del CE.

- **UpdateRequestStateMDs.** Esta variable sirve para pedir actualizaciones del estado de todos los dispositivos registrados en el CE.

- **UpdateRequestTechnicalDataMDs.** Esta variable sirve para pedir actualizaciones de la información técnica de todos los dispositivos registrados en el CE.

- **dateTimeControl.** Fecha en la que fue realizada la última actualización de los recursos propios del CE.

B.2 Grupo MedicalDeviceInfo

Este grupo contiene toda la información relativa a los MDs asociados al CE. Esta información se divide a su vez en cuatro tablas: datos de control, datos técnicos, cambios de estado de los MDs y errores que hayan surgido durante las capturas.

B.2.1 MedicalDeviceControlTable

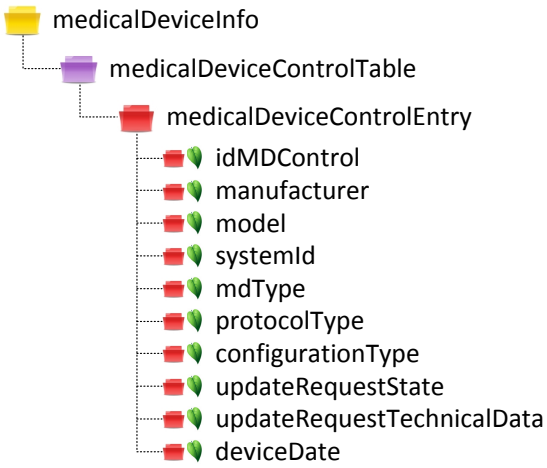


Figura B.3: Estructura MedicalDeviceControlTable MIB MD.

En la figura B.3, se muestra la tabla de control de los MDs. Está tabla guarda l información estática de los MDs. Tiene una entrada por cada MD asociado que se registra la primera vez que el MD se conecta al CE. Los parámetros de esta tabla se describen a continuación:

- **IdMDControl.** Identificador que se le asigna a cada MD dentro de la MIB.
- **Manufacturer.** Fabricante del MD.
- **Model.** Modelo del MD dentro de la gama del fabricante.

- **SystemId.** Identificador del MD proporcionado por el fabricante.
- **MDType.** Tipo de MD que se ha registrado. Hay 4 tipos: termómetro, báscula, medidor de presión arterial o pulsioxímetro.
- **ProtocolType.** Tipo de protocolo utilizado en la comunicación entre MD y manager. En este caso será por defecto el protocolo X.73.
- **ConfigurationType.** Tipo de configuración con la que está trabajando el MD. Puede ser estándar o extendida.
- **UpdateRequestState.** Este parámetro se utiliza para pedir actualizaciones del estado de éste dispositivo en concreto.
- **UpdateRequestTechnicalData.** Este parámetro se utiliza para pedir actualizaciones del estado de éste dispositivo en concreto.
- **DeviceDate.** Fecha en la que se registró este MD.

B.2.2 MedicalDeviceDataTable

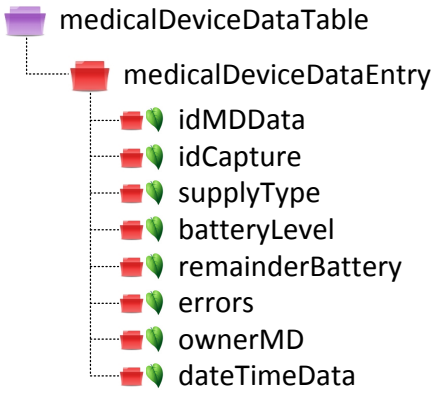


Figura B.4: Estructura MedicalDeviceDataTable MIB MD.

En la figura B.4, se muestra la estructura de la tabla de datos técnicos de los MDs. Como su propio nombre indica, registra los datos técnicos recibidos de los MDs asociados. Esta tabla se indexa por el identificador del MD y de la captura. Puede haber un máximo de 20 capturas por MD. Los objetos de esta tabla se describen a continuación:

- **IdMDData.** Identificador del MD al que corresponde la captura.

- **IdCapture.** Identificador de la captura.

- **SupplyType.** Ttipo de alimentación que está utilizando el MD. Puede variar entre batería o conexión a la red eléctrica.

- **BatteryLevel.** Porcentaje de batería que le queda al MD.

- **RemainderBattery.** Número estimado de horas que puede funcionar el MD con la batería que le queda.

- **Errors.** Muestra si ha habido algún error en el MD al tomar esta medida. Su valor es OK si no ha habido problema, y Error si lo ha habido.

- **OwnerMD.** Persona que ha realizado la medida.

- **DateTimeData.** Fecha en la que se realizó esta entrada en la tabla.

B.2.3 MedicalDeviceStateTable

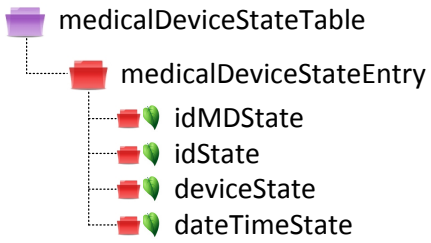


Figura B.5: Estructura MedicalDeviceStateTable MIB MD.

En la figura B.5, se puede observar esta tabla. La tabla medicalDeviceSstateTable muestra los cambios de estados que se han recibido de los MDs asociados al CE. Al igual que la tabla anterior, también puede contener un máximo de 20 cambios de estado por cada MD asociado. La descripción de los objetos que se pueden observar en esta tabla es la siguiente:

- **IdMDState.** Identificador del MD al que corresponde el estado que se registra en esta entrada.

- **IdState.** Identificador del estado registrado en esta entrada.

- **DeviceState.** Valor del nuevo estado del MD.Este valor será una de estas posibilidades: *available*, *notavailable*, *disconnected*, *connected*, *associated* u *operating*.

- **DateTimeState.** Fecha en la que se realizó esta entrada en la tabla.

B.2.4 SpecificErrorsTable

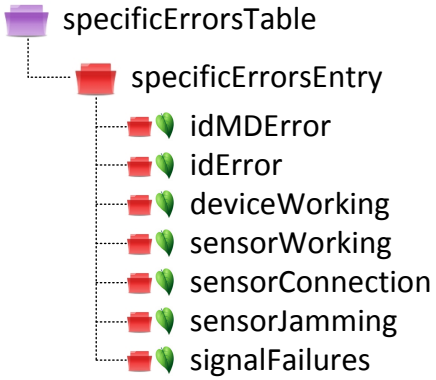


Figura B.6: Estructura SpecificErrorsTable MIB MD.

En la figura B.6, se visualiza la estructura de la tabla. Esta tabla es la encargada de almacenar los tipos de errores que se han producido al recibir los datos técnicos de los MDs asociados. También permite 20 entradas como máximo. Los objetos se detallan a continuación:

- **IdMDError.** Identificador del MD al que corresponde el error que se registra en esta entrada.

- **IdError.** Identificador del error registrado en esta entrada.

- **DeviceWorking.** Indica si hay algún error en el funcionamiento general del MD. En caso de haberlo, si valor será Error; si no lo hay, el valor será OK.

- **SensorWorking.** Indica si hay algún error en el funcionamiento del sensor del MD, en caso de que posea sensor. En caso de haberlo, si valor será Error; si no lo hay, el valor será OK.

- **DeviceConnection.** Indica si hay algún error en la conexión del MD con el CE. En caso de haberlo, si valor será Error; si no lo hay, el valor será OK.

- **SensorJammin.** Indica si hay algún error debido a interferencias en el sensor del MD, en caso de poseer sensor. En caso de haberlo, si valor será Error; si no lo hay, el valor será OK.

- **SignalFailures.** Indica si hay algún error debido a que las señales médicas que utiliza el dispositivo para calcular los valores médicos son pobres o tienen algún problema. En caso de haberlo, si valor será Error; si no lo hay, el valor será OK.

B.3 Grupo AlarmTable

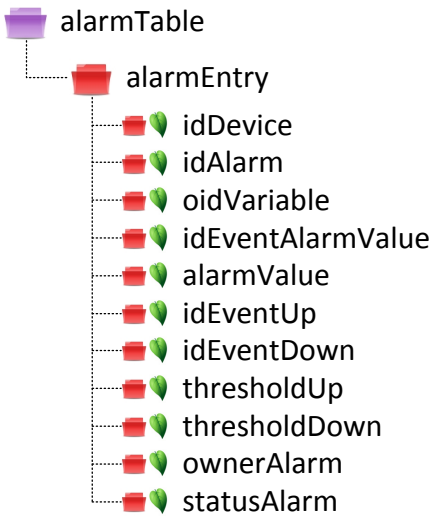


Figura B.7: Estructura Grupo AlarmTable MIB MD.

En la figura B.7, se observa esta tabla. Las entradas de esta tabla son añadidas por el gestor. Para crear entradas en esta tabla, el gestor tiene que enviar una petición inicial de creación de la tabla. Posteriormente, va rellenando el resto de valores aunque no es necesario completar todos los campos. Esta tabla permite la creación de tres tipos de alarmas diferentes: por valor, esta alarma será activada cuando el recurso asociado a la alarma tome un determinado valor; por umbral superior, esta alarma será activada cuando el recurso asociado alcance un valor

superior al umbral; y por umbral inferior, esta alarma será activada cuando el recurso asociado alcance un valor inferior al umbral. La descripción de los objetos pertenecientes a esta tabla es la siguiente:

- **IdDevice.** dentificador del MD para el cual se ha definido esta alarma.
- **IdAlarm.** Identificador de la alarma. Este identificador no tiene por qué ser consecutivo, el gestor que crea la alarma elige su valor.
- **OidVariable.** Muestra el OID (recurso) que vamos a monitorizar para comprobar que sus valores no hacen activarse a la alarma.
- **IdEventAlarmValue.** Identificador del evento asociado a esta alarma cuando la variable que monitorizamos alcanza el valor indicado en AlarmValue.
- **AlarmValue.** Cuando la variable con OID igual al del campo OidVariable alcanza el valor indicado en este objeto, se activa la alarma.
- **IdEventUp.** Identificador del evento asociado a esta alarma cuando la variable que monitorizamos sobrepasa por encima el valor indicado por ThresholdUp.
- **IdEventDown.** Identificador del evento asociado a esta alarma cuando la variable que monitorizamos sobrepasa por debajo el valor indicado por ThresholdDown.
- **ThresholdUp.** Cuando la variable con OID igual al del campo OidVariable sobrepasa por encima el valor indicado en este objeto, se activa la alarma.
- **ThresholdUp.** Cuando la variable con OID igual al del campo OidVariable sobrepasa por debajo el valor indicado en este objeto, se activa la alarma.
- **OwnerAlarm.** Identificador del gestor que ha creado la alarma.
- **StatusAlarm.** Estado de creación de la alarma. Si el valor es 1 (valid), implica que la entrada está creada y completada. Si el valor es 3

(underCreation), implica que la entrada aún está por completar. Los valores 2(creation) y 4 (drop) sirven para crear y borrar la entrada, respectivamente.

B.4 Grupo EventTables

Esta tabla contiene la información de eventos activos en el sistema. Está dividida, a su vez en dos tablas que se explican a continuación.

B.4.1 ConfigurationEventTable

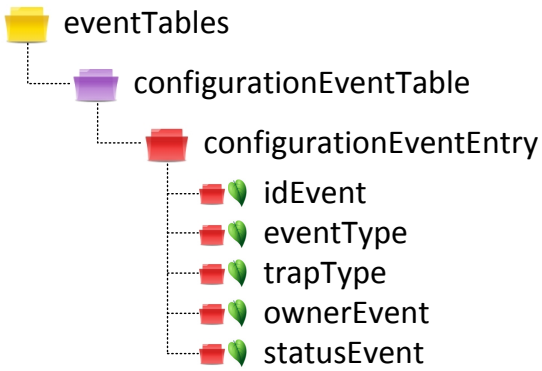


Figura B.8: Estructura ConfigurationEventTable MIB MD.

Esta tabla se muestra en la figura B.8. Es otra de las tablas configurables por el gestor. En ella se crean los eventos que se asocian a alguna alarma y serán lanzados al activársela misma. La descripción de los objetos visibles en esta tabla es la siguiente:

- **IdEvent.** Identificador del evento. El gestor puede darle el valor que quiera a este objeto al configurar el evento.

- **EventType.** Indica el tipo de evento que se ha creado. El tipo de evento describe las acciones que se realizan al activar la alarma. Estas acciones pueden variar entre enviar un trap, crear una entrada en la tabla de logs o ambas acciones .

- **trapType.** Indica el tipo de trap que se va a enviar al gestor. Los tipos de trap son: *SystemOvercharged, WrongDeviceWorking, SpecificError, NewMD* y *Warning.*

- **OwnerEvent.** Identificador del gestor que configuró este evento.

- **StatusEvent.** Estado de creación del evento. Si el valor es 1 (valid), implica que la entrada está creada y completada. Si el valor es 3 (underCreation), implica que la entrada aún está por completar. Los valores 2(creation) y 4 (drop) sirven para crear y borrar la entrada, respectivamente.

B.4.2 LogTable

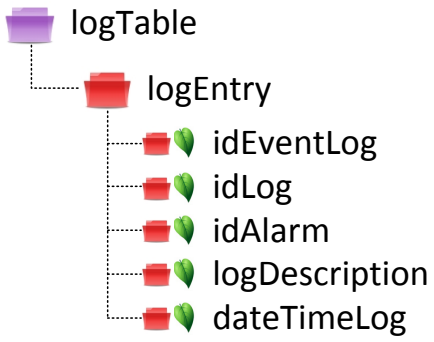


Figura B.9: Estructura LogTable MIB MD.

Uno de los eventos que se pueden configurar en la tabla anterior es escribir una entrada nueva en la tabla de logs. En esta tabla, cuya estructura se muestra en la figura B.9, es donde se registran esas entradas. A continuación se describen los objetos de la tabla:

- **IdEventLog.** Identificador del evento que responsable de esta entrada en la tabla de logs.

- **IdLog.** Identificador del log.

- **IdAlarm.** Identificador de la alarma responsable de la activación del evento y por tanto de la creación del log.

- **LogDescription.** Describe brevemente la causa de la activación de la alarma.

- **DateTimeLog.** Fecha en la que fue creado este log.

B.5 Grupo ManagerTable

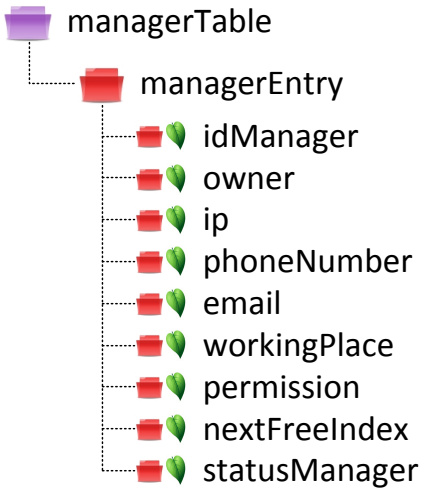


Figura B.10: Estructura Grupo ManagerTable MIB MD.

La última table de la MIB se muestra en la figura B.10. Esta tabla registra la información asociada a los gestores cuyo contenido puede ser añadido por el administrador del sistema. La descripción de este último grupo es la que sigue:

- **IdManager.** Identificador del gestor.

- **Owner.** Nombre del gestor.

- **IP.** Dirección IP del gestor al que se enviarán traps.

- **PhoneNumber.** Teléfono del gestor.

- **Email.** Email del gestor.

- **Workingplace.** Dirección donde se puede comunicar con el gestor de manera física.

- **Permission.** Indica los privilegios del gestor dentro de la MIB.

- **NextFreeIndex.** Indica el siguiente índice libre.

- **StatusManager.** Estado de creación del gestor. Si el valor es 1 (valid), implica que la entrada está creada y completada. Si el valor es 3 (underCreation), implica que la entrada aún está por completar. Los valores 2(creation) y 4 (drop) sirven para crear y borrar la entrada, respectivamente.

B.6 Definición de los Traps

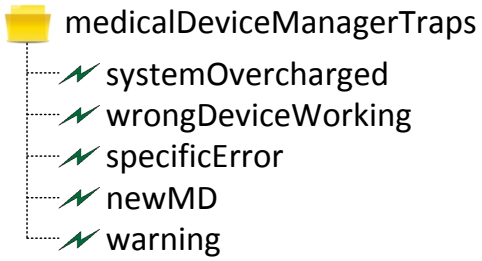


Figura B.11: Definición de Traps MIB MD.

Un trap es un mensaje asíncrono que envía el agente al gestor asociado para informar de algún cambio importante en el mismo. Los traps definidos en esta MIB se muestran en la figura B.11. Hay cinco tipos:

- **SystemOvercharged.** Será enviado cuando alguno de los recursos del CE se encuentre saturado. Las variables que se adjuntan son el ancho de banda, uso de CPU y uso de memoria, además de incluir la fecha a la que se detectó la sobrecarga.

- **WrongDeviceWorking.** Si alguna medida proveniendo de los MDs asociados no está dentro del rango definido como normal, se envía este trap. Justo a este mensaje, se envía el identificador del dispositivo, el identificador de la factura y la fecha a la que ocurrió.

- **SpecificError.** Este trap está asociado a la table con el mismo nombre. Incluye todos los parámetros de la misma.

- **NewMD.** Cada vez que un MD nuevo se registra al CE, se envía este trap. Para más detalle, se adjunta la entrada que ha creado en la tabla MDControl.

- **Warning.** Este trap es genérico y está definido para cubrir el resto de recursos que no tiene Trap asociado. Al ser enviado, incluye el recurso y el valor que ha hecho activarse a la alarma.

Anexo C

Estructura de la aplicación

La aplicación se ubica en la carpeta *Webapps* del servidor *Apache Tomcat 6.0* en una carpeta llamada *Gestor* que se muestra en la figura C.1.

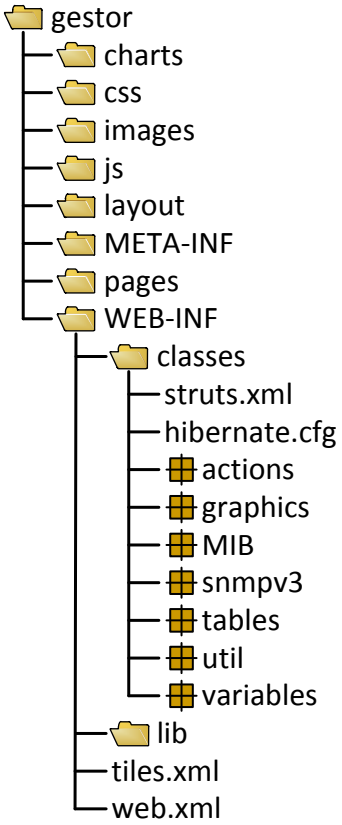


Figura C.1: Estructura de Carpetas de la Aplicación.

C.1 Carpeta Gestor

Esta carpeta contiene otras carpetas que se explican a continuación:

- **charts.-** Contiene los archivos que se encargan de preparar los gráficos para representarlos.

- **css.-** Contiene las hojas de estilo *.css* que definen la aplicación.

- **images.-** Guarda todas las imágenes que se utilizan a lo largo de todas las vistas.

- **js.-** Almacena librerías JavaScript complementarios a las funciones creadas para la aplicación.

- **layout.-** Define los Layout de Tiles, es decir, las bases de las cuales todas las vistas van a heredar.

- **META-INF.-** Tiene los archivos de persistencia.

- **pages.-** Contiene todas las páginas JSP que constituyen la vista del sistema.

- **WEB-INF.-** Contiene toda la información de configuración necesaria para la aplicación Web. En su interior, se guarda el fichero de configuración *web.xml*, que actúa de controlador principal, determinando como asignar a los Servlets, si se necesita autentación, etc. Además de este fichero, se encuentra también el fichero *tiles.xml* que indica el body JSP que se utilizará para cada vista. Esta carpeta también tiene a su vez otras dos carpetas: *lib* y *classes*. La carpeta *lib* almacena todas las librerías necesarias para el correcto funcionamiento de la aplicación. *classes* contiene todas las clases Java que se han implementado además de los archivos de configuración de Struts2 e Hibernate y los ficheros *.properties*. El fichero de configuración de Struts2, *struts.xml*, indica los Interceptores, Actions y a qué página habrá que dirigirse en cada momento. El fichero de configuración de Hibernate, *hibernate.cfg* contiene la información necesaria para conectarse a la base de datos indicando

también dónde se encuentran las clases Java a las que se tiene que mapear. Los ficheros *.properties* guardan los textos de todas las etiquetas en Ingles y en Español. Esta carpeta separa su contenido en distintos paquetes según el tipo de acciones que realizan las clases Java que hay en su interior. Estas carpetas son:

- **actions.-** Contiene las clases Action de Struts2 que realizarán las operaciones oportunas para decidir la respuesta mostrada.
- **graphics.-** Almacena las clases que ayudarán a la creación de los gráficos de datos.
- **MIB.-** Conjunto de clases que tienen la estructura de la MIB para facilitar el manejo de los datos recibidos.
- **snmpv3.-** Estas clases envían los mensajes SNMP al agenteMD adecuados en cada momento.
- **tables.-** Clases Java a las que se mapea la base de datos que usa la aplicación.
- **util.-** Define métodos para la conexión a la base de datos mediante HIbernate.
- **Variables.-** Conjunto de métodos auxiliares que se utilizan para mostrar los datos provenientes de la MIB en el idioma correcto.

Anexo D

Guía de Instalación

En este Anexo se va a explicar los pasos a seguir para instalar correctamente la aplicación. Hay que tener en cuenta que esta aplicación se comunica con un agenteMD, cuyos datos de conexión se almacenan en la base de datos, por lo que esta aplicación no funcionará correctamente si los datos son incorrectos o el agente no es accesible desde el lugar donde está colocada la aplicación.

D.1 Preparar el entorno Java

Para el desarrollo de esta aplicación se ha utilizado el JDK 1.6 (*Java Development Kit*). El primer paso para configurar el entorno es descargarse el kit desde la página de Oracle (17). Una vez descargado, se instala y se define su variable de entorno *JAVA_HOME*. Para ello, hay que ir a:
Inicio → *Panel de Control* → *Sistema* → *Configuración Avanzada del sistema* → *Variables de Entorno ...*
Una vez en la ventana de *Variables de Entorno*, se crea una nueva con los siguientes datos:

Nombre de la variable: JAVA_HOME

Valor de la variable: Directorio donde se ha instalado el JDK (Ejemplo: C:\Program Files\Java\jdk1.6.0_21)

Además de crear esta nueva variable de entorno, hay que modificar una existente llamada *PATH*, añadiendo al final (sin eliminar los valores existentes) este valor *;%JAVA_HOME%\bin*.

D.2 Instalación de MySQL

Descargar XAMPP (18) e instalarlo. Una vez instalado, hay que Iniciar el servidor Apache y MySQL pulsando el botón *Start*. Cuando ambos servidores estén funcionando, hay que acceder desde el navegador a http://localhost/phpmyadmin y crear una nueva base de datos llamada *usuarios*. También hay que crear un usuario con todos los privilegios, desde la pestaña de *Privilegios* → *Agregar un nuevo usuario*. Los datos del usuario a crear son los siguientes:

Nombre de usuario: veronica

Servidor: %

Contraseña: vero2108

Después de crear el usuario adecuado, se ejecuta *hibernate.exe* que es un programa ejecutable que crea las tablas necesarias y añade dos usuarios para las pruebas: *admin* y *manager* y los agentesMD asociados. Las tablas creadas son cuatro:

D.2.1 Tabla user

Esta tabla contiene los usuarios que tienen al acceso al sistema. La estructura de esta tabla es la siguiente:

- **userId.-** Identificador del usuario generado automáticamente por Hibernate.
- **username.-** Nombre del usuario para el acceso al sistema.
- **password.-** Contraseña del acceso al sistema del usuario.

- **category.-** Indica la categoría del usuario: root, manager o doctor.

- **fullName.-** Nombre completo del usuario.

- **email.-** Correo electrónico del usuario.

- **phoneNumber.-** Teléfono de contacto del usuario.

D.2.2 Tabla user_computeengine

Es una tabla relacional creada por Hibernate para almacen los CEs asociados a los usuarios registrados en la tabla *user*.

- **user_userID.-** Código del usuario de la tabla *user*.

- **computeEngines_idComputeEngine.-** Código del CE de la tabla *computeEngines*.

D.2.3 Tabla computeEngine

Esta tabla guarda la información necesaria para la conexión con los agentesMD. Los campos necesarios para ellos, son:

- **idComputeEngine.-** Código del CE generado automáticamente por Hibernate.

- **username.-** Nombre de usuario asociado al CE.

- **idCEMIB.-** Código de la MIB para la aplicación Web.

- **ipUser.-** Dirección IP para la conexión con el agenteMD.

- **port.-** Puerto con el que hay que comunicarse.

- **securityName.-** Nombre de seguridad para la conexión.

- **securityLevel.-** Nivel de seguridad requerido.

- **authPassword.-** Contraseña de autenticación, si es necesaria.

- **privPassword.-** Contraseña privacidad para la conexión.

D.2.4 Tabla traps

Esta tabla registra todas los traps que se reciben de los agente. La estructura es la siguiente:

- **idTrap.-** Identificador del Trap generado automáticamente por Hibernate.

- **IPAgente.-** Dirección IP del agente que ha enviado el trap.

- **idDevice.-** Identificador del MD asociado a la alarma activada.

- **idCaptura.-** Identificador de la captura que ha producido el envío del trap.

- **trapType.-** Tipo de trap recibido.

- **dateTime.-** Fecha en la que se recibió el trap.

- **nuevo.-** Indica si el trap se ha visualizado o no.

D.3 Gestor SNMP Trap

El gestor necesita escuchar constantemente en el puerto 162. Para ello, hay que liberar cualquier aplicación que pueda estar usando ese puerto siguiendo los siguientes pasos:

- Ir al fichero C:\WINDOWS\System32\drivers\etc\services para cambiar la aplicación *snmptrap* a un puerto diferente. Por ejemplo, el puerto 163.

- Hay que asegurarse de que no haya ninguna aplicación utilizando el puerto 162. Para ello, ejecutamos netstat -a desde línea de comandos y, si hay alguna aplicación que lo utilice, se busca en el Administrador de tareas → Terminar tarea correspondiente.

Una vez que el puerto se haya quedado libre, se procede a ejecutar *GestorTraps*. Si ha habido algún error, se creará un fichero *log.txt* en el mismo directorio. Si todo ha ido correctamente, a partir de este momento, todos los traps recibidos se almacenarán en la base de datos traps.

D.4 Configuración Tomcat

Para el desarrollo de esta aplicación, se ha utilizado Apache Tomcat 6.0, disponible en (19).

D.4.1 Instalación de soporte SSL para Tomcat

Para permitir la conexión mediante el protocolo TLSv1 a nuestra aplicación, se necesitan tres cosas: una CA (*Certification Authority*), un certificado público y un almacén de claves. Una CA es una entidad de confianza que se encarga de emitir y revocar certificados digitales. Un certificado público es un certificado que necesita el navegador para reconocer a nuestra CA y permitir la conexión como página de confianza. Y, por último, el almacén de claves donde se guardan las claves generadas para el servidor, encargadas de cifrar la comunicación entre el servidor y los clientes y de asegurar la confianza en el mismo. Para crearlos se utiliza OpenSSL (20) y keytool (21). Estos pasos no son necesarios realizarlos ya que los ficheros se proporcionan en el CD.

D.4.1.1 Generar certificado CA

Para generar el certificado CA, hay que ejecutar, sobre línea de comandos:

```
C:\AutenticacionCA>openssl genrsa -out keys/ca.key 1024
```

Una vez creadas las claves, hay que crear un certificado público X509 que tendrán que instalarse los usuarios en sus navegadores para poder confiar en nuestro servidor. Este certificado se genera así:

```
C:\AutenticacionCA>openssl req -new -x509 -days 1001 -key keys/ca.key -out
 certs/ca.cert
Loading 'screen' into random state - done
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:ES
State or Province Name (full name) [Some-State]:Zaragoza
Locality Name (eg, city) []:Zaragoza
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Universidad de Zarag
oza
Organizational Unit Name (eg, section) []:Universidad de Zaragoza
Common Name (e.g. server FQDN or YOUR name) []:localhost
Email Address []:vdebronik@gmail.com
```

D.4.1.2 Establecer SSL en un servidor

Para establecer una conexión segura en nuestro servidor, se necesita un certificado de servidor que sirva tanto para identificar al propio servidor como para cifrar la comunicación entre el servidor y los usuarios. Para ello, hay que generar claves y almacenarlas en un almacen de claves JKS (*Java Key Store*):

```
C:\AutenticacionCA>keytool -genkey -alias gestorServer -keypass vero2108 -store
pass vero2108 -keystore CA.keystore -keyalg RSA
What is your first and last name?
  [Unknown]:  Veronica Garcia
What is the name of your organizational unit?
  [Unknown]:  Universidad de Zaragoza
What is the name of your organization?
  [Unknown]:  Universidad de Zaragoza
What is the name of your City or Locality?
  [Unknown]:  Zaragoza
What is the name of your State or Province?
  [Unknown]:  Zaragoza
What is the two-letter country code for this unit?
  [Unknown]:  ES
Is CN=Veronica Garcia, OU=Universidad de Zaragoza, O=Universidad de Zaragoza, L
=Zaragoza, ST=Zaragoza, C=ES correct?
  [no]:  yes
```

El siguiente paso es generar un fichero CSR *Certificate Signing Request*) para que la CA emita un certificado asociado a las claves que se han generado

previamente. Para conseguirlo, se ejecuta:

```
C:\AutenticacionCA>keytool -certreq -alias gestorServer -keypass vero2108 -store
pass vero2108 -keystore CA.keystore -file request/gestorServer.csr
```

Ya solo queda enviar la petición a la CA y firmarla. Para ello, se copia el directorio openssl.cnf y se modifica lo siguiente:

- dir = . # *Where everything is kept*
- certs = $dir/certs # *Where the issued certs are kept*
- crl_dir = $dir/crl # *Where the issued crl are kept*
- database = $dir/database.txt # *database index file.*
- new_certs_dir = $dir/certs # *default place for new certs.*
- certificate = $dir/certs/ca.cert # *The CA certificate*
- serial = $dir/serial.txt # *The current serial number*
- crlnumber = $dir/crlNumber.txt # *the current crl number*
- crl = $dir/crl/crl.pem # *The current CRL*
- private_key = $dir/keys/ca.key # *The private key*

Una vez modificado, se crea el certificado para el servidor :

```
C:\AutenticacionCA>openssl ca -in request/gestorServer.csr -out certs/gestorSer
ver.cert -config openssl.cfg -policy policy_anything
Using configuration from openssl.cfg
Loading 'screen' into random state - done
Check that the request matches the signature
Signature ok
```

El certificado generado de esta manera no se puede importar directamente, por lo que hay que comprimirlo de la siguiente manera:

```
C:\AutenticacionCA>openssl x509 -in certs/gestorServer.cert -outform PEM -
out gestorServerPEM.cert
```

Después, hay que importar el certificado público de la CA al almacén de claves marcándolo de confianza:

```
C:\AutenticacionCA>keytool -import -alias gestorCa -keypass vero2108 -file cert
s/ca.cert -storepass vero2108 -keystore CA.keystore
```

Por último, se importa el certificado de servidor:

```
C:\AutenticacionCA>keytool -import -alias gestorServer -keypass vero2108 -
file gestorServerPEM.cert -storepass vero2108 -keystore CA.keystore
Certificate reply was installed in keystore
```

D.4.2 Añadir almacén de claves al servidor

Este paso sí hay que realizarlo. En los pasos anteriores, se ha generado un almacén de claves llamado, *CA.keystore*. Este fichero se encuentra en la carpeta *AutenticacionCA* del CD y hay que copiarlo en la carpeta *conf* de Tomcat. Además de copiar el fichero, hay que modificar *server.xml* de esa misma carpeta comentando la conexión al puerto 8080 si está siendo usado por XAMPP:

```
!--
   <Connector executor="tomcatThreadPool"
              port="8080" protocol="HTTP/1.1"
              connectionTimeout="20000"
              redirectPort="8443" />
 -->
```

Y descomentar la conexión por el puerto 8443 indicando la ruta del fichero y su contraseña.

```
!--
Define a SSL HTTP/1.1 Connector on port 8443
        This connector uses the JSSE configuration, when using APR, the
        connector should be using the OpenSSL style configuration
        described in the APR documentation
 -->

 <Connector port="8443" protocol="HTTP/1.1"
```

```
    SSLEnabled="true"   maxThreads="150"
    scheme="https" secure="true"
    clientAuth="false" sslProtocol="TLS"
    keystoreFile="./conf/CA.keystore"
    keystorePass="vero2108" />
```

D.4.2.1 Añadir certificado público al navegador

Este proceso puede ser diferente dependiendo del navegador. En Chrome, hay que agregar el certificado público *ca.cert*(que se encuentra en la carpeta *AutenticacionCA/certs*) para que no aparezcan alertas de seguridad de la siguiente forma:

Icono Herramientas → Opciones → Avanzada → HTTPS/SSL → Administrar certificados → Entidades de certificación Intermedias → Importar

También hay que agregarlo a la pestaña *Entidades de certificación raíz de confianza*.

D.4.3 Desplegar la aplicación en Tomcat

Este es el último paso que queda por realizar. Para agregar la aplicación al servidor Tomcat, hay que copiar la carpeta *gestor* disponible en el CD a la carpeta *webapps* dentro del servidor.

Para lanzar el servidor, hay que hacerlo sobre línea de comandos:

Menú Inicio → Ejecutar → cmd

Nos colocamos sobre el directorio *bin* de Tomcat ejecutando

```
cd %Ruta Directorio Tomcat%/bin
```

Una vez en el directorio *bin* de Tomcat, lanzamos el servidor con el comando *startup*. Tras unos segundos, el servidor estará funcionando y se podrá acceder a la aplicación a través del navegador, mediante la dirección:

https://localhost:8443/gestor

Anexo E

Guía de Usuario

E.1 Estructura de las páginas

La estructura de las páginas es muy sencilla manteniendo concordancia entre las distintas vistas y destacando en la parte superior las principales secciones que tiene:

- **Inicio.** Muestra la página de bienvenida, donde se pueden ver los últimos eventos recibidos y cambiar la configuración personal.
- **Dispositivos.** En esta pestaña, se podrá acceder a la información relacionada con los dispositivos médicos y sus dispositivos concentradores de datos.
- **Alarmas.** Permite configurar las alarmas y visualizar los datos relacionados con ellas: alarmas configuradas, eventos asociados, tabla de logs o eventos recibidos.
- **Usuarios.** Esta pestaña está solo disponible para los usuarios de tipo administrador. Permite buscar, agregar o eliminar usuarios.

Además, de estas pestañas, a lo largo de todas las vistas, en la parte superior derecha de la página aparecen las siguientes opciones:

- **Ayuda.** Al pulsarse, muestra un popup con información sobre el contenido de la página y las acciones que se pueden realizar en ella.

- **In English.** Permite cambiar de idioma.
- **Cerrar Sesión.** Finaliza la sesión iniciada por el usuario.

E.2 Página de Inicio

La página de inicio es la mostrada en la figura E.1. Desde esta sección, se puede ver los últimos eventos recibidos o cambiar opciones de la configuración personal del usuario que tiene iniciada la sesión. Los últimos eventos mostrados en esta página son aquellos que se han recibido mientras el usuario ha estado desconectado o no se han visualizado todavía. Las opciones de configuración personal son:

- **Visualizar/Editar datos personales.** Está opción permite modificar al usuario los datos personales que tiene registrados en el sistema: nombre de usuario, contraseña, nombre completo, teléfono y email.
- **Cambiar contraseña.** Con esta opción se puede cambiar la contraseña de acceso al sistema.

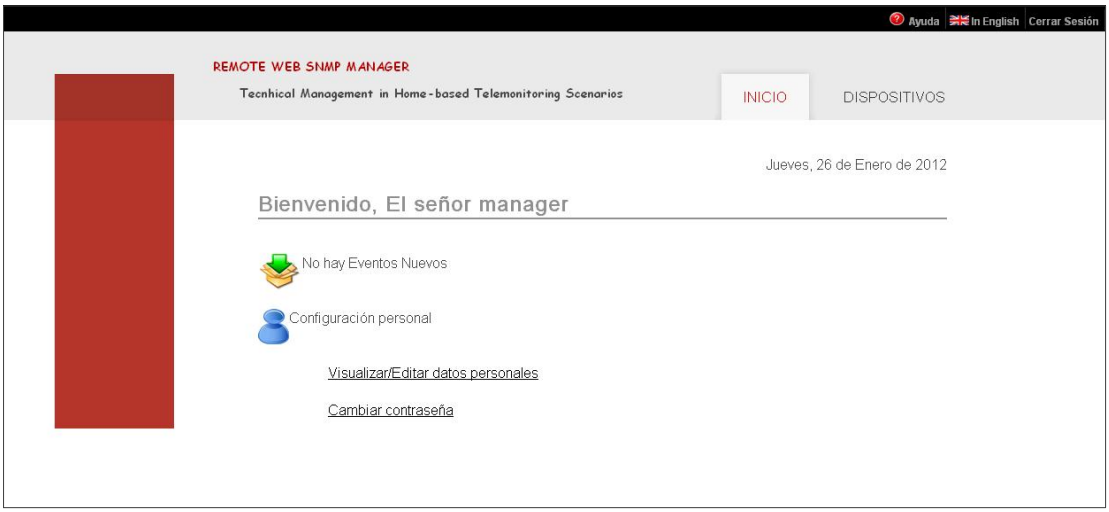


Figura E.1: Página Inicio del sistema.

E.3 Gestión Dispositivos

E.3.1 Datos asociados al dispositivo concentrador.

Desde la pestaña *Dispositivos* se pueden ver todos los hogares que se tienen asociados a su usuario. Para acceder a la información de control de algún dispositivo concentrador o a los dispositivos médicos que éste controla, hay que pulsar encima de la imagen correspondiente. Los datos del dispositivo concentrador se representan en la figura E.2.

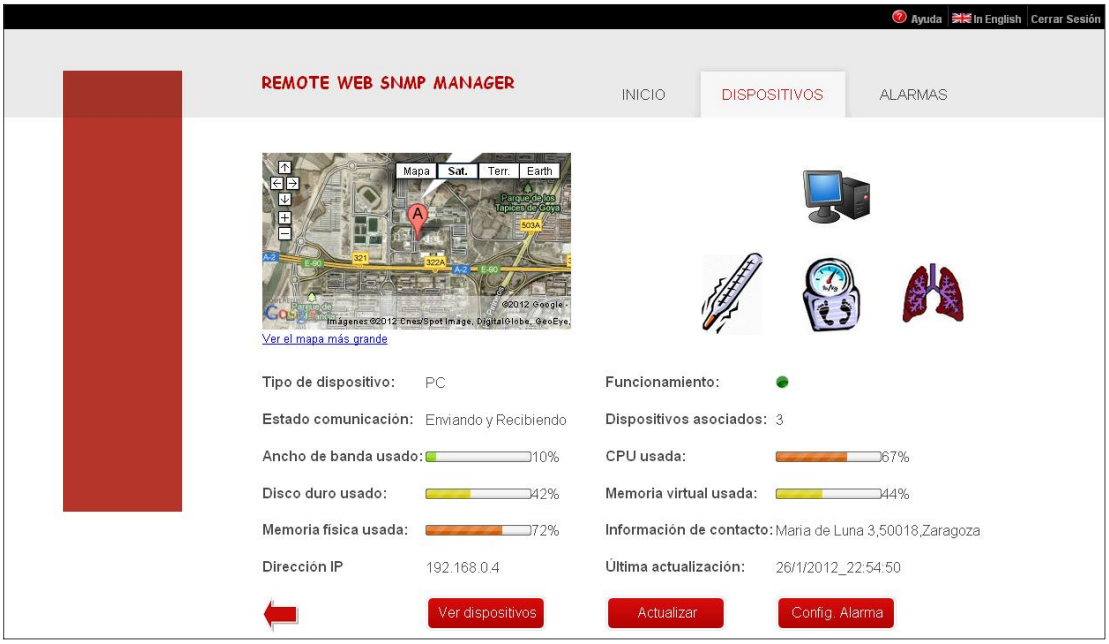


Figura E.2: Recursos del dispositivo concentrador de datos.

Desde esta página, se puede ver el lugar desde el cual se están haciendo las mediciones, el tipo de dispositivo que está recogiendo los datos (cuyos datos técnicos se están visualizando) y los tipos de dispositivos médicos que tiene asociados. Desde aquí, se pueden ir a varios sitios dependiendo del botón/imagen que se pulse, además de volver a la página anterior:

- **Imágenes de los dispostivos.** Al pulsar alguna de las imágenes de los dispositivos que se encuentran en la parte superior de la página, se mostrarán los últimos datos técnicos recogidos (tipo de batería, porcentaje de batería, estado, etc) sobre ese dispositivo.

- **Ver Dispositivos.** A través de este enlace, se obtiene la información más relevante sobre los últimos datos técnicos recibidos de todos los dispositivos asociados. Esta información es: horas de batería, errores en la medida y estado. Se muestra en la figura E.3.

- **Actualizar.** Al pulsar este botón se enviará una petición para actualizar los datos técnicos y el estado de todos los dispositivos asociados además de actualizar los datos del dispositivo concentrador de datos.

- **Config.Alarma.** A través de este botón, se accede a otra página para configurar alarmas asociadas a alguno de los recursos del dispositivo concentrador de datos.

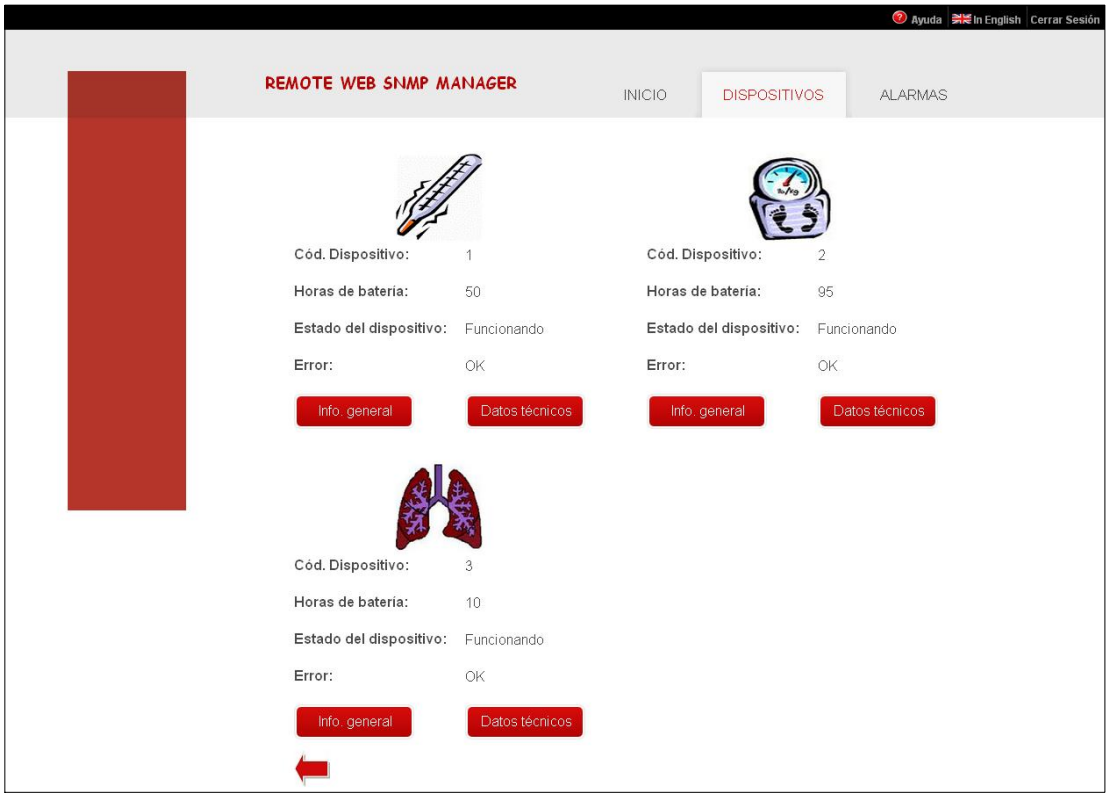


Figura E.3: Datos técnicos relevantes de todos los dispositivos.

E.3.2 Datos técnicos de los dispositivos médicos

El acceso más directo para visualizar los datos técnicos asociados a un dispositivo es a través de las imágenes de los dispositivos que hay en la página anterior (figura E.2) pudiendo también acceder a los datos técnicos de un dispositivo médico en concreto después de haber visualizado la información más relevante sobre los últimos datos técnicos (botón *Ver Dispositivos*, figura E.3). Estos datos se muestran en la figura E.4.

Figura E.4: Datos técnicos de los dispositivos médicos.

Desde esta página se pueden realizar varias acciones:

- **Columna *Hist.* + Botón *GO!*.** Esta columna se utiliza para visualizar el histórico de los datos técnicos asociados al dispositivo actual en forma de tabla. Se pueden seleccionan tantas columnas como se desee mostrándose solo los datos seleccionados.

- **Columna *Graf.* + Botón *GO!*.** Marcando está columna y pulsando posteriormente el botón *GO!*, se visualiza el histórico de los datos técnicos de forma gráfica. Al igual que en la columna anterior, se pueden marcar tantas columnas como se desee mostrándose solo los datos seleccionados.

- **Columna *Alarm.* + Botón *GO!*.** Esta columna solo permite seleccionar un recurso. Una vez seleccionado, al pulsar el botón *GO!*, lleva a otra página para configurar una alarma asociada al dispositivo y recurso seleccionado.
- **Info. General.** Al pulsar este botón, se mostrarán los datos estáticos asociados al dispositivo que se está visualizando. Esto son datos son: fabricante, código de fabricante, protocolo de comunicación, etc.

E.3.3 Información General dispositivos médicos

A esta información se accede mediante el botón *Info. General* que se encuentra en las páginas que contienen los datos técnicos. La información que se presenta en la figura E.5 son los datos estáticos del dispositivo seleccionado. Esta página permite visualizar los últimos datos recibidos de ese dispositivo pulsando el botón *Datos Técnicos*.

Figura E.5: Información General de los dispositivos médicos.

E.4 Alarmas

Esta pestaña se muestra en la figura E.6 que contiene cuatro opciones:

- **Últimos eventos.** A través de este enlace se pueden visualizar los últimos eventos recibidos.
- **Visualizar tabla de logs.** Al pulsar sobre esta imagen se muestra el registro de logs.
- **Visualizar alarmas configuradas.** En este enlace se encuentran todos las alarmas configuradas en el hogar que se está mirando.
- **Configurar Alarma.** Al pulsar esta imagen aparece otra página para la configuración de alarmas.

Figura E.6: Vista pestaña Alarmas.

E.4.1 Últimos eventos recibidos

Para visualizar los últimos eventos recibidos se puede acceder desde la pantalla de *Inicio*, desde la pestaña de *Alarmas* pulsando sobre el el enlace *Últimos eventos* o pulsando sobre el icono de advertencia que aparece cuando llegan eventos nuevos. La información que se puede ver en esa página es la mostra en la figura E.7:

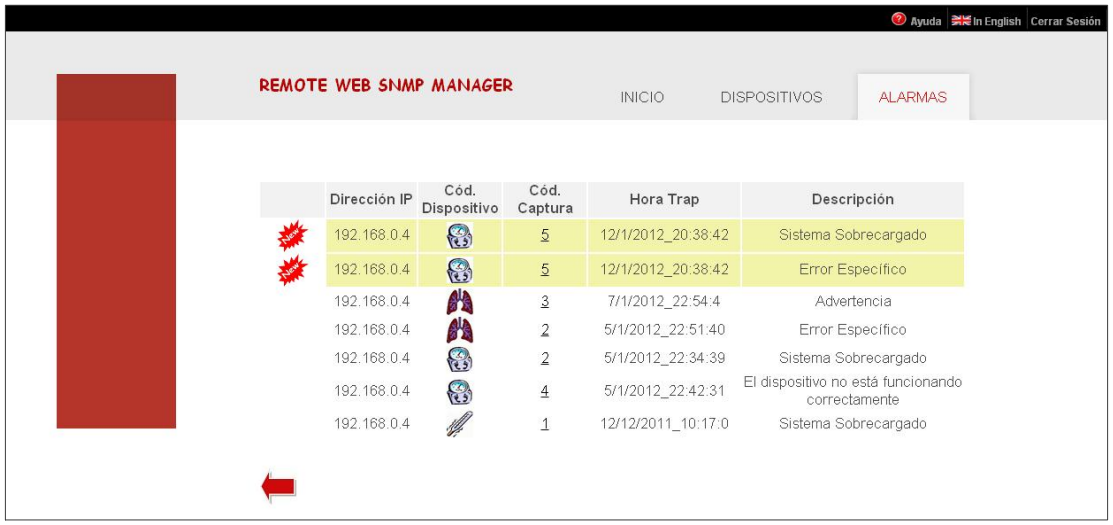


Figura E.7: Últimos eventos recibidos.

- **Imagen *New*.** Indica que el evento de esa fila no ha sido visto anteriormente. Además de esta imagen, las filas que contengan información sobre nuevos eventos, están sombreadas.

- **Dirección IP.** Muestra la dirección IP desde la que se ha recibido el evento.

- **Cód.Dispositivo.** Esta columna permite visualizar el tipo de dispositivo que ha enviado el mensaje. Pulsando sobre la imagen, se accede a la información general del dispositivo.

- **Cód.Captura.** Indica el código correspondiente a la captura que ha activado la alarma. Pulsando sobre el número, se puede ver los datos técnicos correspondientes a dicha captura.

- **Hora Trap.** Muestra la fecha en la que fue recibido el mensaje.

- **Descripción.** Describe el tipo de mensaje recibido.

E.4.1.1 Aviso nuevos eventos

Cuando un icono triángular parpadeante como el de la figura E.8 aparezca en la parte superior derecha de la página, significa que un evento nuevo ha llegado.

Para poder visualizar los datos de los eventos nuevos en detalle, basta con pulsar sobre el icono.

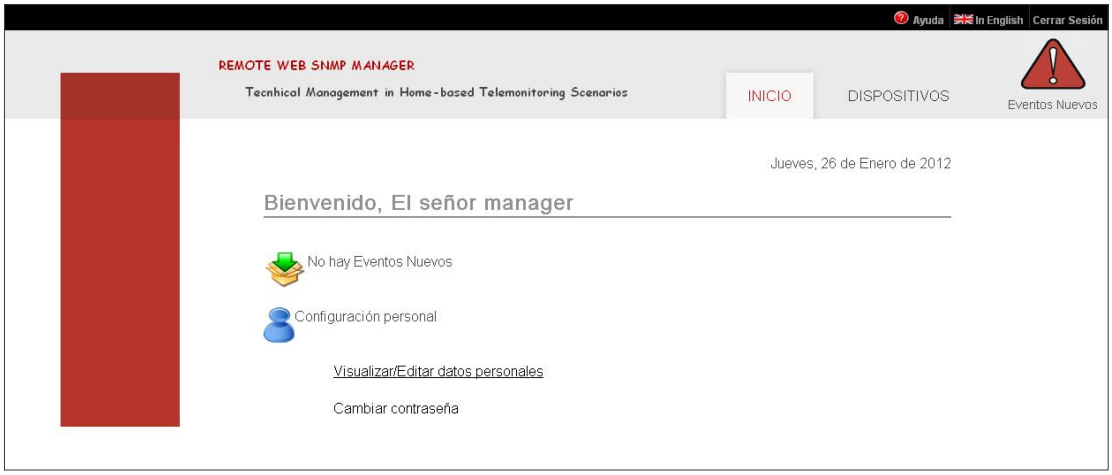


Figura E.8: Aviso llegada de eventos nuevos.

E.4.2 Visualizar tabla de logs

Para visualizar la tabla de logs, basta con pulsar sobre el icono correspondiente en la pestaña de *Alarmas*. La página que se verá es la mostrada en la figura E.9 donde se tiene:

Figura E.9: Logs registrados.

- **Cód.Evento.** Este código representa el evento que ha creado el registro en la tabla de logs. Al pulsar sobre el código, se muestra la información detallada sobre dicho evento.

- **Cód.Alarma.** Indica el código de la alarma que ha sido activada y cuyo evento asociado ha creado el registro. Al pulsar sobre el código, se accede a la información detallada sobre dicha alarma.

- **Descripción.** Comenta el motivo por el que se ha activado la alarma. Puede ser: valor inválido, alarma activada por umbral superior o inferior y alarma activada por valor.

E.4.3 Visualizar alarmas configuradas

Para acceder a esta sección, hay que pulsar sobre la imagen correspondiente en la pestaña de *Alarmas*. Una vez se haya pulsado la imagen, se accederá a otra página donde se dividen las alarmas configuradas en tres grupos según los eventos que tengan asociados serán:

1. **Alarmas por valor:** Estas alarmas son aquellas que se activan cuando el recurso asociado a la alarma alcanza un determinado valor.

2. **Alarmas activada por umbral superior:** Las alarmas que se ven al pulsar este icono son aquellas que son activadas cuando el recurso asociado supera un determinado umbral.

3. **Alarmas activadas por umbral inferior:** Estas alarmas son aquellas que se activan cuando el recurso asociado supera un determinado umbral inferiormente.

Para visualizar el tipo de alarmas deseado, se pulsa sobre el icono correspondiente y se obtiene una página como la de la figura E.10. En ella, los datos representados son:

Figura E.10: Visualizar alarmas configuradas.

- **Dispositivo.** En esta columna se muestra una imagen del tipo de dispositivo al que se ha configurado la alarma. Al pulsar sobre la imagen, se muestra la información general del mismo.
- **Cód.Alarma.** Indica el código de la alarma configurada. Al pulsar sobre él, se obtiene la información detallada de dicha alarma.
- **Cód.Evento.** Es el código del evento asociado a dicha alarma. Al pulsar sobre él, se obtiene los detalles del mismo.
- **Recurso.** Muestra el recurso al que se ha asociado la alarma.
- **Valor.** Indica el valor exacto o los umbrales para los que será activada la alarma.

E.4.4 Configurar una Alarma

Se puede acceder a esta página desde la página de los datos del dispositivos concentrador, desde los datos técnicos de cualquiera de los dispositivos asociados o desde la pestaña *Alarmas* pulsando sobre el icono correspondiente. La vista de esta página se muestra en la figura E.11. Para configurar una alarma correctamente, hay que indicar el tipo de dispositvo, el recurso y asociarle, al menos, un evento. Los detalles de la página se explican a continuación.

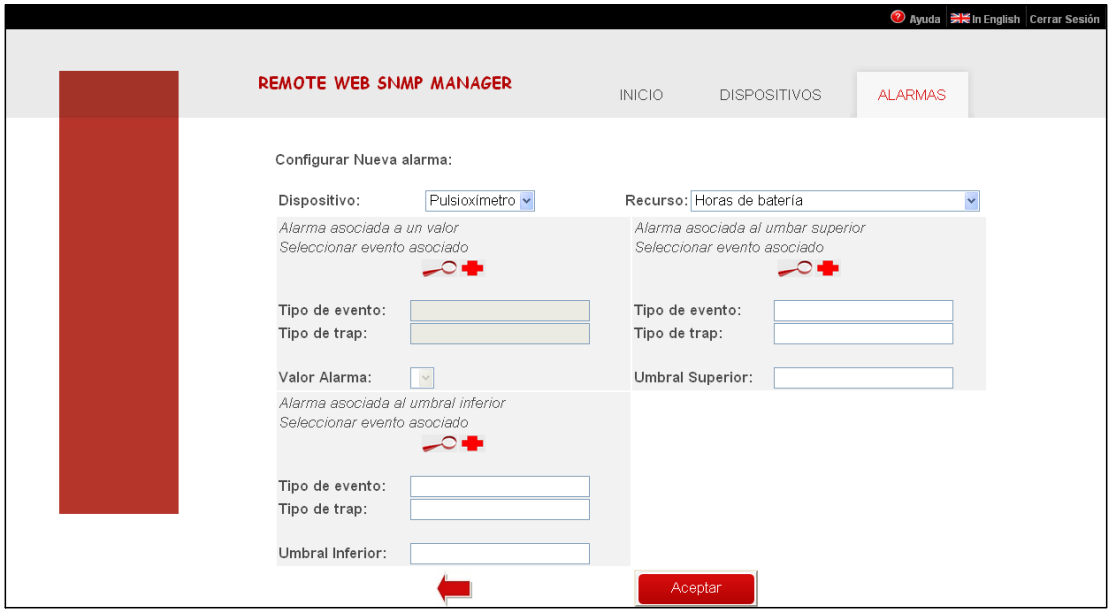


Figura E.11: Configurar Alarma.

- **Dispositivo.** En este desplegable, se indica el dispositivo al que va a ir asociado la alarma.

- **Recurso.** Indica el recurso cuyo valor hará activar la alarma.

- **Alarma asociada a un valor.** Este rectángulo gris se rellenará si se desea configurar una alarma asociada por valor. En ese caso, se han de rellenar todos estos parámetros:

  - **Tipo de evento:** Se rellena con el tipo de evento que se desea realizar cuando la alarma tome determinado valor: mandar un aviso, añadir una nueva entrada en la tabla de logs o ambos.

  - **Tipo de trap:** Se rellena con el tipo de aviso que se desea mandar, en caso de que ese sea el evento que se quiere enviar.

  - **Valor:** Hay que seleccionar el valor que activará la alarma. Dependiendo del recurso elegido, las opciones mostradas variarán.

Para rellenar estos el tipo de evento y el tipo de trap, se utilizan los iconos que aparecen justo encima. La imagen de la lupa mostrará los eventos ya

existentes mientras que la imagen del signo más permitirá configurar un nuevo evento.

- **Alarma asociada al umbral superior.** Para configurar una alarma asociado a un umbral superior, este es el recuadro gris que hay que rellenar. Los parámetros a rellenar y la manera de hacerlo son los explicados para el caso de una alarma asociada a un valor a excepción de que, en este caso, el valor será numérico.

- **Alarma asociada al umbral inferior.** Esta parte de la página se rellena si se desea configurar una alarma asociada al umbral inferior. Al igual que en los casos anteriores, hay que rellenar el tipo de evento, tipo de trap y el valor para poder configurar la alarma correctamente.

Una vez introducidos todos los valores deseados con los que configurar la alarma, se pulsa el botón *Aceptar* y se procederá a crear la alarma.

E.5 Usuarios

Esta pestaña solo es visible para los usuarios de tipo administrador. En ella, se permite:

- **Añadir nuevo usuario.** Pulsando sobre esta imagen, aparece un formulario con los datos a introducir para poder añadir un nuevo usuario: nombre de usuario, contraseña, tipo de usuario, ip, email y teléfono.

- **Eliminar usuario.** Este enlace muestra todos los usuarios que hay en el sistema y permite eliminarlos.

- **Ver usuarios.** Desde este icono se puede acceder a visualizar todos los usuarios registrados en el sistema.

Anexo F

Cuestionarios Evaluación

En este anexo, se adjuntan los dos cuestionarios que se han realizado para la evaluación del sistema:

- Cuestionario Evaluación Guiada
- Cuestionario Evaluación General

FASE I: EVALUACIÓN GUIADA

CUESTIONARIO DE EVALUACIÓN DEL SERVICIO DE TELEMONITORIZACIÓN DOMICILIARA

Fecha: _____ Edad: _____
Perfil: _____ Sexo: □ M □ H

Evaluación general del Sistema

1. La aplicación que va a utilizar permite la gestión técnica remota de los dispositivos médicos que utiliza un paciente en su casa. Tras introducir el usuario y contraseña, usted accede a la página de inicio. ¿Sabría usted acceder a gestionar los dispositivos?

□ Sí □ No Tiempo:

2. Una vez que conoce todos los CEs asociados y accede a la gestión individual de cada uno, ¿le parece que la información acerca del CE se presenta de forma clara?

□ Sí □ No Tiempo:

3. ¿Sabría decir cuántos dispositivos tiene associados y sus tipos este CE?

□ Sí □ No Tiempo:

4. Ahora que ya conoce los dispositivos asociados, ¿sabría acceder a la última información recibida sobre alguno de ellos?

□ Sí □ No Tiempo:

5. Una vez que ya conoce los últimos datos técnicos recibidos, ¿sabría visualizar el histórico de los mismos de forma gráfica?

□ Sí □ No Tiempo:

6. ¿Sabría configurar una alarma asociada al recurso deseado?

□ Sí □ No Tiempo:

7. Dejando a parte los datos técnicos de los dispositivos, ¿sabría ver las alarmas configuradas?

□ Sí □ No Tiempo:

8. ¿Sabría visualizar los eventos asociados a alguna de las alarmas?

□ Sí □ No Tiempo:

9. ¿Sabría acceder a los logs registrados?

□ Sí □ No Tiempo:

10. ¿Sabría ver los eventos recibidos y distinguir cuáles son los últimos?

□ Sí □ No Tiempo:

FASE II: EVALUACIÓN GENERAL

CUESTIONARIO DE EVALUACIÓN DEL SERVICIO DE TELEMONITORIZACIÓN DOMICILIARA

Fecha: _____ Edad: _____
Perfil: _____ Sexo: □ M □ H

Evaluación general del Sistema

1. ¿El sistema es fácil de usar?

□ Sí □ No

2. ¿Es intuitivo?

□ Sí □ No

3. ¿Se están produciendo fallos en el funcionamiento del sistema?

□ Sí □ No

4. ¿Con qué frecuencia?

_____

_____

5. ¿En cuántas ocasiones no ha sido posible visualizar los datos deseados?

_____

_____

6. ¿La información proporcionada es suficiente para el seguimiento del correcto funcionamiento de los dispositivos?

□ Sí □ No

7. ¿Cuál es el grado de satisfacción en la configuración de alarmas?

□ Muy Satisfecho

□ Satisfecho

□ Normal

□ Insatisfecho

□ Muy insatisfecho

8. ¿Cree que el aviso de llegada de nuevos eventos es suficiente?

□ Sí □ No

9. ¿Qué otros avisos añadiría?

_____

_____

_____

10. ¿En qué ocasiones considera que es útil el sistema?

_____

_____

_____

11. Enumere las principales dificultades encotradas.

_____

_____

_____

12. ¿Aconsejaría la adopción de un sistema de estas características en un entorno de salud de forma permanente?

□ Sí □ No

13. ¿Por qué?

_____

_____

_____

14. ¿Cuál es el grado de satisfacción general con el uso del sistema?

□ Muy Satisfecho

□ Satisfecho

□ Normal

□ Insatisfecho

□ Muy insatisfecho

15. Sugerencias:

_____

_____

_____

Anexo G

Diagrama de Gantt

A continuación se presentan las tareas que aparecen en el diagrama de Gantt para este Proyecto Fin de Carrera mostrado en la figura G.1:

1. Documentación de tecnología Web.
2. Documentación arquitectura SNMP.
3. Documentación sobre el estado del arte.
4. Estudio MIB MD.
5. Creación estructura Web.
6. Diseño interfaz web.
7. Implementación del bloque de comunicaciones SNMP.
8. Integración de ambas herramientas y desarrollo de todas las vistas del sistema.
9. Internacionalización de la aplicación.
10. Integración herramientas para visualización gráfica de datos.
11. Implementación de herramientas para captura de mensajes asíncronos.
12. Evaluación del sistema.

13. Redacción de la memoria.

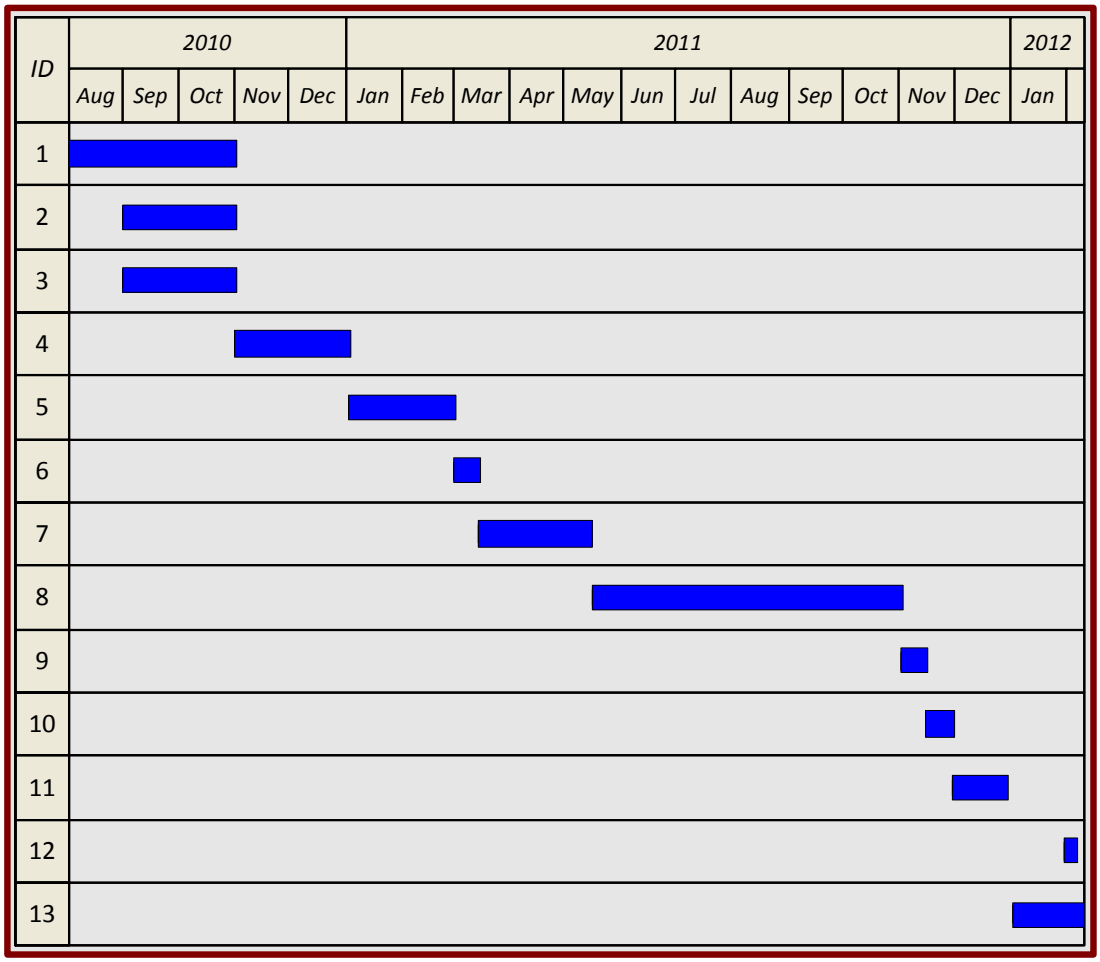


Figura G.1: Diagrama de Gantt del Proyecto Fin de Carrera