



**Universitat Autònoma  
de Barcelona**

# **Monitorización remota de sistemas**

Memoria del proyecto  
de Ingeniería Técnica en  
Informática de Sistemas  
realizado por  
Francisco Ortega Belmonte  
y dirigido por  
Jordi Pons Aróztegui

Escuela Universitaria de Informática  
Sabadell, junio de 2009

El firmante, Jordi Pons Aróztegui,  
profesor de la Escuela Universitaria de Informática de la  
UAB,

**CERTIFICA:**

Que el trabajo al que corresponde la  
presente memoria  
ha sido realizado bajo su dirección  
por Francisco Ortega Belmonte  
Y para que conste firma la presente.  
Sabadell, junio de 2009

-----  
Firmado: Jordi Pons Aróztegui

El firmante, Franc Marquina Torres,  
de CCSAgresso,

**CERTIFICA:**

Que el trabajo al que corresponde la  
presente memoria  
ha sido realizado bajo su supervisión  
por Francisco Ortega Belmonte

Y para que conste firma la presente.  
Sabadell, junio de 2009

-----

Firmado: Franc Marquina Torres

El firmante, Toni Sojo,  
de CCSAgresso,

**CERTIFICA:**

Que el trabajo al que corresponde la  
presente memoria  
ha sido realizado bajo su supervisión  
por Francisco Ortega Belmonte

Y para que conste firma la presente.  
Sabadell, junio de 2009

-----  
Firmado: Toni Sojo

# Agradecimientos

Quiero agradecer a mi familia, pareja y amigos todo el soporte que me han dado en la realización del proyecto y durante toda la carrera.

A Jordi Pons, Franc Marquina y Toni Sojo por guiarme a lo largo de todo el proyecto haciendo que los problemas del día a día del mismo se viesen más fáciles de resolver.

Por último, quiero agradecer a todos los profesores de las diferentes asignaturas que he cursado por la formación que me han dado y que he podido aplicar en la resolución del proyecto.

El proyecto “Monitorización remota de sistemas” surge del convenio de colaboración entre la Escuela Universitaria de Informática de Sabadell y la empresa CCSAgresso.

CCSAgresso tenía la necesidad de mejorar los servicios de monitorización que actualmente ofrece a sus clientes y por esta razón planteó este proyecto.

El objetivo principal del proyecto es dotar a la empresa de un sistema proactivo para la detección de incidencias en las redes de sus clientes.

A lo largo de toda la memoria veremos detalladamente como se ha conseguido este objetivo y los múltiples obstáculos que se han tenido que superar.

# Índice

<b>1. Introducción .....</b>	<b>1</b>
1.1. Proyecto en empresa .....	1
1.2. Motivaciones.....	3
1.3. Objetivos.....	4
1.4. Metodología.....	5
1.5. Contenido de la memoria .....	7
 <b>2. Diagnóstico de la situación .....</b>	 <b>9</b>
2.1. ¿Qué es la monitorización de sistemas?.....	9
2.2. La monitorización en grandes empresas.....	14
2.3. La monitorización actual en CCSAgresso .....	15
2.3.1. Descripción de la situación actual.....	15
2.3.2. Conclusiones de la situación actual.....	16
 <b>3. Estudio de viabilidad .....</b>	 <b>19</b>
3.1. Análisis de requerimientos.....	19
3.1.1. Requerimientos funcionales.....	19
3.1.2. Requerimientos no funcionales.....	21
3.2. Viabilidad técnica y económica.....	22
3.2.1. Recursos.....	22

3.2.1.1. Software.....	22
3.2.1.2. Hardware .....	24
3.2.1.3. Recursos humanos.....	25
3.2.2. Análisis coste-beneficio.....	25
3.2.2.1. Coste del material .....	27
3.2.2.2. Coste del personal.....	28
3.2.3. Presupuesto .....	29
3.2.4. Evaluación de riesgos .....	30
3.3. Modelo de desarrollo.....	31
3.4. Planificación temporal .....	32
3.5. Conclusión .....	36
<b>4. Estudio de alternativas .....</b>	<b>37</b>
<b>5. Sistema desarrollado .....</b>	<b>39</b>
5.1. Nagios como aplicación de monitorización .....	39
5.2. Arquitectura global del sistema .....	41
5.3. Arquitectura de los servidores cliente .....	42
5.4. Arquitectura del servidor de bases de datos .....	44
5.5. Aplicación de visualización.....	46
5.5.1. Casos de uso .....	46
5.5.2. Diagrama de clases.....	47
5.5.3. Codificación.....	48
5.5.4. Descripción de la aplicación .....	49
5.5.4.1. Descripción general .....	49
5.5.4.2. Vista general.....	50
5.5.4.3. Vista detallada .....	51
5.5.4.4. Vista de problemas .....	52
5.5.4.5. Generación de informes.....	53
5.5.4.6. Configuración de la aplicación .....	54
5.6. Pruebas.....	55
<b>6. Conclusiones.....</b>	<b>56</b>



6.1. Grado de cumplimiento de los objetivos .....	56
6.2. Propuestas de ampliación y futuras mejoras .....	57
6.3. Análisis de la planificación.....	58
6.4. Valoraciones personales .....	59
<b>Bibliografía .....</b>	<b>60</b>

# Índice de figuras

2.1. Tabla comparativa de las aplicaciones de monitorización .....	11
2.2. Diagrama de monitorización actual.....	16
3.1. Tabla de coste de material.....	27
3.2. Tabla de coste de personal.....	28
3.3. Modelo de desarrollo lineal .....	32
3.4. Planificación temporal.....	33
3.5. Diagrama de Gantt de la planificación .....	34
3.6. Planificación temporal detallada .....	35
5.1. Diagrama de la arquitectura global del sistema .....	42
5.2. Diagrama de la arquitectura de los servidores cliente .....	43
5.3. Diagrama de la arquitectura del servidor de BBDD .....	45
5.4. Diagrama de casos de uso .....	46
5.5. Diagrama de clases .....	47
5.6. Diagrama de la arquitectura de la aplicación .....	49
5.7. Pantalla de la vista general.....	50
5.8. Pantalla de la vista detallada .....	51
5.9. Pantalla de problemas .....	52

5.10. Pantalla de generación de informes.....	53
5.11. Pantalla de configuración .....	54
6.1. Comparativa de la planificación temporal.....	58



# **Capitulo 1**

## **Introducción**

### **1.1. Proyecto en empresa**

Antes de adentrarnos a analizar los pormenores del proyecto es necesario hacer una pequeña explicación del contexto en el que se desarrolla el mismo. Dicho proyecto, titulado “Monitorización remota de sistemas” está enmarcado en un convenio de prácticas entre la Escuela Universitaria de Informática de Sabadell y la empresa CCSAgresso.

Desde principios del curso 2002/2003 la Escuela y CCSAgresso mantienen un convenio de colaboración que tiene como objetivo elaborar conjuntamente propuestas de formación en el área de las nuevas tecnologías de la información, abiertas a todos los estudiantes, y permitir que los alumnos de tercer curso puedan desarrollar su

proyecto final de carrera incorporándose en alguno de los departamentos de la empresa.

A lo largo de los años, desde el curso 2002/2003 hasta el curso 2008/2009, ya son más de 70 los alumnos que han podido desarrollar su proyecto en el marco de este convenio. Muchos de estos alumnos, al finalizar el proyecto, fueron incorporados a la plantilla de trabajadores de CCSAgresso. Por todo esto podemos decir que esta relación no solo es una relación laboral, si no que ayuda al alumno a abrirse más puertas en el mundo laboral, ya sea en CCSAgresso o en otras empresas.

La duración de la estancia en la empresa es de aproximadamente 560 horas repartidas a lo largo del curso. Durante este periodo los tutores de la empresa asignados al proyecto hacen un seguimiento exhaustivo y guían al alumno en busca de una resolución satisfactoria.

En el caso del proyecto que estamos tratando se ha desarrollado en su totalidad en el Departamento de Soporte de Barcelona.

La empresa CCSAgresso es la filial española del grupo Unit4Agresso, uno de los primeros fabricantes internacionales de software de gestión. Dispone de oficinas, centros de desarrollo y servicios distribuidos por toda la geografía nacional, con una plantilla especializada de aproximadamente 700 personas y una importante cartera de clientes, tanto en empresas de diferentes sectores como en las Administraciones Públicas y la Sanidad.

Gracias a su software de última generación -ekon, Agresso Business World y SPAI- CCS Agresso posee la capacidad tecnológica,

de servicios y postventa necesaria para proporcionar al usuario información ágil y asequible con mejor coste del mercado.

La compañía, líder europea en soluciones de gestión e interlocutor global para empresas y organismos públicos, cuenta con un extenso conjunto de partners repartidos por toda la Península Ibérica que ofrecen al cliente una relación calidad/producto de alto nivel.

Por todo lo analizado con respecto al convenio de colaboración se puede decir que la relación entre la Escuela Universitaria de Informática de Sabadell y CCSAgresso goza de una excelente salud y proporciona una oportunidad a los alumnos de trabajar en una multinacional líder en su sector.

## **1.2. Motivaciones**

En este momento habría que dar una pequeña pincelada del concepto de monitorización de sistemas para tener una leve idea. Este se refiere al análisis periódico del estado de un sistema. Por análisis periódico se entiende la recopilación de información de manera iterativa que nos define el estado de un sistema. Por otro lado cuando hablamos de sistemas informáticos podemos referirnos desde complejas redes de computadores hasta simples aplicaciones informáticas.

En la actualidad la monitorización es una herramienta indispensable para la gestión y el control de un determinado sistema.

Mediante una buena monitorización, aunque no podemos impedir que se produzcan fallos en los sistemas, si que podemos alertar inmediatamente y proceder a la resolución de las incidencias con mayor celeridad.

Es por esto que multitud de empresas utiliza la monitorización como un recurso más para aumentar la fiabilidad de sus sistemas. Además, cada vez más empresas como CCSAgresso, ofrecen a sus clientes la posibilidad de monitorizar sus sistemas y de esta manera mejorar el funcionamiento de los mismos.

Es precisamente de la necesidad de mejorar este servicio que CCSAgresso ofrece a sus clientes de donde nace el proyecto “Monitorización remota de sistemas”. Con este proyecto como veremos más adelante se pretende automatizar todo el proceso de monitorización que actualmente se hace de manera totalmente manual.

### **1.3. Objetivos**

Los objetivos del proyecto han sido definidos por CCSAgresso, concretamente por el Departamento de Soporte de Barcelona, en función de las carencias y necesidades detectadas en el servicio que actualmente se ofrece a los clientes.

Los objetivos se estructuran en dos grandes ejes principales, el primero es el diseño e implementación de un sistema de monitorización remoto, y el segundo es el diseño e implementación de una aplicación



de visualización que muestre el estado de las redes clientes.

La primera rama se centra principalmente en la arquitectura que debe tener el sistema de monitorización. Mediante este sistema se debe poder analizar el estado de las redes de los clientes prácticamente a tiempo real.

El segundo gran eje se basa en la implementación de una aplicación gráfica que visualice de forma sencilla e intuitiva la información generada el sistema de monitorización comentado anteriormente. De esta manera los técnicos de sistemas de CCSAgresso tendrán una idea del estado de la redes clientes con un simple vistazo a la aplicación.

Estos dos grupos de objetivos nos dividen el sistema a desarrollar, como más adelante veremos, en dos partes. Una parte más de dedicada a la parte de sistemas y una segunda parte dedicada por completo al diseño e implementación software.

## **1.4. Metodología**

Los objetivos que CCSAgresso planteó inicialmente, a simple vista, pueden parece muy complejos y ambiciosos, ya que para el cumplimiento de los mismos es necesario estudiar nuevas tecnologías, formarse en diversos campos y para todo ello dispondremos de un tiempo y recursos limitados. Por lo tanto es básico definir una buena metodología donde se asegure la viabilidad del proyecto y el desarrollo

del mismo sea ágil y consistente.

La metodología comenzará con un diagnóstico profundo de la situación actual. Empezando por analizar el uso de la monitorización en las grandes empresas y terminando por estudiar el proceso que se realiza actualmente en CCSAgresso para realizar la monitorización de las redes clientes.

Seguidamente, a partir de la información obtenida en la fase de diagnóstico, realizaremos un estudio de viabilidad donde se analizarán los principales requerimientos del proyecto, los recursos económicos y técnicos que serán necesarios para su elaboración la planificación y el modelo de desarrollo que se seguirá y finalmente se analizará si el proyecto es viable o no.

A continuación iniciaremos el desarrollo del sistema empezando por el análisis, diseño y posterior implementación de la arquitectura del sistema. Una vez terminada dicha arquitectura podremos comenzar el análisis, diseño y codificación de la aplicación de visualización.

La finalización de la implementación, tanto de la arquitectura del sistema como de la aplicación, nos marcará el inicio de la etapa de pruebas. En esta etapa haremos una simulación en formato reducido, es decir, con pocos clientes, donde pondrá a prueba el sistema realizado.

Después de la etapa de pruebas y de la corrección de los posibles errores ocurridos pasaremos a la realización de la documentación de apoyo redactando manuales de usuario, de configuración y de instalación del sistema.

Finalizaremos la metodología empleada realizando un análisis

de diferentes aspectos del proyecto ya finalizado como el grado de cumplimiento de los objetivos iniciales o algunas propuestas para futuras mejoras del sistema.

## **1.5. Contenido de la memoria**

La memoria del proyecto final de carrera se divide en un total de seis capítulos. Cada uno de estos capítulos se identifica con una etapa utilizada en la metodología comentada en el apartado anterior. A continuación haremos un pequeño desglose del contenido de cada uno de los capítulos.

En el primer capítulo, en el cual nos encontramos, se hace una pequeña explicación de lo que significa realizar un proyecto en empresa y se expone de manera más profunda los objetivos del proyecto y la metodología que se utilizará para conseguirlos.

En el segundo capítulo se analizará principalmente el proceso que utiliza actualmente CCSAgresso para monitorizar las redes de sus clientes. Adicionalmente se explicará el concepto de monitorización y se analizará la monitorización actual en las grandes empresas.

El tercer capítulo se corresponde con la fase del estudio de viabilidad. En el se decidirá si el proyecto debe seguir adelante valorando aspectos económicos, técnicos y legales. En este capítulo también estimaremos la planificación temporal y el modelo de desarrollo que se utilizará para llevar a cabo el proyecto.

En el cuarto capítulo veremos un pequeño estudio de las

diferentes alternativas para la realización del proyecto y el porqué de la elección final.

El quinto capítulo contiene una explicación detallada del sistema implementado desde el desarrollo de la arquitectura del sistema, pasando

pasando por el diseño de la aplicación y finalizando la exposición del conjunto de pruebas realizadas.

En el sexto capítulo encontraremos un resumen de las conclusiones extraídas del proyecto. Se analizará el grado de cumplimiento del proyecto, la desviación entre la planificación inicial y la final, y se verán algunas propuestas de ampliación para el futuro.

Para finalizar, en el CD adjunto de la memoria encontraremos el código fuente de la aplicación, concretamente el proyecto completo de la plataforma Eclipse, y los anexos adicionales que están formados por: manual de usuario de Nagios 3.0, manual de usuario de NDOUtils, tablas de la base de datos generada por NDOUtils, manual de instalación del sistema, manual de configuración del sistema y la descripción general del sistema.

## **Capítulo 2**

### **Diagnóstico de la situación**

#### **2.1. ¿Qué es la monitorización de sistemas?**

Para comprender el contenido de la memoria es de vital importancia, yo diría imprescindible, entender el significado del concepto monitorización de sistemas. En el capítulo 1, en el apartado de motivaciones, hicimos un pequeño acercamiento a este concepto. Seguidamente ampliaremos esta información.

Por monitorización de sistemas entendemos todo tipo de análisis periódico de un sistema de cuya información se desprenda su estado actual. Si analizamos más profundamente la definición podemos denotar que cuando nos referimos a análisis periódico queremos decir un escaneo del sistema que se repite cada cierto intervalo de tiempo. Este escaneo puede realizarse de forma automática, mediante software especializado como veremos posteriormente, o de forma manual, es

## *CAPÍTULO 2. DIAGNÓSTICO DE LA SITUACIÓN*

decir, un técnico se conecta al sistema a analizar y hace mediciones de diferentes parámetros para evaluar su estado.

Si continuamos analizando la definición nos encontramos con el concepto de sistema. En monitorización un sistema es cualquier tipo de ente informático, es decir, desde la aplicación informática más sencilla hasta las redes de computadoras más complejas, pasando por computadoras aisladas, impresoras, enrutadores,... Podemos afirmar que en informática casi cualquier cosa imaginable se puede monitorizar.

Cuando en una monitorización el sistema a analizar está a una cierta distancia esta adquiere la característica de remota.

Actualmente el futuro de la monitorización pasa por ser una monitorización remota y automática. Para ello existen en el mercado multitud de herramientas y aplicaciones tanto privadas como de código libre. Algunas de las aplicaciones más famosas en el campo de la monitorización son: Cacti, Cisco Works LMS, Nagios, Open NMS, Osmius, Zabbix, ...

La inmensa mayoría de las herramientas existentes se basan en el protocolo SNMP para conseguir la información del estado del sistema a analizar.

Para finalizar este apartado veremos una tabla comparativa de las aplicaciones de monitorización actuales:

## CAPÍTULO 2. DIAGNÓSTICO DE LA SITUACIÓN

Nombre	Gráficos	Agrupación lógica	Agente	SNMP	Syslog	Scripts externos	Plugins	Creación Plugins	Alertas	WebApp	Monitorización distribuida	Método almacenamiento de datos	Licencia	Esquemas	Control de acceso	Escrito en
AdRem	SI	SI	No	SI	SI	SI	No	N/A	SI	SI	No	SQL	Comercial	SI	SI	
NetCrunch	SI	SI	No	SI	SI	SI	SI	Medio	SI	SI	SI	Flatfile, ODBC	Comercial	No	SI	
Advanced Host Monitor	SI	SI	SI	SI	SI	SI	SI		SI	SI	SI		Comercial	No	SI	
Argus	SI	SI	SI	SI	No	SI	SI	Fácil	SI	SI	Parcial	Barkeley DB	Artística	No	Granular	
CA eHealth	SI	SI	SI	SI	System Edge	System Edge	No	N/A	SI	Limitado	SI	Oracle	Comercial	No	Granular	
Cacti	SI	SI	No	SI	SI	SI	SI	Fácil	SI	SI	SI	MySQL	GPL	SI	SI	PHP
CiscoWorks LMS	¿?	¿?	¿?	¿?	¿?	¿?	¿?	¿?	¿?	¿?	¿?	¿?	Comercial	SI	¿?	¿?
Collectd	SI	No	SI	SI	SI	SI	SI	Fácil	SI	SI	SI	CSV	GPL	No	Apache	C
doppleVUE	SI	SI	No	SI	SI	SI	SI	.Net	SI	SI	SI	SQL	Comercial	SI	SI	
Entuity	SI	SI	No	SI	SI	SI	SI	Medio	SI	SI	SI	MySQL	Comercial	SI	Granular	

2.1 Tabla comparativa de las aplicaciones de monitorización

## CAPÍTULO 2. DIAGNÓSTICO DE LA SITUACIÓN

Nombre	Gráficos	Agrupación lógica	Agente	SNMP	Syslog	Scripts externos	Plugins	Creación Plugins	Alertas	WebApp	Monitorización distribuida	Método almacenamiento de datos	Licencia	Esquemas	Control de acceso	Escrito en
Everest	Si	Si	Si	Si	Si	Si	Si	Fácil	Si	Si	Si	MySQL	Comercial	Si	Si	Python C
FireScope	Si	Si	Si	Si	Si	Si	Si	Fácil	Si	Si	Si	SQL	Comercial	Si	Si	
FreeNATS	Si	Si	Si	No	Si	Si	Si	Medio	Si	Si	No	MySQL	GPL	No	Granular	PHP
Ganglia	Si	Si	Si	Si	No	Si	Si	Medio	No	Si	Si	RRDtool	BSD	Si	No	C
GroundWork Community	Si	No	Si	Si	Si	Si	Si	Medio	Si	Si	Si	MySQL	GPL	No	¿?	
Heroix	Si	Si	Si	Si	Si	Si	Si	Difícil	Si	Si	Si	MaxDB	Comercial	Si	Granular	Java
Hyperic	Si	Si	Si	Si	Si	Si	Si	Fácil	Si	Si	Si	PostgreSQL	GPL	Si	Si	
i-enable rmf	Si	Si	Si	Si	Si	Si	Si	Fácil	Si	Si	Si	MySQL	Comercial	Si	Si	
Intermapper	Si	Si	Si	Si	Si	Si	Si	Medio	No	Si	Si	PostgreSQL	Comercial	Si	Si	
LoriotPro	Si	Si	No	Si	Si	Si	Si	Medio	Si	Si	Si	Flat file	Comercial	Si	Si	
Munin	Si	Si	Si	Si	No	Si	Si	Medio	Si	Si	¿?	RRDtool	GPL	¿?	¿?	



## CAPÍTULO 2. DIAGNÓSTICO DE LA SITUACIÓN

Nombre	Gráficos	Agrupación lógica	Agente	SNMP	Syslog	Scripts externos	Plugins	Creación Plugins	Alertas	WebApp	Monitorización distribuida	Método almacenamiento de datos	Licencia	Esquemas	Control de acceso	Escrito En
Nagios	Si	Si	Si	Si	Si	Si	Si	Fácil	Si	Si	Si	MySQL	GPL	Si	Si	Perl
Nedi	Si	No	No	Si	No	¿?	¿?	¿?	Si	Si	¿?	MySQL	GPL	¿?	¿?	Perl
Netscope	Si	Si	Si	Si	Si	Si	Si	Fácil	Si	Si	Si	SQL	Comercial	¿?	¿?	
Nimsoft	Si	Si	Si	Si	Si	Si	Si	Medio	Si	Si	Si	SQL	Comercial	Si	Si	
OpenNMS	Si	Si	Si	Si	Si	Si	Si	Hard	Si	Si	Si	JRobin	GPL	Si	Si	Java
Pandora FMS	Si	Si	Si	Si	Si	Si	Si	Fácil	Si	Si	Si	MySQL	GPL	Si	Granular	PHP
Orion	Si	Si	Si	Si	Si	Si	Si	Fácil	Si	Si	Si	SQL	Comercial	Si	Si	C
Zabbix	Si	Si	Si	Si	Si	Si	Si	Fácil	Si	Si	Si	Oracle	GPL	Si	Si	C
Zytron	Si	Si	Si	Si	Si	Si	Si	Fácil	Si	Si	Si	SQL	Comercial	Si	Si	Java

## **2.2. La monitorización en grandes empresas**

Actualmente la monitorización en empresas mínimamente grandes es un tema fundamental en lo que se refiere a la administración de los sistemas. Esta monitorización se puede dividir en dos grandes campos: la monitorización interna y la monitorización externa.

La monitorización interna se utiliza básicamente para controlar las posibles incidencias que puedan ocurrir en la red propia de la empresa. Podemos decir que es una herramienta más para mejorar el control y la gestión de una red.

Por otro lado existe lo que se llama monitorización externa, que cada vez es una práctica más habitual en las empresas informáticas. Esta monitorización es ofrecida a los clientes de la empresa para mejorar el rendimiento de sus redes. Mediante este servicio el cliente ve maximizado el control que se realiza sobre su red y las incidencias técnicas se reducen al máximo.

Podemos concluir, sin riesgo a equivocarnos, que esta práctica , que cada vez es más extendida, poblara de manera unánime casi todas las empresas informáticas en un corto espacio de tiempo debido a sus enormes beneficios y su escaso desembolso económico.

## **2.3. La monitorización actual en CCSAgresso**

### **2.3.1. Descripción de la situación actual**

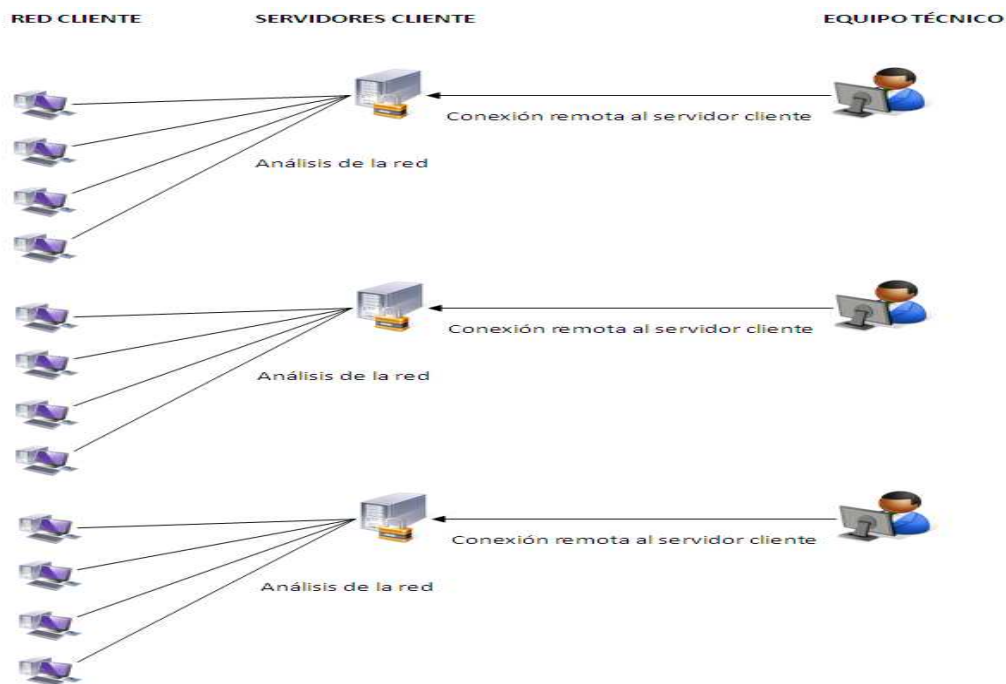
En la actualidad el Departamento de Soporte de CCSAgresso dispone de una cartera de servicios orientados a la monitorización y administración remota de sistemas. Estos servicios se contratan por módulos dependiendo de la instalación del cliente, alguno de estos módulos son:

- Servidores Windows.
- Gestores de Bases de Datos.
- Servidores de Correo Exchange
- Copias de seguridad.
- ...

En estos momentos esta administración y monitorización es totalmente manual. El procedimiento es el siguiente: el técnico de sistemas se conecta al servidor del cliente vía terminal remoto y desde allí realiza el chequeo de la red según los diferentes servicios contratados. Esta rutina se repite periódicamente cada cierto intervalo de tiempo.

Mediante el siguiente diagrama tendremos una idea más clara del proceso que se sigue actualmente para realizar la monitorización:

## CAPÍTULO 2. DIAGNÓSTICO DE LA SITUACIÓN



2.1 Diagrama de la monitorización actual

A continuación y en base a lo explicado en este apartado haremos un pequeño análisis de las conclusiones obvias que se desprenden del sistema actual.

### 2.3.2. Conclusiones de la situación actual

Podemos deducir varias conclusiones del procedimiento que se lleva a cabo actualmente en CCSAgresso. Gracias a estas conclusiones tendremos un punto de partida en la confección del futuro proyecto ya

que nos darán mucho de los puntos flacos que en estos momentos tiene el servicio que se ofrece a los clientes.

Lo primero que podemos afirmar del sistema actual de monitorización y administración remota es que es un sistema muy. Esto es debido, como ya hemos explicado, a que el chequeo de las redes de los clientes se hace de forma totalmente manual.

Esta lentitud del procedimiento implica directamente mayor esfuerzo por parte de los técnicos de sistemas y por lo tanto más horas empleadas en esta tarea. Esto supone, por lo tanto, un desembolso económico que se podría evitar.

Además de la duración del proceso hay que remarcar que el chequeo de las redes se realiza cada cierto intervalo de tiempo, por lo que es fácil predecir que se pueden producir incidencias entre chequeo y chequeo, de las cuales no se tendrían constancia ya que no quedarían registradas.

Por otro lado, al realizarse el chequeo de forma manual se deja en manos de la valía de los técnicos que estas revisiones sean más o menos satisfactorias.

Por último remarcar la dificultad que supone con el sistema actual llevar un registro de todas las incidencias que se van produciendo. Esta tarea actualmente esta asignada a los técnicos de sistemas que manualmente dan de alta en la base de datos cada incidencia nueva que se produce en el cliente.

Con todo lo visto del proceso actual podemos afirmar que se trata de un sistema poco fiable, lento y que supone un coste que es posible reducir de manera sencilla.



## **Capítulo 3**

### **Estudio de viabilidad**

#### **3.1. Análisis de requerimientos**

##### **3.1.1. Requerimientos funcionales**

Al igual que los objetivos, CCSAgresso planteó al principio del proyecto una serie de funcionalidades que el sistema a desarrollar debía proporcionar a los usuarios, que en este caso son los miembros del Departamento de Soporte. Estas funcionalidades tienen como principal objetivo subsanar los defectos del sistema actual que vimos en el último apartado del Capítulo 2.

El primer requerimiento, y quizás el más importante de todos, es el de proporcionar una aplicación gráfica que sea capaz de visualizar el estado a tiempo real de las redes de los clientes según los servicios

que estos tengan contratados. Esta aplicación tiene que tener como mínimo dos modos de visualización, la primera donde se vea un resumen general del estado de todas las redes clientes, y la segunda donde podamos observar el estado de un cliente específico así como el de los componentes que forman su red.

La segunda funcionalidad que se requiere es la de dar la posibilidad, mediante la aplicación, de una conexión directa con la interfaz web del programa de monitorización. De esta manera cuando llegue el aviso de una incidencia el técnico podrá conectarse vía http y analizarla más profundamente.

Otro requisito que el sistema debe cumplir es que la aplicación de visualización sea plenamente configurable. Esto incluye la posibilidad de modificar la frecuencia de monitorización, los datos de conexión de todos los clientes y los datos de conexión de la base de datos central donde se ira almacenando la información.

Como último requerimiento funcional debemos destacar que el sistema a desarrollar debe ser capaz de proporciona reportes del estado de los clientes, estadísticas de uso, históricos de incidencias, alarmas detectadas,... es decir, tratar al máximo la información generada por las diferentes monitorizaciones.



### 3.1.2. Requerimientos no funcionales

De igual manera que los requerimientos funcionales CCSAgresso impuso una serie de restricciones en temas como: los sistemas operativos con los que trabajar, el hardware que se debía utilizar y por último, y mas importante, el programa de monitorización con el que formas el núcleo del sistema.

Empezaremos con el último de los requerimientos, el que tiene que ver con el programa de monitorización. CCSAgresso eligió como aplicación de monitorización la herramienta Nagios después de un profundo estudio que llevó a cabo la Delegación de Granada. En este estudio se concluía que Nagios era uno de los programas de monitorización más potentes, escalables y amigables del mercado. Además una baza a su favor fue que se distribuye mediante una licencia GPL, es decir, es de código libre.

El segundo requerimiento no funcional era la utilización de un determinado hardware. Concretamente se debía emplear un servidor dedicado para cada cliente. Este servidor como veremos más adelante se encargará de realizar la monitorización mediante la herramienta Nagios.

Por último tenemos el requisito de los sistemas operativos con los que trabajar. Cada uno de los servidores cliente debía funcionar con sistemas Windows, debido al desconocimiento de los clientes de otros sistemas, y dentro del sistema operativo tener instalada una máquina virtual Ubuntu. En esta máquina virtual será donde este alojado el programa de monitorización Nagios debido a que en la actualidad esta

aplicación es incompatible con sistemas Windows.

## **3.2. Viabilidad técnica y económica**

### **3.2.1. Recursos**

#### **3.2.1.1. Software**

Para la realización del proyecto se necesita unos determinados recursos software. A continuación enumeraremos todo el software imprescindible para el desarrollo del sistema.

- Servidores cliente:
  - o Sistema Operativo: Windows XP Professional.
  - o Monitorización de sistemas: Nagios 3.0
  - o Máquina Virtual: VMWare con Ubuntu Linux instalado por la incompatibilidad de Nagios 3.0 con los sistemas Windows.
  - o Servidor HTTP: Apache instalado en Ubuntu Linux.
  - o Plugins de Nagios: Los diferentes plugins de Nagios para analizar los diferentes servicios.
  - o NDOUtils: Plugin de Nagios para la comunicación y almacenamiento en la base de datos.
  - o Compiladores: Perl instalado en Ubuntu Linux.

- PC's clientes:
  - o Plugin de Nagios: NSCLIENT++ en caso de que el sistema a analizar sea un sistema Windows.
  
- Servidor de la empresa:
  - o Sistema Operativo: Windows XP Professional.
  - o Monitorización de sistemas: Nagios 3.0
  - o Máquina Virtual: VMWare con Ubuntu Linux instalado por la incompatibilidad de Nagios 3.0 con los sistemas Windows.
  - o Servidor HTTP: Apache instalado en Ubuntu Linux.
  - o NDOUtils: Plugin de Nagios para la comunicación y almacenamiento en la base de datos.
  - o Compiladores: Perl instalado en Ubuntu Linux.
  - o Base de datos: MYSQL instalado en Ubuntu Linux.
  
- PC's técnicos de sistemas:
  - o Sistema Operativo: Windows XP Professional.
  - o Navegador Internet: Internet Explorer o Mozilla Firefox.
  - o Herramienta corporativa: Ekon SAT.
  
- Entornos de programación:
  - o Visual C++, Java y Perl.
  
- Generación de documentación:
  - o Open Office Writer, Open Office Impress y Open Project.

### **3.2.1.2. Hardware**

Para la implementación del sistema final también serán necesarios una cantidad específica de recursos hardware. Entre estos recursos debemos contar cada uno de los servidores dedicados que estarán instalados en las dependencias de nuestros clientes, el servidor central de base de datos y por último el equipo del desarrollador del proyecto.

Para los tres tipos de equipos será necesario dar unos requisitos mínimos que deben cumplir:

- Recursos mínimos de los servidores clientes:
  - o Memoria RAM: 512 Mb
  - o Procesador: Intel Core2Duo 2.66 Hz
  - o Disco Duro: 100Gb
  - o Tarjeta de red
- Recursos mínimos del servidor de la empresa:
  - o Memoria RAM: 1024 Mb
  - o Procesador: Intel Core2Duo 2.66 Hz
  - o Disco Duro: 300Gb
  - o Tarjeta de red
- Recursos mínimos del equipo del desarrollador:
  - o Memoria RAM: 512 Mb
  - o Procesador: Intel Core2Duo 2.66 Hz
  - o Disco Duro: 100Gb
  - o Tarjeta de red

### **3.2.1.3. Recursos humanos**

Durante todo el transcurso del desarrollo del sistema participaran diferentes perfiles de individuos que realizarán diversas funciones. En el proyecto que estamos tratando todos los perfiles estarán aunados en una única persona: el alumno.

A continuación veremos los diferentes perfiles implicados en el proyecto:

- Analista: Análisis y diseño de todo el proyecto.
- Técnicos programadores: Implementación de la aplicación del proyecto a partir del análisis y diseño del analista.
- Técnicos de sistemas: Realización de toda la parte de instalación y configuración de los servidores clientes y empresa.
- Equipo de pruebas: Comprobación de la fiabilidad del sistema una vez finalizado para posibles mejoras.
- Técnicos de mantenimiento: Conocimiento del sistema para una vez instalado hacer un mantenimiento del mismo.

### **3.2.2. Análisis coste-beneficio**

Otro factor importante a analizar para asegurar la viabilidad del

### *CAPÍTULO 3. ESTUDIO DE VIABILIDAD*

proyecto es el coste total, tanto de material como del personal que participe en su desarrollo.

En base a este aspecto se decidirá si el proyecto es viable o no económicamente.

A continuación veremos dos tablas donde se describen tanto las partidas de materiales utilizadas como la relación de horas dedicadas a cada fase del proyecto.

**3.2.2.1. Coste de material**

<b>RECURSO</b>	<b>COSTE TOTAL</b>
<i>Servidor cliente</i>	1300,00 €
<i>Servidor CCS</i>	1500,00 €
<i>Windows XP Prof.</i>	150,00*2=300 €
<i>Ubuntu Linux</i>	0,00*2 €
<i>SGBD MYSQL</i>	0,00 €
<i>Apache</i>	0,00*2 €
<i>Nagios 3.0</i>	0,00*2 €
<i>VMWare</i>	0,00*2 €
<i>Amortización PC programador</i>	$(1200/36)*3=100,00$ €
<i>Amortización PC analista</i>	$(1200/36)*1=33,33$ €
<i>Amortización PC técnico sistemas</i>	$(1200/36)*5=166,67$ €
<i>Amortización Open Office</i>	0,00 €
<i>Amortización Open Project</i>	0,00 €
<i>Visual C++,Java, Perl</i>	0 € (ya amortizados)
<b>Total</b>	3400,00 €

3.1 Tabla de coste de material

**3.2.2.2. Coste de personal**

<b>FAENA</b>	<b>TRABAJO (HORAS)</b>	<b>COSTE</b>
<i>Formación y recopilación de información</i>	40	<b>200 €</b>
<i>Estudio de viabilidad</i>	30	<b>150 €</b>
<i>Diseño del sistema de recopilación de datos</i>	80	<b>400 €</b>
<i>Diseño de la interfaz de usuario</i>	60	<b>300 €</b>
<i>Diseño y configuración de la base de datos</i>	30	<b>150 €</b>
<i>Pruebas locales del sistema</i>	60	<b>300 €</b>
<i>Instalación del servidor del cliente</i>	16	<b>320 €</b>
<i>Instalación del servidor de la empresa</i>	8	<b>160 €</b>
<i>Instalación NSCLIENT en PC cliente</i>	2	<b>40 €</b>
<i>Pruebas finales con la red en funcionamiento</i>	60	<b>300 €</b>
<i>Elaboración de la documentación</i>	50	<b>250 €</b>
<b>Total</b>	<b>436</b>	<b>2570 €</b>

3.2 Tabla de coste de personal



### 3.2.3. Presupuesto

Después de analizar el coste del material y el coste del personal podemos pasar a confeccionar el presupuesto de una instalación del futuro sistema en un número indeterminado de clientes. Con este presupuesto desglosado podremos valorar si el coste de la instalación será lo suficientemente competitivo como para atraer a los clientes.

Detalle del presupuesto

- Servidor cliente..... **n x 1943,50 €**
  - o Memoria RAM: 512 Mb
  - o Procesador: Intel Core2Duo 2.66 Hz
  - o Disco Duro: 100Gb
  - o Tarjeta de red
- Instalación del servidor cliente ..... **n x 702,65 €**
  - o Instalación Windows XP Professional
  - o Instalación de VMWare
  - o Instalación Ubuntu Linux
  - o Instalación y configuración de Nagios
- Servidor empresa ..... **2242,50 €**
  - o Memoria RAM: 1024 Mb
  - o Procesador: Intel Core2Duo 2.66 Hz
  - o Disco Duro: 300Gb
  - o Tarjeta de red
- Parte proporcional de la instalación del servidor empresa ..... **463,45 €**
  - o Instalación Windows XP Professional

- Instalación de VMWare
  - Instalación Ubuntu Linux
  - Instalación y configuración de Nagios
  - Instalación de MYSQL
- Aplicación: Monitorización remota de sistemas ..... 3289,00 €
- Detalle del presupuesto ..... 5994,95+n(2646,15) €

***\*n: número de clientes a los que está destinado el proyecto.***

### 3.2.4. Evaluación de riesgos

En el sistema a desarrollar debemos tener en cuenta algunos factores de riesgo que pueden hacer perder la integridad del sistema y el buen funcionamiento del mismo.

El primer factor importante es el de garantizar la seguridad de los resultados que se envían de los servidores clientes al servidor de la empresa. Dicha información ha de viajar cifrada para evitar que se apodere de ella gente no autorizada.

El segundo factor destacable es el de la posibilidad de pérdidas de conexión entre el servidor cliente y la red que se está analizando. Ante este problema poco podemos hacer debido a los recursos que tenemos pero para una pronta recuperación de la caída lo idóneo es

que cada cierto intervalo se vaya consultando el estado de los servidores clientes.

El último factor de riesgo que podemos analizar es el de posibles pérdidas de conexión entre el servidor de la empresa y los diferentes servidores clientes. De igual manera que en el caso anterior la estrategia para una recuperación rápida sería consultar periódicamente el estado de los cliente, por ejemplo, mediante comandos ping.

### **3.3. Modelo de desarrollo**

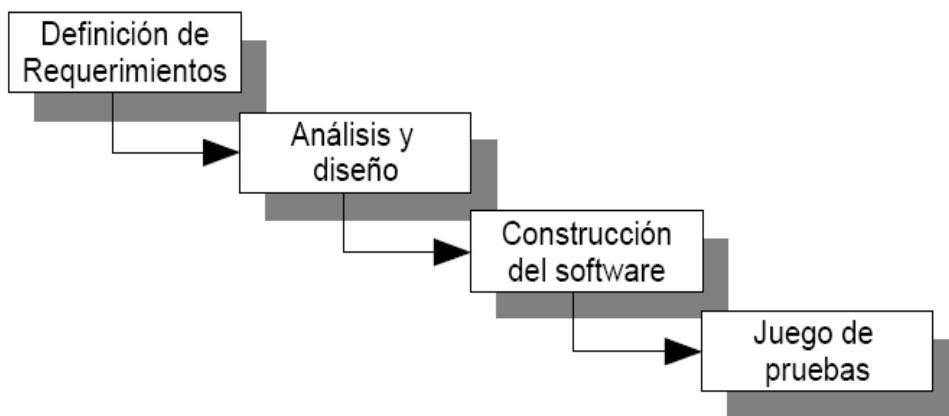
Para todo proyecto de una cierta envergadura es de vital importancia la elección de un modelo de desarrollo que se adecue a la tipología del problema.

En el proyecto que estamos tratando el modelo que más se adapta a sus características es el modelo de desarrollo lineal. Este modelo de desarrollo se basa en cuatro etapas básicas que se aplican todas ellas de forma secuencial, es decir, cada fase comienza cuando la anterior ha finalizado totalmente.

Las cuatro etapas son las siguientes: definición de requerimientos, donde se especifican los requerimientos funcionales y no funcionales; la fase de análisis y diseño, donde se estudia el problema a tratar y se hace una propuesta de solución; la etapa de construcción donde se implementa el diseño llevado a cabo en la etapa anterior; y por último tenemos la fase de pruebas, donde se harán

simulaciones del sistema con un número reducido de clientes.

En el siguiente diagrama vemos las diferentes etapas del modelo en forma de cascada:



3.3 Modelo de desarrollo lineal

### 3.4. Planificación temporal

En este apartado haremos una planificación temporal aproximada basándonos en el modelo de desarrollo visto en el anterior apartado.

Primero veremos una asignación de las horas estimadas para la realización de todas las etapas del modelo. Seguidamente desglosaremos las diferentes fases en tareas más simples y evaluables donde igualmente asignaremos un periodo de tiempo para su finalización. Cabe destacar que nos hemos ayudado para hacer la planificación temporal de software especificado con el que hemos

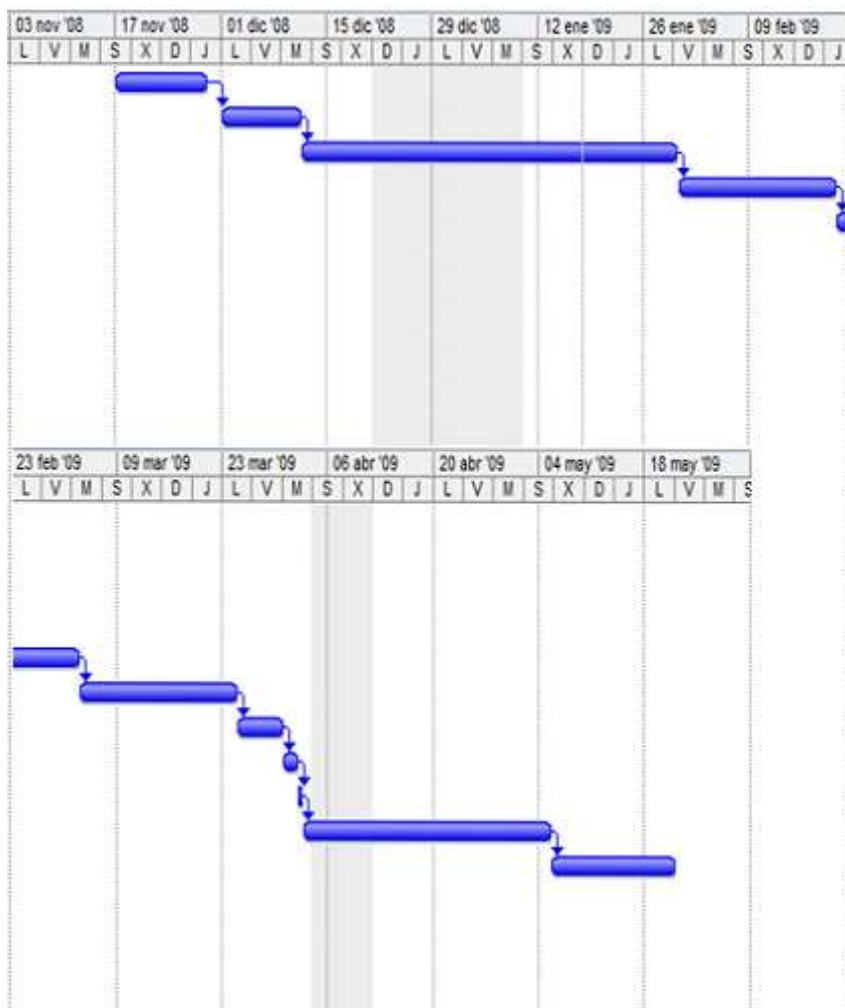
confeccionado un Diagrama de Gantt detallado.

La planificación temporal de las etapas del proyecto quedará de la siguiente manera:

<b>Etap</b>	<b>Descripción</b>	<b>Duración inicial</b>
1	<i>Definición de requerimientos</i>	60 horas
2	<i>Análisis y diseño</i>	170 horas
3	<i>Construcción del software</i>	200 horas
4	<i>Juego de pruebas</i>	70 horas

#### 3.4 Planificación temporal

Por último, en la siguiente imagen podemos ver la distribución temporal de las diferentes tareas mediante un Diagrama de Gantt y una tabla detallada:



3.5 Diagrama de Gantt de la planificación

### CAPÍTULO 3. ESTUDIO DE VIABILIDAD

	Nombre de tarea	Duración	Comienzo	Fin	Prec.
1	<b>Definición de requerimientos</b>	15 días	<b>17/11/08</b>	05/12/08	
2	<i>Recopilación de info.</i>	6 días	17/11/08	24/11/08	
3	<i>Estudio del sistema actual</i>	2 días	25/11/08	26/11/08	2
4	<i>Estudio del sistema a realizar</i>	4,5 días	27/11/08	03/12/08	3
5	<i>Redacción del documento</i>	2,5 días	03/12/08	05/12/08	4
6	<b>Análisis y diseño</b>	42,5 días	09/12/08	24/02/09	
7	<i>Análisis/diseño hardware</i>	17,5 días	09/12/08	20/01/09	5
8	<i>Análisis/diseño software</i>	25 días	20/01/09	24/02/09	7
9	<b>Construcción</b>	50 días	24/02/09	05/05/09	
10	<i>Codificación</i>	37,5 días	24/02/09	16/04/09	8
11	<i>Documentación</i>	12,5 días	17/04/09	05/05/09	10
12	<b>Juego de pruebas</b>	17,5 días	05/05/09	<b>28/05/09</b>	11

3.6 Planificación temporal detallada

### **3.5. Conclusión**

Por todo lo visto en los diferentes apartados del estudio de viabilidad podemos extraer una serie de conclusiones con un riesgo mínimo a equivocarnos.

La primera y más llamativa del estudio es que el proyecto tiene una relación coste-beneficio realmente inmejorable. Esto se debe a varias razones como que todo el proyecto lo realiza una sola persona bajo la supervisión de dos tutores por lo que la mano de obra es muy barata. Otra razón es que casi todo el software utilizado en el proyecto se distribuye mediante la licencia GPL y por lo tanto el coste es cero para la empresa.

La segunda conclusión que podemos extraer del estudio son los múltiples beneficios que tendría la realización del proyecto. La empresa con el sistema desarrollado dispondrá de una herramienta que ahorrará tiempo y costes. Además se eliminarán todas las carencias del sistema actual y se podrá ofrecer un mejor servicio al cliente. Por último podemos destacar que el trabajo de los miembros del equipo de soporte será mucho más automatizado y sobretodo cómodo.

Ante todas estas conclusiones podemos deducir que el proyecto ante el que nos encontramos es perfectamente viable y su realización será un rotundo éxito para la empresa.



## Capítulo 4

### Análisis de alternativas

Una práctica muy recomendable cuando se aborda un proyecto de estas características es hacer una evaluación de las diferentes alternativas que se pueden emplear para el desarrollo del proyecto. A continuación veremos un pequeño resumen de las opciones que se han podido escoger y finalmente justificaremos estas elecciones.

El primer campo donde se presenta la duda entre diferentes alternativas es el que tiene que ver con el almacenamiento de los resultados de la monitorización de las redes. La primera opción es almacenarlos en ficheros log en los servidores clientes. Si lo hacemos de esta manera deberemos exportarlos periódicamente, mediante un canal seguro, al servidor de CCSAgresso donde deberemos analizar y seleccionar la información que nos interese.

La segunda opción es almacenar la información en una base de datos centralizada. El sistema de monitorización Nagios proporciona esta funcionalidad de manera automática mediante un plugin llamado NDOUTils.

El segundo campo donde se produce una dicotomía es el de la arquitectura del sistema de monitorización de redes. La primera opción es la de realizar un sistema con un solo Nagios principal en el servidor de la empresa que trate a todo el conjunto de clientes como una única red.

Otra opción para el tema de la arquitectura del sistema es la utilización de un sistema de monitorización distribuido, es decir, cada servidor cliente tendrá instalado su propio Nagios, el cual enviará los datos al servidor de la empresa. En el servidor de la empresa nos encargaremos de separar todos los datos recibidos de todos los clientes.

Con respecto al almacenamiento de los resultados Se descarta la opción de los ficheros log por la dificultad en el análisis de los mismos ya que su estructura no es para nada trivial. Además el sistema con base de datos está implementado ya en el programa de monitorización Nagios y no requiere de ningún esfuerzo extra. Por lo tanto podemos afirmar que el sistema de almacenamiento en base de datos es bastante más óptimo que el de generación de ficheros log.

En relación a la arquitectura del sistema se tomo la decisión de descartar la opción del sistema de monitorización centralizado. Utilizando este sistema aumentaría la dificultad de discretizar la información recibida de los diferentes clientes. Además un fallo en el servidor de CCS supondría la detención de todas las monitorizaciones de todos los clientes. En cambio, el sistema distribuido nos ofrece mucha más fiabilidad en este aspecto ya que si el servidor de CCS falla podremos recuperar la información de los programas de monitorización instalados en los servidores cliente.

## **Capitulo 5**

### **Sistema desarrollado**

#### **5.1. Nagios: aplicación de monitorización**

Como ya comentamos en el apartado de requerimientos no funcionales del proyecto CCSAgresso eligió como aplicación de monitorización el programa Nagios. Este programa es uno de los más conocidos y utilizados para proyectos de estas características. Seguidamente comentaremos varias de las propiedades de Nagios que han propiciado que finalmente sea el elegido.

Nagios es un sistema open source de monitorización de redes ampliamente utilizado, que vigila los equipos (hardware) y servicios (software) que se especifiquen, alertando cuando el comportamiento de los mismos no sea el deseado. Entre sus características principales figuran la monitorización de servicios de red (SMTP, POP3, HTTP, SNMP...), la monitorización de los recursos de sistemas hardware

(carga del procesador, uso de los discos, memoria, estado de los puertos...), independencia de sistemas operativos, posibilidad de monitorización remota mediante túneles SSL cifrados ó SSH, y la posibilidad de programar plugins específicos para nuevos sistemas.

Se trata de un software que proporciona una gran versatilidad para consultar prácticamente cualquier parámetro de interés de un sistema, y genera alertas, que pueden ser recibidas por los responsables correspondientes mediante (entre otros medios) correo electrónico y mensajes SMS, cuando estos parámetros exceden de los márgenes definidos por el administrador de red.

Fue llamado originalmente Netsaint, nombre que se debió cambiar por coincidencia con otra marca comercial, fue creado y es actualmente mantenido por Ethen Galstad, junto con un grupo de desarrolladores de software que mantienen también varios complementos.

Nagios fue originalmente diseñado para ser ejecutado en GNU/Linux, pero también se ejecuta bien en variantes de Unix. Por último cabe destacar que está licenciado bajo la GNU General Public License Version 2 publicada por la Free Software Foundation.

Por todas estas razones se consideró a Nagios como la herramienta ideal para formar lo que será el núcleo del sistema de monitorización.

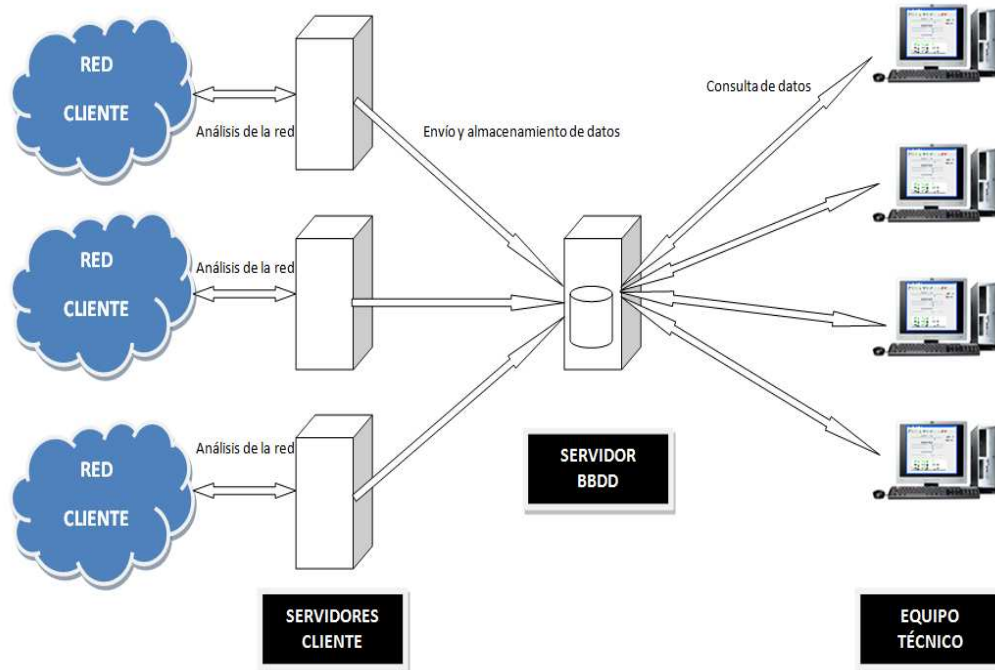
## 5.2. Arquitectura global del sistema

El sistema desarrollado, como podemos ver en la figura 5.1, consta de tres partes bien diferenciadas: la primera, cada uno de los servidores instalados en los clientes; la segunda, el servidor de bases de datos alojado en las instalaciones de CCSAgresso; y por último, cada uno de los equipos de los técnicos dotados de la aplicación de visualización.

La parte de los servidores cliente es la encargada de monitorizar la red del cliente y enviar los datos generados periódicamente al servidor de bases de datos.

El servidor recibe y almacena los datos que le envía cada uno de los clientes. Estos datos son almacenados durante un periodo de tiempo para permitir a la aplicación la consulta y tratamiento de los mismos.

Por último tenemos la parte de la aplicación. Esta está instalada en el equipo de cada técnico y consulta los datos de la base de datos y según estos muestra el estado de cada red cliente prácticamente a tiempo real.



5.1 Arquitectura global del sistema

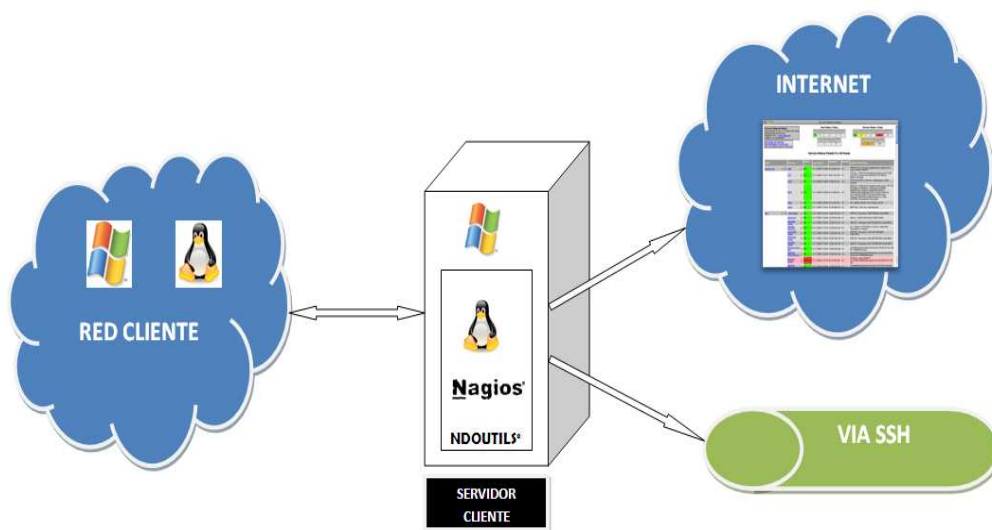
En los siguientes capítulos nos dedicaremos a hacer un análisis más profundo de cada una de las partes de la arquitectura del sistema.

### 5.3. Arquitectura de los servidores cliente

Como hemos introducido en el capítulo anterior en cada cliente tendremos un servidor dedicado única y exclusivamente a la

monitorización de la red y sus servicios.

En cada uno de los servidores estará instalado el programa de monitorización Nagios que será el encargado de escanear periódicamente la red del cliente. Como explicamos en el apartado de requerimientos del capítulo 3 cada servidor cliente funcionará mediante un sistema Windows y dentro de este estará alojada la maquina virtual Ubuntu con la aplicación Nagios. Esta distribución de sistemas operativos se hace de esta manera debido a que Nagios actualmente no es compatible con sistemas Windows y los clientes de CCSAgresso no dominan sistemas operativos tipo Unix.



## 5.2 Arquitectura de los servidores cliente

El funcionamiento de esta parte del sistema es el siguiente:

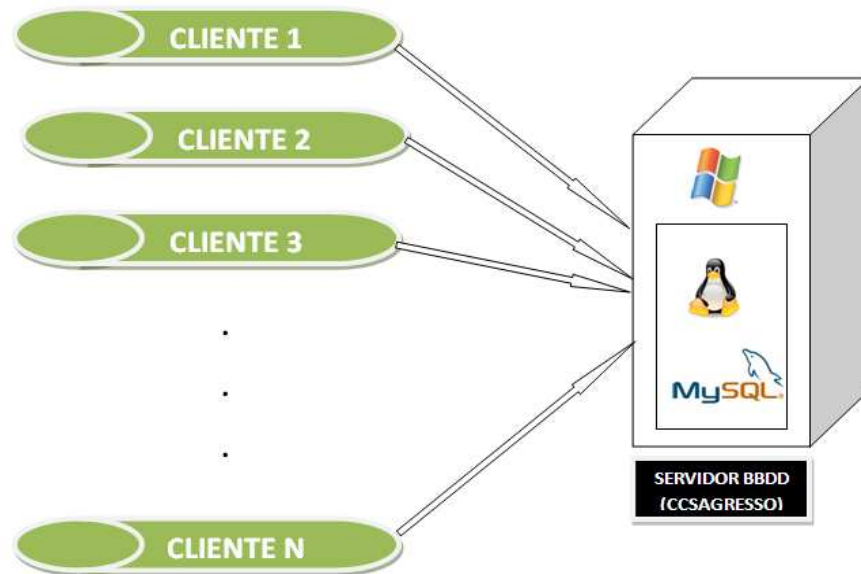
- Nagios escanea la red cliente periódicamente y genera datos con la información del estado de dicha red.
- Esta información es enviada automáticamente vía SSH al servidor central de bases de datos. Cabe destacar que esta funcionalidad la proporciona un plugin de Nagios llamado NDOUtils.
- Por último, como podemos ver en la figura n, Nagios nos proporciona una interfaz Web donde podremos acceder para hacer un análisis más profundo del estado de la red.

### **5.4. Arquitectura del servidor de BBDD**

La segunda parte del sistema es la correspondiente al servidor central de bases de datos. Este servidor estará alojado en las instalaciones de CCSAgresso.

Para el funcionamiento de la base de datos se ha elegido el sistema gestor MySQL, principalmente porque es un sistema de código libre ampliamente conocido y fiable.





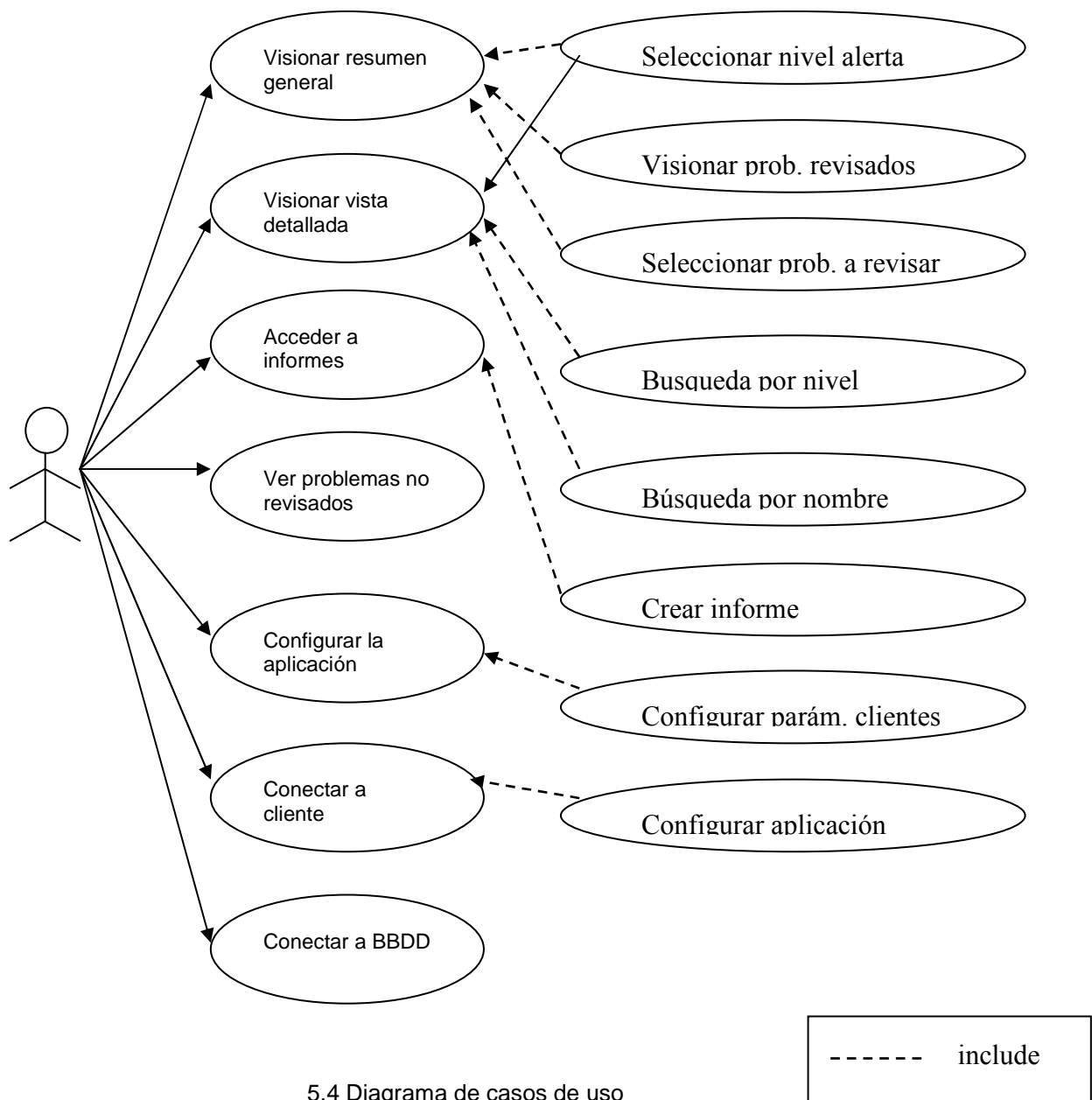
### 5.3 Arquitectura del servidor de BBDD

El funcionamiento de esta parte del sistema es el siguiente:

- La base de datos MySQL recibe de forma periódica los datos generados por los Nagios de los servidores cliente.
- Estos datos serán almacenados para la consulta posterior de la aplicación para visualizarlos, generar históricos,...

## 5.5. Aplicación de visualización

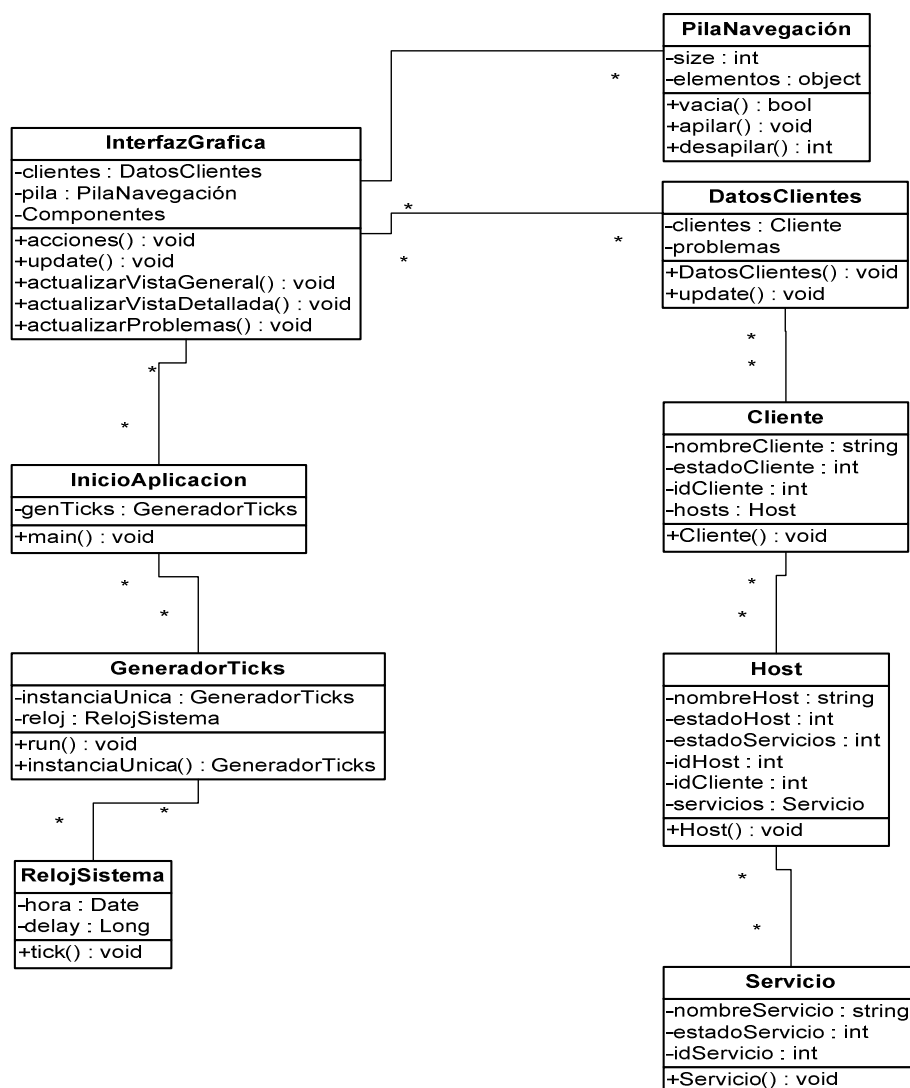
### 5.5.1. Casos de uso



5.4 Diagrama de casos de uso

### 5.5.2. Diagrama de clases

En el siguiente diagrama podemos ver una distribución de todas las clases utilizadas en la aplicación:



5.5 Diagrama de clases

### **5.5.3. Codificación**

En este apartado hablaremos, de forma breve, de todo lo relacionado con la codificación de la aplicación, como puede ser la tecnología utilizada o el paradigma de programación empleado.

La aplicación de visualización esta totalmente desarrollada en tecnología Java. Se hizo la elección de este lenguaje de programación debido a su portabilidad y gran difusión en la informática actual. Gracias a la tecnología Java podemos decir que la aplicación es plenamente compatible con la mayoría de sistemas operativos actuales ya que utiliza una máquina virtual para ejecutarse. También hay que destacar la facilidad en la programación Java debido a que para casi todas las funcionalidades de la aplicación hemos encontrado ejemplos en la API del lenguaje.

Otro punto a comentar dentro de la codificación es el del paradigma de programación utilizado. Como no podía ser de otra forma, al utilizar Java como lenguaje de programación, empleado un paradigma orientado a objetos mediante el cual toda la aplicación esta estructurada a la perfección y facilita la comprensión, seguridad y ampliación futura del código fuente.

## 5.5.4. Codificación

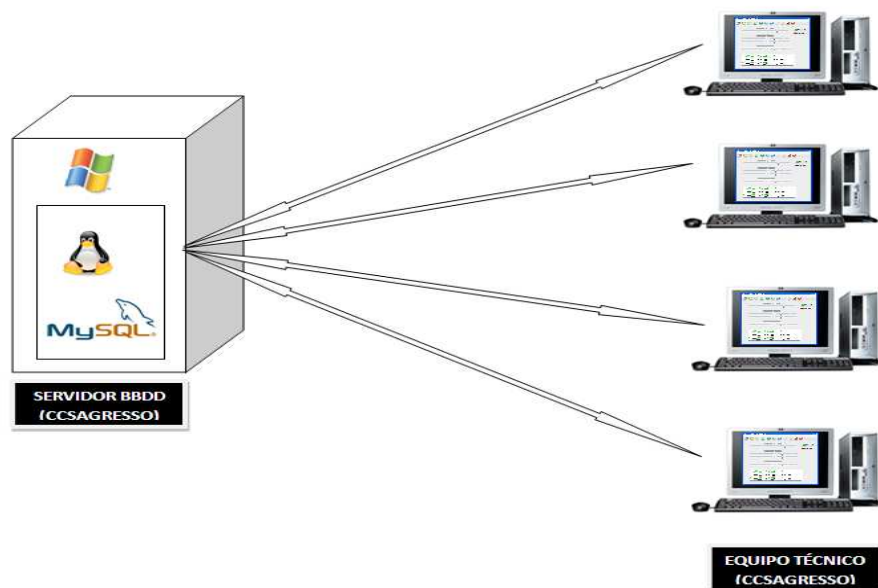
### 5.5.4.1 Descripción de la aplicación

La aplicación, como ya introdujimos en la arquitectura global del sistema y se puede observar en la figura 5.6, estará alojada en cada uno de los equipos de los técnicos de sistemas. Esta consultará permanentemente los datos almacenados en la base de datos y los visualizará de forma sencilla y gráfica para que nuestros técnicos tengan una idea a simple vista de cómo están las redes de los clientes.

La aplicación también es capaz de generar informes, concretamente históricos, para poder saber las incidencias de los clientes, servidores y servicios en un determinado espacio de tiempo.

En los siguientes apartados veremos las diferentes partes de la aplicación que se corresponden con los requerimientos funcionales de CCSAgresso.

5.6 Diagrama de la arquitectura de la aplicación



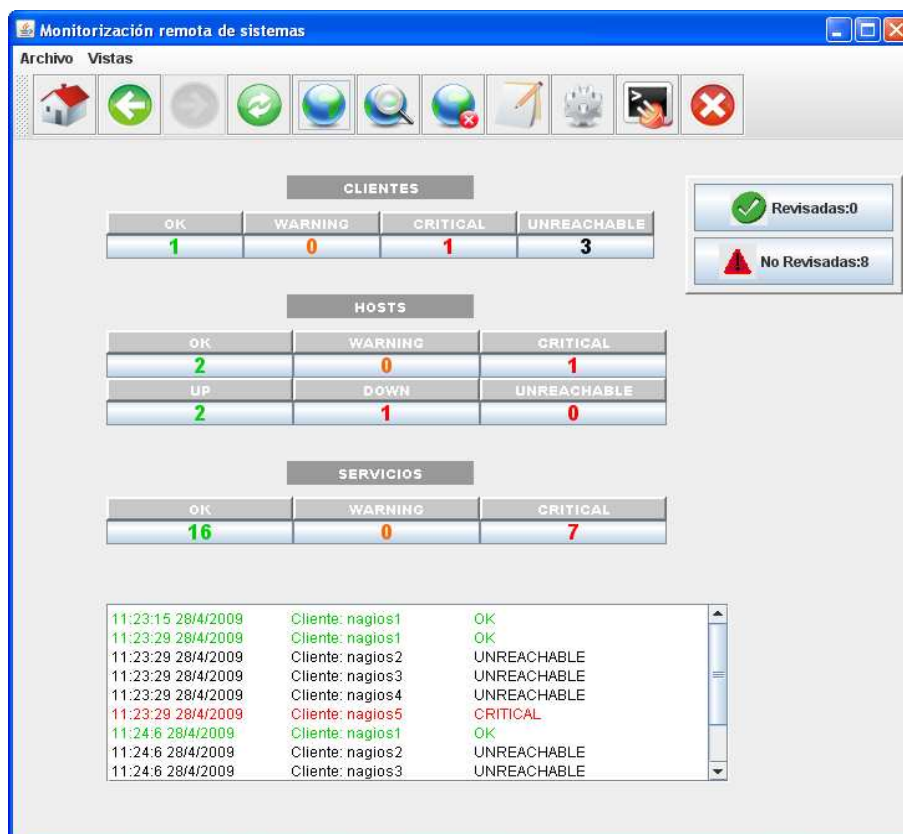
### 5.5.4.2 Vista general

En la captura de pantalla que corresponde con la figura 5.7 podemos ver un resumen general del estado de los clientes, de sus hosts y de sus servicios. Podemos observar que los estados se denotan según diferentes niveles de alerta: Ok, Warning, Critical o Unreachable.

En la parte inferior se puede observar un área de reportes donde se informa de los cambios de estado que se van produciendo.

Por último, en la parte superior derecha tenemos un resumen de las incidencias revisadas y las que quedan por revisar. De esta manera tenemos controlado en todo momento el estado de los problemas y sabremos cuales son los que necesitan un análisis urgente.

5.7 Pantalla de la vista general

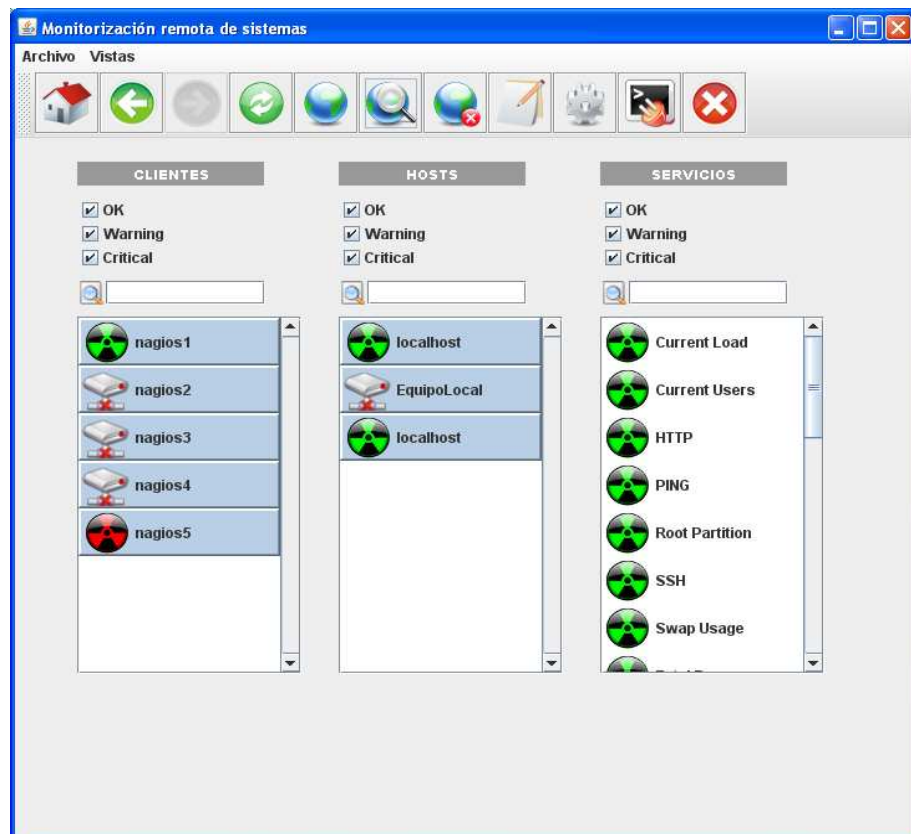


### 5.5.4.3 Vista detallada

Otra funcionalidad que nos proporciona la aplicación es la consulta del estado de un determinado cliente, host o servicio. En la captura de pantalla que se muestra en la figura 5.8 se puede ver como los estados se muestran mediante diferentes iconos según el nivel de alerta correspondiente.

Una segunda funcionalidad que nos proporciona esta parte del software es la de poder realizar búsquedas específicas en el árbol de clientes. Podemos buscar mediante los checks de niveles o mediante la introducción de una cadena de caracteres que coincida con el elemento deseado.

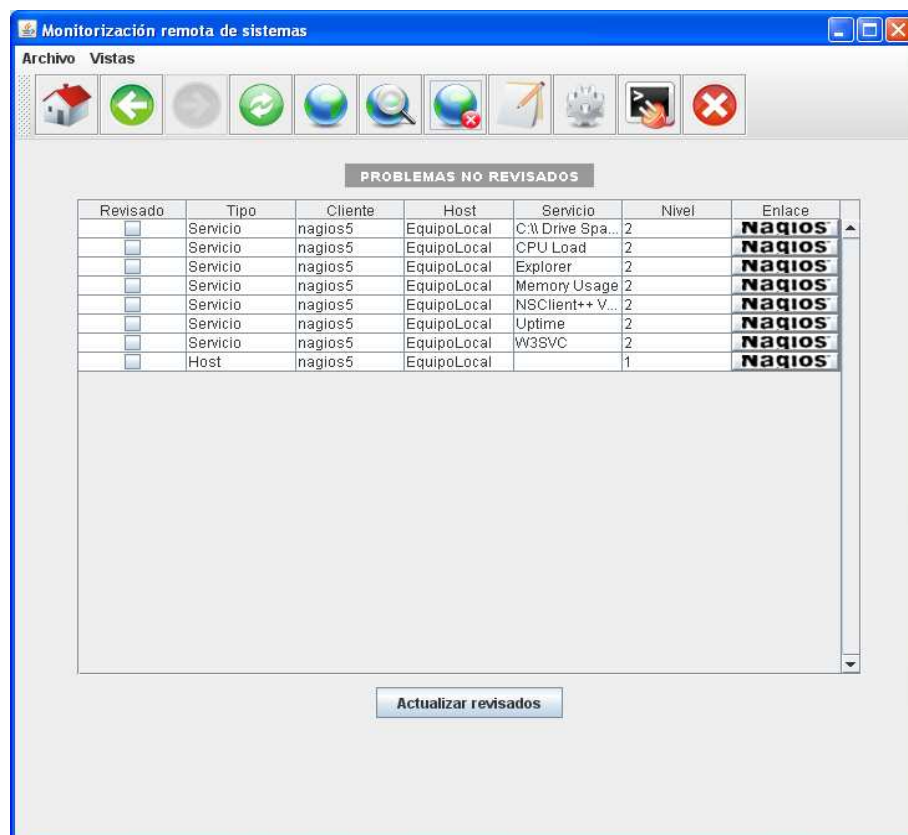
5.8 Pantalla de la vista detallada



#### 5.5.4.4 Vista de problemas

La tercera parte importante del software es la que nos ofrece una relación únicamente de los problemas no revisados. En cada fila de la tabla que vemos en la figura 5.9 se nos detalla que cliente, host y servicio están relacionados con la incidencia en cuestión. Mediante esta tabla podemos proceder a un análisis más profundo de la incidencia accediendo, gracias al enlace de cada fila, al programa de monitorización de cada cliente. Una vez lo hayamos analizado a conciencia podremos activar el check del problema y marcarlo como problema revisado.

5.9 Pantalla de problemas





### 5.5.4.5 Generación de informes

Otra funcionalidad que nos puede ser de gran utilidad es la de generación de históricos que aparece en la figura 5.10. Este panel nos da la capacidad de imprimir tres tipos de históricos: de cliente, de host o de servicio.

En el histórico de clientes obtendremos una relación de las incidencias de todos los hosts y todos los servicios pertenecientes al cliente seleccionado entre las fechas insertadas. De igual manera en el histórico de host generaremos una relación con las incidencias de todos los servicios de ese host entre las fechas de inicio y fin. Por último, el informe más específico es el histórico de servicio, donde se obtendrá una tabla con todos los problemas ocurridos sobre ese servicio entre las dos fechas indicadas.

5.10 Pantalla de generación de informes

Monitorización remota de sistemas

Archivo Vistas

Historico

Historico

Clientes:  Hosts:  Servicios:

nagios1  
nagios2  
nagios3  
nagios4  
nagios5

Desde:  27 / 5 / 2009

Hasta:  31 / 5 / 2009

Generar

Calendario

Mayo 2009

L	M	X	J	V	S	D
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

#### 5.5.4.6 Configuración de la aplicación

Por último, una de las funcionalidades que se requería era que la aplicación fuera configurable en lo que respecta a los datos de los clientes y los datos de la base de datos.

En la figura 5.11 podemos ver que la aplicación también cumple a la perfección esta función. Podemos modificar la IP, usuarios y contraseñas tanto del servidor de cada cliente como del servidor central de base de datos. Además también podemos observar un campo para la modificación de la frecuencia de monitorización. De esta manera podemos controlar en todo momento cada que intervalo nos interesa realizar el escaneo de las redes.

5.11 Pantalla de configuración

Monitorización remota de sistemas

Archivo Vistas

Configuración de parámetros

**Parámetros de clientes**

nagios1  
nagios1  
nagios2  
nagios3  
nagios4  
nagios5

IP: 10.39.43.112

Usuario: root

Password: .....

Guardar

**Parámetros de aplicación**

IP BBDD: 10.39.10.62 Frecuencia: 30000

Usuario BBDD: root

Password BBDD: .....

Guardar

## 5.6. Pruebas

Para comprobar la integridad y fiabilidad del sistema se han hecho fundamentalmente dos grandes conjuntos de pruebas, el primero basado en máquinas virtuales y el segundo sobre máquinas reales.

Para la realización del primer bloque de pruebas se simularon un total de cinco clientes mediante sendas máquinas virtuales Ubuntu. Dentro de cada máquina virtual alojamos la aplicación Nagios configurada para que escanease diferentes servicios de la máquina virtual. La base de datos MySQL también estaba instalada en una máquina virtual Ubuntu.

Una vez toda la configuración estaba hecha y en funcionamiento se inicio la aplicación comprobando principalmente la integridad de la misma. Cuando se detectaron y solucionaron los diferentes problemas se paso a la segunda fase de pruebas.

En la segunda fase los servidores clientes y la base de datos igualmente eran representados por las máquinas virtuales pero ahora Nagios no solo estaba configurado para el análisis de la máquina local sino también se le añadieron varios hosts de la red de CCSAgresso para proceder a su escaneo.

Después de las dos fases de pruebas podemos decir que el producto esta preparado para su implantación y distribución.

## **Capítulo 6**

### **Conclusiones**

#### **6.1. Grado de cumplimiento de los objetivos**

Si recordamos, en el apartado de objetivos del capítulo 1, los objetivos que planteó CCSAgresso se dividían en dos grandes bloques: los objetivos de la arquitectura del sistema y los de la aplicación de visualización.

El objetivo principal de la arquitectura del sistema era que debía poder analizar el estado de las redes de los clientes prácticamente a tiempo real. Como hemos visto en el apartado de descripción de la aplicación podemos visionar los estados de los clientes tanto de forma general como detallada y por lo tanto podemos decir que el objetivo se ha cumplido a la perfección.

Dentro de los objetivos de la aplicación había funcionalidades

que esta debía proporcionar. Algunos de estas funcionalidades eran la generación de informes, configuración de parámetros, posibilidad de conexión directa a los servidores y, por último, visualizar el estado de las redes cliente. Podemos concluir que con todo lo visto en el apartado de explicación de la aplicación todos y cada uno de los objetivos se ha resuelto de forma satisfactoria.

## 6.2. Propuestas de ampliación

En el apartado de mejoras de la aplicación hay un par de ampliaciones que me gustaría comentar.

La primera tiene que ver con la configuración de Nagios. Cuando hablamos de configuración nos referimos a añadir o quitar hosts y servicios a escanear. Actualmente esta configuración se hace modificando los archivos de texto plano que Nagios tiene para este fin. En el mercado existen diversos módulos de código libre que nos proporcionan una interfaz web para la modificación de los parámetros de configuración. De esta forma, incluyendo un acceso desde la aplicación a la interfaz web, podríamos tener también la funcionalidad de la configuración de cada Nagios cliente.

La segunda, y quizás la más fácil de implementar, sería añadir la posibilidad de generación de más tipos de informe. Actualmente solo se generan históricos pero en el futuro nos pueden interesar otros tipos de

reportes como estadísticas de uso, alarmas más destacadas,...

### 6.3. Análisis de la planificación

Al principio de la memoria, en el capítulo del estudio de viabilidad, hicimos una planificación temporal con todas las tareas a realizar durante el proyecto según el modelo de desarrollo.

Una vez acabado el proyecto y habiendo contabilizado el número de horas empleado en cada una de estas tareas podemos hacer un análisis de cuan bueno era la planificación inicial y el porque de las desviaciones.

En la siguiente tabla comparativa tenemos la relación de tiempos esperados y los finalmente obtenidos:

<b>Etapas</b>	<b>Descripción</b>	<b>Duración inicial</b>	<b>Duración final</b>
1	<i>Definición de requerimientos</i>	60 horas	60 horas
2	<i>Análisis y diseño</i>	170 horas	230 horas
3	<i>Construcción del software</i>	200 horas	220 horas
4	<i>Juego de pruebas</i>	70 horas	50 horas
	<b>Total</b>	500 horas	560 horas

6.1 Comparativa de la planificación temporal

Vemos que ha habido una desviación temporal bastante marcada que se debe principalmente a la falta de experiencia en la planificación de proyectos de este tipo y, sobretodo, a las diferentes dificultades encontradas a lo largo del desarrollo del sistema.

## **6.4. Valoraciones personales**

Cuando se me asignó el proyecto “Monitorización remota de sistemas” lo analice y lo vi bastante complejo, no sabía muy bien por donde empezar ya que mis conocimientos sobre redes y programación se podían considerar básicos. Pero gracias a los tutores de la empresa que me fueron guiando mediante reuniones semanales conseguí formar, poco a poco, el sistema que acabo de exponer.

Hacer el proyecto en empresa me ha servido, sobretodo, para madurar profesionalmente de forma acelerada. Ahora sé que puedo desenvolverme con soltura en el mundo de la informática y que puedo continuar mi carrera profesional de forma exitosa.

# Bibliografía

- [1] Wolfgang Barth. Nagios: System and Network Monitoring. No Starch Press, mayo 2006.  
ISBN: 1593270704
  
- [2] Max Schubert. Nagios 3 Enterprise Network Monitoring. Syngress, mayo 2008.  
ISBN: 9781597492676
  
- [3] Wojciech Kocjan. Learning Nagios 3.0. Packt Publishing, octubre 2008.
  
- [4] Chris Burgess. The Nagios Book. NagiosBook.org, mayo 2008.
  
- [5] Bill Lubanovic, Tom Adelstein. Administración de sistemas Linux. Anaya Multimedia, febrero 2005.  
ISBN: 8441522340
  
- [6] Jono Bacon, Jonathan Jesse, Corey Burger, Benjamin Mako Hill. El libro oficial de Ubuntu. Anaya Multimedia, edición 2009.  
ISBN: 8441525080



- [7] Olivier Jan. NAGIOS et la Supervision Open Source. Eni Editions, novembre 2008.

---

Firmado por: Francisco Ortega Belmonte

Sabadell, junio de 2009