Universidad de las Ciencias Informáticas
Facultad 6

Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas

Sistema para monitoreo de eventos en servidores de máxima disponibilidad.

Autor: Rocío de Bárbara Méndez Pérez

Tutor: Yurisnel Corrales Valdés

Ciudad de La Habana, 2011
"Año 53 de la Revolución"

No hay distancia que no se pueda recorrer ni meta que no se pueda alcanzar".
Napoleón Bonaparte

Dedicatoria

… a mis padres quienes confiaron plenamente en mí desde el principio, les dedico mi título que más que mío es de ellos.

…a mis abuelos por brindarme siempre su amor y su apoyo.

…a mi hermano por apoyarme y estar a mi lado en todo momento.

…a mi novio gracias por todo el amor que me brindas…te quiero mucho.

Agradecimientos

A mi papá y a mi mamá *por ser los mejores padres del mundo, por darme todo su amor, por confiar en mí, por sus mejores consejos, por sacrificarse tanto por mi hasta verme graduada y porque a ellos les debo todo lo que soy.*

A mi abuela Flora *por estar siempre conmigo, educarme, y quererme mucho y* ***a mi abuelo Negro*** *que aunque no está ya con nosotros siempre lo llevo en mi corazón.*

A mi hermano *por ser el mejor hermano del mundo, por preocuparse por mí y cuidarme en todo momento.*

A mi novio Orly *que aún en la distancia estuvo muy cerca de mí brindándome todo su apoyo durante este tiempo, por su amor, comprensión y espera. Muchas gracias por compartir conmigo los mejores momentos de mi vida.*

A mis abuelos Inocencia y Eugenio *por su preocupación y por creer en mí.*

A toda mi familia *por ser única, porque siempre están ahí cuando los necesito.*

A mi tía Elsa *por acogerme estos cinco años en su casa y tratarme como si fuera una hija.*

A mis primos *Yaliesky, Roselí, Susana, Mayté, Ian e Iris por ser como hermanos.*

A mis amigas de Carrillo *por apoyarme siempre y confiar en mí, en especial a Mayuli, Liset, Ibelixa y Dania.*

A Dunia *por ser como una tía más para mí y ser en ocasiones mi paño de lágrimas.*

Al grupo de amigos de fin de año *por darme las mejores fiestas.*

A todas las amistades *que durante estos cinco años me han permitido compartir momentos inolvidables.*

A todos mis compañeros de aula *por compartir juntos buenos y alegres momentos, además de reuniones sin opiniones.*

A todos los que me ayudaron en la realización de mi tesis, *en especial al profesor Armando Ortiz, y a mi compañero de proyecto Anyer Gámez.*

A mis amigas inolvidables *Susani, Daya, Harle, Alianis, Yanet, Mailin y Ana por haber compartido conmigo durante todo el tiempo en la universidad y pasar juntas tan buenos momentos. Aunque este sea el*

último año juntas, nos separe la distancia, no tenga correo, ni forma de comunicarnos sepan que siempre las voy a llevar en mi corazón y nunca las voy a olvidar.

Declaración de Autoría

Hoja III]

Declaro que soy el único autor de este trabajo y autorizo al <nombre área> de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

"[Insertar nombre(s) de autor(es)]" "[Insertar nombre(s) de tutor(es)]"

_____ _____

DATOS DE CONTACTO

<Insertar breve currículum e información de contacto del tutor>
<Insertar breve currículum e información de contacto del co-tutor>
<Insertar breve currículum e información de contacto del consultor>
<Insertar breve currículum e información de contacto del asesor>

Resumen

El presente trabajo de diploma propone el desarrollo de un sistema para el monitoreo de eventos en los servidores de máxima disponibilidad existentes en la Universidad de las Ciencias Informáticas. El mismo se enmarca en la realización de una aplicación que permita conocer el estado de los servidores, detectando y enviando alertas a través de un correo electrónico para evitar fallas que puedan afectar la disponibilidad de los servicios.

Incluye una revisión detallada de las herramientas de modelado, la metodología a utilizar en el progreso de la solución y documentación técnica del proceso de desarrollo. La principal novedad consiste en realizar la implementación del sistema de monitoreo para eventos utilizando herramientas libres. El análisis, diseño y posterior implementación de dicho **Sistema para monitoreo de eventos en servidores de máxima disponibilidad** constituye la esencia de este trabajo.

PALABRAS CLAVES: Monitoreo de eventos, Servidores de máxima disponibilidad.

Índice de tablas

Tabla 1 Comparación de las principales soluciones existentes *13*
Tabla 2 Descripción de los actores del sistema. *42*
Tabla 3 Descripción textual del caso de uso del sistema: Configurar_Alertas. *43*
Tabla 4 Descripción textual del caso de uso del sistema: Consultar_Eventos_Ocurridos. *44*
Tabla 5 Descripción textual del caso de uso del sistema: Enviar_Alerta. *45*
Tabla 6 Caminos básicos del flujo *60*
Tabla 7 Caso de prueba del camino básico 1 ..... *60*
Tabla 8 Caso de prueba del camino básico 2 ..... *61*
Tabla 9 Caso de prueba del camino básico 3 ..... *62*
Tabla 10 Caso de prueba del camino básico 4 ..... *63*
Tabla 11 Caso de prueba del camino básico 5 ..... *64*
Tabla 12 Caso de prueba del camino básico 6 ..... *65*

Índice de figuras

Figura 1 Ciclo de vida de AUP *18*
Figura 2 Modelo de dominio *38*
Figura 3 Diagrama de Casos de Uso del Sistema. *43*
Figura 4 Diagrama de Clases del Diseño del CU Enviar_Alerta. *47*
Figura 5 Diagrama de Clases del Diseño del CU Configurar_Alertas. *48*
Figura 6 Diagrama de Clases del Diseño del CU Consultar_Eventos. *48*
Figura 7 Diagrama de Despliegue. *54*
Figura 8 Diagrama de Componentes. *56*
Figura 9 Representación del código del método EnviarAlerta(). *59*
Figura 10 Grafo del flujo asociado al método EnviarAlerta(). *59*

Índice

Introducción *1*
1.1 Introducción *5*
1.2 Conceptos asociados al dominio del problema *5*
1.2.1 El Monitoreo De Servidores *6*
1.3 Análisis de otras soluciones existentes *7*
1.3.1 Weathermap *8*
1.3.2 Real Time Atlas *9*
1.3.3 Nagios *9*
1.3.4 Pandora FMS *11*
1.4 Comparación entre las principales soluciones existentes. *13*
1.5 Conclusiones *15*
2.1 Introducción *15*
2.2 Metodología de desarrollo *15*
2.2.1 Metodología XP (Xtreme Programming) *15*
2.2.2 Microsoft Solution Framework (MSF) *16*
2.2.3 Agile Unified Process (AUP) *18*
2.3 Herramientas Case *20*
2.3.1 Rational Rose *20*
2.3.2 Visual Paradigm *22*
2.4 Lenguaje de programación *23*
2.4.1 Lenguaje de programación Java *24*
2.4.2 Lenguaje de programación C++ *25*
2.4.3 Lenguaje de programación C Sharp *26*
2.5 Framework *28*
2.5.1 QT *28*
2.6 Entorno de desarrollo integrado (IDE) *30*
2.6.1 IDE NetBeans *31*
2.6.2 IDE Eclipse *32*

2.6.3 IDE QT Creator 33
2.7 Conclusiones 35
3.1 Introducción 37
3.2 Análisis de la solución 37
3.2.1 Modelo del dominio 37
Diagrama del Modelo del Dominio 38
3.2.2 Requisitos 39
3.2.3 Modelo de casos de uso del sistema 42
3.3 Diseño de la solución 46
3.3.1 Diagramas de clases del diseño por paquetes 46
3.3.2 Principios del Diseño 49
3.3.3 Patrones de diseño 49
3.4 Conclusiones 51
4.1 Introducción 54
4.2 Modelo de despliegue 54
Descripción de los componentes: 54
Descripción de los protocolos 55
4.3 Modelo de implementación. 55
4.3.1 Diagrama de componentes. 55
4.4 Pruebas del sistema 57
4.4.1 Prueba de caja blanca 57
4.5 Conclusiones 66
Conclusiones generales 68
Recomendaciones 69
Referencias Bibliográficas 70
Bibliografía 74

Introducción

En el mundo actual las tecnologías de la información forman parte indispensable en la vida del ser humano. La informática a nivel mundial se conoce como la nueva revolución tecnológica, pues han sido numerosos los avances alcanzados en dicha materia, debido a la necesidad de información y de nuevos conocimientos que tiene el mundo actual, no únicamente para el desarrollo de los pueblos, sino también para el de las nuevas tecnologías.

Cuba va aparejada al creciente desarrollo de las tecnologías de la información, aportando resultados significativos en las diversas esferas. Se desarrollan varios proyectos productivos en las diferentes provincias del país que revelan el conocimiento y habilidades de los profesionales que se forman hasta el momento en Cuba. En Villa Clara se encuentra el Centro de Innovación Tecnológica (CITEC), con sede en la Universidad Central de Las Villas Marta Abreu, que tiene a su cargo la realización de varios proyectos sobre medio ambiente, ahorro energético, turismo, administración de empresas, etc. ofertando una gran cantidad de productos.

Se encuentra también el Centro de Estudios de Diseño y Fabricación Asistidos por Computadoras (CAD/CAM) en la Universidad de Holguín, vinculado a las plantas productoras de Níquel en el territorio, el abasto de agua a la población, el perfeccionamiento del diseño de implementos agrícolas, la biomecánica computacional. En Cienfuegos el Centro de Información y Gestión Tecnológica (CIGET) brindando entre sus principales servicios informáticos: servicios de implementación de software, diseño, programación e implementación de sistemas automatizados de información, diseño y organización de Intranets, etc.

La Universidad de las Ciencias Informáticas (UCI) ocupa un lugar destacado en la producción de software en el país, en ella se han desarrollado múltiples proyectos productivos que benefician el desarrollo económico y tecnológico del país, y ayudan a la adquisición de nuevos conocimientos. Dicha universidad está estructurada en varias facultades en su sede principal y 3 facultades regionales, todas tienen diversas líneas de investigación y producción, una de ellas es la Facultad 6, en la cual se encuentra el Centro de Geoinformática y Señales Digitales (GEYSED), teniendo como uno de sus principales

departamentos el de Señales Digitales que se dedica a la investigación y desarrollo de software en el área de video y sonido digital.

La tecnología de servidores es un tema suficientemente empleado en diversas empresas del país y particularmente en la UCI en dicho departamento. Cuando un equipo pierde la conectividad, se pierde el acceso a los equipos conectados a él generándose entonces un conjunto de fallas. El monitoreo de eventos en servidores surge para dar solución a dichos problemas encargándose de medir y controlar el funcionamiento de los equipos que están conectados a una red, surgiendo con ello una cantidad importante de protocolos y aplicaciones que reúnen la información de los estados de los equipos y lo presentan al administrador de la red.

En la Universidad de las Ciencias Informáticas existen diversos servidores, incluyendo los servidores de máxima disponibilidad como son el servidor de correo y el servidor de la intranet. Hoy día, en el centro se necesita asegurar la continuidad operacional de aplicaciones de misión crítica, y para ello es de gran importancia conocer la calidad de operación, eficiencia y productividad de los servidores, garantizando una mayor rapidez, confidencialidad y seguridad en los mismos. Es por ello que una herramienta que permita conocer el estado de los equipos de red, o sea, de los servidores, es indispensable para detectar y alertar de las posibles fallas que puedan producirse. Actualmente no existe implementado en el centro un sistema que monitoree los eventos de dichos servidores.

Por lo anteriormente expuesto se plantea el siguiente **problema a resolver:** Ineficiente control y documentación de los eventos en servidores de máxima disponibilidad en la Universidad de las Ciencias Informáticas. Considerando como **objeto de estudio** los procesos para monitoreo de eventos en servidores de máxima disponibilidad, y tomando como **campo de acción** la automatización de los procesos en servidores de máxima disponibilidad en la Universidad de Ciencias Informáticas (UCI), se definió el siguiente **objetivo general:** Desarrollar un sistema para monitoreo de eventos en servidores de máxima disponibilidad en la Universidad de las Ciencias Informáticas.

Las siguientes **tareas de la investigación** están dirigidas a dar cumplimiento a los objetivos del trabajo:

- Caracterizar y valorar los logros y limitaciones en los enfoques existentes sobre el desarrollo de sistemas de monitoreo a servidores de máxima disponibilidad.

- Caracterizar las herramientas más utilizadas en Cuba y el mundo para el monitoreo a servidores de máxima disponibilidad.
- Valorar las herramientas a utilizar en la construcción del sistema.
- Argumentar la Metodología de Desarrollo de software a usar en el proceso.
- Describir las funcionalidades que debe brindar el sistema.
- Diseñar el sistema.
- Implementar el sistema.
- Realizar las pruebas del sistema

Sucesivo a lo anterior expuesto se puede plantear la **idea a defender:** Si se realiza la implementación de un sistema para monitoreo de eventos en servidores de máxima disponibilidad se lograría identificar y detectar posibles errores o fallos que afecten a la disponibilidad de los servicios en los servidores de la Universidad de las Ciencias Informáticas.

Para la realización de este trabajo fue necesario consultar y analizar bibliografía por diferentes vías de investigación y para ello se consultó los métodos científicos de investigación llegando a la conclusión que se utilizaron los que a continuación se describen.

Métodos científicos de investigación:

- ✓ **Teóricos:**

Analítico - sintético: se utilizó para estudiar los temas relacionados con el desarrollo de sistemas para el monitoreo de eventos en servidores de máxima disponibilidad, así como su evolución en la Universidad de las Ciencias Informáticas (UCI), además de la metodología a utilizar en el desarrollo.
Análisis histórico-lógico: fue utilizado para el análisis de documentos, materiales, y temas relacionados a las mejores prácticas en el desarrollo de sistemas para el monitoreo de eventos en servidores de máxima disponibilidad; además para definir la metodología a utilizar.
Modelación: se empleó para modelar el sistema para el monitoreo de eventos en servidores de máxima disponibilidad.
Inductivo-Deductivo: se utilizó porque permite obtener conocimientos generalizados de los sistemas para el monitoreo de eventos en servidores, partiendo desde el análisis de lo particular a lo general.

- ✓ **Empíricos:**

Entrevistas: se utilizó para obtener la información acerca de las necesidades de información, requisitos funcionales y no funcionales con los que debe cumplir la solución. Se entrevistó al líder de proyecto Heliodoro Rodríguez Milián del proyecto Video Vigilancia.

Capítulo 1: Fundamentación teórica.

1.1 Introducción

Este capítulo abarca aspectos y elementos teóricos importantes de la tecnología de servidores, en específico del monitoreo y el control de los eventos en servidores de máxima disponibilidad, y se analizan conceptos y elementos básicos relacionados con el tema, además se estudian soluciones existentes que pueden aportar aspectos importantes en la construcción del sistema a desarrollar.

1.2 Conceptos asociados al dominio del problema

En la actualidad la noción de servidor está asociada al campo de la tecnología, **un servidor** es una computadora que forma parte de una red y que provee servicios a otras computadoras, que reciben el nombre de cliente. Los servidores suelen utilizarse para almacenar archivos digitales. El cliente por lo tanto se conecta a través de la red con el servidor y accede a los archivos en cuestión. En ocasiones la computadora puede cumplir con la funciones de servidor y de cliente de manera simultánea. Entre los distintos tipos de servidores, pueden destacarse los servidores de archivos, los servidores de correo y los servidores web.

Un servidor es un ordenador central que se encarga de prestar un servicio a otros ordenadores que se conectan a él; una aplicación informática o programa que realiza algunas tareas en beneficio de otras aplicaciones llamadas clientes; una computadora en la que se ejecuta un programa que realiza alguna tarea en beneficio de otras aplicaciones llamadas clientes. **(Definicion.de, 2011)**

Un servidor no es necesariamente una máquina de última generación de grandes proporciones, no es necesariamente un superordenador; un servidor puede ser desde una computadora vieja, hasta una máquina sumamente potente (ej.: servidores web, bases de datos grandes, etc. y hasta varios gigabytes de memoria). Todo esto depende del uso que se le dé al servidor. Si lo desea, puede convertir al equipo desde el cual usted está leyendo esto en un servidor instalando un programa que trabaje por la red y a la que los usuarios de su red ingresen a través de un programa de servidor web como Apache.

Servidores de máxima disponibilidad: es la o las máquinas que exigen el máximo de rendimiento y eficiencia para desarrollar una o varias funciones.

Monitoreo: consiste en la observación del curso de uno o más parámetros para detectar eventuales anomalías. **(Definicion.de, 2010)**

El monitoreo es el seguimiento sistemático y periódico de la ejecución de una actividad o proyecto para verificar el avance en la ejecución de la Meta Física (eficacia), la adecuada utilización de recursos para lograr dicho avance (eficiencia) y la consecución de los objetivos planteados durante el proceso de ejecución (efectividad), con el fin de detectar, oportunamente, deficiencias, obstáculos y/o necesidades de ajuste. **(scribd.com, 2010)**

1.2.1 El Monitoreo De Servidores

Para muchas funciones críticas de comercio en línea así como la gran gama de servicios que las empresas hoy día pueden brindar a sus clientes, la disponibilidad y el grado de respuesta de estos servicios como lo son servidores, aplicaciones Web, y demás componentes de red son de vital importancia para el funcionamiento de la organización. Las aplicaciones críticas dependen en gran medida de la disponibilidad de la infraestructura que es soportada por las tecnologías de información existentes, especialmente de los servidores, servicios y sistemas de comunicación, los cuales en su conjunto entregan el soporte necesario a las organizaciones actualmente.

La naturaleza de los servidores exige de un máximo de vigilancia, por consiguiente para mantener un control y documentación de los eventos que ocurren en los servidores de máxima disponibilidad se necesita de una aplicación o una herramienta que se encargue de ello, es ahí donde surge el término de monitoreo de servidores. Con monitorización clásica de sistemas se entiende monitorizar, por ejemplo:

- El espacio en disco, la carga del procesador o la memoria libre de un servidor.
- El ancho de banda de la salida a Internet, los puertos caídos o el master de HSRP de un equipo de comunicaciones.

- Si hay respuesta a PING de un equipo, si funciona el correo o si funciona una página WEB determinada.

El monitoreo de servidores concibe entre sus funciones principales:

- **Capacidad de planeación:** Anticipar los problemas que puedan ser causados por la falta de recursos (disco duro, memoria, capacidad de procesamiento) en sus servidores, aplicativos y Bases de Datos dimensionando sus plataformas en forma adecuada y con datos precisos.
- **Disponibilidad:** La información recopilada en los servidores debe estar disponible en todo momento que el cliente la necesite.
- **Detección de errores:** Identificar de forma rápida y efectiva las causas de los problemas que afectan el rendimiento del servidor.
- **Accesibilidad:** Visualizar la información requerida de sus monitores en el momento y lugar que sea necesario.

Los servicios en un servidor pueden fallar por dos motivos:

- Mal comportamiento de software.
- Mal comportamiento de hardware.

El mal comportamiento de hardware puede estar dado por un equipo que esté funcionando deficientemente por algún problema con la tarjeta de red, el disco duro, u otro dispositivo de hardware. El mal comportamiento de software proporciona la interrupción de servicios, obviamente los servicios los proporciona una máquina, y redirigir el servicio y su actividad a una máquina en perfecto estado es el reino de la **Alta Disponibilidad.**

1.3 Análisis de otras soluciones existentes

A nivel internacional existen numerosas herramientas para el monitoreo de eventos en servidores, algunas de ellas desarrolladas en software libre facilitando la copia, modificación y uso con total libertad. Se estudiaron aplicaciones con distintos estilos de monitoreo para analizar sus ventajas y desventajas. A continuación se proporciona una información detallada de cada una de las aplicaciones referidas.

1.3.1 Weathermap

***WeatherMap* o "*Estado climático de la red*"** es una analogía a una carta sinóptica donde se puede apreciar el comportamiento climático de una zona geográfica. Llevando la analogía al monitoreo de redes de telecomunicaciones, es una herramienta que muestra el estado global de la red, mostrando el nivel de utilización de ancho de banda de los principales enlaces que la conforman. A continuación se mencionan 2 herramientas *WeatherMap*. **(Arriagada, 2011)**

WeatherMap 4RRD

Herramienta escrita en lenguaje PHP, adquiere los valores de utilización de ancho de banda desde una base de datos RRDTool o archivos HTML de MRTG y se muestra con flechas coloreadas sobre un mapa que representa la topología de la red.

Network WeatherMap

Herramienta escrita en lenguaje PERL, los datos provienen de gráficos creados por MRTG y son mostrados con flechas de doble punta sobre un mapa que representa la topología de la red. La imagen resultante es presentada en una página web utilizando DHTML y JavaScript sobre un servidor Web con soporte de OverLib JavaScript.

Estas aplicaciones son sencillas de utilizar e implementar, pero ofrecen pocas funcionalidades y opciones. Las herramientas WeatherMap son bastante básicas con respecto al monitoreo, recolectan una sola variable de los equipos, la que simplemente es desplegada por pantalla, sin un procesamiento que permita ayudar a determinar la causa de las fallas.

Ventajas:

- Fácil de implementar.
- Sencilla de utilizar.
- Genera una visión global de la red.

Desventajas:

- Posee redundancia de alertas.
- No tiene métodos de alarma.
- Es difícil encontrar el motivo de las fallas.

1.3.2 Real Time Atlas

Real Time Atlas es una herramienta que combina mapas cartográficos, la topología de la red de telecomunicaciones y la información recolectadas de los equipos. Fue desarrollado por el centro de operaciones e investigaciones de redes globales (*GRNOC)* de la Universidad de Indiana. El sistema integra herramientas desarrolladas internamente, como la base de datos *GRNOC*, el sistema de información geográfica *UMN Mapserver* y las bibliotecas de proyecciones cartográficas y abstracción geoespacial *GDAL*. **(Arriagada, 2011)**

Ventajas:

- Sencilla de utilizar.
- Genera una visión global de la red.
- Incorpora más variables al monitoreo.
- Utiliza aplicaciones para mediciones de retardo, análisis multicast y trafico BGP.

Desventajas:

- Posee redundancia de alertas.
- No tiene métodos de alarma.
- Es difícil encontrar el motivo de las fallas.
- Utiliza paquetes privados.

1.3.3 Nagios

Nagios es un avanzado sistema de monitoreo de código abierto líder, que permite a las organizaciones identificar y resolver los problemas de infraestructura de Tecnologías de la Información (TI) antes de que afecten los procesos de negocio críticos. Supervisa de forma activa toda la infraestructura de TI, redes, servidores, aplicaciones, proporciona además sofisticados sistemas de alarma y notificación escalado para alertar a los administradores de potenciales problemas que requieren atención.

Esta monitorización permite a los administradores de sistemas abstraerse de la vigilancia continua, permitiéndoles desempeñar otras funciones y áreas sin tener que estar constantemente revisando que todo está funcionando. Para la recepción de alarmas, la aplicación es bastante flexible puesto que éstas pueden recibirse mediante correo electrónico. SMS, a través de un servidor de mensajería Jabber o utilizando un *plugin* para Thunderbird. Dentro de las capacidades de la aplicación se encuentra la de

monitorizar servicios de red, la gestión vía SNMP (que quizás sea uno de los aspectos más importantes) o la monitorización de recursos hardware (carga del procesador, espacio en disco, memoria, estado de los puertos, etc). Cualquier sistema que soporte SNMP es susceptible de ser monitorizado con Nagios (*switches*, *routers*, puntos de acceso, servidores de cualquier tipo, etc). **(nagios.org., 2011)**

La visualización de la información es un aspecto muy interesante, incluyéndose una vista global con el estado de todos los servicios definidos para cada elemento, *host*, el estado de cada *host* o la lista de *host* con problemas. Además, se incluye una sección de informes de disponibilidad, fundamental para verificar que los niveles de servicio obtenidos están dentro de los parámetros de funcionamiento definidos.

Características:

- **Monitoreo:** Todos los componentes de la infraestructura de misión crítica, incluidas las aplicaciones, servicios, sistemas operativos, protocolos de red, parámetros del sistema, y la infraestructura de red son monitoreados.
- **Visibilidad:** Provee de vista central de toda su red de operaciones de TI.
- **Notificaciones:** Las alertas se envían a personal de TI a través de correo electrónico, avisos vía celular con mensajes SMS o cualquier método definido por el usuario. Las alertas pueden ser escaladas para llegar a la gente adecuada.
- **Resolución proactiva:** Brinda la capacidad ejecutar scripts y procesos de mantenimiento para solucionar los problemas, como reinicio de las aplicaciones, sistemas operativos etc. **(nagios.org., 2011)**
- **Reportes:** Proporciona los registros históricos de las interrupciones, las notificaciones, y la respuesta de alerta para su posterior análisis. Utiliza una interfaz basada en web. **(nagios.org., 2011)**
- **Arquitectura modular:** Existen cientos de módulos, scripts y aplicaciones desarrollados por terceros que extiende su funcionalidad, cualquier script desarrollado puede integrarse con Nagios. **(nagios.org., 2011)**
- **Redundancia:** Soporte para implementar hosts de monitores redundante.

Ventajas:

- Sencilla de utilizar.
- Genera una visión global de la red.
- Completo conjunto de herramientas de monitoreo.
- Incorpora método de alarma.

Desventajas:

- Posee redundancia de alertas.
- Es difícil encontrar el motivo de las fallas.

1.3.4 Pandora FMS

Pandora FMS es un software de Código Abierto que sirve para monitorizar y medir todo tipo de elementos. Monitoriza sistemas, aplicaciones o dispositivos. Permite saber el estado de cada elemento de un sistema a lo largo del tiempo. Pandora FMS puede detectar si una interfaz de red se ha caído, una pérdida de memoria en algún servidor de aplicaciones, etc. Pandora FMS puede enviar SMS si un sistema falla. Pandora FMS puede recoger información de cualquier sistema operativo, con agentes específicos para cada plataforma, que recolectan datos y los envían al servidor. Hay agentes específicos para GNU/Linux, AIX, SUN Solaris, HP-UX, BSD/IPSO y Windows 2000, XP y 2003.

Pandora FMS también puede monitorizar cualquier tipo de servicio TCP/IP, sin necesidad de instalar agentes, y monitorizar sistemas de red como balanceadores de carga, routers, switches, sistemas operativos, aplicaciones o impresoras si se necesita hacerlo de forma remota. Pandora FMS también soporta SNMP para recolectar datos. **(pandora.sourceforge.net, 2011)**

Funcionalidades:

- **Detección de topología de red y autodescubrimiento:** Pandora FMS es capaz de reconocer y detectar periódicamente nuevos sistemas no monitorizados, detectando su sistema operativo y

perfil, basado en puertos TCP y asignándolos a una plantilla específica de monitorización de red, dependiendo de la red, sistema operativo o perfil de puerto. El Recon server también detecta topología de red e intentará "juntar" el padre más directo conocido al nuevo host. **(pandora.sourceforge.net, 2011)**

- **Monitorización de funcionamiento y disponibilidad:** Pandora FMS proporciona una solución completa para funcionamiento y disponibilidad, monitorizando los recursos claves a través de la infraestructura, para asegurarse de que todos los dispositivos están listos para responder a los requerimientos del usuario final. **(pandora.sourceforge.net, 2011)**
- **Gestión de errores y eventos:** El sistema de eventos de Pandora FMS mantiene un log de todo lo que ha sucedido: cuando un servicio o un host se cae, o cuando se recupera, cuando se dispara una alerta, cuando se descubren nuevos hosts en la red, etc. Es posible buscar eventos, filtrándolos por grupo, tipo, severidad o status del evento. Todo esto se hace desde la Consola Web. (pandora.sourceforge.net, 2011)
- **Alta disponibilidad:** Pandora FMS tiene una estructura basada en servidores múltiples (Data Server, Plugin Server, Network Server,...), una consola Web y una Base de datos. Tiene redundancia sobre todos sus sistemas. Se puede crear cualquier cantidad de servidores o consolas, así como un clúster MySQL para la Base de datos.

Otras funciones:

- Detectar nuevos sistemas en la red.
- Test de disponibilidad o rendimiento.
- Disparar alertas cuando algo va mal.
- Permite obtener información dentro de los sistemas usando sus propios agentes (disponibles para todos los sistemas operativos del mercado).
- Permite obtener datos remotos usando peticiones de red, incluyendo SNMP.
- Recibir Traps SNMP de dispositivos de red genéricos.
- Generar informes y gráficos en tiempo real.
- Informes SLA (Acuerdo Nivel Servicio).
- Almacenamiento de datos de meses, listos para ser usados en informes.
- Gráficos en tiempo real para cada módulo.

- Alta disponibilidad en todos sus componentes.
- Arquitectura escalar y modular.
- Soporta más de 2500 módulos por servidor.
- Multiusuario, perfiles múltiples, agrupación de agentes.
- Sistema de eventos con validación por el usuario para trabajo en equipo.
- Acceso granular y perfiles de usuario para cada grupo.
- Los perfiles pueden ser personalizados usando hasta ocho atributos de seguridad sin limitación de grupos o perfiles.

Arquitectura:

Pandora FMS es una herramienta muy versátil y modular, y permite trabajar de varias maneras. De forma resumida podemos decir que Pandora FMS trabaja tanto con monitorización remota como con monitorización basada en agentes, y que por supuesto permite combinar ambas. Pandora FMS está desarrollado en diferentes lenguajes: C++ y Perl para los agentes, Perl en el servidor, y PHP/JavaScript en la consola WEB. (pandora.sourceforge.net, 2011)

Pandora FMS tiene un diseño modular, basado en varios subservidores específicos para cada tipo de chequeo. Todos sus componentes son redundantes y pueden funcionar en HA Activo/Activo.

1.4 Comparación entre las principales soluciones existentes.

Después de hacer un análisis exhaustivo de todas las aplicaciones estudiadas se comprobó que existe una gran similitud entre las herramientas de monitoreo Nagios y Pandora FMS, a continuación se demuestra esta afirmación con una tabla de comparación entre ellas.

Tabla 1 Comparación de las principales soluciones existentes

Características	Nagios	Pandora FMS
Arquitectura	Arquitectura modular, desarrollado en PHP	Arquitectura modular y escalar, desarrollado en C++, Perl y PHP/JavaScript.

Visibilidad	Genera una visión global de la red.	Gráficos en tiempo real para cada módulo.
Licencia	GPL2	GPL2 y Licencia Artica ST Enterprise
Disponibilidad	Incluye una sección de informes de disponibilidad.	Alta disponibilidad en todos sus componentes.
Configuración	Poco amigable.	Sencilla para trabajar con ella.

1.5 Conclusiones

Posterior al estudio de las soluciones existentes relacionadas de una forma u otra con el tema de nuestro trabajo de diploma se demostró la semejanza entre Pandora FMS y Nagios, aprovechando que son herramientas libres y debido a las características comunes de estas dos aplicaciones se seleccionó Pandora FMS como una guía en el proceso de creación de una herramienta para monitoreo de eventos en servidores de máxima disponibilidad, con tipologías específicas requeridas en la Universidad de las Ciencias Informáticas.

Capítulo 2: Tecnologías y herramientas utilizadas en el desarrollo del sistema

2.1 Introducción

Luego de realizado un análisis del estado del arte de las principales soluciones existentes relacionadas con el monitoreo de servidores. En el presente capítulo se analizarán y justificarán las tecnologías y herramientas empleadas en el análisis, diseño e implementación de la solución propuesta, como son las metodologías de desarrollo, las herramientas CASE, el IDE y lenguaje de programación a utilizar.

2.2 Metodología de desarrollo

Todo desarrollo de software es muy riesgoso además de ser difícil de controlar, y si no se emplea una metodología que guíe este proceso, los resultados que se obtendrán serán clientes insatisfechos y desarrolladores aún más descontentos. Hoy día en todo el mundo, se proponen diferentes metodologías en dependencia del tiempo de vida y la complejidad del proyecto que se vaya a desarrollar. Una metodología es una agrupación de técnicas y procedimientos que permiten conocer los elementos necesarios para definir un proyecto de software, es la base para la edificación de un producto de este tipo, fundamentalmente, sirve para elevar la calidad del software.

Las metodologías, dadas sus características, se enmarcan en dos grandes grupos, los llamados "métodos pesados" y los "métodos ágiles". La diferencia más notable entre estos dos grupos es que mientras los métodos pesados intentan obtener los resultados apoyándose principalmente en la documentación ordenada y persiguen la consecución de un proyecto cuya planificación está bien definida, los métodos ágiles tienen como base de sus resultados la comunicación e interacción directa con todos los usuarios involucrados en el proceso además están pensadas para pequeños grupos de personas.

2.2.1 Metodología XP (Xtreme Programming)

Xtreme Programming (XP), se basa en el trabajo orientado directamente al objetivo, basándose para esto en las relaciones interpersonales y en la velocidad de reacción para la implementación y para los cambios que puedan surgir durante el desarrollo del proceso. **(Jack, 2007)**

Es una de las metodologías de desarrollo de software más exitosas en la actualidad utilizada para proyectos de corto plazo y equipo. La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. **(Jack, 2007)**

Características de XP:

- **Pruebas Unitarias:** se basa en las pruebas realizadas a los principales procesos, de tal manera que si se adelanta en algo hacia el futuro, se podrán hacer pruebas de las fallas que pudieran ocurrir. Es como adelantar los posibles errores.

- **Refabricación:** se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.

- **Programación en pares:** una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento. Es como el chofer y el copiloto: mientras uno conduce, el otro consulta el mapa. **(Informatizate, 2010)**

Lo fundamental en este tipo de metodología es:

- La comunicación, entre los usuarios y los desarrolladores.
- La simplicidad, al desarrollar y codificar los módulos del sistema.
- La retroalimentación, concreta y frecuente del equipo de desarrollo, el cliente y los usuarios finales.

2.2.2 Microsoft Solution Framework (MSF)

Esta es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos. MSF se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológicas. MSF se compone de varios modelos encargados de planificar las diferentes partes implicadas en el

desarrollo de un proyecto: Modelo de Arquitectura del Proyecto, Modelo de Equipo, Modelo de Proceso, Modelo de Gestión del Riesgo, Modelo de Diseño de Proceso y finalmente el modelo de Aplicación.

Esta metodología proporciona un sistema de modelos, principios, y pautas para dar soluciones a empresas que diseñan y desarrollan de una manera que se asegure de que todos los elementos de un proyecto, tales como personas, procesos y herramientas, puedan ser manejados con éxito. Además propone una secuencia generalizada de actividades para la construcción de soluciones empresariales. El MSF combina los mejores principios del modelo en cascada y del modelo en espiral. Combina la claridad que planea el modelo en cascada y las ventajas de los puntos de transición del modelo en espiral.

MSF tiene las siguientes características:

- **Adaptable:** es parecido a un compás, usado en cualquier parte como un mapa, del cual su uso es limitado a un específico lugar.
- **Escalable:** puede organizar equipos tan pequeños entre 3 o 4 personas, así como también, proyectos que requieren 50 personas o más.
- **Flexible:** es utilizada en el ambiente de desarrollo de cualquier cliente.
- **Tecnología Agnóstica:** porque puede ser usada para desarrollar soluciones basadas sobre cualquier tecnología. **(Informatizate, 2010)**

El Modelo de proceso MSF consta de cinco fases distintas:

- Previsión
- Planeamiento
- Desarrollo
- Estabilización
- Implementación

2.2.3 Agile Unified Process (AUP)

El Proceso Unificado Ágil de Scott Ambler ó Agile Unified Process (AUP) en inglés es una versión simplificada del Rational Unified Process (RUP). Este describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software usando técnicas ágiles y conceptos y aún así se mantiene fiel a las RUP. El AUP aplica técnicas ágiles incluyendo el desarrollo dirigido por pruebas, el modelado ágil, la gestión de cambios ágil, y la refactorización de base de datos para mejorar la productividad. En la figura 1 se muestra el ciclo de vida de dicha metodología. **(Scott W. Ambler, 2011)**

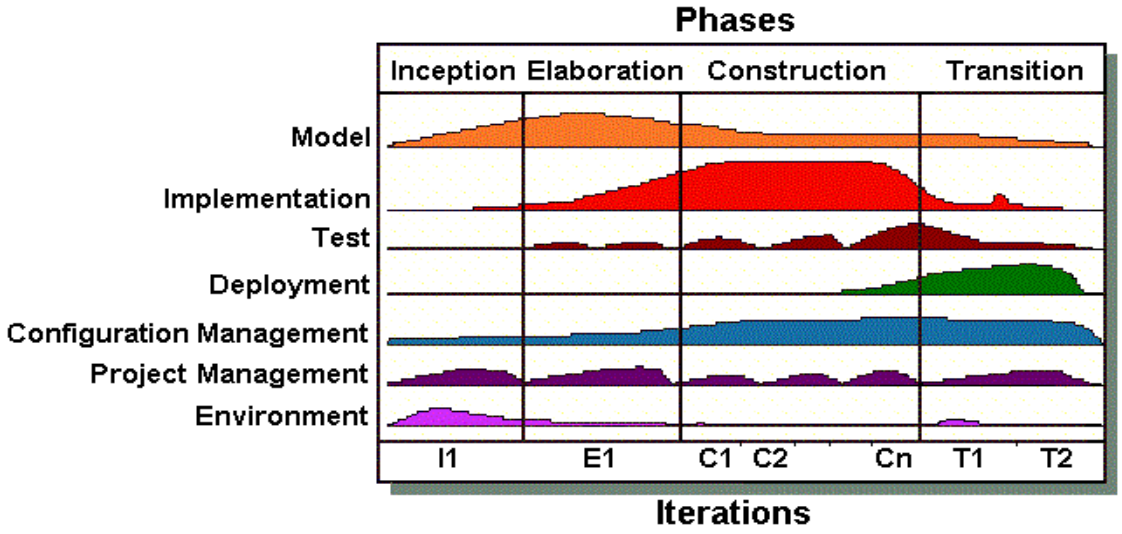

Figura 1 Ciclo de vida de AUP

La naturaleza de AUP se refleja en sus 4 fases:

1. **Creación:** El objetivo es identificar el alcance inicial del proyecto, una arquitectura potencial de su sistema, y para obtener la financiación inicial del proyecto y la aceptación de las partes interesadas.
2. **Elaboración:** El objetivo es probar la arquitectura del sistema.
3. **Construcción:** El objetivo es la construcción de software de forma regular e incremental respondiendo a las necesidades de mayor prioridad de las partes interesadas.
4. **Transición:** El objetivo es validar y desplegar el sistema en su entorno de producción.

Las disciplinas que conforman la metodología se llevan a cabo de manera iterativa, o sea, la definición de las actividades que realizan los miembros del equipo de desarrollo para construir, validar y desplegar software de trabajo que satisfaga las necesidades de sus grupos de interés son de manera iterativa e incremental. **Dichas disciplinas son:**

1. **Modelo:** El objetivo de esta disciplina es entender el negocio de la organización y determinar una solución viable para hacer frente al dominio del problema.
2. **Implementación:** El objetivo de esta disciplina es transformar el modelo del proyecto en el código ejecutable y para llevar a cabo un nivel básico de las pruebas específicamente en las pruebas de unidad.
3. **Prueba:** El objetivo de esta disciplina consiste en realizar una evaluación objetiva para asegurar la calidad. Esto incluye encontrar defectos, validar que el sistema funcione como está previsto, y verificar que se cumplan los requisitos.
4. **Despliegue:** El objetivo de esta disciplina es el plan para la entrega del sistema y para ejecutar el plan para que el sistema esté a disposición de los usuarios finales.
5. **Gestión de la Configuración:** El objetivo de esta disciplina es para administrar el acceso a los artefactos del proyecto. Esto incluye no sólo el seguimiento de versiones de los artefactos a través del tiempo, sino también el control y la gestión de los cambios a los mismos.
6. **Gestión de Proyecto:** El objetivo de esta disciplina es dirigir las actividades que se llevan a cabo en el proyecto. Esto incluye la gestión de riesgos, la dirección de personas (la asignación de tareas, seguimiento de los progresos, etc.), y coordinar con la gente y los sistemas fuera del alcance del proyecto para asegurarse de que se entregue a tiempo y dentro del presupuesto.
7. **Ambiente:** El objetivo de esta disciplina es apoyar el resto de los esfuerzos por garantizar que la orientación adecuada del proceso, (normas y directrices), y herramientas (hardware, software, etc) estén disponibles para el equipo según sea necesario.

Ventajas de AUP:

1. **Simplicidad:** Toda la documentación se describe de manera concisa en el proyecto.
2. **Agilidad:** El proceso unificado ágil se ajusta a los valores y principios de la alianza ágil.

3. **Se centra en actividades de alto valor:** La atención se centra en actividades que en realidad cuentan en un proyecto, obviando las que no son necesarias para el desarrollo del mismo.

De las metodologías de desarrollo analizadas se seleccionó AUP pues es una herramienta de modelado para proyectos pequeños, es una técnica de modelado de procesos ágil, en su modelado simplifica la realidad, proporciona los planos del sistema e incluye elementos que tienen gran influencia, omitiendo aquellos menores que no son relevantes para el nivel de abstracción dado. Los modelos ayudan a visualizar como es o como se quiere que sea el sistema, permiten especificar la estructura o comportamiento del sistema.

2.3 Herramientas Case

Se puede definir a las Herramientas CASE como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un Software. Como es sabido, los estados en el Ciclo de Vida de desarrollo de un Software son: Investigación Preliminar, Análisis, Diseño, Implementación e Instalación. **(scribd.com, 2011)**

Se define Case como conjunto de métodos, utilidades y técnicas que facilitan la automatización del ciclo de vida del desarrollo de sistemas de información, completamente o en alguna de sus fases. Las herramientas CASE también permiten a los analistas tener más tiempo para el análisis y diseño y minimizar el tiempo para codificar y probar. La principal ventaja de la utilización de una herramienta CASE, es la mejora de la calidad de los desarrollos realizados y, en segundo término, el aumento de la productividad. **(scribd.com, 2011)**

2.3.1 Rational Rose

Rational Rose es una herramienta de producción y comercialización establecidas por Rational Software Corporation (actualmente parte de IBM). Rose es un instrumento operativo conjunto que utiliza el Lenguaje Unificado (UML) como medio para facilitar la captura de dominio de la semántica, la arquitectura y el diseño. Permite especificar, analizar, diseñar el sistema antes de codificarlo. **(Plusformacion.com, 2011)**

Sus características principales: (Monografias.com S.A, 2011)

- No es gratuito, se debe hacer un previo pago para poder adquirir el producto.
- La ingeniería de código (directa e inversa) es posible para ANSI C++, Visual C++, Visual Basic 6, Java, J2EE/EJB, CORBA, Ada 83, Ada 95, Bases de datos: DB2, Oracle, SQL 92, SQL Server, Sybase, Aplicaciones WEB.
- Solamente Ingeniería reversa para COM.
- Rational Rose habilita asistentes para crear clases y provee plantillas de código que pueden aumentar significativamente la cantidad de código fuente generado. Adicionalmente, se pueden aplicar los patrones de diseño, Racional Rose ha provisto 20 de los patrones de diseño GOF para Java.
- Admite la integración con otras herramientas de desarrollo (IDEs).

Rational Rose es la herramienta CASE que comercializan los desarrolladores de UML y que soporta de forma completa la especificación del UML 1.1. Esta herramienta propone la utilización de cuatro tipos de modelos para realizar un diseño del sistema, utilizando una vista estática, otra dinámica, una lógica y otra física. Permite crear y refinar estas vistas creando de esta forma un modelo completo que representa el dominio del problema y el sistema de software.

Ventajas

1. Es una de las herramientas CASE más usadas para modelar sistemas con UML y UML2.
2. Es además muy completa y estable al brindar facilidades de uso para modificar y crear nuevos diagramas.

Desventajas:

1. Es una herramienta que presenta una interfaz poco amigable para el usuario.
2. No es un software libre.
3. No se puede crear el entorno del sistema para los diagramas de casos de usos.

2.3.2 Visual Paradigm

Visual Paradigm está desarrollada por Visual Paradigm International, una de las principales compañías de herramientas CASE. Su principal característica y por tanto su mayor ventaja es que es multiplataforma pues permite ejecutarse en diferentes sistemas operativos. Visual Paradigm ofrece gran cantidad de soluciones de software que permite a las organizaciones desarrollar aplicaciones con mayor calidad, rapidez y más baratos.

Visual Paradigm es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. También proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. Presenta licencia gratuita y comercial. Es fácil de instalar, actualizar y es compatible entre ediciones.

Visual Paradigm presenta varias ediciones, cada una destinada a sus propias necesidades: Enterprise, Professional, Community, Standard, Modeler y Personal. También ayuda a los equipos de desarrollo de software a desplegar el proceso de desarrollo de los mismos, logrando maximizar y acelerar tanto las contribuciones individuales como las de equipo, facilita la diagramación visual y el diseño de sus proyectos, además posee alta interoperabilidad.

Características principales: (Monografias.com S.A, 2011)

- Soporte de UML versión 2.1.
- Diagramas de Procesos de Negocio - Proceso, Decisión, Actor de negocio, Documento.
- Modelado colaborativo con CVS y Subversión (control de versiones).
- Interoperabilidad con modelos UML2 (metamodelos UML 2.x para plataforma Eclipse) a través de XMI.
- Ingeniería de ida y vuelta.
- Ingeniería inversa - Código a modelo, código a diagrama.
- Ingeniería inversa Java, C++, Esquemas XML, XML, NET exe/dll, CORBA IDL.

- Generación de código - Modelo a código, diagrama a código.
- Editor de Detalles de Casos de Uso - Entorno todo-en-uno para la especificación de los detalles de los casos de uso, incluyendo la especificación del modelo general y de las descripciones de los casos de uso.
- Diagramas EJB - Visualización de sistemas EJB.
- Generación de código y despliegue de EJB - Generación de beans para el desarrollo y despliegue de aplicaciones.
- Diagramas de flujo de datos.
- Soporte ORM - Generación de objetos Java desde la base de datos.

Visual Paradigm es la herramienta CASE escogida para el desarrollo del software debido a que posee licencia gratuita y comercial, es un producto de mucha calidad, multiidioma, multilenguaje, muy fácil de instalar y actualizar, además de bridar compatibilidad entre ediciones. Visual Paradigm soporta modelado UML y provee el modelado de procesos de negocios. **Además permite:**

- Navegación intuitiva entre código y el modelo. Poderoso generador de documentación y reportes UML PDF/HTML/MS Word.
- Demanda en tiempo real, modelo incremental de viaje redondo y sincronización de código fuente.
- Superior entorno de modelado visual.
- Soporte completo de notaciones UML.
- Diagramas de diseño automático sofisticado.
- Creación de modelos UML.

2.4 Lenguaje de programación

Un lenguaje de programación es un lenguaje diseñado para describir el conjunto de acciones consecutivas que un equipo debe ejecutar. Por lo tanto, un lenguaje de programación es un modo práctico para que los seres humanos puedan dar instrucciones a un equipo. Los lenguajes de programación son herramientas que nos permiten crear programas y software. Entre ellos están C Sharp, C++, Pascal, Java, etc. Los lenguajes de programación facilitan la tarea de programación, ya

que disponen de formas adecuadas que permiten ser leídas y escritas por personas, a su vez resultan independientes del modelo de computador a utilizar. **(CCM-Benchmark Network, 2011)**

2.4.1 Lenguaje de programación Java

Plataforma de software desarrollada por Sun Microsystems. Esta plataforma ha sido desarrollada de tal manera que los programas desarrollados para ella puedan ejecutarse de la misma forma en diferentes tipos de arquitecturas y dispositivos computacionales. El lenguaje mismo adopta la sintaxis de C++, pero su funcionamiento es muy similar al de Smalltalk. Incorpora sincronización y manejo de tareas en el lenguaje mismo (similar a Ada) e incorpora interfaces como un mecanismo alternativo a la herencia múltiple de C++.
La plataforma Java consta de tres partes:

- El lenguaje de programación
- Máquina virtual de Java
- API Java

Además de programas del servidor, Java permite escribir programas de interfaz gráfica o textual. Además se pueden correr programas de manera incorporada o embebida en los navegadores web de Internet, aunque esto nunca llegó a popularizarse como se esperaba en un principio. Los programas en Java generalmente son compilados a un lenguaje intermedio o bytecode, y luego interpretados por una máquina virtual (JVM). Esta última sirve como una plataforma de abstracción entre la máquina y el lenguaje permitiendo que se pueda "escribir el programa una vez, y correrlo en cualquier lado".

Con la programación en Java, se pueden realizar distintos aplicativos, como son applets, que son aplicaciones especiales, que se ejecutan dentro de un navegador al ser cargada una página HTML en un servidor WEB. Otra de las utilidades de la programación en Java es el desarrollo de aplicaciones, que son programas que se ejecutan en forma independiente, es decir con Java, se pueden realizar aplicaciones como un procesador de palabras, una hoja que sirva para cálculos, una aplicación gráfica, etc. Java permite la modularidad por lo que se pueden hacer rutinas individuales que sean usadas por más de una aplicación. **(lenguajes-de-programacion.com, 2011)**

El poder reducir los problemas de acceso a memoria y liberación automática hacen de java un lenguaje poco apropiado para desarrollar aplicaciones de base como Sistemas Operativos. Sin embargo Java puede ser implementado en hardware como la tecnología **jini** que son redes adaptables y escalables. Otras desventajas pueden ser la de poseer la tecnología de la máquina virtual, si se hace referencia a la velocidad. Es considerablemente lento respecto de lenguajes como C o C++.

2.4.2 Lenguaje de programación C++

C++ es un lenguaje imperativo orientado a objetos derivado del **C**. En realidad un superconjunto de **C**, que nació para añadirle cualidades y características de las que carecía. El resultado es que como su ancestro, sigue muy ligado al hardware subyacente, manteniendo una considerable potencia para programación a bajo nivel, pero se la han añadido elementos que le permiten también un estilo de programación con alto nivel de abstracción.

C++ es una mejoría sobre mucha de las características de C, y proporciona capacidades de P.O.O que promete mucho para incrementar la productividad, calidad y reutilización del software. En C, la unidad de programación es la función, con lo cual, se trata de una programación orientada a la acción en cambio en C++ la unidad de programación es la clase a partir de la cual, los objetos son producidos. Se trata, pues, de una programación orientada al objeto. Las bibliotecas estándar de C++ proporcionan un conjunto extenso de capacidades de entrada/salida. Este lenguaje usa entradas/salidas de tipo seguro; no podrán introducirse datos equivocados dentro del sistema. (Informáticos, 2011)

En la actualidad, el C++ es un lenguaje versátil, potente y general. Su éxito entre los programadores profesionales le ha llevado a ocupar el primer puesto como herramienta de desarrollo de aplicaciones. El C++ mantiene las ventajas del C en cuanto a riqueza de operadores y expresiones, flexibilidad y eficiencia. Además, ha eliminado algunas de las dificultades y limitaciones del C original. La evolución de C++ ha continuado con la aparición de Java, un lenguaje creado simplificando algunas cosas de C++ y añadiendo otras, que se utiliza para realizar aplicaciones en Internet.

Los programas escritos en C o C++ tienen otras ventajas sobre el resto. Con la excepción del ensamblador, generan los programas más compactos y rápidos. El código es portable, es decir, un programa ANSI en C o C++ podrá ejecutarse en cualquier máquina y bajo cualquier sistema operativo.

Y si es necesario, proporcionan un acceso a bajo nivel de hardware sólo igualado por el ensamblador. C++ se trata simplemente del sucesor de un lenguaje de programación hecho por programadores (de alto nivel) para programadores, lo que se traduce en un diseño pragmático al que se le han ido añadiendo todos los elementos que la práctica aconsejaba como necesarios, con independencia de su belleza o purismo conceptual. **(usabilidadweb.com.ar, 2011)**

2.4.3 Lenguaje de programación C Sharp

C# o C Sharp es un lenguaje de programación que está incluido en la Plataforma .NET y corre en el Lenguaje Común en Tiempo de Ejecución (CLR, Common Language Runtime). El primer lenguaje en importancia para el CLR es C#, mucho de lo que soporta la Plataforma .NET está escrito en C#. Este lenguaje deriva de C y C++, es moderno, simple y enteramente orientado a objetos, simplifica y moderniza a C++ en las áreas de clases, namespaces, sobrecarga de métodos y manejo de excepciones. Se eliminó la complejidad de C++ para hacerlo más fácil de utilizar y menos propenso a errores.

Principales características de C Sharp:

Sencillez de uso: C# elimina muchos elementos añadidos por otros lenguajes y que facilitan su uso y compresión, como por ejemplo ficheros de cabecera, o ficheros fuentes IDL1. 12. Es por ello que se dice que C# es autocontenido. Además, no se incorporan al lenguaje elementos poco útiles, como por ejemplo macros, herencia múltiple u operadores diferentes al operador de aceso a métodos (operador punto) para acceder a miembros de espacios de nombres.

Modernidad: Al ser C# un lenguaje de última generación, incorpora elementos que se ha demostrado a lo largo del tiempo que son muy útiles para el programador, como tipos decimales o booleanos, un tipo básico string, así como una instrución que permita recorrer colecciones con facilidad (instrucción for each). Estos elementos hay que simularlos en otros lenguajes como C++ o Java.

Orientado a objetos: C# como lenguaje de última generación, y de propósito general, es orientado a objetos. No permite la inclusión de funciones ni variables globales que no estén incluidos en una definición de tipos, por lo que la orientación a objetos es más pura y clara que en otros lenguajes como

C++. Además, C# soporta todas las características del paradigma de la programación orientada a objetos, como son la encapsulación, la herencia y el polimorfismo.

Eficiente: En C#, todo el código incluye numerosas restricciones para garantizar su seguridad, no permitiendo el uso de punteros. Sin embargo, y a diferencia de Java, existen modificadores para saltarse esta restricción, pudiendo manipular objetos a través de punteros. Para ello basta identificar regiones de código con el identificador unsafe, y podrán usarse en ellas punteros de forma similar a como se hace en C++. Esta característica puede resultar de utilidad en situaciones en las que se necesite gran velocidad de procesamiento.

Extensión de modificadores: C# ofrece, a través de los atributos, la posibilidad de añadir a los metadatos del módulo resultante de la compilación de cualquier fuente información adicional a la generada por el compilador que luego podrá ser consultada en tiempo de ejecución a través de la biblioteca de reflexión de .NET.

Otras características de C# son: (canalvisualbasic.net., 2011)

- Provee el beneficio de un ambiente elegante y unificado.
- No soporta herencia múltiple, solamente el *runtime* .NET permite la herencia múltiple en la forma de interfaces, las cuales no pueden contener implementación.
- No maneja apuntadores, para emular la función de los apuntadores se utiliza delegates el cual provee las bases para el .NET event model.
- Por default trabaja con código administrado.
- La Plataforma .NET provee un colector de basura que es responsable de administrar la memoria en los programas C#.
- El manejo de errores está basado en excepciones.
- Soporta los conceptos como encapsulación, herencia y polimorfismo de la programación orientada a objetos.

Después de hacer un análisis de 3 principales lenguajes de programación se eligió el lenguaje de programación C++ porque es un lenguaje de alto nivel, con una gran robustez y genera los programas más compactos y rápidos. Se pueden definir entradas/salidas definidas por el usuario así como de tipos estándar, esta extensibilidad es una de las características más valiosas de este lenguaje. Además

permite un tratamiento común a dichas entradas. Este tipo de estado facilita la reutilización de código y un mejor desarrollo del software en general. Sus características de C permiten acceder al control del Hardware.

La posibilidad de orientar la programación a objetos que ofrece este lenguaje permite al programador diseñar aplicaciones desde un punto de vista más cercano a la vida real. Además permite la reutilización del código de una manera más lógica y productiva. También es de alta portabilidad y brevedad. Además C++ brinda muchas facilidades para la programación orientada a objetos y para el uso de plantillas o programación genérica.

2.5 Framework

Un framework simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, un framework proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. Por último, un framework facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas. **(Fabien Potencier, 2008)**

2.5.1 QT

Qt es una biblioteca multiplataforma para desarrollar interfaces gráficas de usuario y también para el desarrollo de programas sin interfaz gráfica como herramientas de la consola y servidores. Funciona en todas las principales plataformas, y tiene un amplio apoyo. El API de la biblioteca cuenta con métodos para acceder a bases de datos mediante SQL, así como uso de XML, gestión de hilos, soporte de red, una API multiplataforma unificada para la manipulación de archivos y una multitud de otros para el manejo de ficheros, además de estructuras de datos tradicionales.

Qt es utilizada en KDE, un entorno de escritorio para sistemas como GNU/Linux o FreeBSD, entre otros. Qt utiliza el lenguaje de programación C++ de forma nativa, adicionalmente puede ser utilizado en varios otros lenguajes de programación a través de bindings. Distribuida bajo los términos de GNU Lesser General Public License (y otras), Qt es software libre y de código abierto. **(linuxlandit.blogspot.com, 2011)**

Algunas de sus características son:

1. Compatibilidad multiplataforma con un sólo código fuente.
2. Performance de C++.
3. Disponibilidad del código fuente.
4. Excelente documentación.
5. Arquitectura lista para plugins.

Componentes del framework Qt:

- Las librerías Qt (clases en C++).
- Qt Designer, para crear formularios visualmente.
- Qt Assistant, acceso rápido a la documentación.
- Qt Linguist, traducción rápida de programas.
- qmake, simplifica el proceso de construcción de proyectos en las diferentes plataformas soportadas.

(Meyer, 2007)

Algunos de los módulos que forman Qt son: (Zona QT, 2011)

- Bases de Datos - Qt SQL
- Core - Qt Core
- Comunicación en red - Qt Network
- Interfaz Gráfica de usuario - Qt GUI
- Multimedia - Phonon, Qt Multimedia
- Quick - Qt Declarative, QML
- Webkit - Qt Webkit
- XML - Qt XML

Ventajas de QT

1. Permite implementar el código de un dispositivo de destino o una plataforma a otra mediante la reutilización de código.
2. Facilita centrarse en la construcción de su aplicación en lugar de mantener las API.
3. Es multiplataforma.

Qt también provee poderosas herramientas de desarrollo, entre ellas destaca un completo entorno de desarrollo, llamado Qt Creator, que incluye un editor de texto con autocompletado, diseñador de interfaces gráficas, gestión de proyectos, sistema de depuración, integración con sistemas de control de versiones y muchas más características.

Qt está disponible bajo 3 diferentes licencias:

1. **GPL:** Aplicación de código abierto, los cambios realizados al código fuente de Qt deben ser compartidos con la comunidad.
2. **LGPL:** Es posible crear aplicaciones de código cerrado, los cambios realizados al código fuente de Qt deben ser compartidos con la comunidad.
3. **Comercial:** Es posible crear aplicaciones de código cerrado, los cambios realizados al código fuente de Qt pueden mantenerse cerrados**. (Zona QT, 2011)**

En el desarrollo del trabajo de diploma se utilizará como framework QT porque utiliza C++ de manera nativa, Qt es un framework muy poderoso, comparable con Swing de Java o .NET de Microsoft, además ofrece una suite de aplicaciones para facilitar y agilizar las tareas de desarrollo, posee también una GUI integrada y diseñador de formularios, resaltado y auto-completado de código, soporte para refactorización de código, posee importantes módulos que se utilizarán en el desarrollo del proyecto como es el módulo Qt Network, una de las razones principales es que es un framework de código abierto y puede integrarse fácilmente con C++ como lenguaje de programación.

2.6 Entorno de desarrollo integrado (IDE)

Un IDE es una herramienta de software que facilita el trabajo de los programadores, ofreciéndoles un gran número de funcionalidades en tiempo de implementación, como pueden ser componentes visuales, completamiento de código y editores de código especializados. Estas herramientas de software tienen asociados siempre uno o varios lenguajes de programación, los cuales permiten desarrollar productos informáticos para uno u otro entorno. Es un editor de código que además puede servir para depurar y facilitar las diferentes tareas necesarias en el desarrollo de cualquier tipo de aplicación.

2.6.1 IDE NetBeans

El IDE NetBeans primero fue un proyecto estudiantil de la Universidad Charles de Praga. El proyecto fue adquirido por Sun en 1999. En el año 2000, NetBeans pasa a ser de código abierto. Desde entonces, Netbeans crece en la facilidad de uso, estabilidad y características. Miles de desarrolladores de todo el mundo utilizan NetBeans. Netbeans se creó como un IDE Java pero se está moviendo hacia una mayor versatilidad. A excepción de las tres plataformas Java, Mobile, Desktop y Empresa, NetBeans proporciona herramientas para la creación de proyectos en C / C + + y Ruby.

NetBeans es un entorno de desarrollo integrado de código abierto para los desarrolladores de software. Este IDE introduce soporte de idiomas, además de que incluye los tipos de proyecto para C y C++ con bibliotecas estáticas y dinámicas, también soporta archivos de Fortran. Los usuarios pueden crear de manera fácil proyectos desde cero con solo arrastrar y soltar un archivo binario al IDE. El sistema del proyecto busca automáticamente los archivos de origen, los agrega al proyecto, configura ayuda para el código e incluso explora las dependencias del proyecto.

El IDE NetBeans está dividido en varias partes, llamado simplemente ventanas. Estas son probablemente las ventanas más comunes:

1. **Ventana del proyecto:** permite crear y gestionar proyectos de software.
2. **Ventana del editor de fuente:** para editar la fuente y otros archivos editables.
3. **Ventana de salida:** para mostrar mensajes y normalmente para generar y ejecutar dichos mensajes.

Ventajas del IDE NetBeans: (Bodnar, 2011)

1. Proporciona excelentes herramientas para el desarrollo de proyectos en C/C++ que hacen la programación más eficiente.
2. Proporciona características tales como la finalización de código, resaltado de sintáxis, plantillas de código y compilador y depurador integrado.
3. Es multiplataforma y multilenguaje
4. Es multilenguaje.

2.6.2 IDE Eclipse

Eclipse es un IDE (*Integrated Development Environment*, entorno integrado de desarrollo) para Java muy potente. Es libre y fue creado originalmente por IBM. Se está convirtiendo en el estándar de facto de los entornos de desarrollo para Java. Y es que Eclipse no es tan sólo un IDE, se trata de un marco de trabajo modular ampliable mediante complementos (*plugins*). De hecho, existen complementos que nos permiten usar Eclipse para programar en PHP, Perl, Python, C/C++, etc**. (Guía Ubuntu, 2011)**

Es un ambiente de desarrollo de fuente abierta y orientado principalmente a tecnología Java. Si bien las funciones de Eclipse son más bien de carácter general, las características del programa se pueden ampliar y mejorar mediante el uso de plug-ins. Asimismo, a través de estos "plugins" libremente disponibles es posible añadir un sistema de control de versiones a través de Subversion y a la vez lograr una integración mediante Hibernate.

Algunos proyectos de IDE's con Eclipse son:

- AspectJ es una extensión del lenguaje Java orientado a aspectos.
- Proyecto de herramientas de desarrollo en C/C++ (CDT) trabaja para proveer un Ambiente integrado de desarrollo completamente funcional para C y C++ para la plataforma Eclipse.
- Subproyecto IDE de COBOL para Eclipse (COBOL) construye un Ambiente Integrado de Desarrollo (IDE) completamente funcional para COBOL en la plataforma Eclipse.
- Herramientas de Desarrollo de Java (JDT) provee las herramientas que implementan un IDE de Java, soportando el desarrollo de cualquier aplicación Java, incluyendo los plug-ins de Eclipse.
- Photran (photran) es un IDE completamente funcional para Fortran con soporte para Refactorización.
- PHP Development Tools trabaja para proveer un IDE completamente funcional para PHP para la plataforma Eclipse.
- Wolfram Workbench es un IDE basado en Eclipse (también disponible como plugin para Eclipse) para el lenguaje Mathematica.
- PyDev un IDE completamente funcional para python con soporte para Refactorización, y depurador gráfico. **(Guía Ubuntu, 2011)**

Ventajas en la utilización de Eclipse

1- El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos (en inglés plug-in) para proporcionar toda su funcionalidad al frente de la Plataforma de Cliente rico, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no.

2- Este mecanismo de módulos es una plataforma ligera para componentes de software. Adicionalmente a permitirle a Eclipse extenderse usando otros lenguajes de programación como son C/C++ y Python, permite a Eclipse trabajar con lenguajes para procesado de texto como LaTeX, aplicaciones en red como Telnet y Sistema de gestión de base de datos.

3- La arquitectura plug-in permite escribir cualquier extensión deseada en el ambiente, como sería Gestión de la configuración. Se provee soporte para Java y CVS en el SDK de Eclipse. Y no tiene por qué ser usado únicamente para soportar otros Lenguajes de programación.

4- La definición que da el proyecto Eclipse acerca de su Software es: "una especie de herramienta universal - un IDE abierto y extensible para todo y nada en particular". **(EcuRed, 2010)**

2.6.3 IDE QT Creator

Qt Creator es un entorno multiplataforma de desarrollo integrado (IDE) adaptado a las necesidades de los desarrolladores de Qt. Qt Creator ejecuta en Windows, Linux/X11 y Mac OS X sistemas operativos de escritorio, y permite a los desarrolladores crear aplicaciones de escritorio y plataformas de dispositivos móviles. Posee un avanzado editor de código que provee soporte para edición de C++ y QML (JavaScript), sensible al contexto de ayuda, finalización de código y mucho más.

Principales características de Qt Creator:

- Posee un avanzado editor de código C++.
- Además soporta los lenguajes: C#/.NET Languages (Mono), Python: PyQt y PySide, Ada, Pascal, Perl, PHP y Ruby.
- Posee también una GUI integrada y diseñador de formularios.
- Herramienta para proyectos y administración.
- Ayuda sensible al contexto integrado.

- Depurador visual.
- Resaltado y auto-completado de código.
- Soporte para refactorización de código.

Qt Creator es distribuido bajo tres tipos de licencias: Qt Commercial Developer License, Qt GNU LGPL v. 2.1, Qt GNU GPL v. 3.0 y está disponible para las plataformas: Linux, Mac OSX; Windows, Windows CE, Symbian y Maemo. **(pixelcoblog.com, 2011)**

Qt Creator está totalmente integrado con Qt Designer para diseñar formas de la interfaz de usuario como lo haría con la versión independiente. La integración de Qt Designer incluye también la gestión y finalización del proyecto de código. Qt Creator abastece no sólo a los desarrolladores que están acostumbrados a utilizar el ratón, sino también a los desarrolladores que se sienten más cómodos con el teclado. Una amplia gama de métodos abreviados de teclado y navegación están disponibles para ayudar a acelerar el proceso de desarrollo de su aplicación. **(tuxmaya.wordpress.com, 2011)**

Como un IDE, Qt Creator difiere de un editor de texto, ya que sabe cómo crear y ejecutar aplicaciones. Se entiende el código como código, no sólo como texto sin formato. Esto permite:

- Escribir código bien formateado.
- Anticipar lo que va a escribir y completar el código.
- Visualización en línea de error y mensajes de advertencia.
- Navegar semánticamente a las clases, funciones y símbolos.
- El consumidor recibe ayuda sensible al contexto de las clases, funciones y símbolos.
- Cambiar el nombre de símbolos de una manera inteligente, de modo que otros símbolos con el mismo nombre que pertenecen a otros ámbitos no se cambia el nombre.
- Mostrar las ubicaciones en el código donde se declara una función o una llamada.

Para la realización de este trabajo de diploma se seleccionó el Entorno de desarrollo integrado (IDE) QT Creator pues anteriormente se escogió Qt como framework de desarrollo y además porque Qt Creator es una herramienta también en software libre, se puede integrar con C++ como lenguaje de programción el cual es muy potente y uno de los más robustos y completos que se haya conocido. Y

además este IDE viene acompañado de un conjunto de herramientas para facilitar su uso, permite especificar la configuración para ejecutar aplicación y puede crear un proyecto desde cero o importar un proyecto existente.

2.7 Conclusiones

Luego de un profundo análisis de las principales herramientas y tecnologías que pudieran ser utilizadas para la construcción del sistema, se seleccionó como metodología de desarrollo AUP, y posteriormente como herramienta CASE para el modelado se utilizará Visual Paradigm, definiendo como lenguaje de programación C++, framework QT y entorno de desarrollo integrado QT Creator, considerando que son éstas las más idóneas para el cumplimiento de los objetivos trazados durante el presente trabajo de diploma.

Capítulo 3: Análisis y diseño de la solución.

3.1 Introducción

En el siguiente capítulo se aborda la descripción de los pasos a seguir durante el análisis y el diseño de la solución. Contiene la especificación de los requisitos funcionales y los no funcionales que debe cumplir el sistema, con sus pertinentes descripciones, se realiza la modelación del sistema, así como el diseño que poseerá el mismo. Se construirá el diagrama de Caso de Uso del Sistema, el modelo de dominio el cual dará paso al modelo de diseño, que constituye un plano bastante cercano a la implementación, y contribuirá a una arquitectura estable y sólida. Como parte del modelo del diseño se hará una propuesta de los diagramas de clases del diseño.

3.2 Análisis de la solución

El análisis es el primer paso en la construcción del software. En este proceso el analista se reúne con el cliente y/o usuario (un representante institucional, departamental o cliente particular), e identifican las metas globales, se analizan las perspectivas del cliente, sus necesidades y requerimientos, sobre la planificación temporal y presupuestal, y otros puntos que puedan ayudar a la identificación y desarrollo del proyecto.

3.2.1 Modelo del dominio

El entorno organizacional del Sistema para el monitoreo de eventos en servidores de máxima disponibilidad no funciona como un modelo de negocio, el flujo de información es difuso, tiene múltiples orígenes; no se puede identificar claramente las responsabilidades de cada involucrado y es difícil establecer reglas de funcionamiento, por lo que se hace necesario realizar un Modelo del Dominio. Su objetivo es ayudar a comprender a todos los involucrados en el desarrollo de la aplicación, a utilizar un vocabulario común y para realizar una correcta captura de los requisitos. Se encarga de seleccionar los tipos más importantes de objetos o eventos que acontecen en el Sistema para monitoreo de eventos en servidores de máxima disponibilidad y las relaciones entre estos.

Según (Larman, C. 2008) “El Modelo del Dominio es una representación visual del entorno real “. Es decir, un diagrama con los objetos que existen (reales) relacionados con la aplicación que se va a acometer y las relaciones que hay entre ellos.

Diagrama del Modelo del Dominio

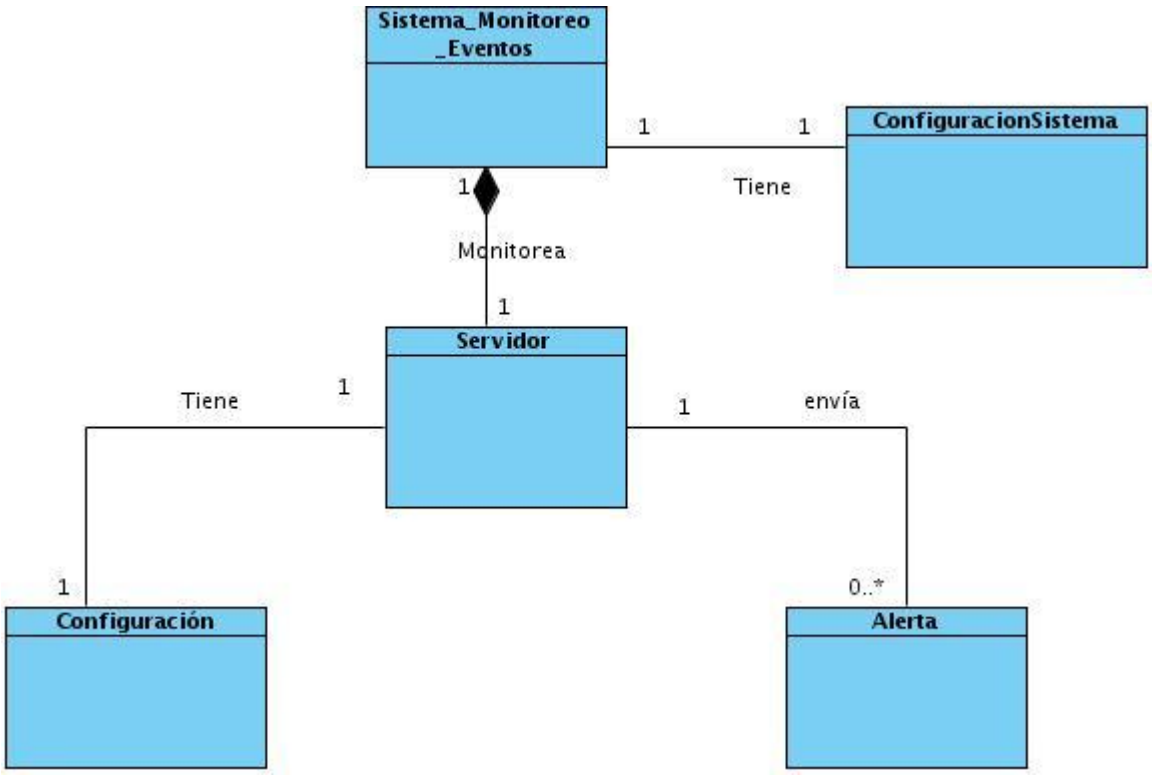

Figura 2 Modelo de dominio

Descripción de los roles y clases

Sistema para Monitoreo de Eventos: aplicación que permite el monitoreo de eventos en servidores de máxima disponibilidad.

Servidor: servidor que va a ser monitoreado por el Sistema.

Alerta: la alerta o las alertas que puede enviar el servidor al administrador por las fallas que puedan ocurrir.

Configuración de alertas: formulario donde el administrador debe establecer la configuración de las alertas según las necesidades y características del servidor.

Configuración de sistema: formulario donde el administrador debe establecer la configuración que tendrá el sistema para poder enviar las alertas.

3.2.2 Requisitos

Los requisitos o requerimientos son los que de forma general especifican lo que debe hacer la solución. Los requisitos tienen muchas formas y con muchos niveles de abstracción, deben ser lo más claros posibles y comprender claramente cómo se cumplen. Existen dos tipos de requisitos, se encuentran los funcionales y los no funcionales. Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir mientras que los no funcionales son propiedades o cualidades que el producto debe tener, son como las características que hacen el producto atractivo, usable, rápido y confiable.

Según el glosario de términos del *Institute of Electrical and Electronics Engineers* (IEEE) se define como requisito:

1. Una condición o necesidad de un usuario para resolver un problema o alcanzar un objetivo.

2. Una condición o capacidad que debe estar presente en un sistema o componentes de sistema para satisfacer un contrato, estándar, especificación u otro documento formal.

3. Una representación documentada de una condición o capacidad que ha de cumplir un sistema.

Requisitos Funcionales (RF):

El sistema debe permitir:

RF1. Determinar el espacio en disco del servidor: el sistema calcula el espacio en disco del servidor que se está monitoreando.

RF2. Determinar la carga del procesador del servidor: el sistema determina la carga del procesador del servidor que se está monitoreando.

RF3. Determinar la memoria libre del servidor: el sistema calcula la memoria libre del servidor.

RF4. Determinar la cantidad de conexiones que tiene el servidor: el sistema determina las conexiones que tiene el servidor.

RF5. Enviar alertas al administrador de los eventos que puedan ocurrir en el servidor: el sistema verifica el estado del servidor, o sea, la carga del procesador, la memoria libre, el espacio en disco y la cantidad de conexiones del mismo y si encuentra alguna anomalía envía la alerta al administrador.

RF6. Guardar los archivos de los eventos ocurridos en el servidor: el sistema guarda periódicamente los archivos resultantes del monitoreo de eventos.

RF7. Configurar las alertas en el servidor: el administrador del servidor configura las alertas en el mismo de acuerdo a las necesidades y características propias del servidor.

RF8. Consultar los datos guardados de los eventos ocurridos en el servidor: el administrador del servidor consulta cuando desee los archivos guardados por el servidor de las anomalías ocurridas en una fecha determinada.

RF9. Configurar el sistema: el administrador del servidor configura el sistema para enviar las alertas a través de un correo electrónico.

Requisitos No Funcionales (RNF)

- Disponibilidad

RNF1. El sistema debe estar disponible el tiempo que se especifique y en óptimas condiciones para su uso.

RNF2. El sistema debe contar con un diseño del modelo físico sencillo, con una estructura y distribución que permita trabajar con rapidez y eficiencia.

- Fiabilidad

RNF3. Ante cualquier falla en el sistema se deben mostrar los errores sin dar detalles de información, que puedan comprometer la seguridad e integridad del mismo. Debe contarse con un sistema de salvas externas de la información para casos de desastres.

- Rendimiento

RNF4. La velocidad de procesamiento de la información y el tiempo de respuestas a de ser a lo sumo dos o tres segundos aproximadamente.

- Confiabilidad

RNF5. El sistema debe ser capaz de mantener la integridad de los datos.

RNF6. La herramienta de implementación debe tener soporte para la recuperación de los datos ante algún fallo.

- Eficiencia

RNF7. El tiempo de respuesta de las consultas debe ser rápido.

- Seguridad

RNF8. Garantizar que la información sea editada únicamente por las personas, que tienen permisos para realizar esta actividad.

- Interfaz

RNF9. El sistema debe tener una interfaz amigable y sencilla de utilizar.

RNF10. Las interfaces de salida no deben tener otra información que no sea concerniente a lo que deben mostrar.

RNF11. El sistema debe estar conectado a un dispositivo de red.

- Legales

RNF12. La plataforma seleccionada para el desarrollo de la aplicación, está basada en la licencia GNU/GPL.

3.2.3 Modelo de casos de uso del sistema

Definición de los Actores

Tabla 2 Descripción de los actores del sistema.

Actor	Descripción
Sistema	Responsable para gestionar los contenidos de la aplicación exceptuando la configuración de las alertas.
Administrador	Responsable de la configuración de las alertas que debe enviar el sistema y responsable de las consultas de los archivos guardados por el servidor de los eventos ocurridos
Reloj	Responsable de realizar cada 5 segundos el envío o no de las alertas.

Diagrama de Casos de Uso del Sistema

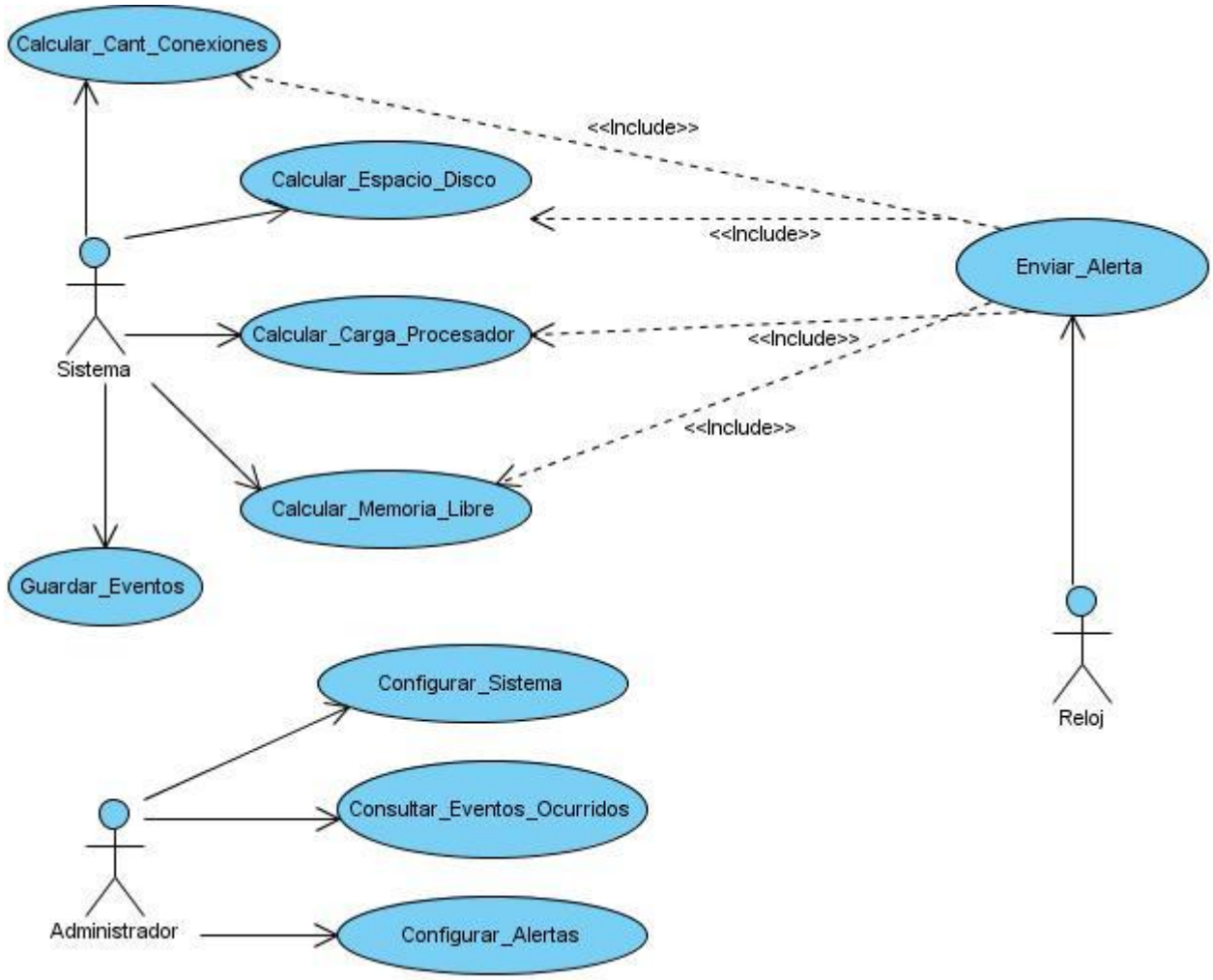

Figura 3 Diagrama de Casos de Uso del Sistema.

Descripción de Casos de Uso del Sistema

A continuación se muestran las descripciones textuales de los casos de usos más significativos del sistema a desarrollar. Las demás se pueden encontrar en el anexo #2.

Tabla 3 Descripción textual del caso de uso del sistema: Configurar_Alertas.

CU-1	Configurar_Alertas
Propósito	Configura las alertas que serán enviadas al administrador de acuerdo con las necesidades y capacidades del servidor.

Actores	Administrador
Resumen	El caso de uso se inicia cuando el administrador selecciona la opción de Configurar Alertas que le serán enviadas de acuerdo con las necesidades y capacidades del servidor que se esté monitoreando
Precondiciones	Conocer las condiciones del servidor que se va a monitorear.
Prioridad	Alta
Referencias	RF7

Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1-El administrador selecciona en el menú la opción Configurar Alertas.	**1.1-**El sistema muestra un formulario con los datos que debe llenar el administrador para configurar las alertas.
2-El administrador completa los datos que son necesarios para la configuración.	**2.1-**El sistema muestra un mensaje de que han sido configuradas las alertas.

Poscondiciones	Establecer la configuración de las alertas en el servidor.

Tabla 4 Descripción textual del caso de uso del sistema: Consultar_Eventos_Ocurridos.

CU-2	Consultar_Eventos_Ocurridos
Propósito	Consultar en el momento que sea necesario por el administrador los eventos ocurridos en el servidor en un período determinado que son archivados.
Actores	Administrador
Resumen	El caso de uso se inicia cuando el administrador selecciona la opción de Consultar Eventos Ocurridos y posteriormente el período que desee examinar de los archivos guardados como resultado del monitoreo del servidor.
Precondiciones	-
Prioridad	Alta

Referencias	RF8	
Flujo normal de eventos		
Acción del actor		**Respuesta del sistema**
1-El administrador selecciona en el menú la opción Consultar Eventos Ocurridos.		**1.1-**El sistema muestra un formulario con los datos que debe llenar el administrador según el período de los eventos que desee consultar.
2-El administrador introduce los datos que son necesarios para la consulta.		**2.1-**El sistema muestra un archivo con los eventos ocurridos que fueron pedidos por el administrador de acuerdo al período consultado.
Poscondiciones	El administrador conoce de los eventos ocurridos en el servidor en determinado período de tiempo.	

Tabla 5 Descripción textual del caso de uso del sistema: Enviar_Alerta.

CU-3	Enviar_Alerta
Propósito	Enviar en el momento necesario alertas mediante correo electrónico al administrador del servidor de las anomalías que puedan afectar a la disponibilidad de los servicios en el servidor.
Actores	Reloj
Resumen	El caso de uso se inicia cada 5 segundos cuando el sistema realiza un análisis de los eventos que ocurren en el servidor y encuentra alguna anomalía que pueda afectar a la disponibilidad de los servicios en el mismo.
Precondiciones	Para la realización de este caso de uso es necesario conocer la carga del procesador, la memoria libre, la cantidad de conexiones y el espacio en disco del servidor.
Prioridad	Alta
Referencias	RF5

Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1-El actor envía un correo al administrador del servidor con la alerta correspondiente.	1.1-El sistema chequea la conexión de red. 1.2-El sistema chequea el protocolo de correo esté funcionando correctamente. 1.2- El sistema verifica la dirección de correo del administrador.
Poscondiciones	Se envía un correo al administrador del servidor con la alerta correspondiente.

3.3 Diseño de la solución

El modelo de diseño es planteado como un modelo de objetos que describe la realización física de los casos de uso, centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar, constituyendo una entrada principal en la actividad de implementación. En el modelo del diseño se realiza la modelación del sistema y se encuentra su forma para que se soporten los requisitos tanto funcionales como no funcionales existentes. Además se crean las bases apropiadas para permitir que se realicen a cabalidad las actividades de la implementación.

3.3.1 Diagramas de clases del diseño por paquetes

Es una representación más concreta que el diagrama de clases del análisis. Representa la parte estática del sistema, las clases y sus relaciones. Están compuestas por clases asociadas y atributos, interfaces con sus operaciones y constantes, métodos, información sobre los tipos de atributos, navegabilidad y dependencias. A continuación se reflejan los diagramas de clases del diseño de los casos de usos más significativos, los demás pueden encontrarse en el Anexo #1.

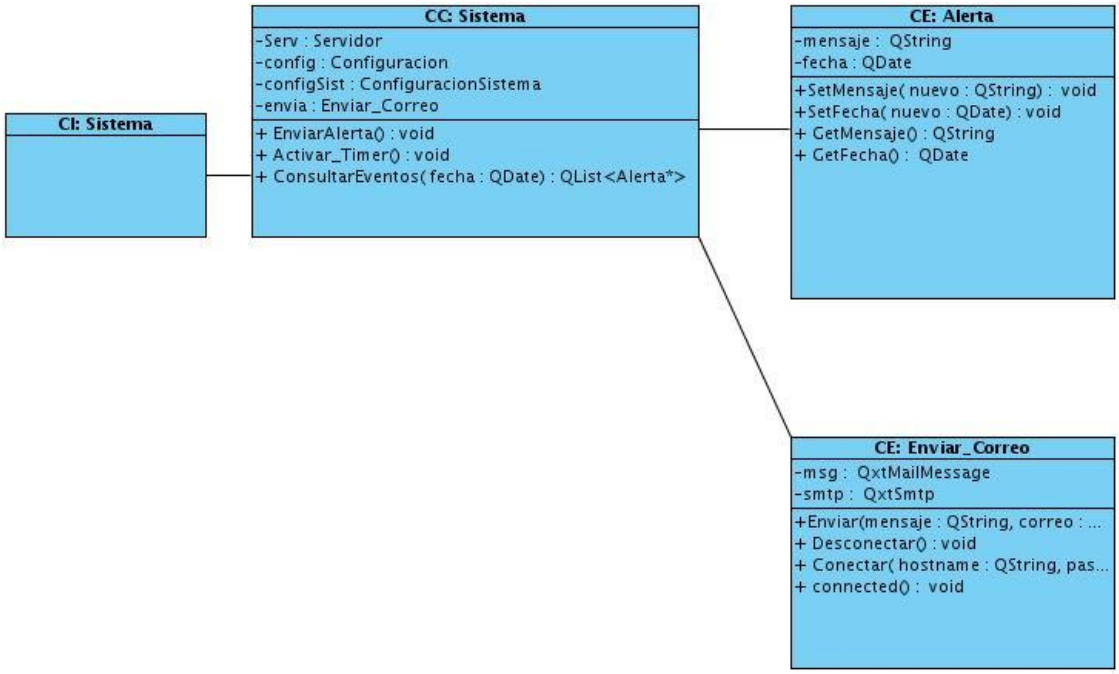

Figura 4 Diagrama de Clases del Diseño del CU Enviar_Alerta.

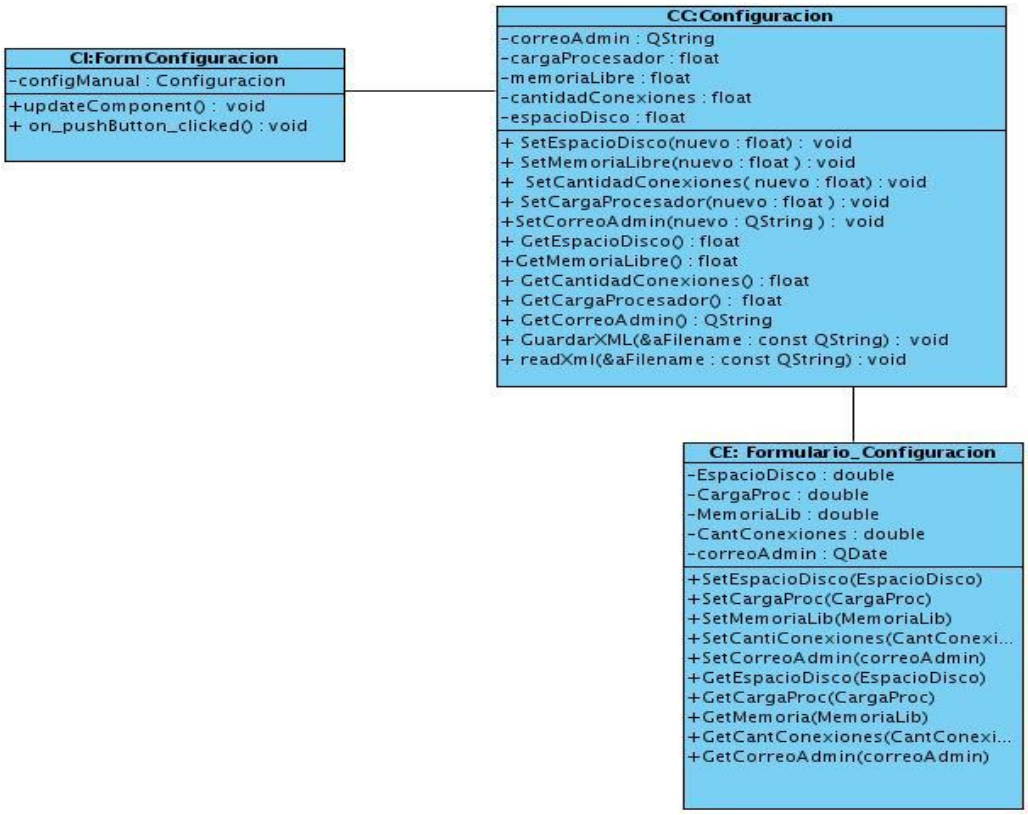

Figura 5 Diagrama de Clases del Diseño del CU Configurar_Alertas.

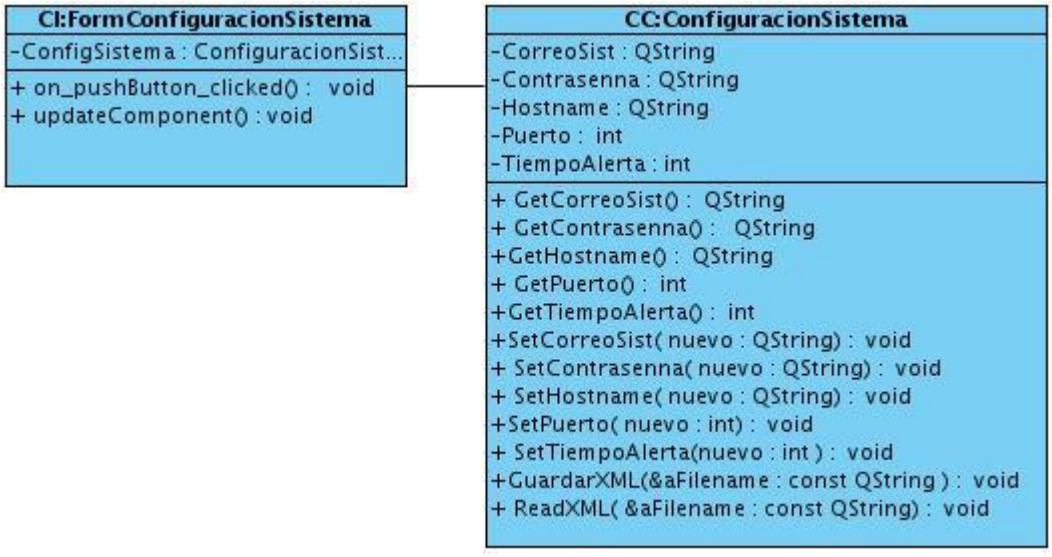

Figura 6 Diagrama de Clases del Diseño del CU Consultar_Eventos.

3.3.2 Principios del Diseño

Estándares de interfaz de la aplicación

Para lograr una interfaz de usuario amigable, atractiva y funcional para el usuario final, se necesita definir principios de diseño de interfaz de usuario, pues esta es una actividad fundamental dentro del diseño de un software. Se debe tener en cuenta que a partir de la aceptación que tengan los usuarios depende, en gran medida el éxito del sistema.

A continuación se definen los principios a tener en cuenta en el desarrollo de la interfaz de usuario de la aplicación:

- Mostrar al usuario toda la información y herramientas necesarias para cada etapa en su trabajo.
- Brindar una interfaz sencilla, de manera tal que cualquier persona con un mínimo dominio de la computación pueda trabajar con la aplicación.
- Garantizar la legibilidad de manera que exista contraste de los colores de los textos con el fondo y el tamaño de la fuente sea lo suficientemente adecuado a la vista del usuario.
- Menús y etiquetas de botones deben comenzar con la palabra más importante.

Concepción general de la ayuda

El sistema a desarrollar contará con una ayuda general para hacer más fácil la manipulación del mismo por parte de los usuarios finales. Esta ayuda se podrá encontrar en el menú principal para lograr una localización rápida. La estructura que se propone para organizar su contenido es la siguiente:

- Ayuda en línea.
- Acerca del producto

3.3.3 Patrones de diseño

Un patrón de diseño es una descripción de un problema y su solución, estos patrones de forma general constituyen parejas problema/solución a los cuales se le asigna un nombre y pueden ser empleados en otros contextos, con sugerencias sobre la manera de usarlo en situaciones nuevas. Los mismos son la base para la búsqueda de soluciones ante algunos problemas que existen en el desarrollo del software.

- **Patrones arquitectónicos:** estos patrones definen la estructura general del software, indican las relaciones entre los subsistemas y los componentes del software y definen las reglas para

especificar las relaciones entre los elementos (clases, paquetes, componentes, subsistemas) de la arquitectura.

- **Patrones de diseño**: estos patrones se aplican a un elemento específico del diseño como un agregado de componentes para resolver algún problema de diseño relaciones entre los componentes o los mecanismos para efectuar la comunicación de componente a componente.
- **Idiomas**: a veces llamados patrones de código, estos patrones específicos de lenguaje por lo general implementan un elemento algorítmico o un componente, un protocolo de interfaz específico o un mecanismo de comunicación entre los componentes. **(ramos.utfsm.c, 2011)**

Patrones GoF (gang of four) (Gracia, 2003)

Patrones de creación:

1. **Singleton:** Garantiza que una clase sólo tenga una instancia, y proporciona un punto de acceso global a ella.

Patrones estructurales:

1. **Facade:** Proporciona una interfaz unificada para un conjunto de interfaces de un subsistema. Define una interfaz de alto nivel que hace que el subsistema sea más fácil de usar.

Patrones de comportamiento:

1. **Observer:** Define una dependencia de uno-a-muchos entre objetos, de forma que cuando un objeto cambia de estado se notifica y actualizan automáticamente todos los objetos.

Patrones GRASP

1. **Experto:** Es un patrón que se usa más que cualquier otro al asignar responsabilidades; es un principio básico que suele ser útil en el diseño orientado a objetos.
2. **Creador:** Guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es

encontrar un creador que debemos conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento.

3. **Bajo Acoplamiento:** El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con qué las conoce y con qué recurre a ellas. Acoplamiento bajo significa que una clase no depende de muchas clases.
4. **Alta Cohesión**: Asignar una responsabilidad de modo que la cohesión siga siendo alta. Una clase tiene responsabilidades moderadas en un área funcional y colabora con las otras para llevar a cabo las tareas, la alta cohesión simplifica el mantenimiento y los mejoramientos. La ventaja que significa una gran funcionalidad también soporta un aumento de la capacidad de reutilización.

3.4 Conclusiones

Para una posterior implementación del sistema se establecieron los requisitos funcionales y no funcionales que debe cumplir la herramienta, se identificaron también las clases del diseño para lograr que el programador pueda realizar una correcta implementación del sistema a desarrollar y se señalaron además los principios de diseño que debían añadirse a los requisitos funcionales y los patrones que se utilizarán al implementar la herramienta.

4.1 Introducción.

En el presente capítulo se abordarán aspectos referentes a la implementación y prueba del sistema a desarrollar, o sea, se modela el diagrama de despliegue para contribuir a una mejor organización de los componentes y sus dependencias, se construye también el modelo de implementación donde los componentes son utilizados para modelar la vista estática del sistema, mostrando la organización y las dependencias lógicas entre los mismos. Finalmente para verificar la calidad y el adecuado funcionamiento del software se realizan las pruebas pertinentes utilizando técnicas que guíen el proceso de la prueba.

4.2 Modelo de despliegue

El propósito del modelo de despliegue es capturar la configuración de los elementos de procesamiento, y las conexiones entre estos elementos en el sistema. El modelo consiste en uno o más nodos relacionados con dispositivos y conectores. A continuación se muestra el modelo de despliegue del sistema a desarrollar.

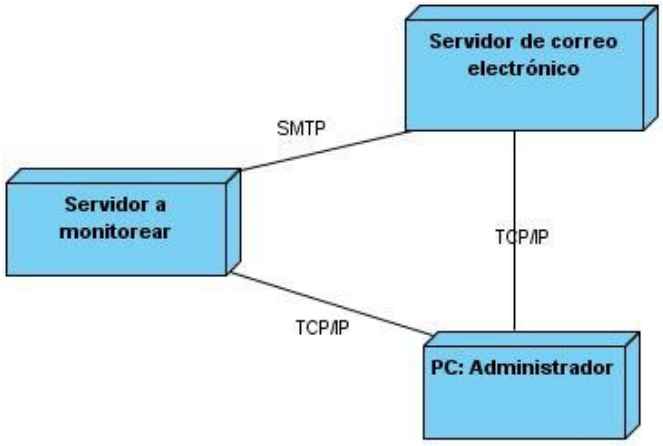

Figura 7 Diagrama de Despliegue.

Descripción de los componentes:

Nodo Servidor a monitorear: Este nodo representa el servidor en el que va a estar el sistema instalado para la monitorización del mismo.

Nodo Servidor de correo electrónico: Este nodo representa el servidor de correo que va a ser utilizado para el envío de alertas al administrador.
Nodo PC Administrador: Este nodo representa la PC del administrador, a la cual le van a llegar los correos electrónicos con las alertas que puedan ocurrir en el servidor.

Descripción de los protocolos

SMTP: Simple Mail Transfer Protocol o Protocolo Simple de Transferencia de Correo Electrónico es un conjunto de reglas que rigen el formato y la transferencia de datos en un envío de Correo Electrónico (e-mail). La comunicación SMTP se realiza por TCP y normalmente en el puerto 25, es una comunicación tipo orden - respuesta delimitada por CRLF (Retorno de carro + Salto de línea) que podemos realizar de manera manual utilizando el comando telnet o con un cliente de correo**. (programacionweb.net, 2011)**

TCP/IP: son las siglas de Protocolo de Control de Transmisión/Protocolo de Internet (en inglés Transmission Control Protocol/Internet Protocol*),* un sistema de protocolos que hacen posibles servicios Telnet, FTP, E-mail, y otros entre ordenadores que no pertenecen a la misma red. **(masadelante.com, 2011)**

4.3 Modelo de implementación.

El modelo de implementación describe cómo los elementos del modelo de diseño y cómo las clases, **s**e implementan en términos de componentes, como ficheros de código fuente, ejecutables, etc. Describe cómo los elementos del Modelo de Diseño se implementan en términos de componentes y cómo estos se organizan de acuerdo a los nodos específicos en el Modelo de Despliegue.

4.3.1 Diagrama de componentes.

Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes de software, sean estos componentes de código fuente, binarios o ejecutables. Desde el punto de vista del diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión de software, la reutilización, y las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo. **(eva.uci.cu, 2011)**

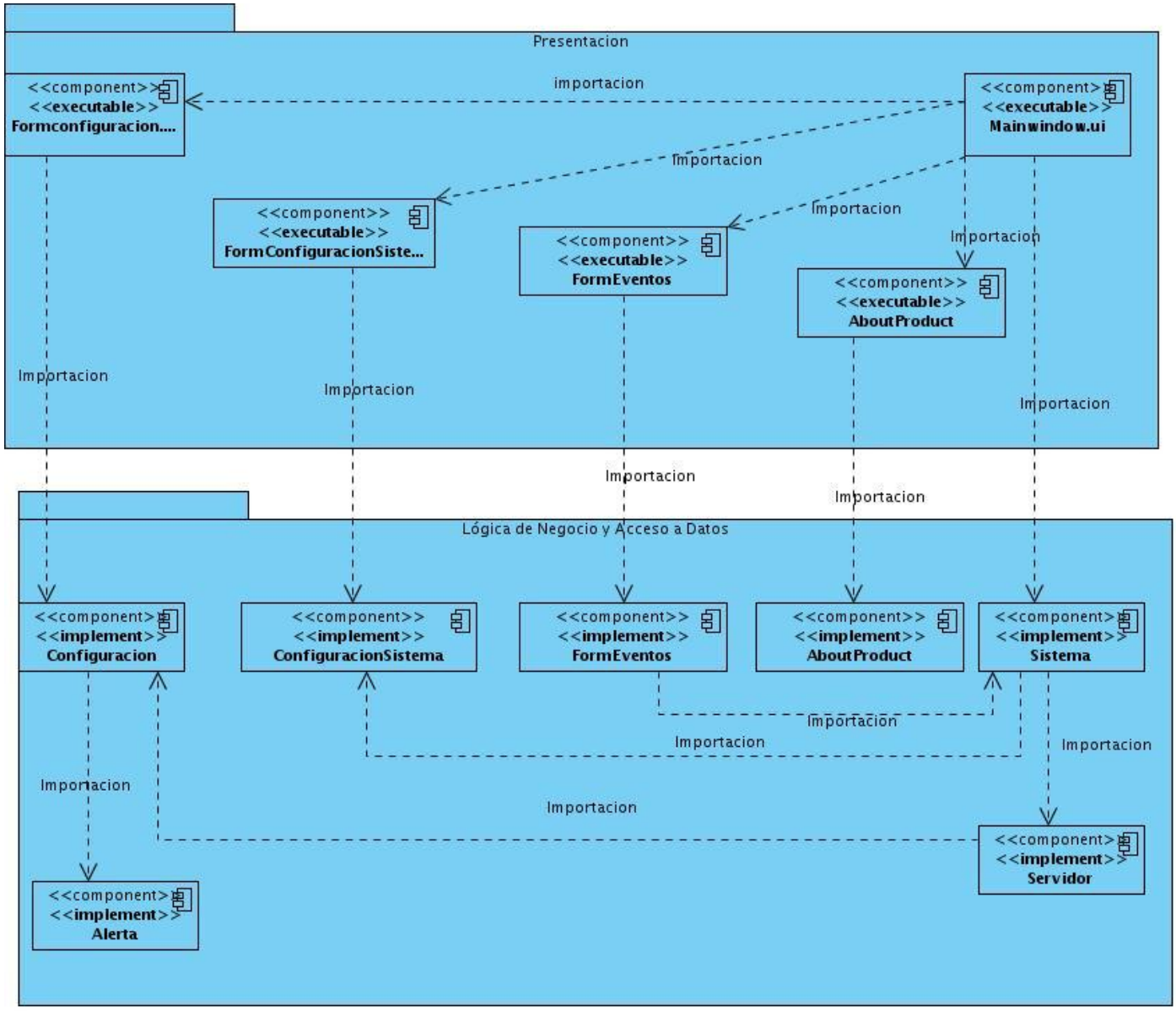

Figura 8 Diagrama de Componentes.

4.4 Pruebas del sistema

Es de suma importancia verificar la calidad y el adecuado funcionamiento del software de ahí la existencia del proceso de pruebas. La etapa de prueba es tan o más importante que todas las realizadas hasta el momento puesto que en ella se refleja la calidad con que ha sido llevada a cabo la construcción del sistema. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad teniendo como objetivo principal medir el grado en que el sistema cumple con los requerimientos, o sea, comprobar que el producto se comporta como se desee. Para que las pruebas tengan éxito es necesario utilizar técnicas que guíen el proceso. Para probar el sistema se utilizaron las técnicas de caja blanca.

4.4.1 Prueba de caja blanca

Se decidió aplicar las pruebas de caja blanca porque es el tipo de prueba que se le aplica al código del software, generalmente se hace caja blanca cuando no hay pocas interfaces o no hay. Las pruebas de caja blanca están dirigidas principalmente a las funciones internas del sistema. Se realizan probando los caminos lógicos del sistema y examinando el estado del programa en varios puntos para determinar si el estado real coincide con el esperado. Para realizar estas pruebas es necesario conocer la lógica del programa.

Mediante los métodos de prueba de la caja blanca, el ingeniero de software puede obtener casos de prueba que garanticen que:

1) Se ejerciten por lo menos una vez todos los caminos independientes para cada módulo.
2) Se ejerciten todas las decisiones lógicas en sus vertientes verdaderas y falsa.
3) Ejecuten todos los bucles en sus límites y con sus límites operacionales.

4) Se ejerciten las estructuras internas de datos para asegurar su validez.

A continuación se muestra una porción del código programado. Sobre este ejemplo se realizarán pruebas de caja blanca. Este código corresponde al caso de uso Enviar_Alerta, el mismo es el responsable de enviar una alerta si existe algún fallo que afecte la disponibilidad de los servicios del servidor.

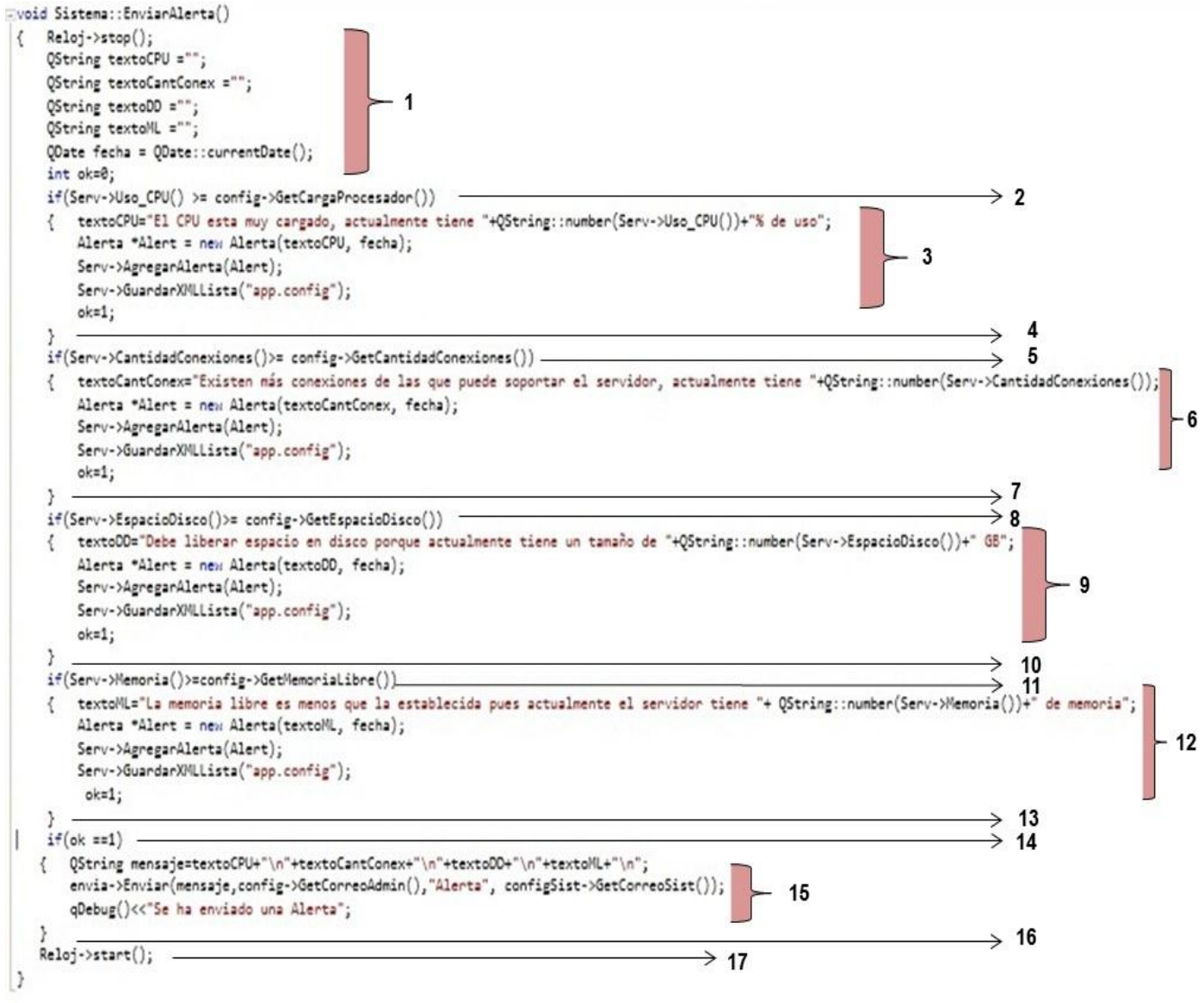

Figura 9 Representación del código del método EnviarAlerta().

La siguiente figura muestra el grafo de flujo representado para el código anterior:

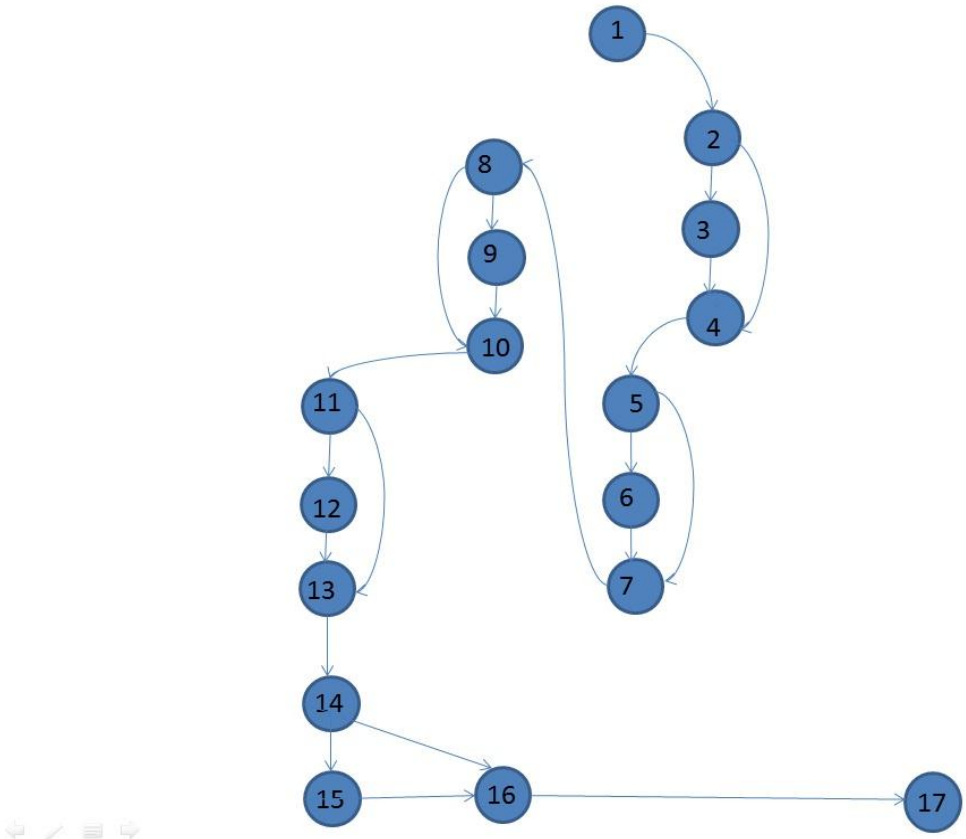

Figura 10 Grafo del flujo asociado al método EnviarAlerta().

Cálculo de la complejidad ciclomática:

Complejidad ciclomática [V (G)] = Cantidad de Aristas [A] – Cantidad de nodos [N] + 2.

V (G) = A – N + 2

V (G) = 21 – 17 + 2

V (G) = 6

El cálculo realizado arrojó como resultado 6, por lo que se puede plantear que la complejidad ciclomática del código anteriormente expuesto es de 6, lo que significa que existen seis posibles

caminos por donde el flujo puede circular, representando este valor el límite mínimo del número total de casos de pruebas que se deben realizar para el procedimiento tratado.

Tabla 6 Caminos básicos del flujo

Número	Camino básico
1	1-2-4-5-6-7-8-9-10-11-12-13-14-15-16-17
2	1-2-3-4-5-7-8-9-10-11-12-13-14-15-16-17
3	1-2-3-4-5-6-7-8-10-11-12-13-14-15-16-17
4	1-2-3-4-5-6-7-8-9-10-11-13-14-15-16-17
5	1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17
6	1-2-4-5-7-8-10-11-13-14-16-17

Luego de tener elaborado el Grafo de Flujos y los caminos a recorrer, se preparan los casos de prueba que forzarán la ejecución de cada uno de esos caminos. Para realizar los casos de pruebas es necesario tener en cuenta los siguientes aspectos.

- **Descripción**: Se realiza la entrada de datos necesaria para validar que los parámetros obligatorios no pasen nulos al método.
- **Entrada**: Se exponen los parámetros que son necesarios para realizar el método.
- **Resultados Esperados**: Se define el resultado esperado que debe devolver el método.

Tabla 7 Caso de prueba del camino básico 1

Caso de prueba para el camino básico 1	
Camino 1	1-2-4-5-6-7-8-9-10-11-12-13-14-15-16-17

Descripción	Los datos de entrada cumplirán con los siguientes requisitos: El objeto Reloj se inicializa, el parámetro fecha tiene la fecha actual del sistema, los métodos a los que se llaman toman valores y el parámetro ok se inicializa en 0.
Entrada	Valor de fecha: 31/05/11 Valor de: Uso_CPU()=50% Valor de: CantidadConexiones()=4 Valor de: EspacioDisco()=20GB Valor de: Memoria()=12% Valor de: GetCargaProcesador()=70% Valor de: GetCantidadConexiones()=2 Valor de: GetEspacioDisco()=10GB Valor de: GetMemoriaLibre()=10% Valor de: ok=1
Resultados esperados	Se envía 1 correo al administrador, especificando las razones del envío de la alerta, o sea, en el cuerpo del correo se detalla la cantidad de conexiones que tiene el servidor, el espacio en disco y finalmente la memoria libre que tiene el mismo y dicha alerta se guarda en el fichero app.config.

Tabla 8 Caso de prueba del camino básico 2

Caso de prueba para el camino básico 2	
Camino 2	1-2-3-4-5-7-8-9-10-11-12-13-14-15-16-17

Descripción	El objeto Reloj se inicializa, el parámetro fecha tiene la fecha actual del sistema, los métodos a los que se llaman toman valores y el parámetro ok se inicializa en 0.
Entrada	Valor de fecha: 31/05/11 Valor de: Uso_CPU()=80% Valor de: CantidadConexiones()=4 Valor de: EspacioDisco()=20GB Valor de: Memoria()=12% Valor de: GetCargaProcesador()=70% Valor de: GetCantidadConexiones()=8 Valor de: GetEspacioDisco()=10GB Valor de: GetMemoriaLibre()=10% Valor de: ok=1
Resultados esperados	Se envía 1 correo al administrador, especificando las razones del envío de la alerta, o sea, en el cuerpo del correo se detalla el espacio en disco que tiene el servidor, la carga del procesador y finalmente la memoria libre que tiene el mismo y dicha alerta se guarda en el fichero app.config.

Tabla 9 Caso de prueba del camino básico 3

Caso de prueba para el camino básico 3	
Camino 3	1-2-3-4-5-7-8-9-10-11-12-13-14-15-16-17
Descripción	El objeto Reloj se inicializa, el parámetro fecha tiene la fecha actual del sistema, los métodos a los que se llaman toman valores y el parámetro ok se inicializa

	en 0.
Entrada	Valor de fecha: 31/05/11 Valor de: Uso_CPU()=80% Valor de: CantidadConexiones()=12 Valor de: EspacioDisco()=8GB Valor de: Memoria()=12% Valor de: GetCargaProcesador()=70% Valor de: GetCantidadConexiones()=8 Valor de: GetEspacioDisco()=10GB Valor de: GetMemoriaLibre()=10% Valor de: ok=1
Resultados esperados	Se envía 1 correo al administrador, especificando las razones del envío de la alerta, o sea, en el cuerpo del correo se detalla la cantidad de conexiones que tiene el servidor, la carga del procesador y finalmente la memoria libre que tiene el mismo y dicha alerta se guarda en el fichero app.config.

Tabla 10 Caso de prueba del camino básico 4

Caso de prueba para el camino básico 4	
Camino 4	1-2-3-4-5-6-7-8-9-10-11-13-14-15-16-17
Descripción	El objeto Reloj se inicializa, el parámetro fecha tiene la fecha actual del sistema, los métodos a los que se llaman toman valores y el parámetro ok se inicializa en 0.

Entrada	Valor de fecha: 31/05/11 Valor de: Uso_CPU()=80% Valor de: CantidadConexiones()=12 Valor de: EspacioDisco()=20GB Valor de: Memoria()=8% Valor de: GetCargaProcesador()=70% Valor de: GetCantidadConexiones()=8 Valor de: GetEspacioDisco()=10GB Valor de: GetMemoriaLibre()=10% Valor de: ok=1
Resultados esperados	Se envía 1 correo al administrador, especificando las razones del envío de la alerta, o sea, en el cuerpo del correo se detalla la cantidad de conexiones que tiene el servidor, la carga del procesador y finalmente el espacio en disco que tiene el mismo y dicha alerta se guarda en el fichero app.config.

Tabla 11 Caso de prueba del camino básico 5

Caso de prueba para el camino básico 5	
Camino 5	1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17
Descripción	El objeto Reloj se inicializa, el parámetro fecha tiene la fecha actual del sistema, los métodos a los que se llaman toman valores y el parámetro ok se inicializa en 0.
Entrada	Valor de fecha: 31/05/11 Valor de: Uso_CPU()=80% Valor de: CantidadConexiones()=12

	Valor de: EspacioDisco()=20GB Valor de: Memoria()=12% Valor de: GetCargaProcesador()=70% Valor de: GetCantidadConexiones()=8 Valor de: GetEspacioDisco()=10GB Valor de: GetMemoriaLibre()=10% Valor de: ok=1
Resultados esperados	Se envía 1 correo al administrador, especificando las razones del envío de la alerta, o sea, en el cuerpo del correo se detalla la cantidad de conexiones que tiene el servidor, la carga del procesador, el espacio en disco y finalmente la memoria libre que tiene el mismo y dicha alerta se guarda en el fichero app.config.

Tabla 12 Caso de prueba del camino básico 6

Caso de prueba para el camino básico 6	
Camino 6	1-2-4-5-7-8-10-11-13-14-16-17
Descripción	El objeto Reloj se inicializa, el parámetro fecha tiene la fecha actual del sistema, los métodos a los que se llaman toman valores y el parámetro ok se inicializa en 0.
Entrada	Valor de fecha: 31/05/11 Valor de: Uso_CPU()=50% Valor de: CantidadConexiones()=7 Valor de: EspacioDisco()=10GB Valor de: Memoria()=8%

	Valor de: GetCargaProcesador()=70% Valor de: GetCantidadConexiones()=8 Valor de: GetEspacioDisco()=20GB Valor de: GetMemoriaLibre()=10% Valor de: ok=0
Resultados esperados	No se envía ningún correo al administrador pues no ocurre ninguna alerta.

Resultados esperados de la prueba:

Después de aplicarse las pruebas de caja blanca, en este caso las pruebas del camino básico se llegó a la conclusión de que los resultados obtenidos fueron satisfactorios pues se pudo verificar que el flujo de trabajo de la función está correcto ya que cumple con las condiciones necesarias que se habían planteado.

4.5 Conclusiones

En este último capítulo se modeló completamente el sistema para monitoreo de eventos en servidores de máxima disponibilidad. Para ello se realizó el modelo de implementación contribuyendo a una mejor organización de los componentes y sus dependencias, creando el diagrama de despliegue y el diagrama de componentes. Posterior a la implementación del sistema a desarrollar se realizaron las pruebas de caja blanca al software para verificar la calidad y el adecuado funcionamiento del mismo.

Conclusiones generales

Luego de concluida toda la investigación que trajo consigo el desarrollo del sistema para monitoreo de eventos en servidores de máxima disponibilidad se pueden arribar a las siguientes conclusiones:

- ✓ La utilización de métodos teóricos y empíricos facilitó ampliar el conocimiento acerca del estado del objeto de estudio.
- ✓ Las tareas investigativas que se trazaron contribuyeron a que la organización de la investigación se tornara fácil.
- ✓ El análisis exhaustivo que se realizó de las herramientas y tecnologías actuales permitió que se escogieran las herramientas adecuadas para la construcción del sistema.
- ✓ El modelado de todo el ciclo de desarrollo del software permitió visualizar el sistema que se deseaba construir, y condujo a que todas las actividades fueran orientadas hacia la calidad.
- ✓ La aplicación realizada permite alertar al administrador del servidor a través de un correo electrónico de fallas que puedan afectar la disponibilidad de los servicios en los servidores de máxima disponibilidad.
- ✓ Con la puesta en práctica de la aplicación se lograría monitorear los eventos en los servidores de máxima disponibilidad de la Universidad de las Ciencias Informáticas.

Recomendaciones

A partir de los resultados o beneficios que proporciona este trabajo de diploma, se proponen las siguientes recomendaciones:

- ✓ Investigar otras posibles funcionalidades y que se realice la implementación de las mismas.
- ✓ Realizar un instalador con el código fuente de la aplicación.
- ✓ Realizar una ayuda más completa que abarque aspectos más relevantes del sistema.
- ✓ Realizar otra aplicación del lado del administrador que le permita al mismo realizar la configuración completa del sistema sin tener que ir necesariamente al servidor.

Referencias Bibliográficas

Scott W. Ambler. 2011. Ambysoft. [En línea] 2011. [Citado el: 15 de 4 de 2011.] www.ambysoft.com/unifiedprocess/agileUP.html.

Arriagada, Francisco. 2011. Sistema de monitoreo con procesamiento inteligente de alarmas. 2011.

Bodnar, Jan. 2011. ZetCode. [En línea] 2011. [Citado el: 15 de 5 de 2011.] zetcode.com/articles/netbeanscdevelopment.

canalvisualbasic.net. 2011. Canal Visual Basic.net. [En línea] 2011. [Citado el: 5 de 2 de 2011.] www.canalvisualbasic.net/manual-net/c-sharp/#cSharp.

CCM-Benchmark Network. 2011. kioskea.net. [En línea] 2011. [Citado el: 16 de 12 de 2010.] es.kioskea.net/contents/langages/langages.php3.

Definicion.de. 2010. Definicion.de. [En línea] 2010. [Citado el: 10 de 11 de 2010.] http://definicion.de/monitoreo.

—. 2011. Definicion.de. [En línea] 2011. [Citado el: 6 de 11 de 2010.] http://definicion.de/servidor.

EcuRed. 2010. EcuRed. [En línea] 2010. [Citado el: 26 de 12 de 2010.] www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado ...

eva.uci.cu. 2011. eva.uci.cu. [En línea] 2011. [Citado el: 5 de 2 de 2011.] eva.uci.cu/mod/resource/view.php?id=35381.

Fabien Potencier, François Zaninotto. 2008. Symfony 1.1, la guía definitiva. s.l. : E. Apress. 2008.

Gracia, Joaquin. 2003. Ingeniero Software. [En línea] 2003. [Citado el: 14 de 5 de 2011.] www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php.

Guía Ubuntu. 2011. Guía Ubuntu. [En línea] 2011. [Citado el: 20 de 1 de 2011.] www.guia-ubuntu.org/index.php?title=Eclipse..

Informáticos, U.C.M. Servicios. 2011. Manual Basico de Programacion en C++. 2011.

Informatizate. 2010. [En línea] 2010. [Citado el: 15 de 12 de 2010.] www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html...

Jack. 2007. Bitácora de un programador. [En línea] 2007. [Citado el: 10 de 12 de 2010.] jackopc.blogspot.com.

lenguajes-de-programacion.com. 2011. Lenguajes de programación. [En línea] 2011. [Citado el: 6 de 2 de 2011.] www.lenguajes-de-programacion.com/programacion-java.shtml.

linuxlandit.blogspot.com. 2011. Linuxlandit & the Conqueror Worm. [En línea] 2011. [Citado el: 5 de 5 de 2011.] linuxlandit.blogspot.com/2011/01/qt-simulator-is-fast-and-lightweight.html.

masadelante.com. 2011. masadelante.com. [En línea] 2011. [Citado el: 1 de 4 de 2011.] www.masadelante.com/faqs/tcp-ip..

Meyer, Lisandro Damian Nicanor Perez. 2007. Introduccion al desarrollo multiplataforma con qt 4. 2007.

Monografias.com S.A. 2011. Monografías.com. [En línea] 2011. [Citado el: 1 de 2 de 2011.] http://www.monografias.com/trabajos73/herramientas-case-proceso-desarrollo-software/herramientas-case-proceso-desarrollo-software2.shtml#ejemplosda..

nagios.org. 2011. Nagios. [En línea] 2011. [Citado el: 1 de 12 de 2010.] www.nagios.org..

pandora.sourceforge.net. 2011. Pandora FMS. [En línea] 2011. [Citado el: 5 de 12 de 2010.] pandora.sourceforge.net.

pixelcoblog.com. 2011. Pixelco blog. [En línea] 2011. [Citado el: 28 de 1 de 2011.] pixelcoblog.com/qt-creator-completo-entorno-de-desarrollo-multiplataforma..

Plusformacion.com. 2011. Plusformacion.com. [En línea] 2011. [Citado el: 25 de 4 de 2011.] http://www.plusformacion.com/Recursos/r/Herramientas-CASE-para-proceso-desarrollo-Software.

programacionweb.net. 2011. Programacion Web.net. [En línea] 2011. [Citado el: 25 de 3 de 2011.] http://www.programacionweb.net/articulos/articulo/?num=412..

ramos.utfsm.c. 2011. Ramos. [En línea] 2011. [Citado el: 20 de 5 de 2011.] www.ramos.utfsm.cl/doc/1125/sc/Clase2.ppt.

scribd.com. 2010. es.scribd.com. [En línea] 2010. [Citado el: 1 de 12 de 2010.] //es.scribd.com/doc/50371367/5-Bases-Para-El-Abordaje-Gerencial-de-Proyectos-de-MFC.

—. 2011. Scribd. [En línea] 2011. [Citado el: 4 de 2 de 2011.] http://www.scribd.com/doc/3062020/Capitulo-I-HERRAMIENTAS-CASE.

tuxmaya.wordpress.com. 2011. Tux Maya Blog//GNU Linux. [En línea] 2011. [Citado el: 1 de 2 de 2011.] tuxmaya.wordpress.com/2010/01/23/programacion-en-c-y-qt-qt-creator..

Universidad Tecnológica Naciona. 2011. Universidad Tecnológica Nacional, Facultad de Tucumán. [En línea] 2011. [Citado el: 3 de 2 de 2011.] www.frt.utn.edu.ar/sistemas/paradigmas/lenguajes.htm#_Java...

usabilidadweb.com.ar. 2011. Usabilidad.com.ar. [En línea] 2011. [Citado el: 10 de 2 de 2011.] www.usabilidadweb.com.ar/cpp.php..

Veasuip.com. 2011. Veasuip.com. [En línea] 2011. [Citado el: 20 de 11 de 2010.] http://www.veasuip.com/conceptos/concepto_servidor.html.

Zona QT. 2011. Zona QT. [En línea] 2011. [Citado el: 10 de 5 de 2011.] www.zonaqt.com/tutoriales/tutorial-de-qt-4-por-zona-qt-0.

Bibliografía

Scott W. Ambler. 2011. ***Ambysoft. [En línea] 2011. [Citado el: 15 de 4 de 2011.] www.ambysoft.com/unifiedprocess/agileUP.html.***

Arriagada, Francisco. 2011. ***Sistema de monitoreo con procesamiento inteligente de alarmas. 2011.***

Bodnar, Jan. 2011. ***ZetCode. [En línea] 2011. [Citado el: 15 de 5 de 2011.] zetcode.com/articles/netbeanscdevelopment.***

canalvisualbasic.net. 2011. ***Canal Visual Basic.net. [En línea] 2011. [Citado el: 5 de 2 de 2011.] www.canalvisualbasic.net/manual-net/c-sharp/#cSharp.***

CCM-Benchmark Network. 2011. ***kioskea.net. [En línea] 2011. [Citado el: 16 de 12 de 2010.] es.kioskea.net/contents/langages/langages.php3.***

Definicion.de. 2010. ***Definicion.de. [En línea] 2010. [Citado el: 10 de 11 de 2010.] http://definicion.de/monitoreo.***

—. 2011. ***Definicion.de. [En línea] 2011. [Citado el: 6 de 11 de 2010.] http://definicion.de/servidor.***

EcuRed. 2010. ***EcuRed. [En línea] 2010. [Citado el: 26 de 12 de 2010.] www.ecured.cu/index.php/Eclipse,_entorno_de_desarrollo_integrado ...***

eva.uci.cu. 2011. ***eva.uci.cu. [En línea] 2011. [Citado el: 5 de 2 de 2011.] eva.uci.cu/mod/resource/view.php?id=35381.***

Fabien Potencier, François Zaninotto. 2008. ***Symfony 1.1, la guía definitiva. s.l. : E. Apress. 2008.****Gracia, Joaquin. 2003.* ***Ingeniero Software. [En línea] 2003. [Citado el: 14 de 5 de 2011.] www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php.***

Guía Ubuntu. 2011. ***Guía Ubuntu. [En línea] 2011. [Citado el: 20 de 1 de 2011.] www.guia-ubuntu.org/index.php?title=Eclipse..***

Informáticos, U.C.M. Servicios. 2011. ***Manual Basico de Programacion en C++. 2011.***

Informatizate. 2010. ***[En línea] 2010. [Citado el: 15 de 12 de 2010.] www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html...***

Jack. 2007. ***Bitácora de un programador. [En línea] 2007. [Citado el: 10 de 12 de 2010.] jackopc.blogspot.com.***

Larma, Craig. 2011. ***UML y Patrones. 2011.***

lenguajes-de-programacion.com. 2011. ***Lenguajes de programación. [En línea] 2011. [Citado el: 6 de 2 de 2011.] www.lenguajes-de-programacion.com/programacion-java.shtml.***

linuxlandit.blogspot.com. 2011. ***Linuxlandit & the Conqueror Worm. [En línea] 2011. [Citado el: 5 de 5 de 2011.] linuxlandit.blogspot.com/2011/01/qt-simulator-is-fast-and-lightweight.html.***

masadelante.com. 2011. ***masadelante.com. [En línea] 2011. [Citado el: 1 de 4 de 2011.] www.masadelante.com/faqs/tcp-ip..***

Meyer, Lisandro Damian Nicanor Perez. 2007. ***Introduccion al desarrollo multiplataforma con qt 4. 2007.***

Monografias.com S.A. 2011. ***Monografías.com. [En línea] 2011. [Citado el: 1 de 2 de 2011.] http://www.monografias.com/trabajos73/herramientas-case-proceso-desarrollo-software/herramientas-case-proceso-desarrollo-software2.shtml#ejemplosda..***

nagios.org. 2011. ***Nagios. [En línea] 2011. [Citado el: 1 de 12 de 2010.] www.nagios.org..***

pandora.sourceforge.net. 2011. ***Pandora FMS. [En línea] 2011. [Citado el: 5 de 12 de 2010.] pandora.sourceforge.net.***

pixelcoblog.com. 2011. ***Pixelco blog. [En línea] 2011. [Citado el: 28 de 1 de 2011.] pixelcoblog.com/qt-creator-completo-entorno-de-desarrollo-multiplataforma..***

Plusformacion.com. 2011. ***Plusformacion.com. [En línea] 2011. [Citado el: 25 de 4 de 2011.] http://www.plusformacion.com/Recursos/r/Herramientas-CASE-para-proceso-desarrollo-Software.***

programacionweb.net. 2011. ***Programacion Web.net. [En línea] 2011. [Citado el: 25 de 3 de 2011.] http://www.programacionweb.net/articulos/articulo/?num=412..***

ramos.utfsm.c. 2011. ***Ramos. [En línea] 2011. [Citado el: 20 de 5 de 2011.] www.ramos.utfsm.cl/doc/1125/sc/Clase2.ppt.***

scribd.com. 2010. ***es.scribd.com. [En línea] 2010. [Citado el: 1 de 12 de 2010.] //es.scribd.com/doc/50371367/5-Bases-Para-El-Abordaje-Gerencial-de-Proyectos-de-MFC.***

—. 2011. ***Scribd. [En línea] 2011. [Citado el: 4 de 2 de 2011.] http://www.scribd.com/doc/3062020/Capitulo-I-HERRAMIENTAS-CASE.***

tuxmaya.wordpress.com. 2011. ***Tux Maya Blog//GNU Linux. [En línea] 2011. [Citado el: 1 de 2 de 2011.] tuxmaya.wordpress.com/2010/01/23/programacion-en-c-y-qt-qt-creator..***

Universidad Tecnológica Naciona. 2011. ***Universidad Tecnológica Nacional, Facultad de Tucumán. [En línea] 2011. [Citado el: 3 de 2 de 2011.] www.frt.utn.edu.ar/sistemas/paradigmas/lenguajes.htm#_Java...***

usabilidadweb.com.ar. 2011. ***Usabilidad.com.ar. [En línea] 2011. [Citado el: 10 de 2 de 2011.] www.usabilidadweb.com.ar/cpp.php..***

Veasuip.com. 2011. ***Veasuip.com. [En línea] 2011. [Citado el: 20 de 11 de 2010.] http://www.veasuip.com/conceptos/concepto_servidor.html.***

Zona QT. 2011. ***Zona QT. [En línea] 2011. [Citado el: 10 de 5 de 2011.] www.zonaqt.com/tutoriales/tutorial-de-qt-4-por-zona-qt-0.***