Grado en Ingeniería Informática

TRABAJO FINAL DE GRADO

Aplicación web para la monitorización en tiempo real de servicios y dispositivos de red

Autor:
Salvador Martí Solsona

Supervisor:
Rafael Forcada
Tutor académico:
Rafael Berlanga

Fecha de lectura: 15 de Julio de 2014
Curso académico 2013/2014

Resumen

La consolidación de los sistemas informáticos como herramientas de valor para las organizaciones ha generado la necesidad de controlar en todo momento el aprovechamiento de los recursos TIC que se disponen así como la prevención de cualquier tipo de incidencia sobre los mismos.

Esta necesidad de control aumenta en el caso de empresas de base tecnológica como ActualTec S.L. que proporciona una serie de servicios de Internet y en la nube que requieren de una mínima calidad de servicio en términos de disponibilidad y continuidad.

Por lo tanto, es este el motivo de peso para desarrollar un sistema que permita seguir ciertos servicios y dispositivos de red y mostrar su evolución en tiempo real, manteniendo siempre un recuerdo histórico de los mismos. Este sistema pretende complementar la monitorización que ofrece Nagios, centrada en el control de servidores y servicios mediante el envío de alertas en el momento en el que se producen situaciones anómalas.

De esta forma se pretende reducir el tiempo de detección de cualquier tipo de problema o incluso anticiparse a ellos, pudiendo actuar con mayor rapidez y optimizar los costes y la eficiencia de la organización.

Palabras Clave

Monitorización, tiempo real, gráficas, patrón de diseño MVC (Modelo-Vista-Controlador), protocolo SNMP (Simple Network Management Protocol).

Keywords

Monitoring, real time, graphs, MVC (Model-View-Controller) design patter, SNMP (Simple Network Management Protocol) protocol.

Índice general

Capítulo 1: Introducción

Desde la asignatura EI1054 - *Pràctiques Externes i Projecte de Final de Grau* del Grado en Ingeniería Informática de la Universitat Jaume I de Castellón se presenta este proyecto bajo el nombre de “Aplicación web para la monitorización en tiempo real de servicios y dispositivos de red”.

El proyecto ha sido desarrollado en la empresa ActualTec Innovación Tecnológica S.L. con el fin de complementar la monitorización que realiza el software Nagios [1] sobre las nubes privadas asociadas a las clínicas de radiología.

Nagios ha sido implementado para monitorizar cada una de las máquinas que conforman estas nubes privadas y configurado para enviar notificaciones y/o alertas en el instante en el que algún servidor o servicio cae o padece algún tipo de anomalía. Además, desde la organización se busca una alternativa para representar algunos parámetros concretos de forma gráfica y ver su evolución en el tiempo, ya que el sistema implementado no presta esta funcionalidad. Se desea pues, disponer de una herramienta más para la prevención y detección de futuras incidencias.

Para acabar, se pretende describir los distintos capítulos que componen la estructura de la memoria:

- Descripción del proyecto: se pone en contexto al lector de la situación en la organización desde la que se parte, se exponen los objetivos del proyecto así como la metodología que se ha seguido y las herramientas software utilizadas para el desarrollo del mismo.

- Planificación del proyecto: se detallan las tareas que se han tenido que llevar a cabo para conseguir cumplir los objetivos del proyecto.

- Análisis y diseño del sistema: se analizan los requerimientos y se describe el diseño general y específico de la aplicación.

- Configuración e implementación: se detallan los procesos de configuración más importantes que se han necesitado para el correcto funcionamiento del sistema así como los detalles de implementación mas relevantes.

- Pruebas y documentación: se exponen las pruebas mas significativas realizadas y la documentación que se habilita a la organización.

- Conclusiones: se describen las conclusiones finales después de la realización del proyecto.

Capítulo 2: Descripción del proyecto

2.1. Contexto

ActualTec Innovación Tecnológica S.L. es una empresa de base tecnológica ubicada en el Parque Científico, Tecnológico y Empresarial de la Universitat Jaume I en Castellón. Actualmente la empresa trabaja paralelamente en dos líneas de negocio: ActualWeb, marca comercial que ofrece servicios de Internet como diseño y programación de páginas web, registro de nombres de dominio, correo electrónico, alojamiento web, posicionamiento en buscadores, tiendas virtuales, etc y ActualMed, marca comercial que se dedica al tratamiento de imágenes médicas digitales para el diagnóstico médico, ofreciendo un servicio de gestión y almacenamiento de imágenes en la nube bajo la marca ActualPacs. Los clientes, en este caso radiólogos o clínicas radiólogas, pueden organizar las imágenes médicas de cada paciente formando lo que se llaman estudios, visualizarlas directamente desde el navegador de cualquier dispositivo conectado a Internet mediante un visor o incluso descargarse el conjunto de imágenes o estudio localmente.

Para poder ofrecer todos estos servicios la empresa posee un pequeño centro de datos situado en el mismo edificio de Espaitec donde despliegan toda la infraestructura física sobre la que se construyen y se sirven los servicios de Internet y la nube. En los servidores que allí disponen se crean máquinas virtuales o PACS [2] construyendo la nube privada de cada cliente.

Entregar toda esta funcionalidad requiere de una mínima calidad de servicio SLA (Service Level Agreement) [3] firmada con el cliente en términos de disponibilidad y continuidad, de forma que es necesario contar con mecanismos de control que sean capaces de notificar al administrador cualquier tipo de anomalía sobre ciertos parámetros importantes que pongan en peligro el buen funcionamiento pactado en el SLA. No cumplir con ellos puede implicar enormes perdidas de dinero e incluso perder definitivamente al cliente.

Es importante pues la instalación y configuración de un sistema de monitorización que sea capaz de controlar cada una de las instancias creadas, comprobando que siguen levantadas y que sus servicios más importantes funcionan y responden bien a las peticiones de sus clientes (Apache, SSH, etc). También es preciso chequear aspectos del hardware asignado a cada máquina, como la carga media del procesador o la cantidad de espacio en disco que está utilizando cada cliente, entre otros muchos parámetros.

Son bastantes las alternativas que se plantean a la hora de elegir el sistema de monitorización más adecuado para la organización: Osmius [4], PandoraFMS [5], God [6], Monit [7], Nagios, etc., son ejemplos de este tipo de software. God y Monit son propuestas que carecen de cierta funcionalidad básica como servicio de notificaciones e/o interfaz gráfica de usuario. Osmius, PandoraFMS y Nagios son, cualquiera de ellas, herramientas más maduras y completas, por lo que se adaptan mejor a las necesidades de la empresa y pueden ser integradas como sistema de monitorización.

Por otro lado también se necesita controlar ciertas variables con un nivel de detalle mayor en el tiempo. Estos parámetros son: ancho de banda de entrada y de salida que se consume en la red de la empresa, número de accesos a las páginas web que se gestionan, número de accesos a sus bases de datos, así como estadísticas de

ActualPacs: número total de estudios, estudios por día, número de bytes totales utilizados y número de bytes utilizados por día.

Los sistemas de monitorización anteriores, como Nagios, son herramientas basadas en alertas y suelen carecer de este nivel de detalle o resolución, y si la implementan suele ser como una extensión o módulo y con ciertas limitaciones. De forma que para conseguir monitorizar y controlar estas variables se plantea el desarrollo de un sistema capaz de realizar un seguimiento en tiempo real y que actualice el estado de cada parámetro automáticamente utilizando gráficas accesibles desde una interfaz web.

Para realizar un seguimiento en tiempo real se requiere mantener una comunicación constante entre el emisor o fuente de los datos y el receptor. Uno de los protocolos más utilizados para este fin es el SNMP (Simple Network Management Protocol) [8].

SNMP se basa en dos elementos principales: un supervisor y agentes. El supervisor es la máquina desde donde el administrador realizará peticiones de gestión. A su vez, los agentes (snmpd) se definen como software que se encuentra en un dispositivo administrado. Los agentes tienen el propósito de recopilar información sobre los distintos objetos.

Routers, switches, hubs y servidores son ejemplos de dispositivos administrados que disponen de estos objetos. Los objetos administrados pueden ser información de hardware, estadísticas, parámetros de configuración, etc., los cuales se encuentran recopilados en una base de datos denominada MIB (Management Information Base). SNMP permite la conexión entre el supervisor y los agentes para recolectar los objetos necesarios en el MIB.

2.2. Objetivos del proyecto

Los objetivos que se deben cumplir para satisfacer las necesidades del proyecto son los siguientes:

- Realización de un sistema de monitorización en tiempo real. Implementación y configuración de toda la infraestructura necesaria para la obtención, procesamiento y almacenamiento automático de los parámetros planteados por la empresa.

- Desarrollo de una interfaz web para la visualización de las medidas a monitorizar. Investigación y análisis de librerías de dibujo de gráficas. Elección de la más oportuna y desarrollo de las gráficas necesarias.

2.3. Metodología

Para desarrollar e implementar la aplicación web se ha seguido el patrón de diseño MVC (Modelo-Vista-Controlador) [9]. Aunque originalmente fue diseñado para aplicaciones de escritorio, ha sido ampliamente adaptado para aplicaciones web y sistemas de monitorización como Nagios.

Los motivos por los cuales se decide emplear este patrón son los siguientes:

- Aplicaciones fáciles de mantener, de mejorar y de comprender.
- Incorporación sencilla de nuevas características.
- Trabajo simultáneo entre desarrolladores.
- Cambios transparentes entre las distintas capas.

Por ello, se intenta separar los datos de la aplicación, la lógica de control y la interfaz de usuario en tres componentes distintos.

Por un lado el modelo son los datos que el sistema opera, es el responsable de la recuperación de los mismos, así como de su procesamiento, validación y cualquier otra tarea relativa a su manipulación.

Por otro lado la vista se encarga de presentar los datos del Modelo en un formato adecuado para interactuar, usualmente como interfaz de usuario.

El controlador es el último elemento y se encarga de gestionar las peticiones que lanzan los usuarios del sistema, invocando al modelo y seleccionando el tipo de respuesta más adecuada según las preferencias del cliente, delegando a la Vista el proceso de presentación. Se puede decir que el controlador es el componente que hace de intermediario entre el modelo y la vista.

Los sistemas de monitorización actuales han intentando adaptarse también, en la medida de lo posible, a las directrices que marca este patrón. Nagios, ya implementado como sistema de alertas en la empresa, dispone de una arquitectura servidor/agente. Esto significa que existe un servidor dedicado dentro de la intranet que tiene instalado y configurado el núcleo del sistema junto con una base de datos de respaldo, mientras que las máquinas remotas necesitan de ciertos plugins (agentes) para chequear los recursos y servicios que se soliciten. Este conjunto formado por el núcleo, la base de datos y los plugins se correspondería con el modelo. Además, Nagios dispone de una interfaz web donde se muestran el estado de los servicios y servidores monitorizados. Como controlador, Nagios cuenta con una serie de programas CGI encargados de consultar el modelo y de obtener la información que solicita la vista.

Sin embargo, no por ello se ha dejado de seguir con las denominadas fases típicas del desarrollo de un proyecto informático: análisis, diseño e implementación han formado parte de la planificación inicial, final y de la metodología seguida a lo largo del transcurso del proyecto.

2.4. Herramientas utilizadas

En este punto se va a detallar las herramientas software necesarias para el desarrollo de este proyecto.

2.4.1. Software del servidor

Utilizamos un servidor remoto que desde la empresa habilitan exclusivamente para el desarrollo del proyecto. Por defecto cuenta con un sistema operativo libre de código abierto como es la distribución CentOS 6.4 con licencia GNU GPL.

Se determina instalar y configurar los siguientes paquetes para el correcto desarrollo del proyecto:

- **mysql-server**: Sistema de Gestión de Bases de Datos utilizado para gestionar todas las bases de datos necesarias.
- **Apache**: servidor web con licencia GPL (httpd en CentOS).
- **vsftpd**: servidor FTP con licencia GPL para la transferencia de paquetes desde el cliente.
- **PHP**: version 5.3.3 junto con las librerías php-mysql y php-snmp como lenguaje de programación del back-end de la aplicación.

2.4.2. Software del PC Local

Utilizamos una estación de trabajo para diseñar e implementar el proyecto de manera local para después enviar diferentes versiones al servidor remoto. Se instaló un sistema operativo como es Ubuntu 12.04 LTS por el hecho de que dispone de un conjunto de herramientas y paquetes de trabajo con el que se está más familiarizado y por compatibilidad con el servidor.

Se determina instalar y configurar los siguientes paquetes y programas para el correcto desarrollo del proyecto:

- **mysql**: Sistema de Gestión de Bases de Datos utilizado para gestionar todas las bases de datos necesarias.
- **Apache**: servidor web con licencia GPL.
- **vsftpd**: servidor FTP con licencia GPL para la transferencia de paquetes desde el cliente hasta el servidor.
- **PHP**: version 5.3.3 junto con las librerías php-mysql y php-snmp como lenguaje de programación del back-end de la aplicación.
- **phpmyadmin**: herramienta para gestionar mediante interfaz web las distintas base de datos mysql creadas.
- **OpenSSH**: conjunto de aplicaciones que tienen como objetivo permitir el establecimiento de conexiones cifradas a través de una red. Utilizado para el acceso, la gestión y la configuración del servidor de gráficas.

- **Aptana Studio 3**: entorno de desarrollo web integrado. Es software libre.
- **Google Chrome**: navegador web gratuito desarrollado por Google.

2.4.3. Software del portátil

También se utiliza un ordenador portátil para el desarrollo de la memoria y la realización de pruebas como un cliente alternativo al de la estación de trabajo. Por defecto lleva instalado el sistema operativo Windows 8 de Microsoft.

Las herramienta necesaria para el desarrollo del proyecto es la siguiente:

- **Microsoft Office 2010**: Suite ofimática de Microsoft. Uso de Microsoft Word para la redacción de la memoria así como de Microsoft Project para ajustar la planificación del proyecto.

Capítulo 3: Planificación del proyecto

En este capítulo se detalla la planificación de tareas que se ha necesitado para llevar a cabo correctamente el proyecto y una estimación del coste total. La planificación inicial ha sufrido ciertos cambios a lo largo de la estancia en prácticas en algunas etapas y tareas, concretándose fases iniciales y generando tareas específicas.

3.1. Definición y planificación de tareas

El desarrollo del proyecto se divide en una serie de tareas que se detallan en la Tabla 1. Se pretende marcar con un tic positivo aquellas tareas que ya se habían contando en la planificación inicial y con una aspa aquellas que han surgido a lo largo del proyecto.

Fase / Tarea	Descripción	Planificada inicialmente	Horas Iniciales	Horas Finales
Planificación y estudio			**80h**	**64h**
Identificar alcance y objetivos	Definir los objetivos principales del proyecto así como su alcance	✓	10h	10h
Definir metodología a seguir	Concretar las tareas a realizar y situarlas en su fase correspondiente	✓	42h	8h
Estudio inicial	Análisis de librerías de *charting*, estudio del protocolo SNMP, JQuery, BootStrap, Nagios, lectura y estudio de la documentación del router Cisco RV320 de la empresa y análisis de las herramientas disponibles.	✓	28h	48h
Entrega Propuesta Técnica	**Hito 0**: Entrega de la propuesta técnica.	✓	0h	0h
Análisis y diseño			**85h**	**57h**
Definir requisitos de uso	Definir que necesidades va a tener que satisfacer el nuevo sistema	✓	25h	10h
Definir requisitos tecnológicos	Concretar que requisitos tecnológicos y de plataforma necesito para llevar a cabo el proyecto (maquinas, sistemas operativos, recursos, herramientas)	✓	25h	7h
Diseño del sistema	Diseño del sistema intentando seguir el patrón de diseño MVC	✗	-	15h
Diseño de la BBDD	Identificación de las tablas necesarias y sus relaciones entre si	✗	-	16h
Diseño de interfaz de usuario	Diseño de la interfaz web a modo de panel de control siguiendo los principios mas importantes.	✓	35h	9h
Configuración e Implementación			**85h**	**158h**

Fase / Tarea	Descripción	Planificada inicialmente	Horas Iniciales	Horas Finales
Configuración del servidor remoto y dispositivos de red	Instalación y configuración de los paquetes y software necesarios en el servidor para el correcto funcionamiento del sistema, así como habilitar y configurar el protocolo SNMP en el router de la empresa.	✓	*	22h
Programación	Desarrollo e implementación de los distintos códigos, funciones y métodos, ya sea PHP y scripts para el modelo y el controlador y HTML y JavaScript para la vista.	✓	*	130h
Mantenimiento del sistema Nagios	Realizar pequeñas tareas de mantenimiento para el sistema Nagios	✗	-	6h
Implantación y pruebas			**45h**	**22h**
Implantación	Implantación del sistema desarrollado en el servidor dedicado	✓	10h	6h
Pruebas	Realización de todo tipo de pruebas para anotar su comportamiento y corregir cualquier tipo de error durante las fases anteriores.	✓	35h	16h
Entrega final	**Hito 1**: Entrega del sistema acabado	✓	0h	0h
Documentación y entrega del PFG			**135h**	**90h**
Informes quincenales	Realización de los informes quincenales durante la estacia en prácticas en la empresa	✓	4h	5h
Preparación de la presentación oral	Preparación de la presentación oral	✓	30h	24h
Memoria Técnica	Redacción de la memoria técnica	✓	100h	60h
Presentación oral y memoria técnica	**Hito 2**: Realización de la presentación oral. Entrega de la memoria del PFG	✓	1h	1h

Tabla 1: Definición de tareas.

Con el objetivo de aclarar ciertos aspectos del contenido de la Tabla 1 se puede decir que las tareas marcadas con un guión (-) significan que al no haberse planificado inicialmente no se estimó el tiempo de duración de las mismas. Por otro lado las tareas marcadas con un asterisco (*) indican que en la planificación inicial conformaban una única tarea.

El diagrama de Gantt correspondiente a las tareas anteriormente descritas es el siguiente:

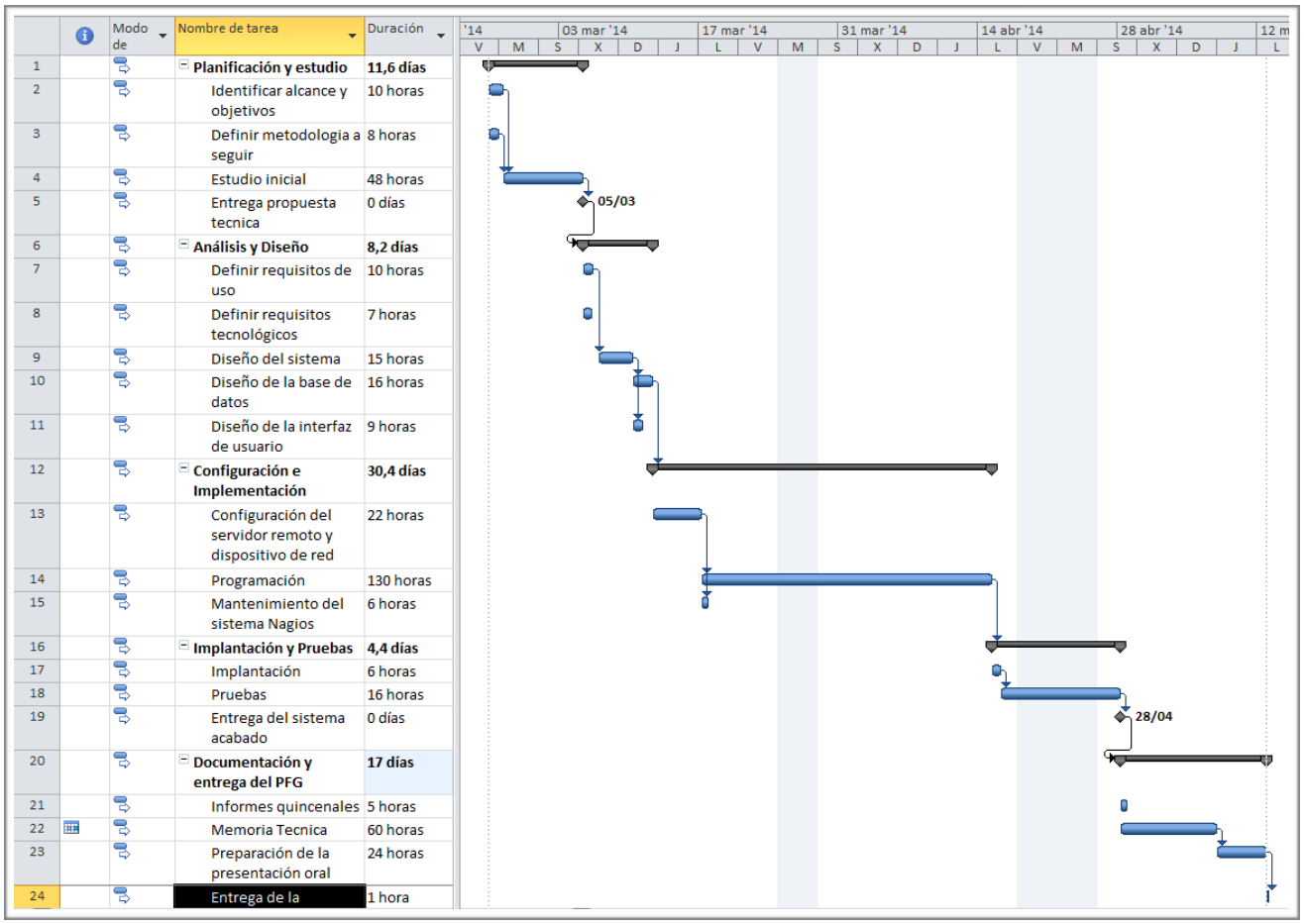

Imagen 1: Diagrama de Gantt sobre la planificación temporal definitiva.

3.2. Estimación del coste económico

En este apartado se estiman los gastos económicos teniendo en cuenta las horas invertidas, los costes del hardware utilizado y los del software necesario.

En primer lugar se tiene que estimar el coste de una hora de trabajo realizado. Esta estimación queda detallada en la Tabla 2.

Tipo de trabajo	Precio/Hora	Horas invertidas	Total
Análisis	20 €	17	340 €
Diseño	20 €	40	800 €
Programación	12 €	130	1560 €
			2700 €

Tabla 2: Estimación del coste/hora de trabajo.

En la Tabla 3 se muestran los gastos relacionados con el hardware.

Material	Coste
Estación de trabajo local	600 €

Tabla 3: Gastos en hardware.

En la Tabla 4 siguiente se detallan los gastos del software.

Material	Coste
Sistema operativo servidor	0 €
Sistema operativo estación local	0 €
Herramientas de programación, diseño, base de datos, conexión remota, configuración, etc. en Linux	0 €
Herramienta ofimática del pc portátil	539 €

Tabla 4: Gastos en software.

Y, finalmente, en la Tabla 5 se especifican los gastos totales:

Variables	Coste
Coste total del trabajo por horas	2700 €
Materiales hardware	600 €
Herramientas software	539 €
TOTAL	**3839 €**

Tabla 5: Gasto total estimado del proyecto.

Capítulo 4: Análisis y diseño del sistema

4.1. Análisis de requisitos

Junto con los objetivos, ActualTec S.L. plantea una serie de requisitos que el sistema a desarrollar debe incorporar. Esta nueva funcionalidad tiene como principal fin subsanar y mejorar en la medida de lo posible todos los defectos del sistema primitivo de gráficas que poseen.

4.1.1. Requisitos funcionales

En este punto quedan detallados los requisitos funcionales que se plantean ante el desarrollo del nuevo sistema:

- El sistema debe obtener y almacenar automáticamente por cada minuto los siguientes datos de ActualWeb: paquetes transmitidos y recibidos que pasan por las interfaces operativas del router, número total de accesos a las páginas web que gestionan, número total de cada tipo de consultas realizadas a las bases de datos de web01.serverwhite.com y web03.serverwhite.com.

- El sistema debe obtener y almacenar automáticamente por cada día los siguientes datos de ActualPacs: número total de estudios, número de estudios por día, número total de bytes y número de bytes por día.

- El sistema debe controlar el crecimiento desmesurado de las tablas que se rellenan por cada minuto.

- El sistema debe controlar el crecimiento desmesurado de las tablas que se rellenan por cada hora.

- El sistema debe presentarse al usuario con la apariencia de un panel de control.

- El sistema debe organizar en la interfaz de usuario cada conjunto de gráficas por cada tipo de información que trata de visualizar.

- El sistema debe generar como mínimo dos gráficas por cada tipo de información. Una con mayor resolución y actualizándose en tiempo real y otra con menor resolución pero manteniendo un histórico de datos en el tiempo.

- La interfaz de usuario debe disponer de un enlace a un menú de selección de gráficas donde el usuario podrá elegir dinámicamente qué gráficas quiere tener en modo pantalla completa.

- La interfaz de usuario debe disponer de un enlace a la interfaz web de administración de Nagios.

- Las gráficas deben adaptar el eje Y automáticamente a los múltiplos y submúltiplos de las unidades utilizadas para representar cada valor.

- Las gráficas deben apuntar siempre al dato mas reciente que se disponga de la tabla de la que obtiene la información.

- Las gráficas deben ser capaces de purgar datos y reajustar su apariencia automáticamente cuando empiecen a haber gran cantidad de datos distintos representados.

- Las gráficas deben ser interactivas, es decir, ofrecer en la medida de lo posible un feedback al usuario cuando éste coloque el cursor sobre las gráficas.

4.1.2. Requisitos no funcionales

En este punto quedan detallados los requisitos no funcionales que se plantean ante el desarrollo del nuevo sistema:

- La interfaz web debe visualizarse y funcionar correctamente en cualquier navegador, especialmente en Google Chrome, Mozilla Firefox y Safari.

- Las gráficas deben ser generadas mediante alguna librería adecuada del lenguaje de programación JavaScript.

- El tiempo de carga de cualquier gráfica no debe superar los tres segundos independientemente de la cantidad de datos que deba de mostrar.

4.2. Análisis de librerías para generar gráficas

JavaScript es un lenguaje de programación utilizado principalmente para generar páginas web dinámicas.

Una página web dinámica puede incorporar efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario.

JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. Es decir, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios.

Este lenguaje dispone de numerosas bibliotecas ya desarrolladas que facilitan la vida al programador web. Por lo que respecta a la construcción de gráficas existen más de una veintena de librerías conocidas. Si se visita el siguiente enlace:

- http://en.wikipedia.org/wiki/ Comparison_of_JavaScript_charting_frameworks

se observa una tabla comparativa de las diferentes librerías de charting. De aquí se han seleccionado cuatro para su posterior análisis, intentando elegir aquellas que más casillas en verde (más posibilidades) ofrecen a priori al programador. La Tabla 6 muestra esta comparativa.

Libreria \Funcionalidad	Tipo de licencia	Documentación	Adaptabilidad	Tiempo Real	Purgar	Interativa
ZingChart	PAGO + GRATUITA	BUENA	SI	SI	SI	SI
amCharts	PAGO + GRATUITA	BUENA	SI	SI	Solo con el paquete Stock Charts	SI
nvd3	GRATUITA	POBRE	SI	SI	NO	SI
RGraph	GRATUITA	COMPLEJA	SI	SI	NO	SI

Tabla 6: Librerias de charting seleccionadas para su posterior análisis.

Los requisitos que se imponía desde la organización eran los comentados en el apartado anterior: adaptabilidad del eje Y, actualización en tiempo real, purgación de datos y respuesta interactiva de la gráfica.

RGraph [10] es la libreria a priori más sencilla de este subconjunto. Es gratuita y permite construir gráficas de diferentes tipos soportando cierta interactividad con el usuario, siempre y cuando el programador desarrolle su propio código de respuesta al evento. Su documentación no es mala pero algo compleja de seguir, lo que dificulta y retrasa el tiempo de aprendizaje. Por otro lado, debido a su sencillez el aspecto de las gráficas que genera con respecto al resto de librerías es el menos logrado. Por estos motivos descartamos RGraph como librería para el desarrollo del proyecto.

Otra librería interesante es NVD3 [11]. Esta basada en la librería D3.js [12] y con ella es posible crear gráficas estéticamente más interesantes que con RGraph. Dispone en su website de una herramienta de edición en tiempo real de su código, donde se puede visualizar cualquier cambio y realizar pruebas. De manera nativa incluye atributos que habilitan o deshabitan cierto tipo de interactividad (como mostrar un cuadro con los valores correspondientes en cada instante al pasar el puntero sobre la gráfica). Por contra no dispone de ningún tipo de manual ni de documentación de referencia. Es una librería gratuita.

La siguiente libreria es ZingChart [13]. Se trata de una de las mejores que se disponen actualmente. En su página web se informa de que es la que utilizan empresas como Apple, Adobe, Intel, Microsoft, Nasdaq, etc. Su documentación es bastante buena, disponiendo de una API bien documentada aunque se pone a disposición en su website de muy pocos tutoriales, olvidándose de temas más avanzados como gestionar los cambios de las gráficas que necesitan actualizarse en tiempo real. Al igual que NVD3, los eventos más habituales se gestionan de forma nativa. Se trata de una librería con licencia de pago, poniendo a disposición de todo el mundo una versión libre limitada de funcionalidad.

La última libreria es amCharts [14]. Se divide en tres sub-librerías:

- CHARTS —> para un uso a nivel de usuario.

- STOCK CHARTS —> optimizada para las necesidades y el desarrollo en empresas.
- MAPS —> gráficas en las que se necesita representar datos sobre un mapa del mundo.

Como ZingCharts y NVD3, es posible lograr gráficas sencillas y estéticamente interesantes de forma asequible. Su documentación incluye una extensa información sobre su API, además de encontrar numerosos y detallados tutoriales que explican la gestión de gráficas en tiempo real, el purgado de datos, así como la recuperación de información de una base de datos relacional, entre otros muchos aspectos. Su desventaja es que, al igual que ZingCharts, es una librería cuya versión estándar es de pago. Aún así, podemos descargarnos la versión gratuita y trabajar con total garantía.

Finalmente la decisión ha sido continuar el proyecto con la librería amCharts, sobretodo porque cumple con todos los requisitos que se solicitan, y como opinión personal proporciona la mejor documentación incidiendo en gran número de detalles.

4.3. Diseño del sistema

En esta sección se describe el diseño de las diferentes piezas que conforman la aplicación web desarrollada.

Como se ha comentado en el apartado de Metodología, se ha seguido el patrón MVC para el desarrollo del sistema. Realmente se ha realizado una adaptación del patrón para un sistema de monitorización, donde la vista simplemente se encarga de visualizar el estado del modelo, sin tener privilegios para poder realizar cambios sobre este último. De esta forma, el controlador va a tener que encargarse únicamente de obtener los datos solicitados y transformarlos a un formato entendible para la vista. Este formato va a ser JSON [15].

Por otro lado, el modelo va a tener que cambiar por la acción en segundo plano de un conjunto de scripts que obtendrán información de diferentes dispositivos de red y servicios y que van a ir actualizando la base de datos del sistema. Se puede observar este comportamiento en la Imagen 1.

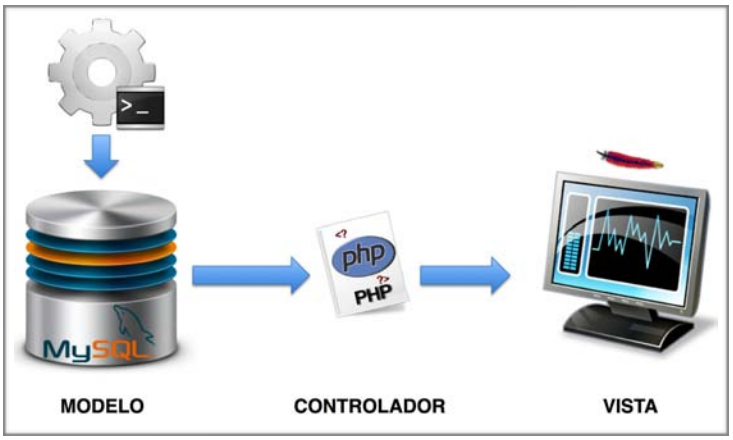

Imagen 1: Esquema general del funcionamiento del sistema.

4.3.2. Diseño específico

Todo el esquema anterior se aplica a un servidor que va a dedicarse exclusivamente a esa labor. Esta máquina va a disponer de una IP dentro de la red interna de la organización y va a ser accesible únicamente desde ésta, al igual que el servidor de Nagios. Para ello la empresa dispone de los mecanismos físicos y lógicos de seguridad necesarios para evitar ataques desde fuera. Son los siguientes:

- Las máquinas se ejecutan en un centro de datos localizado en el propio edificio de Espaitec que cuenta con entrada restringida para un pequeño grupo de administradores de red, siendo accesible únicamente mediante acreditación, pasando la tarjeta por el hardware del pomo de la puerta.

- Disposición de un router que actúa a modo de firewall que esta configurado para evitar cualquier intrusión desde Internet a las máquinas de la organización.

Como se ha comentado en el anterior punto, el servidor de gráficas ha de recibir los datos necesarios de fuentes externas. Estas fuentes son: dispositivos de red como el router principal de la empresa, servicios como apache y mysql, así como distintas páginas web que son generadas dinámicamente en el momento en el que se solicitan.

A continuación se detalla cómo se produce la obtención de los datos:

- Para obtener los paquetes de entrada y de salida se van a realizar consultas al router mediante el protocolo SNMPv3.

- Para obtener el número de accesos a las páginas web que gestiona ActualWeb es necesario descargarse la pagina web01.serverwhite.com/status generada por el módulo mod_status de Apache, tratarla correctamente y obtener el conjunto de datos necesarios.

- Para obtener ciertas estadísticas relevantes de ActualPacs se descargará y procesará la página web 192.168.0.50:81/stats que es generada dinámicamente por dicho servidor interno cada vez que se solicita.

- Para obtener el número y tipo de peticiones a las bases de datos de web01.serverwhite.com y web03.serverwhite.com se habilita un usuario y contraseña para cada base de datos, con privilegios para acceder remotamente desde el servidor de gráficas.

El diseño específico del sistema se ilustra en la Imagen 2.

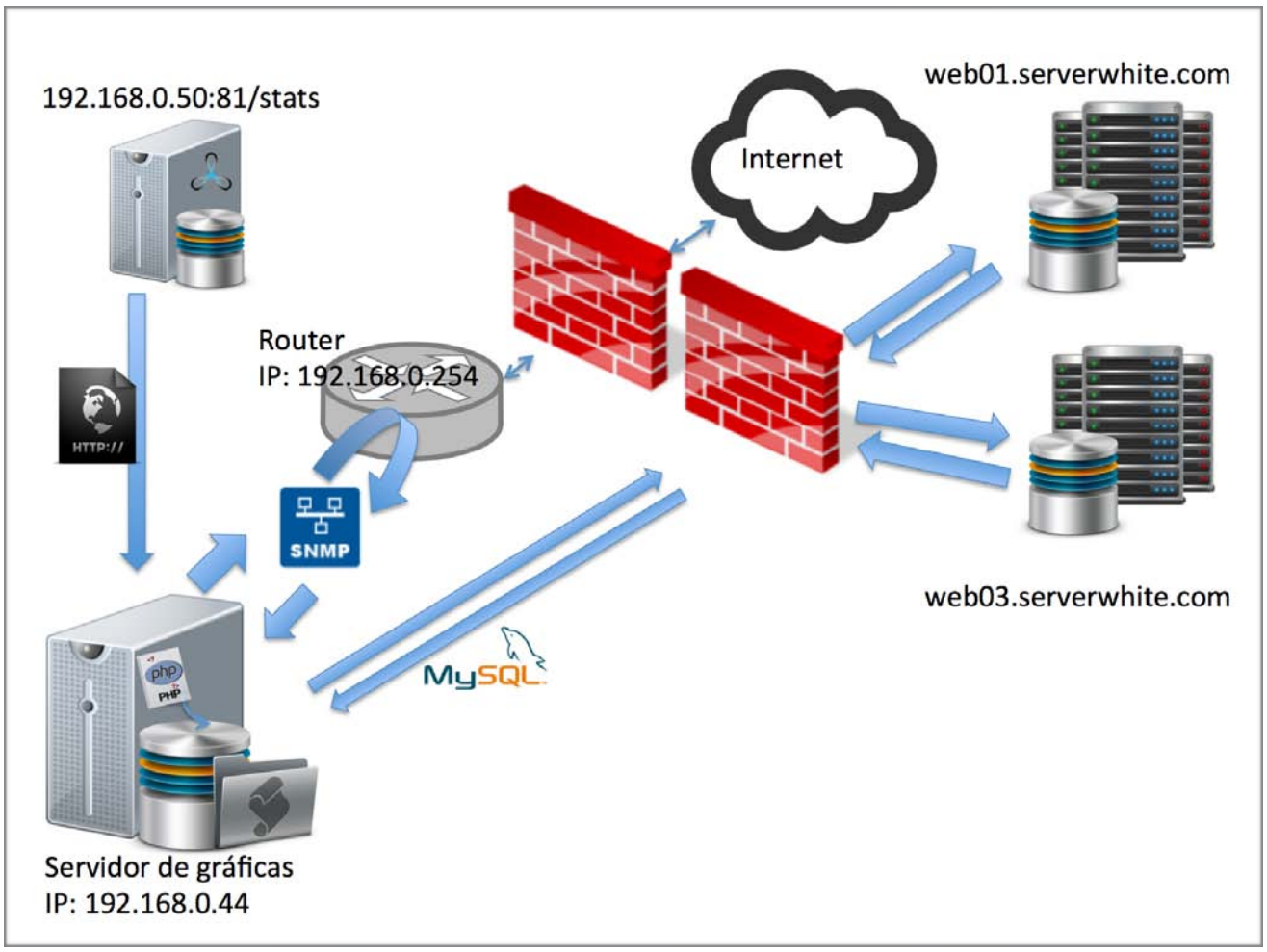

Imagen 2: Diseño específico del sistema de monitorización.

4.3.2. Diseño de la base de datos

Para el almacenamiento de todos los datos solicitados se ha creado una base de datos relacional mediante el SGDB MySQL [16] llamada 'crond'. En este epígrafe se describen las tablas más importantes que contiene. Es conveniente informar de que en este sistema no existe ningún tipo de relación entre ninguna tabla puesto que la aplicación no lo requiere. Cada tabla servirá para almacenar un tipo específico de información que posteriormente se utilizará para generar unas gráficas concretas.

Se pueden clasificar las tablas en función del periodo de tiempo en el que se van rellenando automáticamente. Por un lado se presentan las tablas utilizadas para generar gráficas que se actualizan en tiempo real (cada minuto y cada hora). Por otro lado se muestran las tablas que son rellenadas entre periodos más grandes (cada día y cada año).

Tablas para medidas de monitorización en tiempo real

Las tablas que se actualizan cada minuto de la base de datos 'crond' se muestran en la Imagen 3.

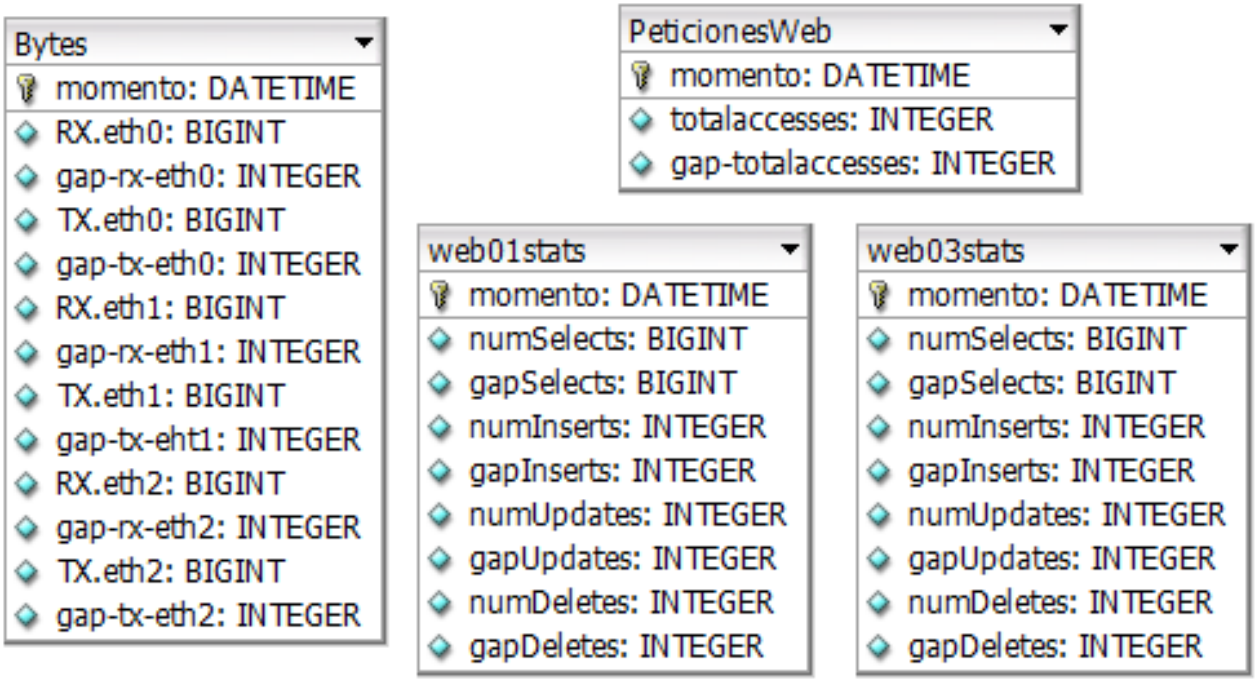

Imagen 3: Tablas actualizadas cada minuto de la base de datos 'crond'.

La idea común de todas ellas es que todos los datos que se van a recoger actúan a modo de 'contador', de forma que para obtener el dato representativo de un periodo de tiempo se debe de calcular la diferencia entre el dato más actual y el último registrado. Estos datos intentan llevar la palabra *gap* (diferencia, intervalo, en inglés) al principio de cada nombre de columna. A continuación se detallan las tablas utilizadas:

- **Bytes**: Cada fila almacena el instante de tiempo en el que se produce la inserción (momento), los paquetes recibidos y transmitidos totales (en bits) que pasan por cada

interfaz del router de la empresa (RX.ethX y TX.ethX), así como el valor exacto durante el último minuto (gap-rx-etX y gap-tx-ethX).

- **PeticionesWeb**: Cada fila almacena el instante de tiempo en el que se produce la inserción (momento), el número total de accesos web que gestiona la empresa (totalaccesses), así como el valor exacto de accesos durante el último minuto (gap-totalaccesses).

- **web01stats**: Cada fila almacena el instante de tiempo en el que se produce la inserción (momento), el número total de peticiones de cada tipo que procesa el SGBD de la máquina web01.serverwhite.com (numSelects, numInserts, numUpdates y numDeletes) así como el valor exacto de cada tipo de accesos durante el último minuto (gapSelects, gapInserts, gapUpdates y gapDeletes).

- **web03stats**: Cada fila almacena el instante de tiempo en el que se produce la inserción (momento) que es la clave primaria, el número de peticiones de cada tipo que procesa el SGBD de la máquina web03.serverwhite.com (numSelects, numInserts, numUpdates y numDeletes) así como el valor exacto de cada tipo de accesos durante el último minuto (gapSelects, gapInserts, gapUpdates y gapDeletes).

Cada tabla comentada anteriormente dispone de otra tabla de respaldo donde se almacena por cada *gap* la media aritmética de todos sus valores asociados y recuperados durante la última hora. Las tablas se muestran en la Imagen 4.

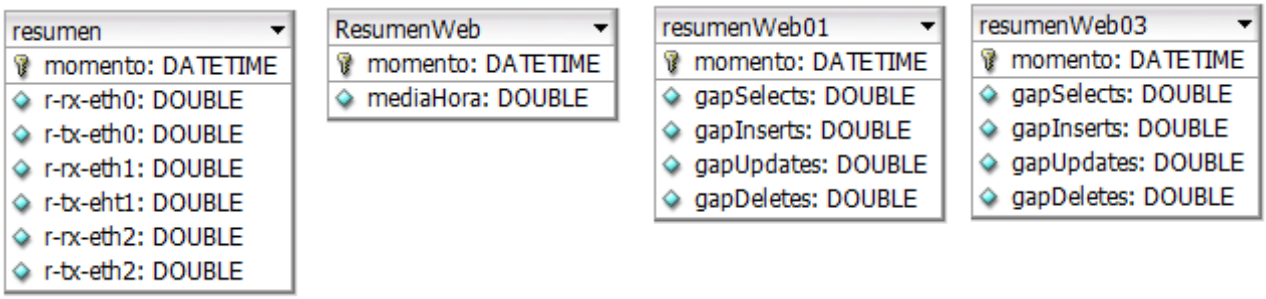

Imagen 4: Tablas actualizadas cada hora de la base de datos 'crond'.

- **resumen**: Almacena la media aritmética de los paquetes recibidos y transmitidos por cada interfaz durante la última hora. Es la tabla de respaldo de Bytes.

- **resumenWeb**: Almacena la media aritmética de todos los accesos (*gap-totalaccesses)* realizados la última hora. Es la tabla de respaldo de PeticionesWeb.

- **resumenWeb01**: Almacena las medias aritméticas de los diferentes tipos de acceso registrados durante la última hora. Es la tabla de respaldo de web01stats.

- **resumenWeb03**: Almacena las medias aritméticas de los diferentes tipos de acceso registrados durante la última hora. Es la tabla de respaldo de web03stats.

Tablas para medidas de monitorización históricas

Estas tablas se caracterizan porque su periodo de actualización es bastante mayor que las comentadas con anterioridad.

Por un lado se dispone de una tabla que almacena información relevante sobre ActualPacs y, por otro, un grupo de tablas que son respaldo anual de las tablas resumen comentadas anteriormente.

En primer lugar, en la Imagen 5 se muestra la tabla asociada a ActualPacs.

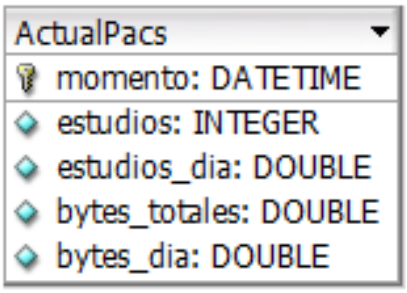

Imagen 5: Tabla ActualPacs, actualizada cada día.

- **ActualPacs**: Se almacena diariamente la cantidad de estudios totales, estudios por día, bytes totales utilizados y bytes por día utilizados.

En la Imagen 6 se muestran las tablas de respaldo anuales asociadas a las tablas resumen.

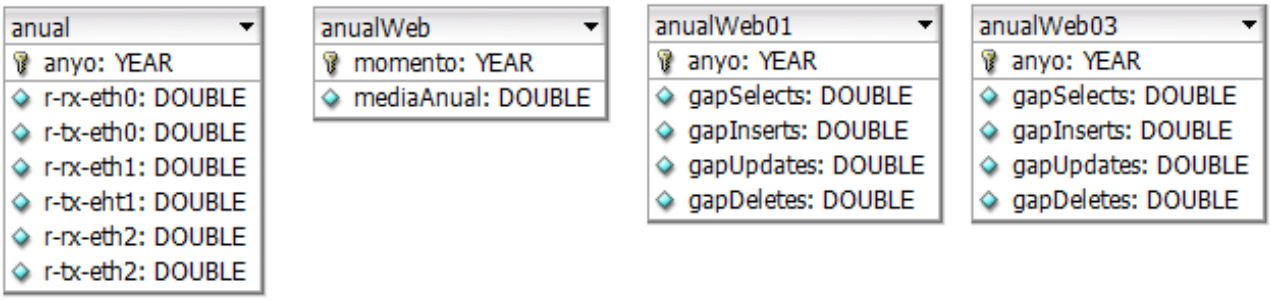

Imagen 6: Tablas actualizadas cada año de la base de datos 'crond'.

- **anual**: Almacena la media aritmética de los paquetes recibidos y transmitidos por cada interfaz durante el último año. Es la tabla de respaldo de resumen.

- **anualWeb**: Almacena la media aritmética de todos los accesos (*gap-totalaccesses)* realizados el último año. Es la tabla de respaldo de resumenWeb.

- **anualWeb01**: Almacena las medias aritméticas de los diferentes tipos de acceso registrados durante el último año. Es la tabla de respaldo de resumenWeb01.

- **anualWeb03**: Almacena las medias aritméticas de los diferentes tipos de acceso registrados durante el último año. Es la tabla de respaldo de resumenWeb03.

4.4. Diseño de la interfaz web

La interfaz web desarrollada pretende ofrecer un aspecto de panel de control al usuario final, para ello se ha decidido utilizar el framework Twitter Bootstrap [17] por los siguientes motivos:

- Disposición de plantillas basadas en HTML5 + CSS3 [18] con buena estética y adaptables a todo tipo pantallas y/o dispositivos.
- Compatible con jQuery [19].
- Framework fácil de integrar en el proyecto web.

El directorio físico donde reside el proyecto web en el servidor dedicado es el siguiente:

- /var/www/html/

Los directorios y ficheros que contiene son los siguientes:

- amcharts/ (ficheros que conforman la librería amcharts).
- css/ (ficheros de hoja de estilo utilizados).
- doc/approach.pdf (propuesta técnica del proyecto).
- html/panel.html (página del panel de control).
- imagenes/ (imágenes utilizadas en la interfaz web).
- index.html (portada de la aplicación web de monitorización).
- js/ (se encuentra la librería jQuery, scripts utilizados para la generación de gráficas así como scripts para la manipulación del DOM [20] del panel de control).
- php/ (scripts que tratan de satisfacer la lógica de negocio, realizando la función del controlador).
- .htaccess (fichero de configuración distribuida donde quedan definidas las directivas de configuración para cada directorio y recurso del proyecto web).

La visualización del panel de control sigue el esquema descrito en la Tabla 7.

Barra de Navegación	
Menú. Barra Lateral	**CONTENIDO**

Tabla 7: Estructura frontal del panel de control.

En la Imagen 7 se observa la portada de la aplicación web.

Imagen 7: Portada de la aplicación web.

Y en las Imágenes 8 y 9 se muestran capturas de pantalla del panel de control.

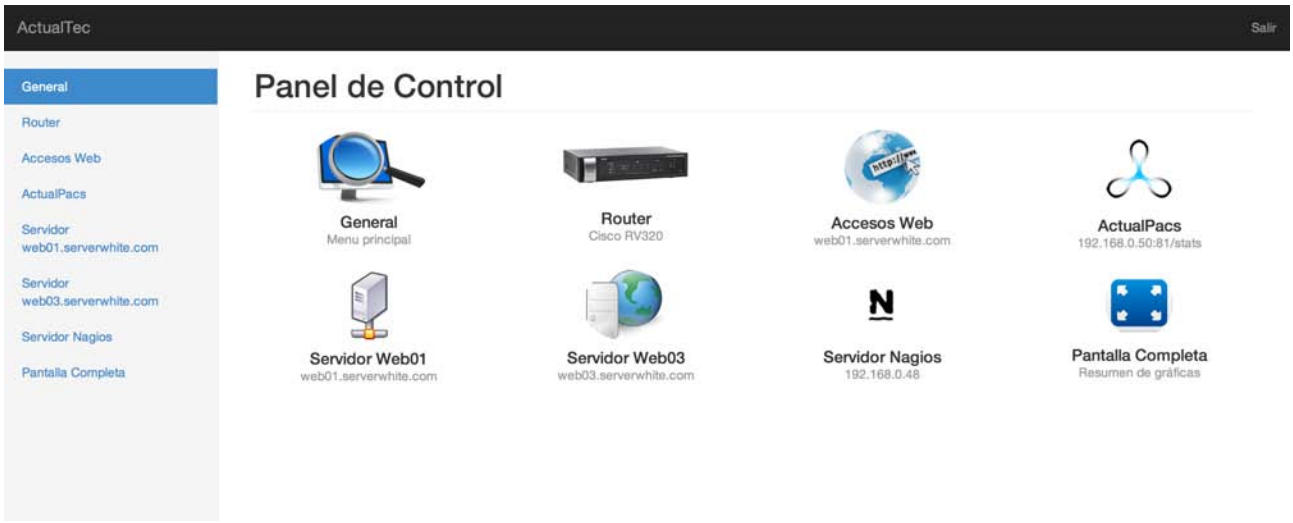

Imagen 8: Página de inicio del panel de control.

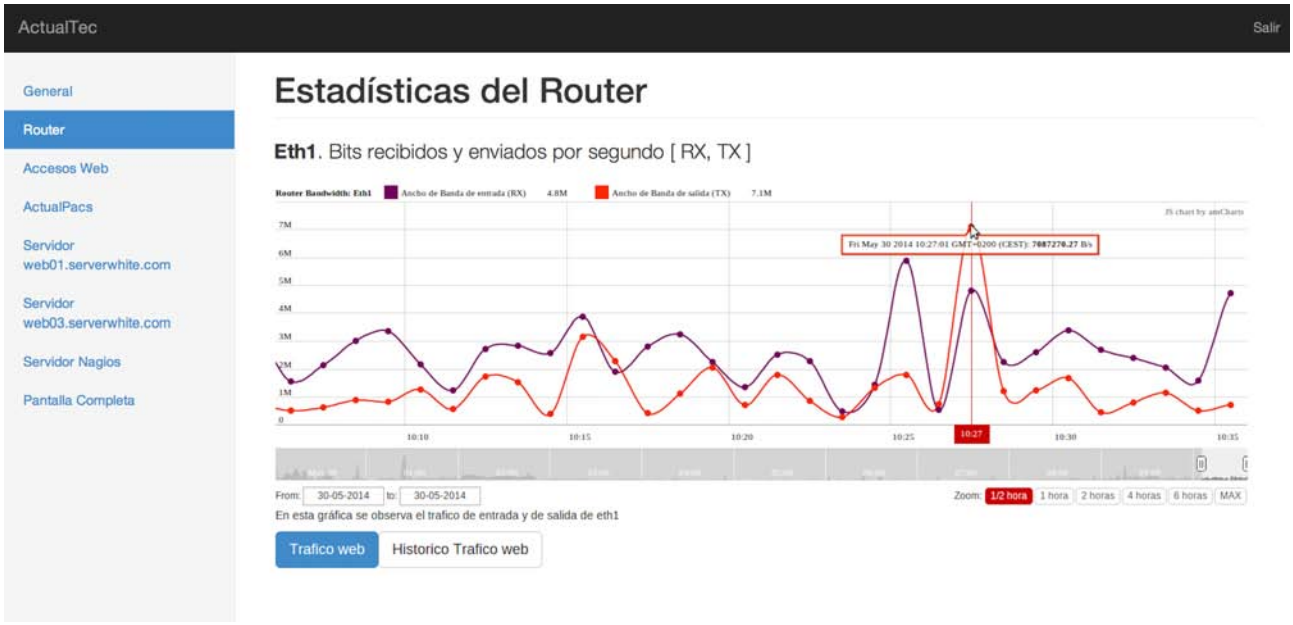

Imagen 9: El panel de control mostrando una gráfica del tráfico de entrada y de salida de eth1.

A continuación se comenta la funcionalidad básica que ofrece el panel de control gracias al menú situado en la barra lateral:

- General: Vuelta al inicio del panel de control.
- Router: Visualización y selección de las distintas gráficas que muestran el estado del tráfico que pasa por el router de la empresa.
- Accesos Web: Visualización y selección de las distintas gráficas que muestran el número de accesos a páginas web gestionadas por ActualWeb.
- ActualPacs: Visualización y selección de las distintas gráficas que muestran los datos más importantes de ActualPacs.
- Servidor web01.serverwhite.com: Visualización y selección de las distintas gráficas que muestran el número de consultas a la base de datos de este servidor así como el porcentaje de cada tipo de peticiones.
- Servidor web03.serverwhite.com: Visualización y selección de las distintas gráficas que muestran el número de consultas a la base de datos de este servidor así como el porcentaje de cada tipo de peticiones.
- Servidor Nagios: Acceso directo a la interfaz de monitorización de Nagios.
- Pantalla completa: Selección de gráficas personalizadas y posterior generación de las mismas en una pagina web optimizada para ser configurada a pantalla completa.

Capítulo 5: Configuración e implementación

Una vez descrito el análisis y el diseño de todo el sistema, en este capítulo se detallan las configuraciones e implementaciones necesarias para cubrir con todas las funcionalidad requerida.

5.1. Configuración de SNMPv3

SNMP es el protocolo por excelencia para la gestión y la monitorización de dispositivos de red y sus funciones. Trabaja en la capa de aplicación bajo el protocolo UDP (User Data Protocol) y por defecto suele permanecer a la escucha en el puerto 161.

SNMP dispone de tres versiones (v1, v2 y v3). Las versiones v1 y v2 son las más aceptadas a día de hoy en las empresas a pesar de que presentan ciertos riesgos de seguridad:

- Por un lado los mecanismos de autenticación que implementan se basan exclusivamente en un nombre de comunidad (*public*, por defecto), un modo de acceso (ninguno, sólo lectura, lectura-escritura o sólo escritura) a una parte o todo el árbol MIB y opcionalmente un rango de direcciones IP que pertenecen al grupo, de forma que la autenticación de los clientes se realiza solo comprobando la cadena de octetos de la comunidad y la dirección IP en el caso de que sea oportuno, generando posibles amenazas de suplantación de identidad.

- Por otro lado los mensajes viajan sin ningún tipo de cifrado por el canal de comunicaciones por lo que cualquier atacante podría capturar el tráfico mediante *sniffers*, obtener muestras de los paquetes y extraer información o modificar los mensajes en tránsito.

La versión v3 intenta salvar estas amenazas proporcionando importantes características de seguridad y configuración remota:

- Confidencialidad: Utilización de algoritmos como DES o AES para cifrar los mensajes, garantizando un mayor nivel de privacidad.

- Autenticación: Empleo de usuario, contraseña y firma digital mediante algoritmos como MD5 y SHA1. Se intenta garantizar de esta forma que los mensajes provienen de una fuente segura y su integridad.

Por los motivos comentados anteriormente se decide emplear y configurar la versión SNMPv3 del protocolo para la recuperación de datos procedentes del router. En concreto se pretende obtener la cantidad de paquetes recibidos y transmitidos para posteriormente calcular el ancho de banda utilizado en cada momento.

Como se había comentado en el capítulo 2, el protocolo se basa en dos elementos principales: un supervisor y unos agentes. El supervisor es la máquina desde donde el administrador realizará peticiones de gestión. A su vez, los agentes (snmpd) se definen como software que se encuentra en un dispositivo administrado y que recopilan información sobre los distintos objetos.

El servidor de gráficas va a realizar el papel de supervisor, realizando peticiones cada minuto al router. Por defecto el sistema operativo CentOS 6 viene equipado con una serie de programas que implementan este protocolo y que permiten consultar a cualquier agente accesible. Estos programas son: snmpwalk y snmpget. El primero de ellos es utilizado para recuperar todo un sub-árbol del MIB, mientras que el segundo programa pretende recuperar entidades aisladas y concretas. Ambos comandos permiten al usuario elegir la versión del protocolo a utilizar así como introducir las credenciales necesarias para poder acceder al agente, entre otras opciones. De esta forma únicamente se debe de configurar el router para que trabaje con el protocolo.

En el siguiente apartado se detallan los pasos necesarios que se han seguido para configurar correctamente el protocolo SNMPv3 en el router.

5.1.1. SNMPv3 en el router

El router Cisco RV320 [21] proporciona una interfaz web donde es posible realizar tareas de gestión y configuración de una forma más fácil e intuitiva que mediante línea de comandos. Una vez se autentica el usuario, es posible acceder al menú principal. Una vez aquí ya es posible configurar el protocolo, para ello en primer lugar se accede al submenú 'Administración del Sistema' y se selecciona la opción SNMP.

En la Imagen 10 se puede observar la situación que se plantea a partir de este punto.

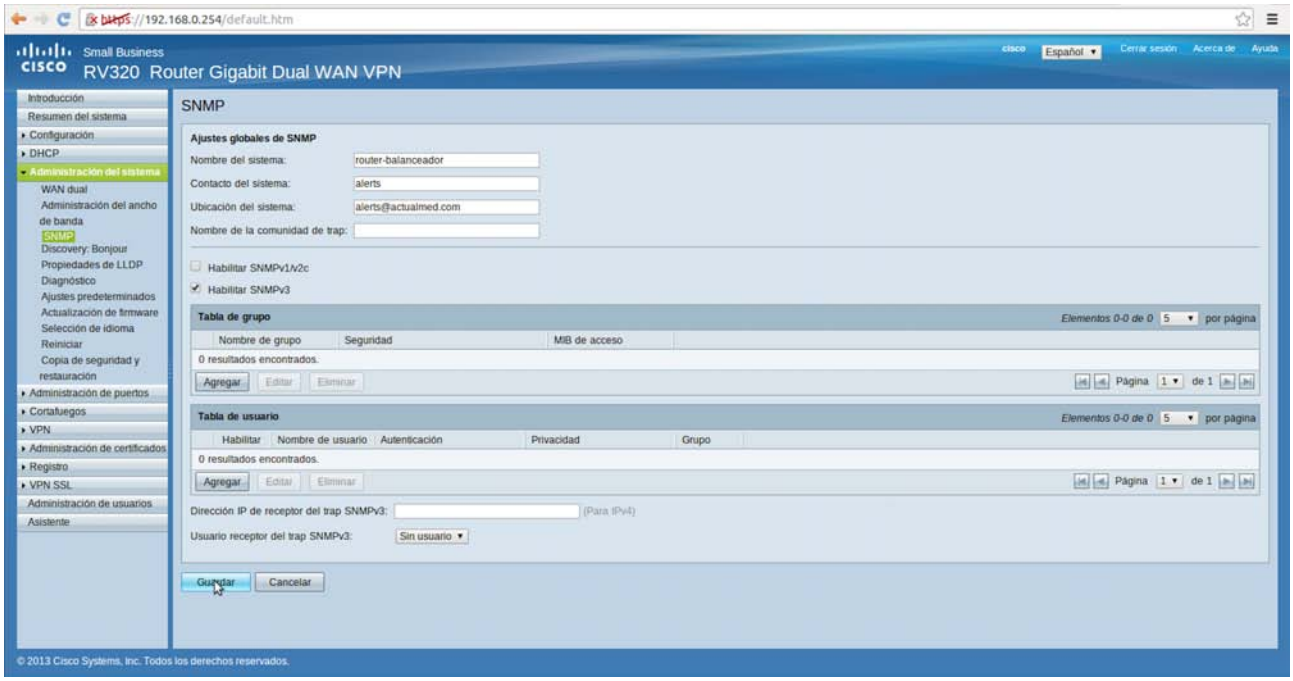

Imagen 10: Interfaz web de configuración del protocolo SNMP.

El siguiente paso consiste en introducir ciertos datos básicos como el nombre del sistema, un nombre del administrador de red al que se le deben notificar las novedades sobre el dispositivo y ubicación del sistema, es decir, cualquier información de contacto sobre el administrador de red. Puede tratarse de una dirección de correo electrónico, un número de teléfono, o un número de radiolocalizador.

Se debe marcar la casilla ‘Habilitar SNMPv3’, se guarda y se pulsa en el botón ‘Agregar’ de la Tabla de grupo. En la Imagen 11 se puede observar la nueva situación que se plantea.

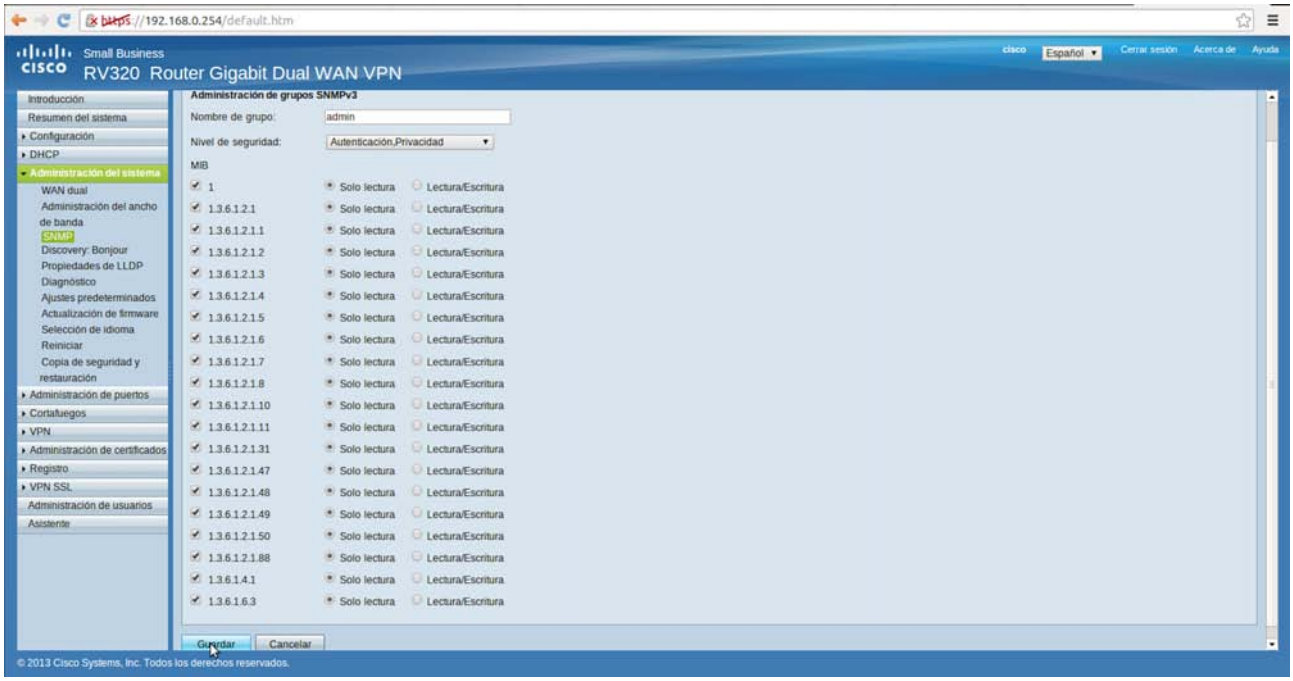

Imagen 11: Definición de la comunidad, su nivel de seguridad y permisos.

Seguidamente se procede a introducir el nombre del grupo o comunidad, el nivel de seguridad y a qué sub-árboles de la base de datos MIB del router se le da acceso y con qué permisos. En este caso se decide darle acceso a la nueva comunidad a todo el MIB con permiso de lectura únicamente. Se procede a guardar la nueva configuración y volver al menú anterior. Es momento de crear el nuevo usuario por lo que se debe pulsar en el botón ‘Agregar’ de la Tabla de usuario. En la Imagen 12 se puede observar la nueva situación que se plantea.

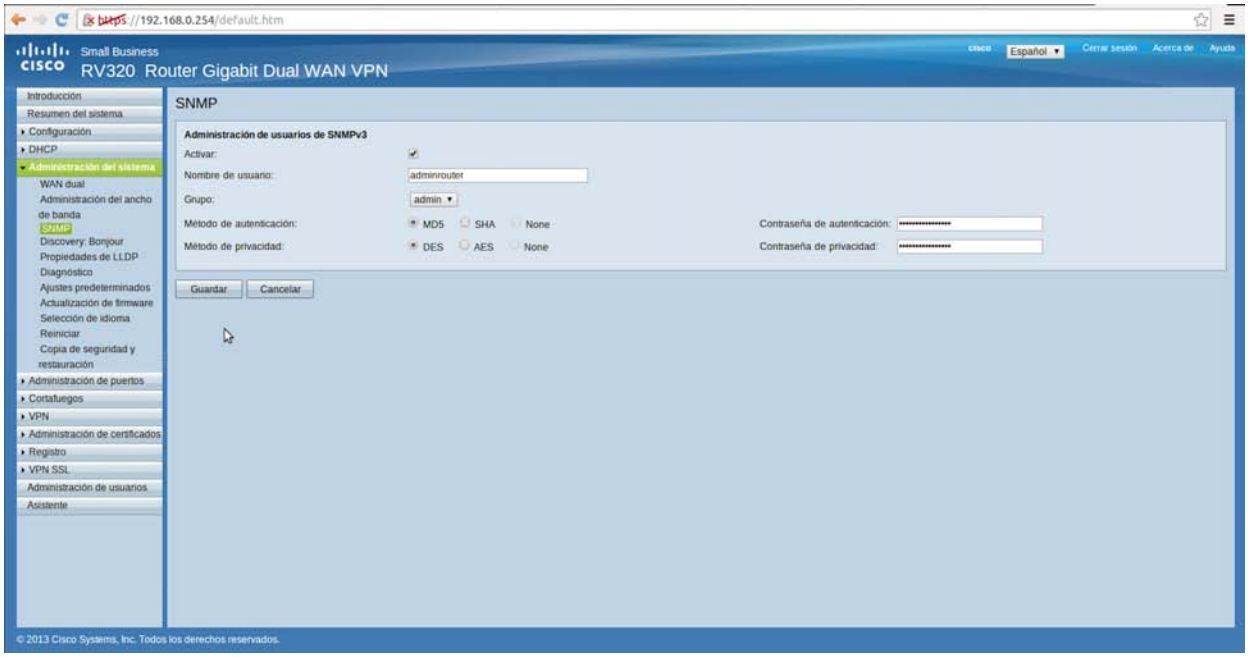

Imagen 12: Definición del usuario, su contraseña y grupo al que pertenece.

A continuación se procede a la definición del usuario, añadiéndole un nombre y asignándole el grupo anteriormente creado. Antes de guardar se debe de introducir las contraseñas de autenticación y de privacidad que posteriormente se solicitan para el acceso remoto.

Y prácticamente esto es todo, en este momento simplemente se debe de volver al menú principal de configuración de SNMP, comprobar que todo es correcto y guardar la configuración. Ya se puede salir de la interfaz web y realizar las primeras pruebas de conexión.

En la Imagen 13 se observa cómo queda la configuración de SNMP:

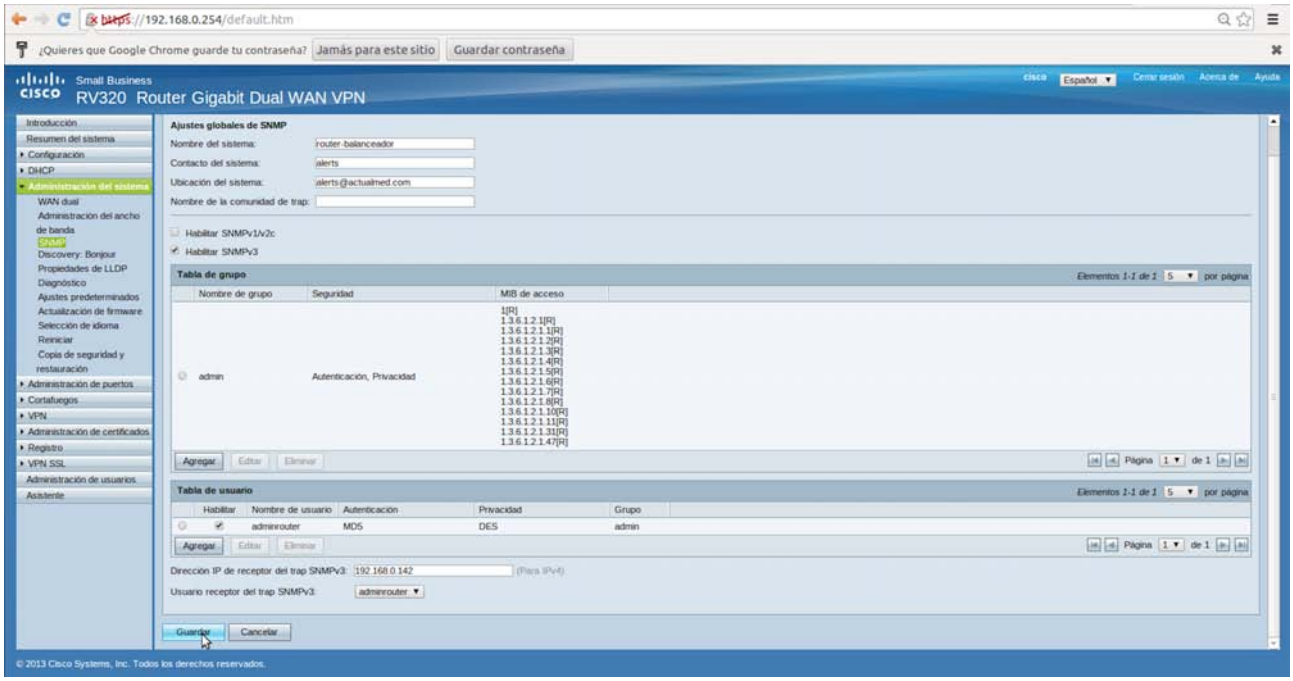

Imagen 13: Estado final del proceso una vez definida la comunidad y los usuarios.

5.2. Restricciones de seguridad

Se pretende dotar de ciertas restricciones de seguridad para el acceso desde el navegador web a los distintos directorios que conforman el proyecto. De este modo se quiere evitar que se liste el contenido de los directorios y mostrar páginas de error personalizadas en caso de intento de acceso mediante el navegador web.

Para conseguirlo el primer paso es configurar Apache para que habilite el fichero .htaccess en el proyecto web. Se tiene que editar el fichero de configuración etc/ httpd/conf/httpd.conf añadiendo la línea *AllowOverride All* sobre el directorio /var/www/html tal cual se puede observar en la Imagen 14, debiendo reiniciar Apache a continuación para que se aplique correctamente la nueva configuración:

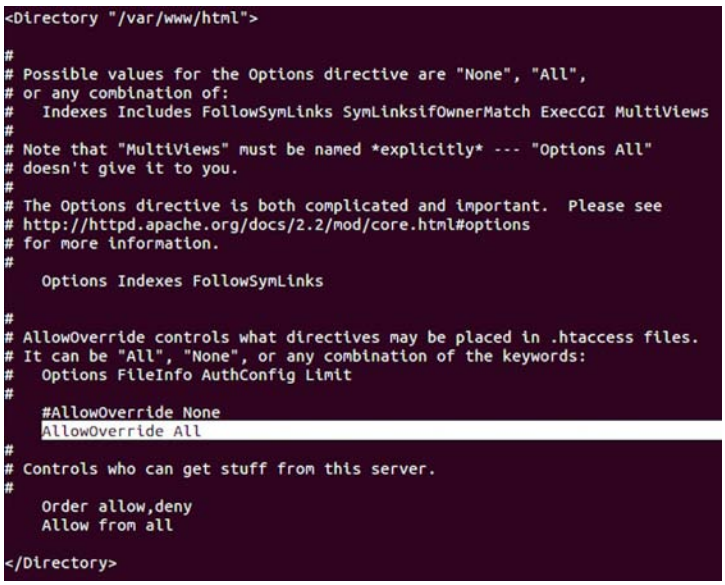

Imagen 14: Añadida la línea *AllowOverride All* que habilita el uso del fichero .htaccess.

Una vez realizados estos pasos es momento de crear el fichero .htaccess sobre /var/www/html/ (directorio donde cuelga todo el proyecto web) y añadir las directivas de seguridad que se crean convenientes. En la Imagen 15 se puede observar el contenido de este fichero:

```
php_flag safe_mode on
php_flag expose_php off
php_flag display_errors off

#Páginas de error personalizadas que se deben de mostrar en caso de error 403 y 404
#sobre /var/www/error

ErrorDocument 403 /error/error403.html
ErrorDocument 404 /error/error404.html

#Inpide listar el contenido de los directorios que forman el proyecto
Options All -Indexes
```

Imagen 15: Contenido del fichero .htaccess.

5.2. Detalles de implementación

En este apartado se pretende describir los detalles de implementación más relevantes de cada uno de los módulos de software en el que se divide la aplicación.

5.2.1. Implementación del modelo

Como se ha mencionado anteriormente, el sistema esta diseñado internamente basándose en el patrón de diseño MVC.

El modelo esta constituido por la base de datos ‘crond’, compuesta por un conjunto de tablas que almacenan datos que van a ser utilizados para construir gráficas. Estas tablas son actualizadas periódicamente por la acción de una serie de scripts que trabajan en

modo por lotes o *batch*. Algunos de ellos son los encargados de establecer conexión con el dispositivo de red, base de datos o página web correspondiente y solicitar y/o procesar los datos necesarios para realizar correctamente la tarea de monitorización.

Se puede hacer una clasificación de estos programas según la acción que realicen sobre la base de datos:

- Insertar un nuevo registro en la tabla correspondiente.
- Vaciar una tabla.
- Insertar un nuevo registro en una tabla y vaciar otra.

Todos los ficheros están escritos en PHP [22] y se encuentran ubicados físicamente en el directorio /root/cron/ del servidor de gráficas, véase la Imagen 16.

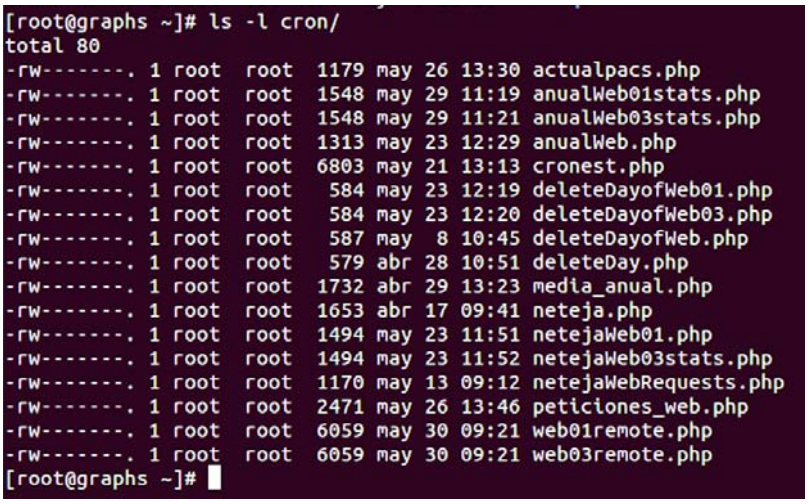

Imagen 16: Ficheros PHP del modelo.

Por cada uno de ellos se ha generado un shell script (con extensión .sh) cuya única función es llamar a la ejecución del programa al que pertenece. Todos, sin excepción, siguen la siguiente estructura:

```
#!/bin/sh
php /root/cron/<programa.php>
```

Código 1: Estructura de los shell scripts.

Estos scripts se encuentran organizados en distintos directorios que cuelgan de /root/scripts dependiendo del grupo de tablas a las que afecta sus llamadas y, en consecuencia, de los parámetros que se quiere monitorizar. En la Imagen 17 se puede observar la estructura seguida para organizarlos.

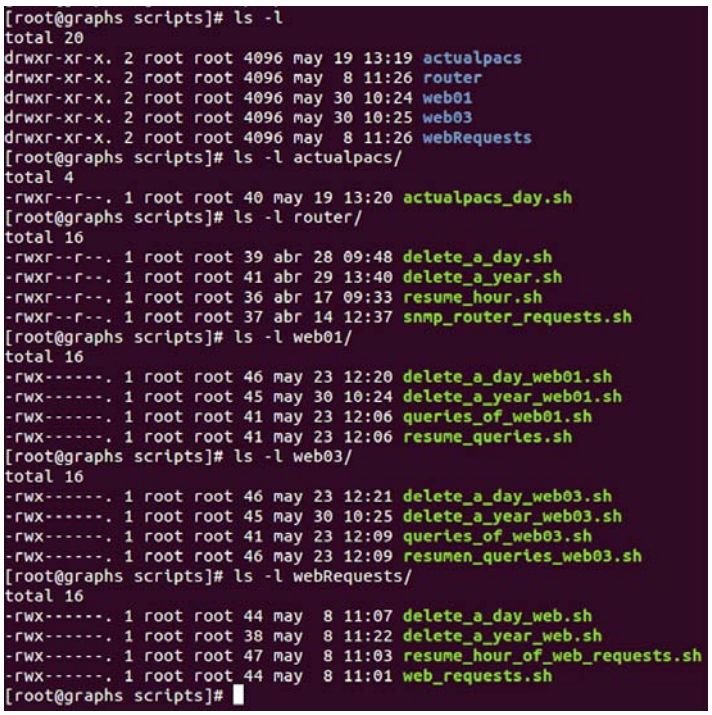

Imagen 17: Scripts del modelo organizados según la variable de monitorización que gestionan.

Y en la Tabla 8 se clasifican estos scripts en función de su acción sobre las tablas de la base de datos.

Acción\ Directorio	Insertar	Vaciar	Insertar y vaciar
router	snmp_router_requests.sh resume_hour.sh	delete_a_day.sh	delete_a_year.sh
webRequest	web_requests.sh resume_hour_of_web_requests.sh	delete_a_day_web.sh	delete_a_year_web.sh
actualpacs	actualpacs_day.sh	-	-
web01	queries_of_web01.sh resume_queries_web01.sh	delete_a_day_web01.sh	delete_a_year_web01.sh
web03	queries_of_web03.sh resume_queries_web03.sh	delete_a_day_web03.sh	delete_a_year_web03.sh

Tabla 8: Clasificación de los scripts en función de su acción sobre las tablas de la base de datos.

A continuación se detalla cada grupo de scripts según la clasificación propuesta anteriormente.

Scripts de inserción de nuevos registros

Son los responsables de mantener actualizadas las tablas de la base de datos. Por lo tanto, debe existir al menos un script de este tipo para cada parámetro que se desea monitorizar.

Se van a dividir en dos grupos, por un lado el conjunto formado por:

- snmp_router_requests.sh que llama a la ejecución de cronest.php

- web_requests.sh que llama a la ejecución de peticiones_web.php
- actualpacs_day.sh que llama a la ejecución de actualpacs.php
- queries_of_web01.sh que llama a la ejecución de web01remote.php
- queries_of_web03.sh que llama a la ejecución de web03remote.php

Y por otro lado el conjunto ‘resume’ formado por:

- resume_hour.sh que llama a la ejecución de neteja.php
- resume_hour_of_web_requests.sh que llama a la ejecución de netejaWebRequests.php
- resume_queries_web01.sh que llama a la ejecución de netejaWeb01.php
- resume_queries_web03.sh que llama a la ejecución de netejaWeb03.php

El primer grupo de scripts realizan una serie de tareas muy similares, son las siguientes:

- Conectan a la base de datos ‘crond’ como usuario ‘root’.
- Recuperan la información de monitorización correspondiente.
- En los casos oportunos deben de procesar la información obtenida para quedarse únicamente con los datos de interés.
- Introducen los datos en las tablas correspondientes. Si las tablas no está vacías deben recuperar el último registro para calcular las diferencias necesarias y obtener los valores reales por unidad de tiempo.
- Cierra conexión a la base de datos.

En la Tabla 9 se puede observar a qué tablas afecta en particular cada script y los datos que recupera:

Scripts	Tablas actualizadas	Datos que obtiene	Periodo de Actualización
snmp_router_requests.sh	Bytes	Total de paquetes RX y TX que pasan por las interfaces eth0, eth1 y eth2 del router en un momento dado.	Cada minuto
web_requests.sh	PeticionesWeb	Total de accesos a páginas gestionadas por ActualWeb en un momento dado.	Cada minuto

Scripts	Tablas actualizadas	Datos que obtiene	Periodo de Actualización
actualpacs_day.sh	ActualPacs	Total de estudios, estudios/dia, total de bytes y bytes/dia.	Cada día
queries_of_web01.sh	web01stats	Total de consultas Select, Insert, Delete y Update contabilizadas en un momento dado.	Cada minuto
queries_of_web03.sh	web03stats	Total de consultas Select, Insert, Delete y Update contabilizadas en un momento dado.	Cada minuto

Tabla 9: Listado de scripts junto con las tablas a las que actualizan y los datos que obtienen.

Para finalizar con este grupo de scripts en los Códigos 2, 3 y 4 se puede observar ejemplos de código PHP necesario para obtener los datos de las fuentes externas.

```
snmp3_get(string $host, string $sec_name , string $sec_level , string $auth_protocol, string $auth_passphrase, string $priv_protocol, string $priv_passphrase, string $object_id);

$bytes_salida = snmp3_get('192.168.0.254', 'adminrouter', 'authPriv', 'MD5', '**********', 'DES', '**********', 'IF-MIB::ifOutOctets.6');
$bytes_entrada = snmp3_get('192.168.0.254', 'adminrouter', 'authPriv', 'MD5', '**********', 'DES', '**********', 'IF-MIB::ifInOctets.6');
```

Código 2: Función de PHP snmp3_get() utilizada para obtener los octetos de entrada y de salida de cada interfaz en cronest.php.

```
$doc = new DOMDocument();
$doc->loadHTMLFile("http://web01.serverwhite.com/status");

$doc = new DOMDocument();
$doc->loadHTMLFile("http://192.168.0.50:81/stats/");
…
```

Código 3: Clase de PHP DOMDocument() utilizada para descargar estas páginas web y procesarlas para obtener los datos de monitorización necesarios en peticiones_web.php y actualpacs.php.

```
$link = mysql_connect( 'web01.serverwhite.com', '***usuario***', '***password***' );
if ( !$link ) {
  die( 'Could not connect: ' . mysql_error() );
}

$updates= "SHOW GLOBAL STATUS like 'Com_update'";
$result = mysql_query( $updates );
…
```

Código 4: Código PHP utilizado en web01remote.php para conectarse a la base de datos remota y ejecutar consultas sobre ella.

El siguiente grupo de scripts tratan de controlar el crecimiento de las tablas actualizadas cada minuto. Actúan por cada hora de la siguiente manera:

- Conectan a la base de datos ‘crond’ como usuario ‘root’.
- Mediante sentencias SQL calculan la media de los datos relevantes añadidos durante la última hora.
- Introducen estos valores promedio en las tablas resumen correspondientes.
- Cierra conexión a la base de datos.

En la Tabla 10 se puede observar a qué tablas afecta en particular cada script.

Scripts	Tabla resumida	Tabla actualizada
resume_hour.sh	Bytes	resumen
resume_hour_of_web_requests.sh	PeticionesWeb	ResumenWeb
resume_queries_web01.sh	web01stats	resumenWeb01
resume_queries_web03.sh	web03stats	resumenWeb03

Tabla 10: Listado de scripts junto con las tablas a las que resumen y actualizan.

Scripts que vacían tablas

Estos scripts son ejecutados cada día a las 0:00h por el servidor y evitan la sobrecarga de aquellas tablas que disponen de un nuevo dato cada minuto. Como queda reflejado en la Tabla 4, los scripts son los siguientes:

- delete_a_day.sh que llama a la ejecución de deleteDay.php
- delete_a_day_web.sh que llama a la ejecución de deleteDayofWeb.php
- delete_a_day_web01.sh que llama a la ejecución de deleteDayofWeb01.php
- delete_a_day_web03.sh que llama a la ejecución de deleteDayofWeb03.php

En la Tabla 11 se puede observar a qué tablas afecta en particular el vaciado de cada script.

Scripts	Tabla vaciada
delete_a_day.sh	Bytes
delete_a_day_web.sh	PeticionesWeb
delete_a_day_web01.sh	web01stats
delete_a_day_web03.sh	web03stats

Tabla 11: Listado de scripts junto con las tablas a las que vacían.

Scripts que vacían tablas y añaden registros

Estos scripts tratan de controlar el crecimiento de las tablas resumen actualizadas cada hora. Son los siguientes:

- delete_a_year.sh que llama a la ejecución de media_anual.php
- delete_a_year_web.sh que llama a la ejecución de anualWeb.php
- delete_a_year_web01.sh que llama a la ejecución de anualWeb01stats.php
- delete_a_year_web03.sh que llama a la ejecución de anualWeb03stats.php

Son ejecutados cada primer día del año y actúan de la siguiente manera:

- Conectan a la base de datos ‘crond’ como usuario ‘root’.
- Mediante sentencias SQL calculan la media de los datos relevantes añadidos durante el último año.
- Introducen los datos en las tablas anuales correspondientes.
- Vacían la tabla resumen consultada.
- Cierra conexión a la base de datos.

En la Tabla 12 se puede observar a qué tablas afecta en particular cada script.

Scripts	Tabla resumida y vaciada	Tabla actualizada
delete_a_year.sh	resumen	anual
delete_a_year_web.sh	ResumenWeb	anualWeb
delete_a_year_web01.sh	resumenWeb01	anualWeb01
delete_a_year_web03.sh	resumenWeb03	anualWeb03

Tabla 12: Listado de scripts junto con las tablas a las que vacían y actualizan.

Fichero /etc/crontab

Para que los scripts sean ejecutados automáticamente se deben de añadir correctamente las tareas en el fichero de configuración de linux /etc/crontab.

Cada tarea debe de especificar el periodo de ejecución, el usuario que tiene permiso para hacerlo así como el script concreto a ejecutar.

Para acabar con este apartado el Código 5 muestra parte del fichero /etc/crontab real configurado:

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# For details see man 4 crontabs

# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name command to be executed


#SCRIPTS PARA LA MONITORIZACIÓN DEL ROUTER

0 0 1 1 * root /root/scripts/router/./delete_a_year.sh
0 0 * * * root /root/scripts/router/./delete_a_day.sh
59 * * * * root /root/scripts/router/./resume_hour.sh
*/1 * * * * root /root/scripts/router/./snmp_router_requests.sh
…
```

Código 5: Parte del fichero /etc/crontab del servidor de gráficas.

5.2.2. Implementación del controlador

El controlador está formado por una serie de ficheros PHP encargados de obtener los datos requeridos del modelo, darles el formato oportuno y devolverlos a la vista. Estos ficheros PHP son llamados desde las funciones JavaScript que generan las gráficas con el objetivo de proveerlas de los datos necesarios.

Se encuentran ubicados físicamente dentro del proyecto web, en el directorio /var/www/html/php y al igual que los scripts del modelo se encuentran organizados en subdirectorios en función de qué datos recuperen. En la Tabla 13 se muestra esta organización en directorios y los ficheros que contiene cada uno de ellos.

/var/www/html/php/	Ficheros
router/	routerData.php - updating_ethx.php - updating_ethx_hours.php - xhours.php
webRequests/	webData.php - updating_web_requests.php - updating_web_requests_hours.php - xhoursWeb.php
web01/	web01Data.php - updating_web01.php - updating_web01_hours.php - xhoursWeb01.php - web01DataType.php
web03/	web03Data.php - updating_web03.php - updating_web03_hours.php - xhoursWeb03.php - web03DataType.php
actualpacs/	actualpacsData.php
./	multipanel.php

Tabla 13: Ficheros php que forman el Controlador organizados en subdirectorios.

Podemos clasificarlos en función de la cantidad de datos que devuelven a las funciones JavaScript:

- Scripts que devuelven todos los datos de la tabla correspondiente.
- Scripts que devuelven el último registro de la tabla correspondiente.

A continuación se detalla cada grupo de ficheros PHP según esta clasificación.

Scripts que devuelven todos los datos de la tabla

Estos scripts se encargan en primer lugar de recuperar todo el contenido de la tabla en el momento en el que se ejecutan y finalmente formatean el resultado a JSON. Son básicos para la construcción de las gráficas puesto que las funciones JavaScript que las generan necesitan de unos datos de partida para poder operar correctamente. La ejecución de estos programas permite inyectar a estas funciones JavaScript los datos necesarios para generar gráficas actualizadas en todo momento.

Se pretende generar al menos dos gráficas por cada parámetro de monitorización: una con mayor resolución y otra con menor resolución, es por ello que se necesitarán dos scripts diferentes que inyecten a cada función javascript el conjunto de datos concreto. De esta forma existen dos scripts por cada parámetro a monitorizar (excepto para ActualPacs): uno que recupera todos los datos de la tabla principal (*Data.php) y otro que lo hace de la tabla de resumen (xhours*.php). En la Tabla 14 se resume este comportamiento.

Ficheros	SELECT * FROM
routerData.php	Bytes
xhours.php	resumen
webData.php	PeticionesWeb
xhoursWeb.php	ResumenWeb
web01Data.php	web01stats
xhoursWeb01.php	resumenWeb01
web03Data.php	web03stats
xhoursWeb03.php	resumenWeb03
actualpacsData.php	ActualPacs

Tabla 14: Clasificación de los ficheros php que cargan todos los datos en función de la tabla en la que operan.

La Imagen 18 muestra un posible resultado en formato JSON de la ejecución del script webData.php. Se trata de un resultado con una resolución de un dato por minuto.

```
[{“momento”:2014-05-30 00:00:01, “pm-web-accesses”:351830, “gap-totalaccesses”:NULL},
{“momento”:2014-05-30 00:01:06, “pm-web-accesses”:352961, “gap-totalaccesses”:1131},
{“momento”:2014-05-30 00:02:02, “pm-web-accesses”:353701, “gap-totalaccesses”:740}]
```


Imagen 18 : Posible resultado de ejecutar webData.php.

La Imagen 19 muestra un posible resultado en formato JSON de la ejecución del script xhoursWeb01.php. Se trata de un resultado con una resolución de un dato por hora.


```
[{“momento”:2014-05-23 12:59:01, “gapSelects”:18853, “gapInserts”:2417, “gapUpdates”:
4101, “gapDeletes”:51}, {“momento”:2014-05-23 13:59:01, “gapSelects”:16693.084,
“gapInserts”:2672.8305, “gapUpdates”:4191.7288, “gapDeletes”:46.6102}, {“momento”:
2014-05-23 14:59:01, “gapSelects”:7413.4576, “gapInserts”:2583.7966, “gapUpdates”:
4059.22, “gapDeletes”:50.084}, {“momento”:2014-05-23 15:59:01, “gapSelects”:6430.2542,
“gapInserts”:2486.0678, “gapUpdates”:3997.0339, “gapDeletes”:42.7797}]
```


Imagen 19: Posible resultado de ejecutar xhoursWeb01.php.

Scripts que devuelven el último registro de la tabla

Estos scripts se encargan de obtener el último registro insertado en la tabla y formatean el resultado a JSON. Son vitales para la configuración de gráficas que se actualizan en tiempo real puesto que sirven para inyectar el valor más actual almacenado en la base de datos siempre que no se produzca una carga completa realizando una llamada al grupo de scripts anteriores.

Se pretende actualizar de manera automática el grupo de gráficas definido en el anterior apartado, por tanto se necesitará de un script que recupere el último registro de la tabla principal y otro que lo haga de la tabla resumen. En la Tabla 15 se resume este comportamiento.

Ficheros	SELECT * FROM …. ORDER BY 1 DESC LIMIT 1
updating_ethx.php	Bytes
updating_ethx_hours.php	resumen
updating_web_requests.php	PeticionesWeb
updating_web_requests_hours.php	ResumenWeb
updating_web01.php	web01stats
updating_web01_hours.php	resumenWeb01
web01DataType.php	resumenWeb01
updating_web03.php	web03stats
updating_web03_hours.php	resumenWeb03
web03DataType.php	resumenWeb03

Tabla 15: Lista de Ficheros php que cargan el último registro en función de la tabla en la que operan.

Para acabar con este apartado, comentar que los scripts web01DataType.php y web03DataType.php recuperan la última fila de resumenWeb01 y resumenWeb03 respectivamente pero formatean el resultado a JSON siguiendo una estructura distinta a la que siguen el resto de scripts. Esto es porque van a inyectar los datos necesarios a gráficas de tipo tarta que siguen una estructura diferente a las que se han utilizado hasta el momento (lineales).

La Imagen 20 muestra un posible resultado en formato JSON de la ejecución del script web01DataType.php. Se trata de un resultado con una resolución de un dato por hora:

```
[{“query”:“Select”, “value”:6430.2542}, {“query”:“Insert”, “value”:2486.0678},
{“query”:“Update”, “value”:3997.0339}, {“query”:“Delete”, “value”:42.7797}]
```

Imagen 20: Posible resultado de ejecutar web01DataType.php.

Script multipanel.php

Cuando el usuario hace click sobre la opción ‘Pantalla Completa’ del menú, se carga un formulario a modo de checkbox con todas las gráficas desarrolladas. El usuario puede marcar las que necesite y hacer click al botón ‘Generar’. El resultado de este formulario es comprobado por el programa multipanel.php anotándose qué gráficas han sido seleccionadas para ser visualizadas. Finalmente, con esta información es capaz de generar dinámicamente una página HTML con las gráficas que se han solicitado. La Imagen 21 muestra un posible resultado tras la ejecución del mismo fichero.

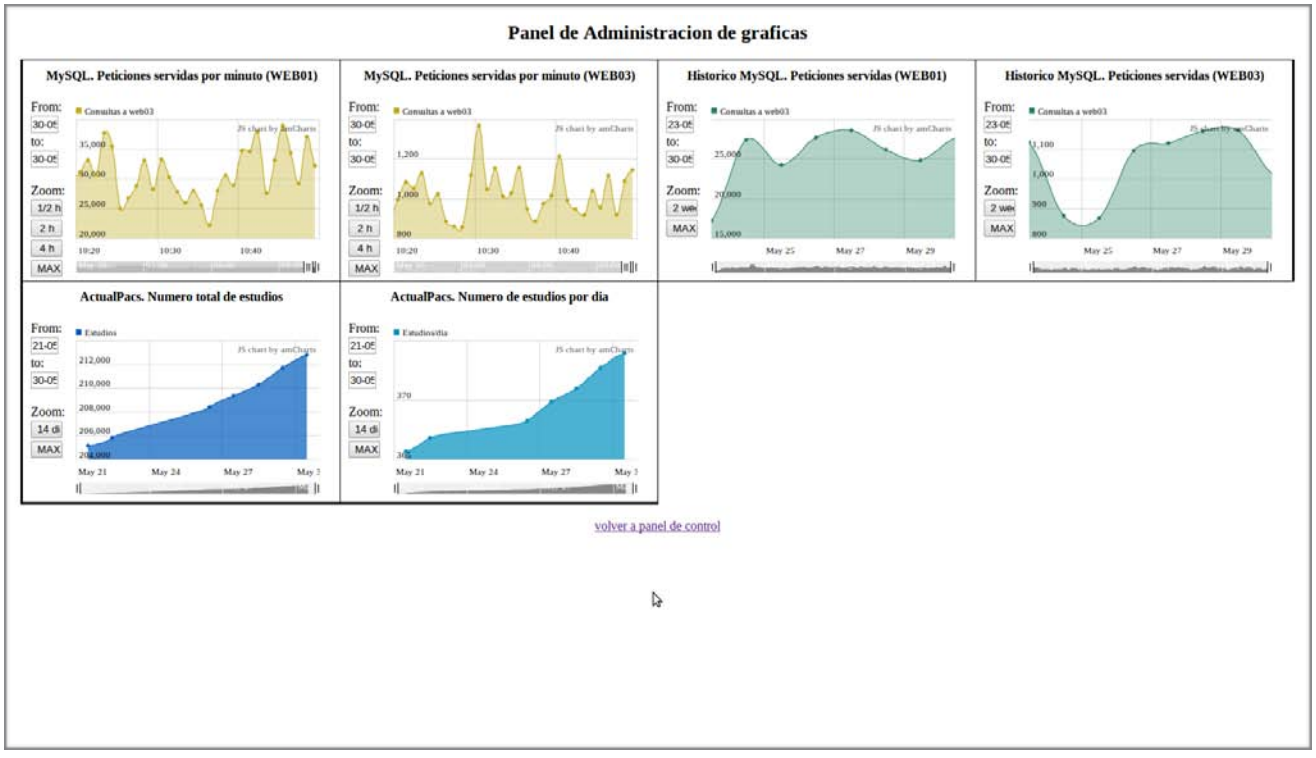

Imagen 21: Página web generada como resultado de la ejecución de multipanel.php.

5.2.3. Desarrollo de las gráficas

El desarrollo de las gráficas ha sido un proceso sistemático desde el momento en el que se conocen gran parte de los atributos y métodos que gestiona la librería amCharts. Básicamente son funciones que son llamadas mediante jQuery en el momento en el que el usuario accede a una sección diferente de la interfaz o hace click sobre algún botón en concreto. Todas siguen el mismo esquema:

- Cargan los datos del modelo: Se llama al controlador por medio de una función propia de la librería y se obtiene un objeto JSON con todos los datos disponibles de una tabla.

- Configuración específica: Para cada gráfica se indica que datos de los obtenidos van a formar parte del eje X y cuales del eje Y. Se configuran aspectos propios de cada gráfica como el tipo (lineal, barras, tarta…), sus colores, la resolución que ofrecerá al usuario, etc.

- Cargar el último dato: Se construye una función capaz de llamarse cada periodo de tiempo indicado y que mediante el controlador recupera el último registro insertado en la tabla y lo añade a su lista de datos. La gráfica de esta forma queda actualizada. Esto es así porque se hace coincidir el periodo de tiempo en el que se recupera un dato de la fuente externa y el tiempo que se llama a esta función.

Capítulo 6: Pruebas y documentación

En este capítulo se describen el grueso de pruebas realizadas durante el desarrollo del proyecto así como la documentación proporcionada a la organización para el posterior mantenimiento o ampliación del sistema de monitorización.

6.1. Pruebas

Se ha desarrollado una fase de pruebas por cada módulo o capa de software que forma el sistema de monitorización. Para ello se ha dispuesto de un ordenador local donde se ha intentado reproducir un entorno de producción real y de esta forma comprobar el correcto funcionamiento de cada módulo. Una vez verificada cada parte se procede a implantar versiones beta en el servidor, donde se podrían detectar otros problemas no planeados con anterioridad.

A continuación se presentan, formando diversos apartados, las pruebas más relevantes de cada uno de los módulos de software de la aplicación.

6.1.1. Pruebas en el modelo

Para desarrollar el modelo se ha utilizado, entre otras herramientas, el software phpmyadmin para la creación de la base de datos y sus respectivas tablas. Mediante esta herramienta gráfica se ha podido ejecutar, a modo de prueba y error, las distintas sentencias SQL necesarias para insertar registros, resumir, borrar y exportar tablas. Una vez verificada las sentencias se incorporaban directamente al código PHP de los distintos scripts que forman el modelo.

También se han realizado las pruebas de conectividad oportunas para la recepción de datos con SNMPv3 desde el router. Para ello se ha utilizado los programas de Linux comentados en el capítulo 4: snmpwalk y snmpget. En la Imagen 22 se muestra un ejemplo.

Imagen 22: Pruebas de conectividad y recepción de datos desde el router mediante SNMPv3.

Además, mientras se desarrolla el código de los scripts se intenta ir paso a paso y comprobar mediante la función de PHP ‘echo’ que se van obteniendo los resultados deseados.

Finalmente otra de las pruebas realizadas es verificar durante un periodo de tiempo relativamente corto que todos los scripts se ejecutan automáticamente siguiendo el periodo de tiempo indicado, visualizando gráficamente los cambios desde la interfaz que proporciona gráficamente phpmyadmin. Si todo funciona según lo previsto se vuelca la misma configuración y ficheros en el servidor y se comprueba para periodos más amplios de tiempo.

6.1.2. Pruebas en el controlador

Las pruebas que se han realizado para el controlador simplemente se han limitado a comprobar, mediante la ejecución de los ficheros PHP y posterior visualización de los resultados en un navegador, que se obtenían y se formateaban correctamente los datos de las tablas a JSON.

6.1.3. Pruebas en la vista

Las pruebas sobre la vista se han basado principalmente en el análisis de la conducta de las gráficas, y han ayudado en mayor medida a detectar problemas en la gestión de la información del modelo, sobretodo al observar comportamientos completamente anómalos en las gráficas.

Una vez construidas las primeras gráficas se intenta probar el comportamiento de las mismas ante una carga de datos bastante grande procedente de la base de datos. Se observa que se pierden prestaciones: las gráficas tardan un tiempo excesivo en cargarse y se merma la interactividad: las gráficas ya no responden tan bien o simplemente no lo hacen. Se genera en este momento un problema importante ya que los datos en el entorno real no van a parar de crecer y se debe poder visualizar las gráficas sin ningún problema. Se vuelve a realizar un pequeño estudio en este instante de la estancia en prácticas y se concluye que se estaba trabajando con la librería equivocada. Como se ha comentado en el anterior capítulo del análisis, amCharts dispone de tres paquetes distintos que satisfacen necesidades dispares. En un primer momento se estaba utilizando la librería básica CHARTS destinada a un uso a nivel de usuario para representaciones más bien pequeñas. Es preciso, de esta forma, el cambio y el uso de la librería STOCK CHARTS, optimizada para la representación de grandes cantidades de datos que gestionan las empresas. A partir de aquí se observan mejorías importantes en cuanto a la gestión de grandes cantidades de datos, así como nueva funcionalidad interesante como la elección por parte del usuario de distintas resoluciones de la gráfica.

Continuando con el análisis del comportamiento de las gráficas se ha observado que en determinados instantes puntuales se producían picos sin ningún sentido lógico en ciertas gráficas como la del tráfico del router y la de los accesos a las páginas web. Se realiza un pequeño estudio, y se concluye que estas variables que se recuperan, al comportarse como contadores, cuando llegan al último valor que es posible representar vuelven a

medir desde 0, de forma que la diferencia se dispara y por ello se producen estos picos. La solución es tratar estas excepciones correctamente en los scripts del modelo correspondientes. En la Imagen 23 se puede observar tal efecto sobre la interfaz eth2 del router:

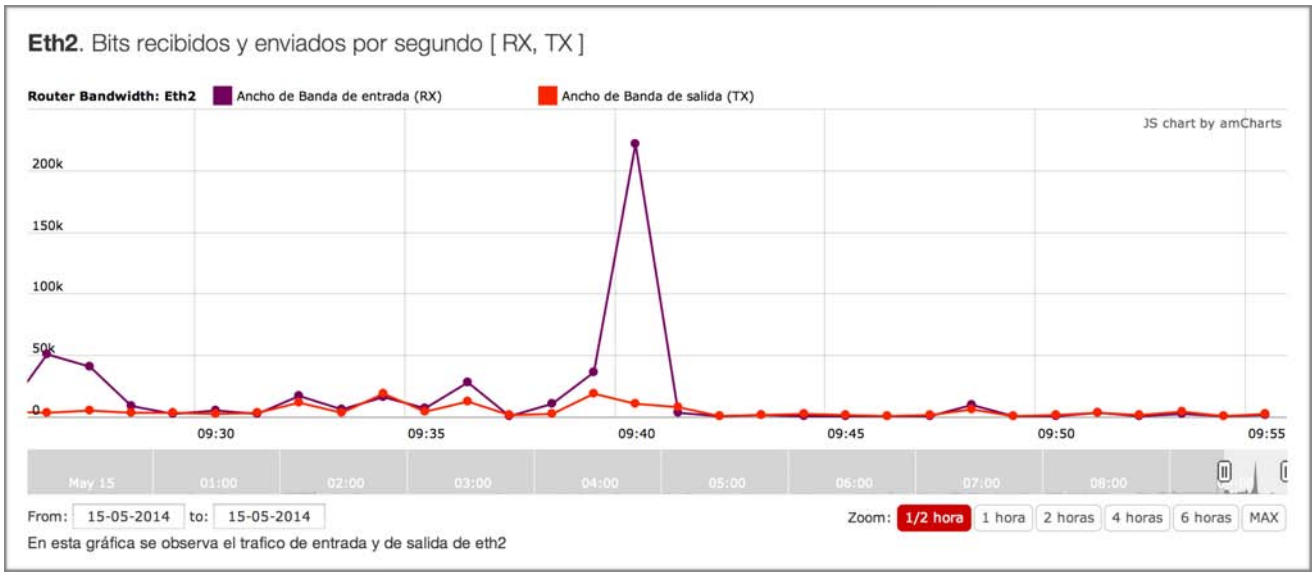

Imagen 23: Pico puntual anómalo en el estado del trafico que pasa por eth2.

Finalmente, junto con mi supervisor, se detecta que las gráficas que muestran el estado del tráfico que pasa por las interfaces eth0 y eth2 del router no parecen tener sentido. La gráfica de eth0 muestra siempre la inversa de eth1 y la gráfica de eth2 contiene un tráfico muy pequeño. Se solicita ayuda enviando un mensaje en inglés al foro de soporte de Cisco pero nunca se llega a recibir respuesta. Finalmente se llega a la decisión consensuada de eliminar las gráficas correspondientes a las interfaces eth0 y eth2 y lanzar una serie de pruebas sobre eth1 para confirmar que es la única gráfica que muestra datos con sentido: se lanza la descarga de 4 imágenes .iso de 700 MB a la vez y se observa como el tráfico de entrada se dispara conforme lo habitual, se procede a subir a dropbox 2 ficheros de 200 MB a la vez y se observa como el tráfico de salida se dispara conforme a lo habitual. En la Imagen 24 se puede observar este incremento del tráfico de bajada.

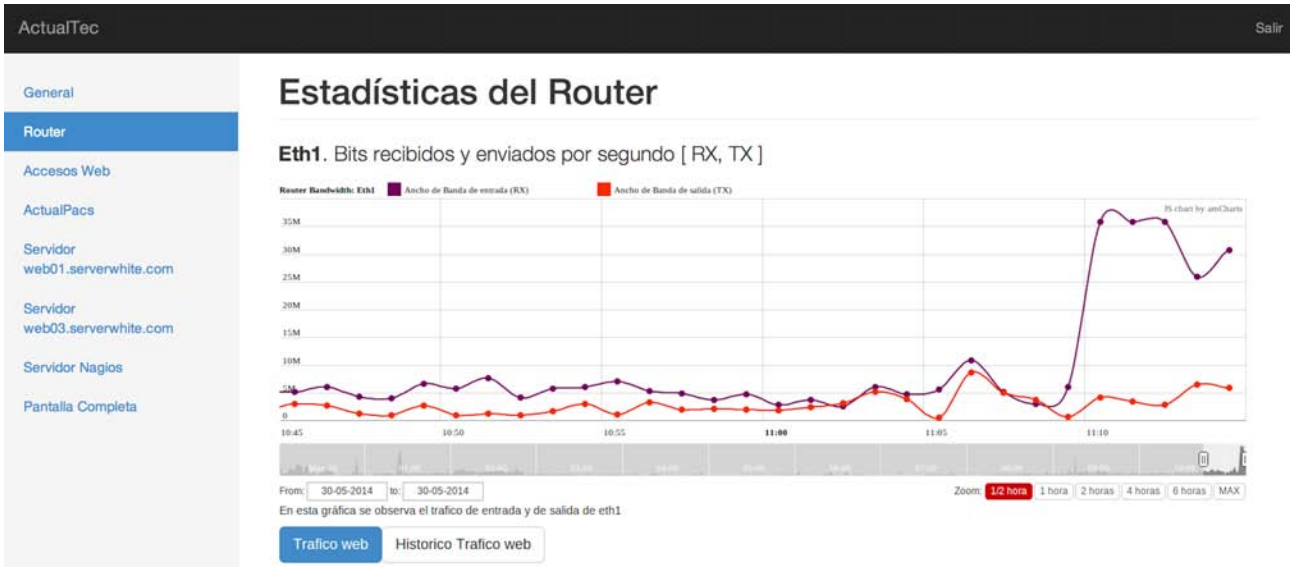

Imagen 24: Incremento de tráfico de bajada durante la descarga de 4 ficheros de más de 700 MB.

6.2. Documentación

A modo de documentación se llega a un acuerdo con el supervisor de la estancia en prácticas para entregar, una vez concluida y revisada, la memoria del TFG así como compartir un documento de Google Drive donde se irán añadiendo y explicando las peculiaridades de implementación y configuración del sistema con vistas a reducir el tiempo para mantenimiento o ampliación de la funcionalidad del sistema.

Capítulo 7: Conclusiones

Gracias al esfuerzo depositado durante todo el desarrollo del proyecto, ha sido posible la creación de una aplicación web capaz de hacer frente a la monitorización de varios servicios y dispositivos de red de la empresa.

El modelo esta automatizado por la acción de una serie de scripts que lo gestionan a corto y largo plazo intentando mantener siempre datos representativos y evitando el uso desmesurado de memoria en disco.

El rendimiento general de la parte gráfica es bastante bueno: la navegación por las distintas secciones de la pagina web se puede considerar fluida y las gráficas cargan rápido y ofrecen un feedback o interactividad asequible para el usuario: se muestran globos con valores concretos si se coloca el puntero sobre un punto determinado de la gráfica, se aumenta o se reduce la resolución arrastrando el puntero entre dos puntos, se activa o se desactiva distintas lineas de monitorización desde la leyenda, etc.

Sin embargo por falta de tiempo y distintos problemas de configuración no se ha podido homogeneizar la obtención de los datos desde las fuentes externas. Lo ideal hubiera sido recibir todos los datos mediante el protocolo SNMPv3 pero para ello se debe desarrollar una serie de módulos que obtengan los datos y los inserten en la base de datos MIB del dispositivo agente. Esto era una tarea complicada ya que estos datos se encontraban en servidores importantes por lo que no se me daba el permiso necesario para desarrollar, configurarlos y realizar las pruebas oportunas. A parte, el supervisor intentó configurar el firewall para permitir consultas SNMP a las maquinas web01.serverwhite.com y web03.serverwhite.com pero se tuvieron problemas y no se consiguieron resolverlos durante mi estancia en prácticas.

Como trabajo futuro se espera la creación y configuración de estos módulos en las máquinas agente que permita esta homogeneidad, así como el desarrollo de la infraestructura necesaria para la integración de cualquier otro parámetro de monitorización futuro que se necesite evaluar.

A nivel personal, ha sido un proyecto que me ha motivado puesto que ya había establecido contacto con diversas herramientas de monitorización durante la carrera y encontraba interesante poder ampliar mis conocimientos en el tema y encaminar mi perfil profesional hacia la administración y control de los sistemas, que es la parte de la informática que más me atrae.

Bibliografia

[1] Wolfgang Barth, *Nagios. System and Network Monitoring.* Open Source Press GmbH, 2006

[2] PACS: http://www.actualmed.com/blog/2010/10/20/servidor-pacs-dicom-server/ (Última Visita: 10/07/2014)

[3] SLA: http://es.wikipedia.org/wiki/Acuerdo_de_nivel_de_servicio (Última Visita: 10/07/2014)

[4] Osmius: http://www.osmius.com/ (Última Visita: 10/07/2014)

[5] PandoraFMS: http://pandorafms.com/ (Última Visita: 10/07/2014)

[6] God: http://godrb.com/ (Última Visita: 10/07/2014)

[7] Monit: http://mmonit.com/monit/ (Última Visita: 10/07/2014)

[8] Douglas Mauro, Kevin Schmidt. *Essential SNMP, 2n Edition.* O'Reilly. September 2005

[9] MVC: http://es.wikipedia.org/wiki/Modelo-vista-controlador (Última Visita: 10/07/2014)

[10] RGraph: http://www.rgraph.net/ (Última Visita: 10/07/2014)

[11] NVD3: http://nvd3.org/ (Última Visita: 10/07/2014)

[12] D3.js: http://d3js.org/ (Última Visita: 10/07/2014)

[13] ZingChart: http://www.zingchart.com/ (Última Visita: 10/07/2014)

[14] amCharts: http://www.amcharts.com/ (Última Visita: 10/07/2014)

[15] JSON: http://json.org/ (Última Visita: 10/07/2014)

[16] MySQL: http://www.mysql.com/ (Última Visita: 10/07/2014)

[17] Twitter BootStrap: http://getbootstrap.com/2.3.2/ (Última Visita: 10/07/2014)

[18] Juan Diego Gauchat. *El gran libro de HTML5, CSS3 y Javascript.* Publidisa. 2012

[19] jQuery: http://jquery.com/ (Última Visita: 10/07/2014)

[20] DOM: http://es.wikipedia.org/wiki/Document_Object_Model (Última Visita: 10/07/2014)

[21] Cisco Small Business. Router RV320/RV325 Gigabit Dual WAN VPN. *Guia de Administración*.

[22] PHP: https://php.net/manual/es/index.php (Última Visita: 10/07/2014)