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.

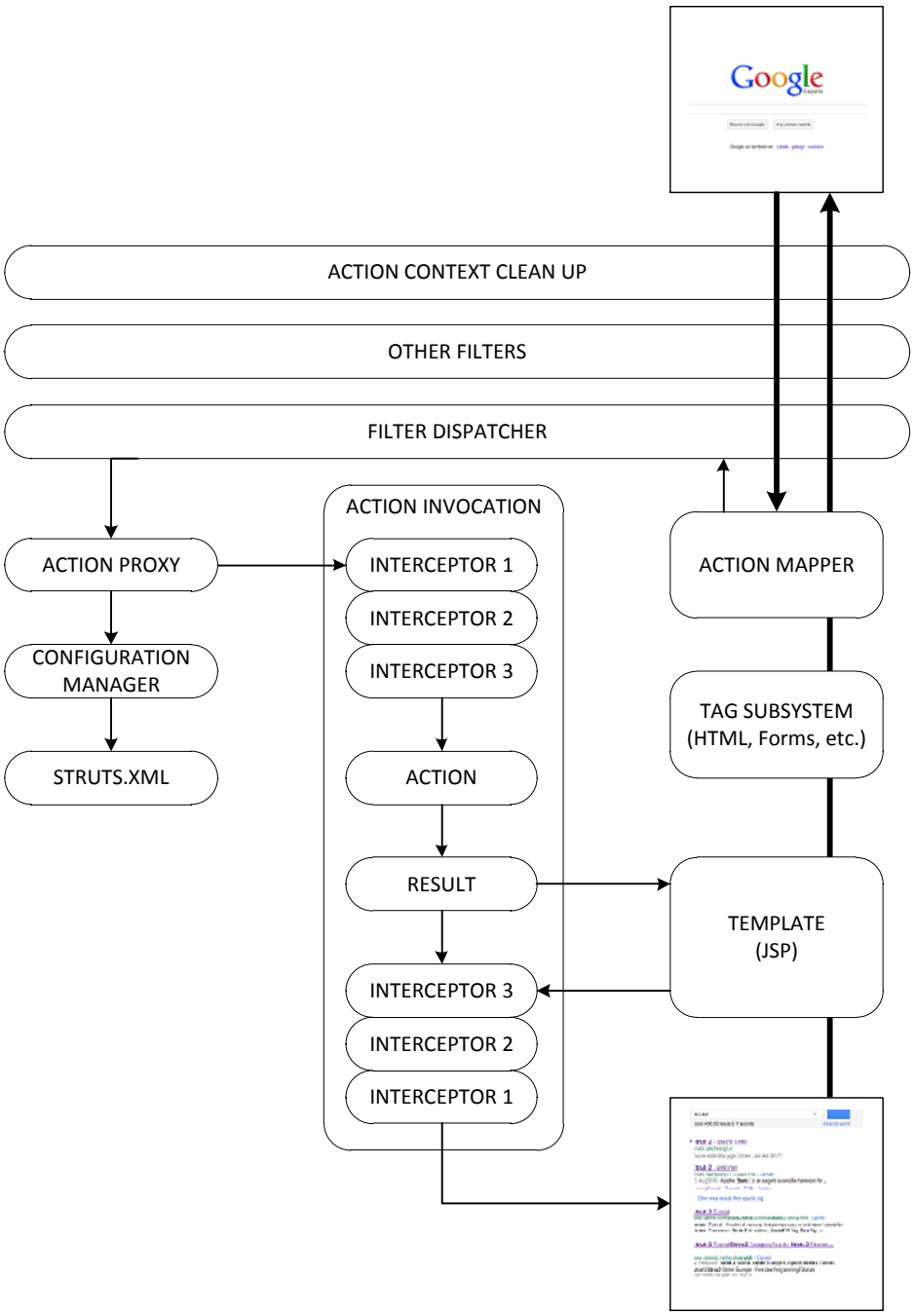

Figura A.1: Arquitectura Framework Struts2.

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:

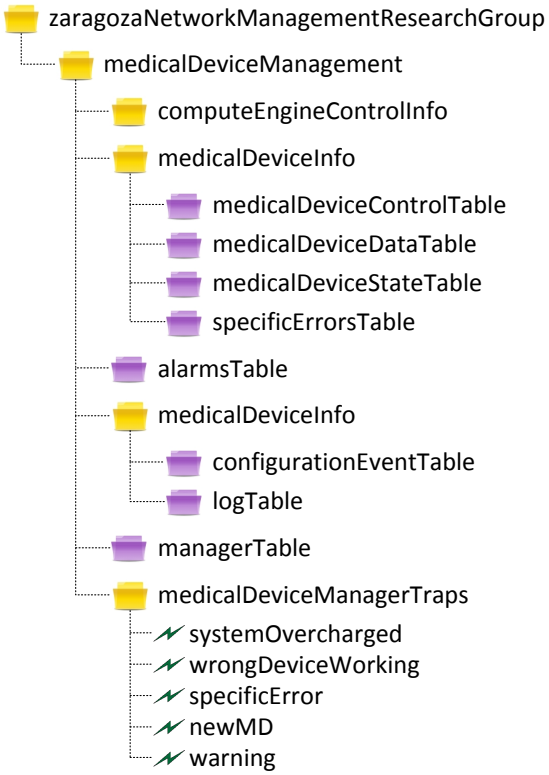

Figura B.1: Estructura MIB MD.

- **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

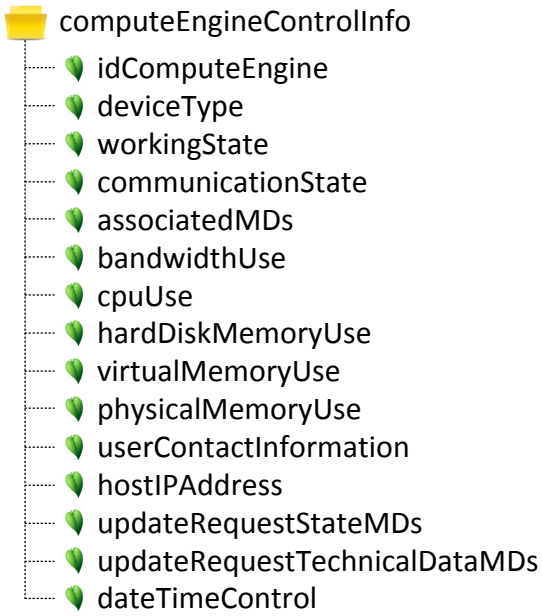

Figura B.2: Estructura Grupo ComputeEngineControlInfo MIB MD.

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

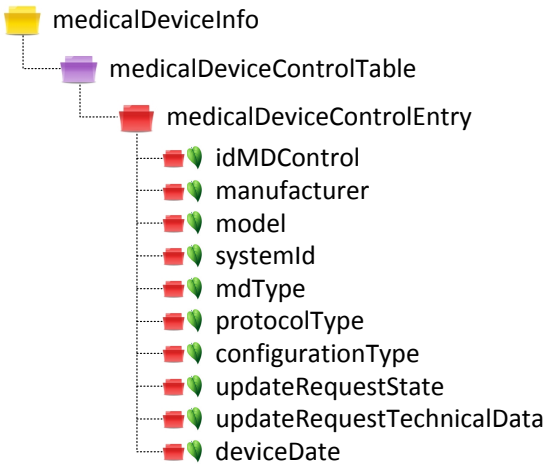

Figura B.3: Estructura MedicalDeviceControlTable MIB MD.

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

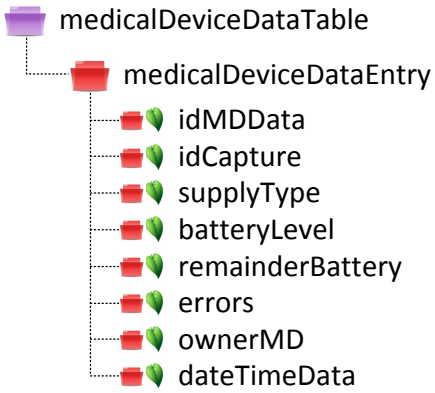

Figura B.4: Estructura MedicalDeviceDataTable MIB MD.

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

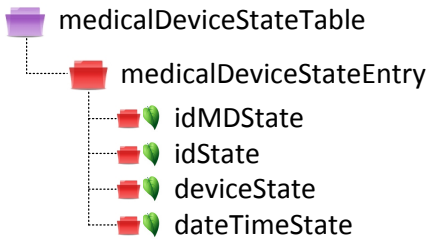

Figura B.5: Estructura MedicalDeviceStateTable MIB MD.

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

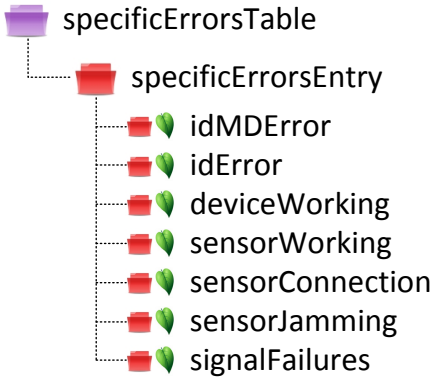

Figura B.6: Estructura SpecificErrorsTable MIB MD.

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

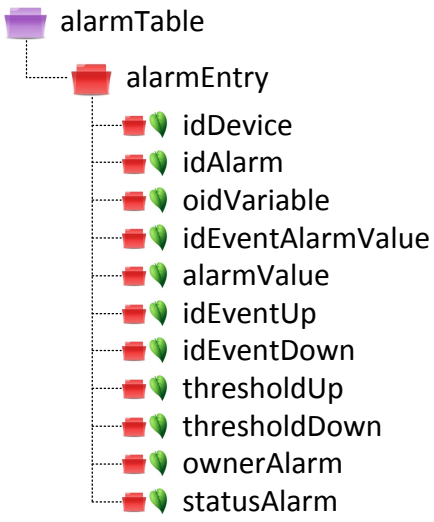

Figura B.7: Estructura Grupo AlarmTable MIB MD.

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

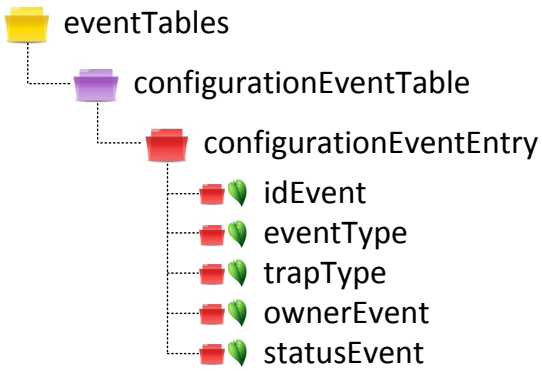

Figura B.8: Estructura ConfigurationEventTable MIB MD.

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

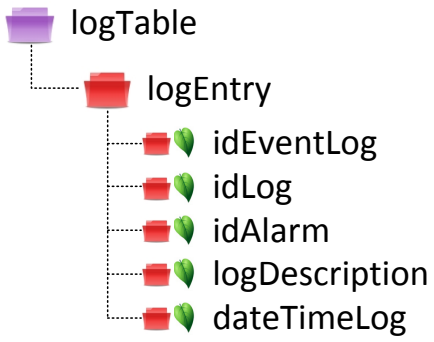

Figura B.9: Estructura LogTable MIB MD.

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

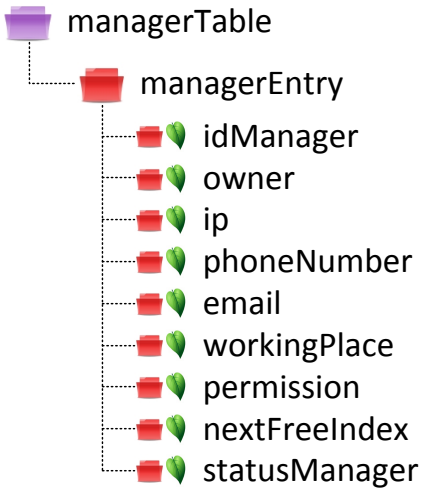

Figura B.10: Estructura Grupo ManagerTable MIB MD.

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

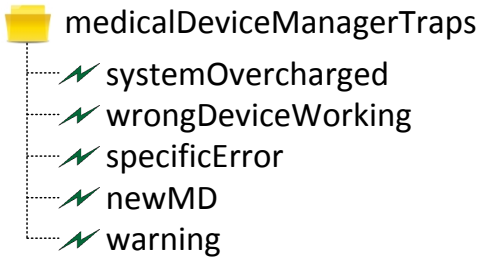

Figura B.11: Definición de Traps MIB MD.

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.

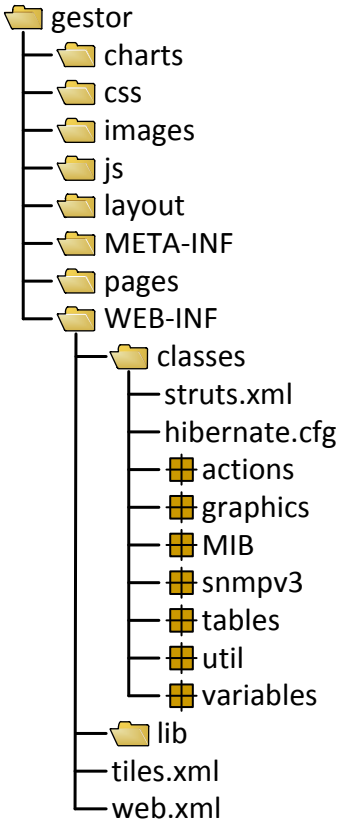

Figura C.1: Estructura de Carpetas de la Aplicación.

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.

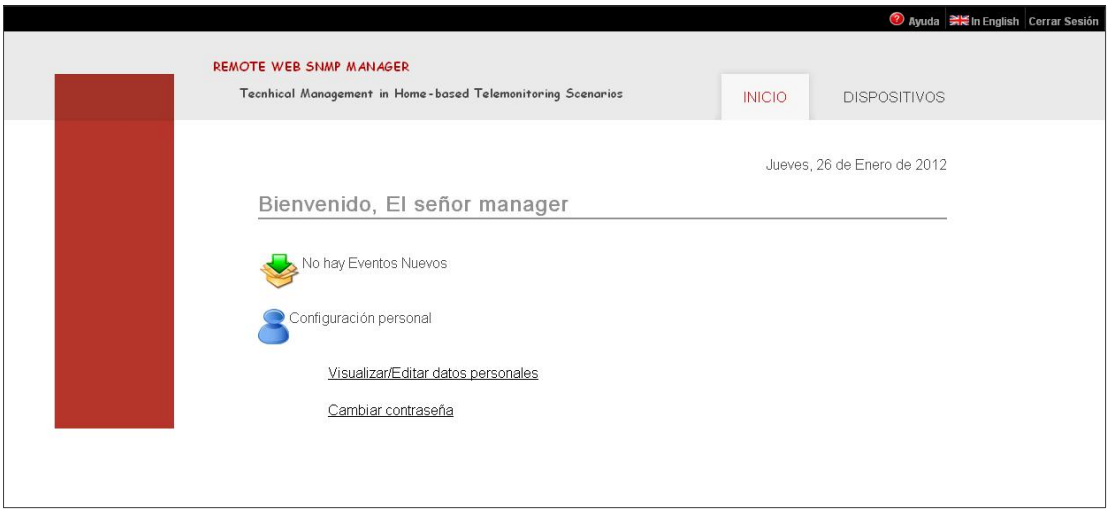

Figura E.1: Página Inicio del sistema.

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.

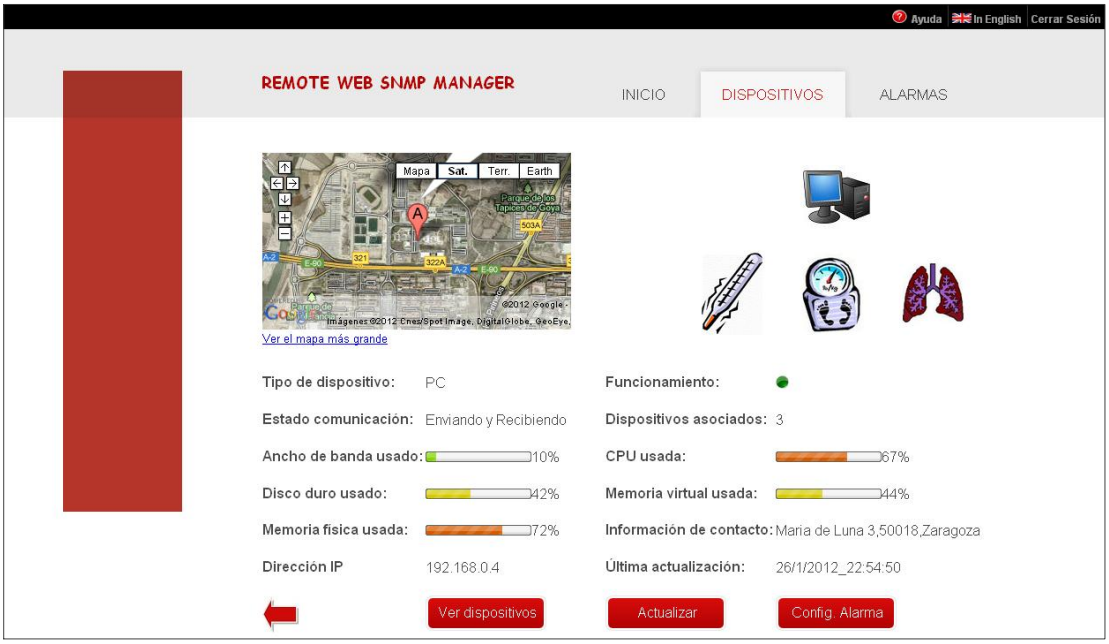

Figura E.2: Recursos del dispositivo concentrador de datos.

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.

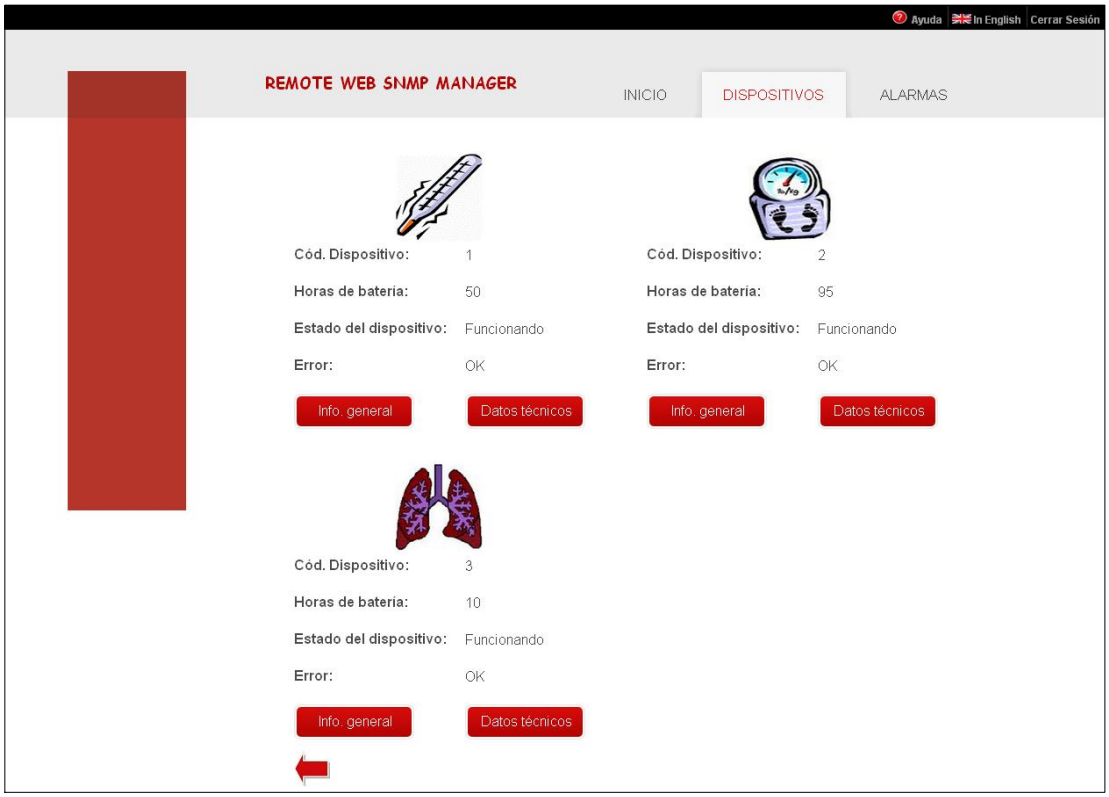

Figura E.3: Datos técnicos relevantes de todos los dispositivos.

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:

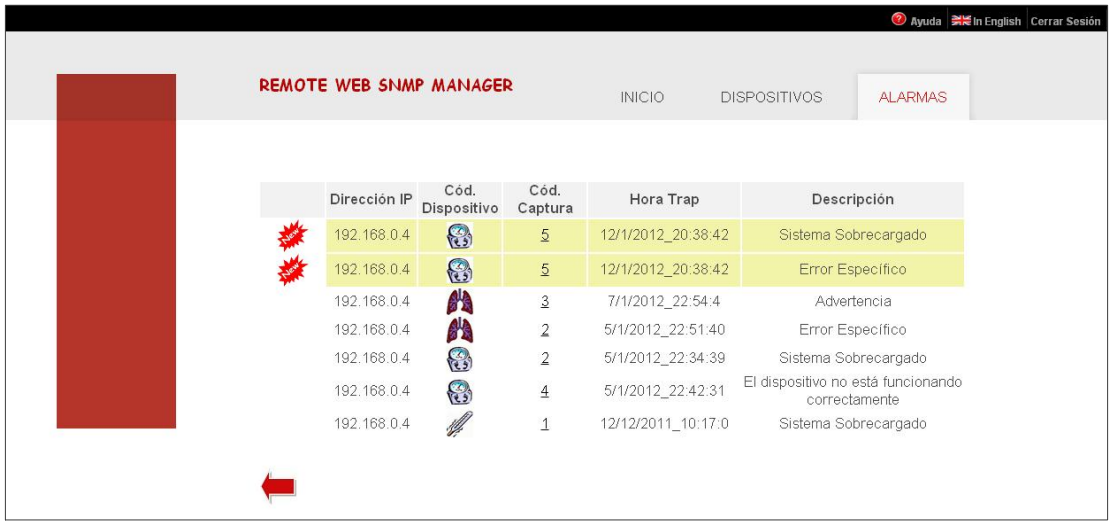

Figura E.7: Últimos eventos recibidos.

- **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.

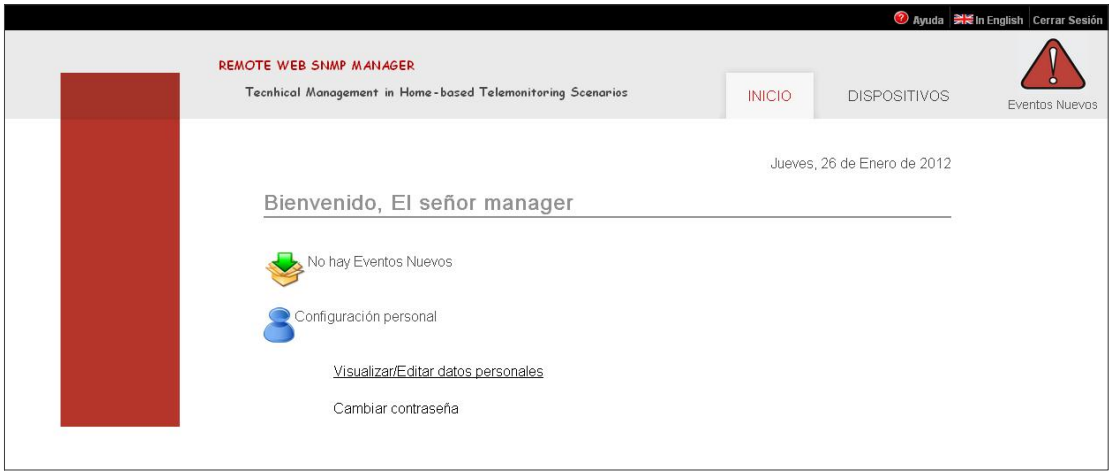

Figura E.8: Aviso llegada de eventos nuevos.

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.

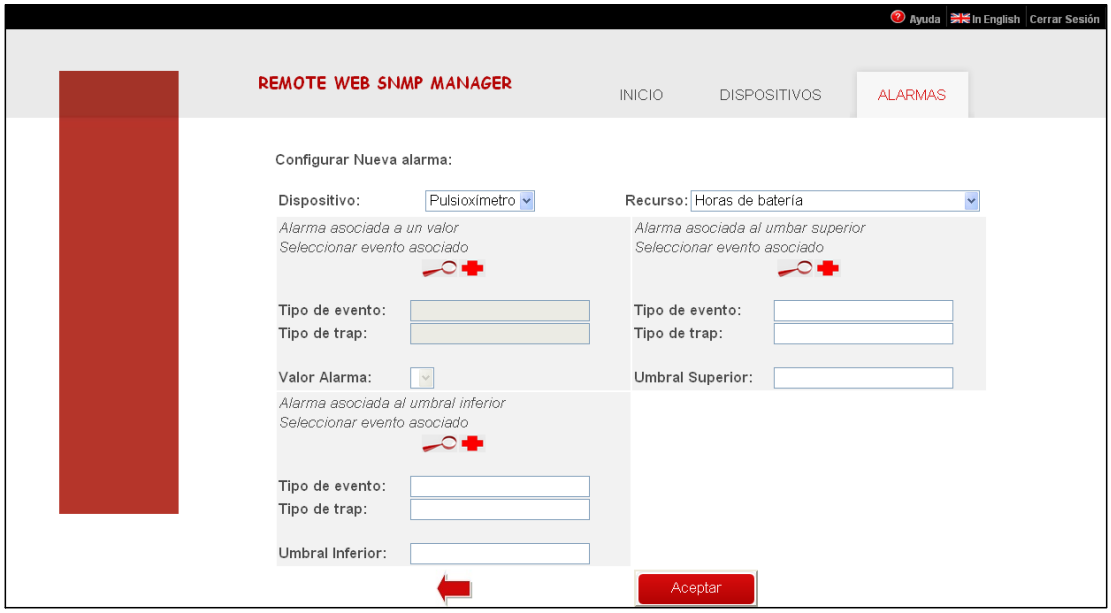

Figura E.11: Configurar Alarma.

- **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.

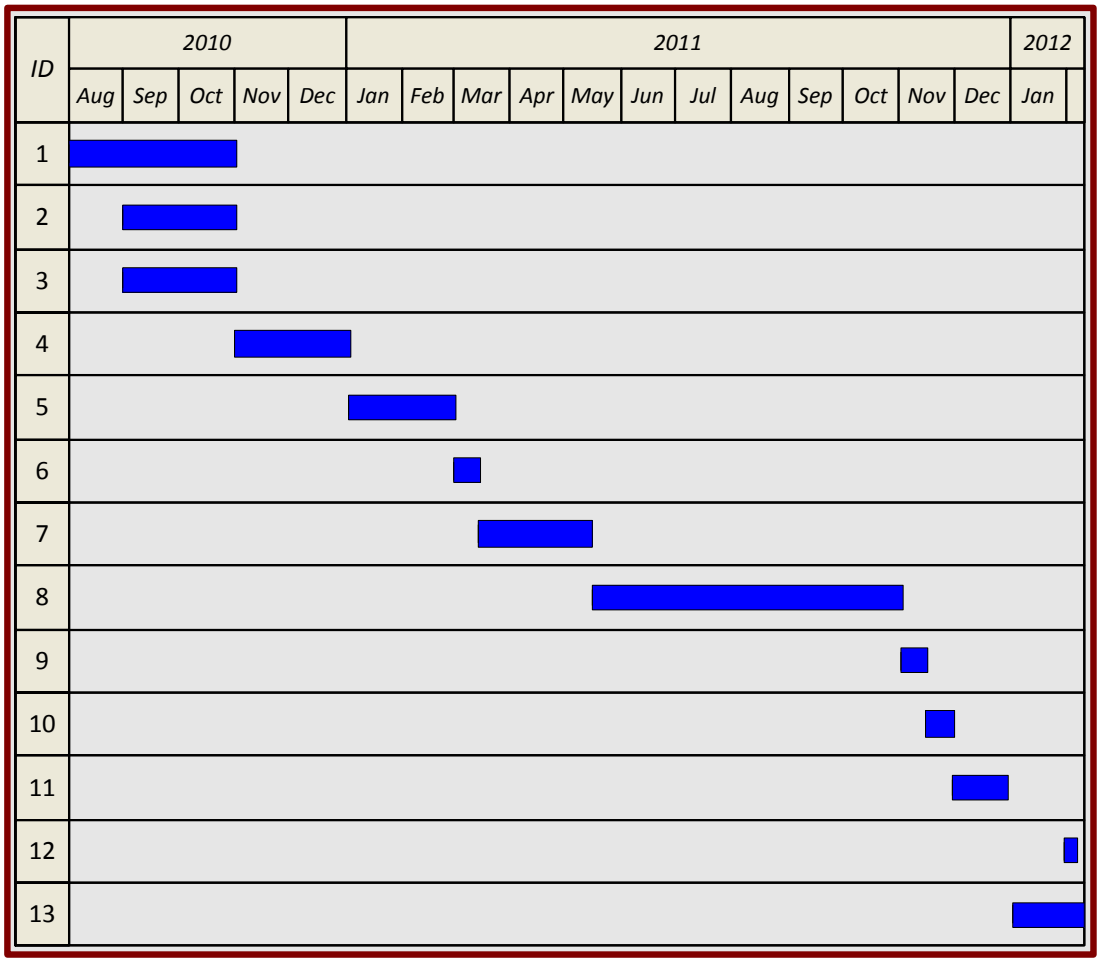

Figura G.1: Diagrama de Gantt del Proyecto Fin de Carrera