Universidad de las Ciencias Informáticas

Facultad1

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Tema: Herramienta para la administración de estaciones de trabajo Nova-Xerberos.

Autores:

Alejandro Veitía Ramos

Osvaldo Ordaz Pedroso

Tutores:

Ing. Raydel Miranda Gómez

MSc. Alain Guerrero Enamorado

La Habana, Cuba

Junio de 2013

Año 55 de la Revolución

Declaración de autoría:

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Osvaldo Ordáz Pedroso

_____

Firma del Autor

Alejandro Veitía Ramos

_____

Firma del Autor

Ing. Raydel Miranda Gómez

_____

Firma del Autor

MSc. Alain Guerrero Enamorado

_____

Firma del Autor

Se pueden adquirir conocimientos y conciencia a lo largo de toda la vida, pero jamás en ninguna otra época de su existencia una persona volverá a tener la pureza y el desinterés con que, siendo joven, se enfrenta a la vida.

Fidel Castro.

Agradecimientos

Osvaldo

Agradezco especialmente a mis padres por estar conmigo siempre.
A mis compañeros del aula 10108 que fue mi primer grupo y en especial a Lazara Yanet.
A los del grupo 01108 y a los nuevos integrantes que se unieron en 4to año. A mi amigo compañero y camarada Raciel por todos estos años de amistad. A mi amigo de la infancia Guillermo Luzua por su ayuda y paciencia, y a mi jefe del MININT Juan Heriberto Pérez De Corcho Mirabal Pedroso por su ayuda.
Por último y no menos importante, mi compañero de tesis Alejandro Veitía Ramos.

Agradecimientos

Alejandro

A mis padres, por brindarme su ayuda, confianza, dedicación y comprensión en el desarrollo de la tesis, apoyarme en las decisiones que he tomado en la vida y alentarme a seguir adelante.

A mis familiares que de una forma u otra se han preocupado por el bienestar mío y me han regalado su cariño.

A mis compañeros de los 5 años de universidad, en especial Adriam, Alfredo, Yannier, Osdany, al grupo 10106 y 1510.

A todo los que contribuyeron en la elaboración de la tesis especialmente a: Keiver, Dayron, Alexis.

Agradecimiento especial Osvaldo Ordaz Pedroso, mi compañero de tesis.

Dedicatoria

Osvaldo:

A mi mamá y a mi papá por apoyarme en todo.....

Alejandro:

A mis padres por darme todo el apoyo necesario para terminar la carrera.

A mis abuelos Domingo y Laura, donde estén, me sigan guiando y apoyando con mucha fe, para triunfar en la vida, siempre con las enseñanzas que me supieron dar… Los quieres.

Resumen

La Universidad de las Ciencias Informáticas (UCI), cuenta con varias facultades dentro de las cueles se encuentra la Facultad 1, la misma presenta un conjunto de laboratorios para la docencia y la producción de software, además de contar con varias estaciones de trabajo que son administradas y configuradas por un mínimo grupo de técnicos informáticos.

El objetivo del presente trabajo es desarrollar la aplicación versión (v1.0) Nova-Xerberos, para la administración y configuración de las estaciones de trabajo de los laboratorios, permitiendo mejorar el proceso de control y auditoría de las máquinas, la gestión de aplicaciones, así como de configurar la base de datos, que puede ser local o externa.

Para realizar el desarrollo de la aplicación se hace uso de varias herramientas, tales como: QtCreator, PostgresSQL, el lenguaje de programación C++ y la tecnología de comunicación Zeromq, una vez implementada la aplicación se realizan pruebas funcionales y se corrigen los errores detectados, garantizando de esta manera que todas las funcionalidades se comporten correctamente.

Palabras claves: Administración, configuración y estaciones de trabajo.

Índice

Índice de tabla

Índice de figuras

Introducción

La Universidad de las Ciencias Informáticas (UCI), cuenta con varias facultades dentro de las cueles se encuentra la Facultad 1, esta es la encargada de realizar el proceso de migración a software libre y código abierto, tanto a la universidad como a las empresas que soliciten este servicio. En estos momentos se cuenta con 3 laboratorios de producción que se encargan del desarrollo de aplicaciones para distribuciones de software libre.

Actualmente dicha facultad presenta 27 laboratorios destinados para la docencia y la producción de software, contando con más de 950 estaciones de trabajo, aplicando en su mayoría la distribución de GNU/Linux Nova. Se dispone de menos de 15 técnicos informáticos por turno, para llevar a cabo el servicio de administración de las máquinas, existiendo un déficit de personal, el cual provoca una carga excesiva de trabajo por la cantidad de estaciones que deben controlar y la frecuencia con la que deben realizar dichas operaciones de control, lo que trae como consecuencia una pérdida de tiempo.

Por ejemplo: El personal debe realizar un estricto control del hardware de cada máquina, de este trabajo manual se generan auditorías que pueden contener varios problemas entre otros, la información errónea de las propiedades de las estaciones de trabajo.

Una vez planteado lo anterior, se identifica el siguiente **problema científico**:

¿Cómo mejorar el proceso de configuración y administración de las estaciones de trabajo en los laboratorios de la facultad 1?

Objeto de estudio del presente trabajo de investigación:
Investigación de las tecnologías informáticas existentes para la administración remota de estaciones de trabajo, enmarcándose en el **campo de acción** para la administración remota de estaciones de trabajo con sistema GNU/Linux.

Objetivo general:
Desarrollar una aplicación informática que permita configurar y administrar remotamente las estaciones de trabajo.

Objetivos específicos:

En el marco de esta investigación se ha desglosado el objetivo general en siguientes objetivos:

- Realizar un estudio para la identificación de mecanismos para la interconexión y comunicación de ordenadores.
- Realizar un estudio de las tecnologías para la administración de ordenadores.
- Desarrollar una herramienta para la administración remota.
- Validar la herramienta desarrollada.

Idea a defender:

El desarrollo de una aplicación para la administración remota de ordenadores con sistema GNU/Linux, se minimizará la pérdida de tiempo y el exceso de trabajo en el proceso de configuración y administración de las estaciones de trabajo en los laboratorios de la facultad 1.

Tareas de investigación:

Para dar cumplimiento a los objetivos específicos se planifican lo siguiente:

- Identificación de herramientas de administración remota.
- Identificación de los mecanismos que existen para administrar remotamente las estaciones de trabajo.
- Identificación y selección de las tecnologías base para la comunicación entre los componentes de la aplicación.
- Identificación de los requisitos funcionales y no funcionales del sistema.
- Diseño de la herramienta.
- Implementación de la herramienta para dar solución a la problemática existente**.**
- Diseño y realización de las pruebas necesarias a la herramienta para garantizar el correcto funcionamiento del mismo.

Métodos de investigación científica:

- **El Analítico-Sintético:** Este método permitió procesar la información de diferentes fuentes bibliográficas que se especializan en las herramientas de administración de estaciones de trabajo con sistemas GNU/Linux, para identificar las características generales y funcionalidades de las mismas.

- **La Observación:** Este método se utiliza porque es un procedimiento fácil de llevar a cabo y permite percibir directamente los hechos de la realidad objetiva.
- **Histórico-Lógico:** Mediante este método se realizó una revisión histórica y análisis de la trayectoria del desarrollo de aplicaciones similares, para tomar una posición al respecto en cuanto a características y funcionalidades.

Estructura del documento

Capítulo 1. Fundamentación Teórica: Este capítulo se centra en un estudio de las aplicaciones de administración de forma remota de estaciones de trabajo, donde se exponen las distintas herramientas, lenguajes y tecnologías a utilizar para la implementación de la aplicación.

Capítulo 2. Diseño y análisis de la solución propuesta: En este capítulo se plasma una propuesta de solución de la interfaz, realizando un modelado del sistema, se identifican cuáles son las funcionalidades que debe realizar y sus características.

Capítulo 3. Implementación y pruebas: Este capítulo se exponen los artefactos de la implementación y las pruebas a las cuales fue sometida la misma, en vistas de elevar la calidad del sistema e identificar posibles errores.

Capítulo 1: Fundamentación Teórica

En el presente capítulo se exponen de manera general los aspectos teóricos relacionados con la administración remota. También cuenta con un análisis de las herramientas**,** tecnologías y metodologías utilizadas para dar solución al problema planteado, así como de los conceptos asociados al dominio del problema que se presentarán en este capítulo, teniendo como objetivo facilitar un mejor entendimiento de la situación.

1.1 Mecanismos de Comunicación

1.1.1 Middleware

Middleware: Es un software de conectividad que consiste en un conjunto de servicios que permiten interactuar a múltiples procesos que se ejecutan en distintas máquinas a través de una red. [1]

Ofrece una ventaja entre las funcionalidades y métodos para que sean usadas en otras aplicaciones, siendo las mismas transparente a la implementación de estos servicios de comunicación en el cliente, brindando una independencia total y bajo acoplamiento en la implementación de las aplicaciones.

1.1.2 Transmisión síncrona

Es la técnica que consiste en el envío de una trama de datos que configura un bloque de información comenzando con un conjunto de bits de sincronismo y terminando con otro conjunto de bits de final de bloque. En este caso, los bits de sincronismo tienen la función de sincronizar los relojes existentes tanto en el emisor como en el receptor. [2]

Dicha transmisión se realiza con un ritmo que se genera centralizadamente en la red y es el mismo para el emisor como para el receptor.

1.1.3 Transmisión asíncrona

Cuando el proceso de sincronización entre emisor y receptor se realiza en cada palabra de código transmitido. Esta sincronización se lleva a cabo a través de unos bits especiales que definen el entorno de cada código.

En una transmisión asíncrona no hay ninguna relación temporal entre la estación que transmite y la que recibe. Es decir, el ritmo de presentación de la información al destino no tiene por qué coincidir con el ritmo de presentación de la información por la fuente. Es un tipo de relación típica para la transmisión de datos. [2]

A continuación se mencionarán algunos mecanismos que utilizan estos tipos de transmisión.

1.1.4 RPC

RPC (del inglés Remote Procedure Call, Llamada a Procedimiento Remoto) es un protocolo que permite a un programa de ordenador ejecutar código en otra máquina remota sin tener que preocuparse por las comunicaciones entre ambos. De esta manera el programador no tenía que estar pendiente de las comunicaciones, estando éstas encapsuladas dentro de las RPC. [3]

Las RPC son muy utilizadas dentro del paradigma cliente-servidor, siendo el cliente el que inicia el proceso solicitando al servidor que ejecute cierto procedimiento o función y enviando éste de vuelta el resultado de dicha operación al cliente.

1.1.5 MOM

Message Oriented Middleware (MOM por su sigla en inglés) es un tipo de software específico que fue creado para abordar algunas deficiencias de los RPC, con el uso de la mensajería. Al igual que RPC, MOM proporciona una *Application Programming Interface* (API por su sigla en inglés) estándar, a través de hardware, plataformas del sistema operativo y redes. [4]

Este software, que opera con los principios de colas de mensajes, permite la transmisión segura de información entre redes diferentes. Las aplicaciones que se ejecutan en esas redes, realizan una entrega garantizada de mensajes, donde se vela que los mismos se entreguen en el orden en que se envían. También incluye control de acceso, cifrado y privacidad de mensaje. Con MOM los mensajes se envían y se reciben por lo general de forma asíncrona. [5]

MOM es un software intermediario que se encarga de intercambiar mensajes entre aplicaciones. Además, a diferencia de RPC, MOM proporciona un envío y recibo de mensajes asíncrona, ya que esta tecnología permite que la aplicación cliente siga trabajando sin tener que esperar que el mensaje sea procesado por el servidor. Dado que los mensajes son almacenados en una cola de mensaje, a la espera de ser leídos, permitiendo así una transmisión segura entre ambas aplicaciones.

El mecanismo de comunicación escogido es MOM, por todas las características antes mencionadas. Este mecanismo presenta varias tecnologías, las cuales son mencionadas a continuación.

1.1.6 Tecnologías de MOM

RabbitMQ

RabbitMQ es un sistema multiplataforma de mensajería, que permite realizar de forma sencilla la persistencia de los datos y posibilita la confirmación de la entrega de los mensajes. La aplicación servidora de RabbitMQ está desarrollada con el lenguaje de programación Erlang y utiliza el Framework[1]Open Telecom Plataform. Los mensajes son enviados a través de Intercambiadores antes de arribar a una Cola. Existen aplicaciones clientes para varios lenguajes de programación. El código fuente está disponible bajo la Licencia Pública de Mozilla y cuenta con una comunidad de personas que contribuye a su desarrollo. [6]

ActiveMQ

ActiveMQ es un software que se distribuye bajo la Licencia Apache, de código abierto, que aplica plenamente el Servicio de Mensajes de Java 1.1 (JMS). Actúa como mediador entre aplicaciones emisoras y receptoras. Proporciona comunicación asíncrona entre aplicaciones, brinda una cantidad de API para diferentes lenguajes como PHP, C/C++, .NET y Ruby. Soporta diferentes protocolos de conexión como HTTP[2], TCP[3], SSL[4], entre otros y también tiene la capacidad de utilizar cualquier base de dato. [7]

ZeroMQ

ZeroMQ es una biblioteca de mensajería fácil de usar mediante la programación, que está desarrollado en C++. Brinda sockets que transportan mensajes a través de varios transportes como en proceso (inproc),

[1]Es una plataforma que predefine patrones a seguir por los desarrolladores.
[2]Protocolo de Transferencia de Hipertexto. Del inglés Hypertext Transfer Protocol.
[3]Protocolo de Control de Transmisión. Del inglés *Transmission Control Protocol.*
[4]Siglas del inglés *Secure Socket Layer.*

entre proceso (ipc), TCP (tcp) y multicast[5] (pgm o epgm). Se pueden conectar sockets[6]entre aplicaciones con patrones como pregunta-respuesta, publicación-suscripción y entre otros. Esta tecnología es muy rápida y cuenta con una cantidad elevada de API en diferentes idiomas, tales como: C++, C#, PHP, Python, Delphi, Erlang, Perl, Ruby, Java y funciona en la mayoría de sistemas operativos. Además ZeroMQ se distribuye bajo la Licencia Lesser General Public License (por su sigla en inglés LGPL) v3 de código abierto. [8]

Una vez analizado estas tecnologías se hace necesaria la selección de una de ellas. A continuación se presenta una tabla que responde a esta problemática.

Tecnología	Presenta una buena Documentación	Es un software libre	Baja latencia	No requiere de la instalación de un servidor
Activemq	P	P	NP	NP
Rabbitmq	P	P	NP	NP
Zeromq	P	P	P	P

Tabla 1. Características a tener en cuenta para la selección de la tecnología

Descripción de las letras NP y P:
P: significa que presenta esa característica.
NP: significa que no presenta esa característica.
Después de mostrar esta tabla se hace evidente que la tecnología seleccionada es Zeromq.

1.2 Herramientas para la administración remota de estaciones de trabajo

En la actualidad uno de los mayores problemas que se enfrenta cualquier empresa o institución con un número considerable de estaciones de trabajo, es poder administrar adecuadamente las mismas por igual, razón por la cual desarrolladores se han encargado de proponer herramientas informáticas para lograr estos objetivos.

[5]Envío de paquetes de información a varias estaciones de trabajo de manera simultánea.
[6]Es un método de comunicación entre una aplicación cliente y una servidora.

Las herramientas de administración remota de estaciones de trabajo, permite controlar a distancia cualquier ordenador que esté conectado a la red, posibilitando al usuario gestionar los problemas que afectan a sus ordenadores, desde su propio puesto de trabajo.

También estas herramientas brindan múltiples posibilidades para tener un mayor control sobre las estaciones de trabajo, no solo disminuye los desplazamientos del usuario hacia las distintas estaciones de trabajo, sino que también puede disminuir el tiempo y los recursos humanos.

La mayoría de estas herramientas, utilizan la arquitectura cliente-servidor, donde un cliente realiza peticiones a otro programa (el servidor) que le da respuesta.

1.2.1 Modelo cliente-servidor

La arquitectura cliente-servidor es un modelo de aplicación distribuida, en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, que le da respuesta, esto significa que todas las gestiones que se realizan se concentran en el servidor, de manera que en él se disponen los requerimientos provenientes de los clientes que tienen prioridad. [9]

Figura 1. Modelo Cliente-Servidor

A continuación se presentarán algunas herramientas, que utilizan este tipo de arquitectura y que se

dedican a la administración remota de estaciones de trabajo.

1.2.2 Herramientas para GNU/Linux

Cfengine

Cfengine es una herramienta de gestión centralizada de equipos que utiliza una arquitectura cliente/servidor. Cfengine está principalmente diseñado para la administración y gestión de ficheros de configuración, pero también puede gestionar puntos de montaje, cron y paquetes instalados. También soporta la gestión de equipos desconectados, si el servidor ordenara un cambio y un cliente afectado por ese cambio estuviese desconectado, al arrancar este observaría que su número de revisión de configuración es inferior. Entonces solicitaría los cambios al servidor para sincronizar su estado. [10]

El proyecto Cfengine lleva activo desde 1993 y es software libre con licencia GPL, la misma utiliza un lenguaje de alto nivel (Ruby) para definir las acciones a realizar y la configuración se realiza íntegramente desde consola y mediante ficheros de configuración.

Puppet

Puppet es un sistema muy parecido a Cfengine, de hecho nace con muchas referencias a Cfengine. Puppet es una herramienta para la monitorización y administración centralizada de equipos, multiplataforma, que permite gestionar archivos, usuarios, grupos, paquetes y servicios desde una localización central. Puede ejecutar comandos o scripts, montajes de sistemas de archivos o hacer configuraciones de red. Además utiliza una arquitectura cliente/servidor, donde todos los clientes pueden hablar con uno o más servidores centrales. [11]

Por otra parte esta herramienta a pesar de ser muy potente, presenta algunas desventajas como: poca documentación, pues continuamente están sacando versiones nuevas debido a que es un proyecto nuevo, no tiene un soporte técnico y tampoco posee una interfaz amigable que ayude al administrador a configurar los equipos.

Landscape

Landscape es una herramienta desarrollada por la empresa Canonical, para la monitorización y administración centralizada de equipos con Ubuntu. Este software consta de una arquitectura

cliente/servidor, donde toda la configuración y gestión de los equipos remotos se realiza mediante una interfaz web alojada en el servidor. [12] Por otra parte al estar diseñada específicamente para Ubuntu, se integra de maravilla con las particularidades del sistema operativo.

Presenta funcionalidades como:

- Instalación y gestión de paquetes.
- Ejecución de scripts en máquinas remotas.
- Gestión usuarios, grupos, tareas programadas.

En cambio su mayor desventaja es su costo:

Unidades	Costo (USD)
1	$150
10	$129
25	$120
50	$112
100	$105
250	$102

Tabla 2. Costo por Unidades

Estos costos son en entornos empresariales y para entornos educativos solo sería, el 30% del costo.

SistClon

Es un software orientado al mantenimiento y la administración remota de un conjunto de terminales clientes. Su principal objetivo es la clonación de imágenes de sistemas operativos, pero además puede utilizarse para realizar auditorías de hardware para sistemas operativos GNU/Linux. Posee soporte para diferentes tipos de motherboards y usa la tecnología de clientes ligeros para iniciar las estaciones de trabajos. A continuación se mostraran algunas funcionalidades de esta aplicación:

Funcionalidades de SistClon

- Obtiene la información del hardware de las estaciones de trabajos conectados al sistema.

- Administra, controla y actualiza las estaciones de trabajos de manera remota mediante la ejecución de comandos y scripts [13].

1.2.3 Herramientas para Windows

Hyena

Hyena está diseñado para simplificar y centralizar casi todas las tareas de gestión, ofreciendo nuevas capacidades para la administración del sistema. Hyena presenta muchos recursos como la administración de usuarios, grupos (tanto locales como globales), dominios, computadoras, servicios, dispositivos, archivos, impresoras, sesiones, abrir archivos, espacio en disco y los derechos de usuario. Por otra parte, se puede utilizar cualquier cliente de Windows para la gestión de cualquier Windows NT, Windows 2000, Windows XP/Vista o Windows Server 2003/2008 [14].

Desktop Orbiter

Desktop Orbiter es un sistema para Windows, que proporciona una multitud de herramientas para gestionar y monitorizar el uso de cada PC remotamente, así el administrador podrá apagar, bloquear o reiniciar cualquier PC, hacer una "foto" de la pantalla, desactivar ratón, teclado, enviar un mensaje e incluso hasta podrá subir o bajar el volumen. Además de las acciones anteriores también existen filtros para impedir la ejecución de determinados programas. [15]

Por otra parte esta herramienta utiliza una arquitectura cliente/servidor y está compuesto por dos programas, desktop orbiter que es el servidor y desktop satellite que es el cliente. [16]

Después de este breve resumen, se mostrarán varias características de esta aplicación:

- Monitoreo de la actividad del usuario, manteniendo la privacidad.
- Posibilidad de implementar filtros de sitios web y filtros de acceso a software.
- Supervisión y administración de máquinas

NetSupervisor

NetSupervisor es una potente herramienta para Windows, de gran utilidad para administradores de redes, que permite supervisar de forma global todo cuanto sucede en la red. Esta aplicación presenta todas las funciones necesarias para la vigilancia y control de redes, sin descuidar por ello el diseño de una interfaz,

amigable y fácil de usar. [17]

Esta herramienta permite diseñar, administrar y monitorizar todo tipo de redes, incluso redes a través de Internet, todo ello desde una única interfaz, con un considerable ahorro de tiempo y esfuerzo.

A continuación se mostrarán algunas características de NetSupervisor:

- Permite diseñar, administrar y monitorear cualquier red.
- Supervisa cualquier equipo conectado a una red.
- Inventario de equipos y propiedades de los mismos.
- Detección de caída de red.
- Representación gráfica de los datos de respuesta para cada equipo.
- Buscador avanzado de equipos en la red.

Luego de realizar un análisis de estas herramientas que permiten la administración remota de estaciones de trabajo, se descarta el sistema Landscape, dado que establece un costo por la utilización del sistema, ver tabla 2. También se descartó la herramienta Puppet, debido a que utiliza el sistema de comunicación XMLRPC, que es un middleware, con transmisión de datos síncrona. Se desechó Cfengine por ser una herramienta no extensible, ya dispone de un solo lenguaje de programación para definir las acciones a realizar en cada máquina, además de no poseer una interfaz gráfica. Por lo tanto se decide que la mejor opción es desarrollar una herramienta que sea, libre de costo alguno, asíncrona, extensible y que presente una interfaz gráfica fácil de usar por el usuario.

1.3 Herramienta y lenguaje a utilizar

Un punto importante en el desarrollo de cualquier software es la selección correcta de las herramientas, lenguajes y tecnologías a utilizar, pues con ello se estaría asegurando a largo plazo una mejor producción, una disminución del tiempo de trabajo y un producto final con calidad.

1.3.1 Lenguaje de programación

Lenguaje de programación: Es cualquier lenguaje artificial que puede utilizarse para definir una secuencia de instrucciones para el procesamiento de un ordenador o computadora.

Actualmente existen muchos lenguajes de programación de alto nivel dentro de los más conocidos se

encuentra C++.

El **C++** es un lenguaje de programación, diseñado a mediados de los años 1980, por Bjarne Stroustrup, como extensión del lenguaje de programación C. Las principales características del C++ son el soporte para programación orientada a objetos y el soporte de plantillas o programación genérica y está considerado por muchos como el lenguaje más potente, debido a que permite trabajar tanto a alto como a bajo nivel. [18]

Fue seleccionado C++ porque es el lenguaje con que está desarrollada la aplicación SistClon que es el antecesor de Xerberos, lo que generara un buen entendimiento de futuras implementaciones. También se elige por ser un lenguaje potente, multiplataforma y libre que brinda facilidades a la hora de realizar una programación a bajo nivel. Este lenguaje posee un tratamiento de memoria que permite que el tiempo de ejecución de este sea menor que los programas escritos en Java o C#.

1.3.2 Herramienta de desarrollo

Actualmente existen muchas herramientas de programación que brindan al implementador la comodidad para desarrollar software. Uno de los objetivos a tener en cuenta es seleccionar un IDE correcto y que apoye al desarrollador en términos como el completamiento de código y el detector de errores**.**

Qt

Qt es un framework multiplataforma creado por la compañía Quasar Technologies. El nombre de Qt, significaba Quasar Technologies toolkit. Esta compañía después cambiaría de nombre a TrollTech y finalmente a Trolltech, que actualmente es propiedad de Nokia, la función más conocida de Qt es la de la creación de interfaces de usuario, sin embargo no se limita a esto, ya que también provee varias clases para facilitar ciertas tareas de programación como el manejo de sockets, soporte para programación multihilo, comunicación con base de datos, manejo de cadenas de caracteres, entre otras [19].

Se seleccionó este framework por su función más conocida que es la creación de interfaces de usuario, y por su IDE integrado QtCreator que utiliza como lenguaje de programación C++, además de su fuerte documentación como ayuda al desarrollador.

1.3.3 Herramienta y lenguaje de modelado

Lenguaje de Modelación Unificado o Unified Modeling Language, en inglés (UML).

El **UML** sirve para especificar, visualizar y documentar esquemas de sistemas de software orientado a objetos. UML no es un método de desarrollo, lo que significa que no sirve para determinar qué hacer en primer lugar o cómo diseñar el sistema, sino que simplemente, ayuda a visualizar el diseño y a hacerlo más accesible para otros. UML está controlado por *Object Management Group* (OMG) y es el estándar de descripción de esquemas de software. [20]

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. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. [21] La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. Presenta un 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 usos.

1.3.4 Gestor de base de datos

Un gestor de base de datos o sistema de gestión de base de datos (SGBD o DBMS) es un software que permite introducir, organizar y recuperar la información de las bases de datos.

Existen actualmente SGBD libres y comerciales, dentro de los libres se encuentran PostgreSQL.

PostGreSQL

Fue el pionero en muchos de los conceptos existentes en el sistema objeto-relacional actual, incluido, más tarde en otros sistemas de gestión comercial. PostGreSQL es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. A pesar de esto, PostGreSQL no es un sistema de gestión de bases de datos puramente orientado a objetos. [22]

Se escogió este SGBD porque es una tecnología libre, multiplataforma, por su documentación de carácter excepcional y además soporta la mayor parte de los tipos de datos de SQL: integer, boolean, char, varchar, date, entre otros.

1.4 Metodología de desarrollo

La metodología de desarrollo de software son los procedimientos, técnicas y herramientas utilizadas para el desarrollo de un sistema. Provee al programador una guía para desarrollar una aplicación, pues enseña a un equipo de trabajo a dividir un proyecto en etapas, muestra qué tareas se llevan a cabo en cada etapa, qué restricciones deben aplicarse, qué técnicas, herramientas y cómo se controla un proyecto. Las metodologías se han dividido en dos grandes grupos: Ágiles/Ligeras y Pesadas/Tradicionales.

1.4.1 OpenUP

OpenUP - un proceso ágil y unificado, que contiene el conjunto mínimo de prácticas que ayudan a los equipos a ser más eficaces en el desarrollo de software. OpenUP abraza una filosofía pragmática y ágil que se centra en la naturaleza colaborativa de desarrollo de software. Se trata de una herramienta agnóstica, bajo la ceremonia proceso que puede utilizarse tal cual o ampliarse para tratar una amplia variedad de tipos de proyecto. [23]

Se escogió OpenuP por ser una metodología ágil, basada en el desarrollo iterativo e incremental, apropiado para proyectos pequeños y de bajos recursos. Por otra parte es centrado en la arquitectura y es dirigido por caso de uso, lo que permitirá desarrollar el sistema de acuerdo con la prioridad de los procesos a llevar a cabo por el mismo, dotando al desarrollador de una base real del estado de la implementación, la cual siempre debe de estar en correspondencia con la arquitectura definida para el sistema.

1.5 Conclusiones Parciales

En el desarrollo de este trabajo se usará UML como lenguaje de modelado, integrado con Visual Paradigm como herramienta CASE y OpenUP como metodología de desarrollo, el *Framework* Qt, con su lenguaje nativo C++ y su IDE integrado Qt Creator, el cual permite la integración de manera sencilla con la biblioteca Zeromq, la que será utilizada para establecer la comunicación entre las aplicaciones y como gestor de base de datos se utilizará PostgreSQL para el almacenamiento de la información.

Capítulo 2: Diseño y análisis de la solución propuesta

En el presente capítulo se inicia la construcción de la solución propuesta: el desarrollo de una herramienta para la administración remota de estaciones de trabajo, con la guía de OpenUP como metodología seleccionada. Se describen los principales artefactos que comprenden la modelación del dominio y el levantamiento de requisitos. Específicamente se muestra el modelo de dominio y los principales conceptos asociados a él. Se detallan los requisitos funcionales y los no funcionales que debe cumplir el sistema, así como el actor del mismo, el diagrama de casos de uso y su descripción.

2.1 Modelo de dominio

Después de efectuar un análisis profundo del problema descrito en la introducción de este trabajo, se llega a la conclusión de que en la presente investigación no se definen concretamente todos los procesos y existen múltiples responsabilidades, por lo que se decide dar un nuevo enfoque para la solución del problema. Para ello se realiza un modelo de dominio. Este modelo contribuye posteriormente a identificar algunas clases que se utilizarán en el sistema.

El Modelo de Dominio (o Modelo Conceptual) es una representación visual de los principales conceptos u objetos del mundo real, significativos para un problema o área de interés. Este es de gran ayuda para desarrolladores y usuarios, ya que de esta forma se utiliza un vocabulario común y pueden entender el contexto en que se enmarca el sistema. [24]

A continuación la Figura 1 muestra el diagrama de clases del dominio:

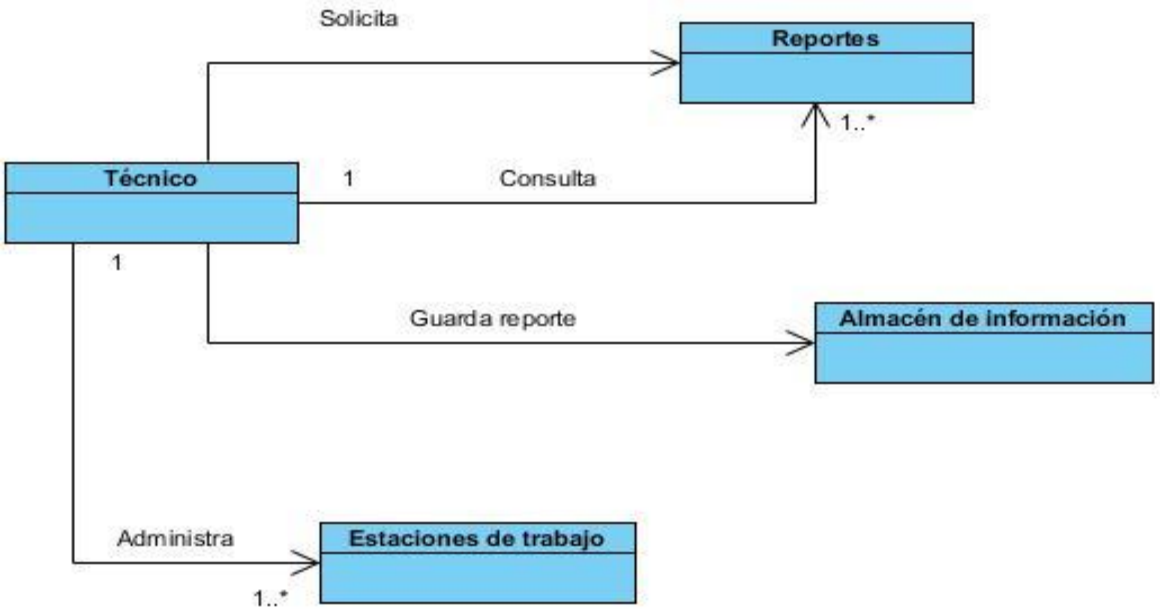

Figura 2. Modelo del dominio

Glosario de términos.

Técnico: persona capacitada que interactúa con la aplicación.

Reportes: componente encargado de manipular la información del hardware de las estaciones de trabajo.

Almacén de información: lugar donde se almacena los reportes.

2.2 Levantamiento de Requisitos

Al terminar de establecer la relación entre las clases que conforman el dominio del sistema y describir los principales conceptos a través del modelo de dominio, se presentan los principales artefactos del flujo de trabajo con la definición de los requisitos funcionales y los no funcionales para la herramienta de administración remota y centralizada.

2.2.1 Requisitos funcionales

Los requisitos funcionales (RF) son capacidades o condiciones que el sistema debe cumplir. Este tipo de requerimiento específica algo que el sistema debe ser capaz de realizar.

A continuación se enumeran las funcionalidades que debe cumplir el sistema:

RF_1: Configurar la base de dato.
RF_2: Generar reporte de hardware.
RF_3: Reiniciar sistema.
RF_4: Apagar sistema.
RF_5: Encender estación de trabajo.
RF_6: Registrar IP
RF_7: Mostrar IP.
RF_8: Eliminar IP.
RF_9: Gestionar aplicaciones:

- Actualizar aplicaciones.
- Instalar aplicación.
- Desinstalar aplicación.

RF_10: Configurar repositorio.

2.2.2 Requisitos no funcionales

Los requisitos no funcionales (RNF) son propiedades que debe tener el sistema. Estos velan por temas como la seguridad y fiabilidad, así como la presentación de una interfaz amigable y usable para el usuario.

A continuación se mencionan los requisitos no funcionales:

Requisitos de usabilidad

RNF_1: La herramienta podrá ser usada por usuarios que tengan un nivel técnico de informática.

Requisitos de eficiencia

RNF_2: El tiempo de respuesta de las conexiones será como máximo de 3 segundos.

RNF_3: La velocidad de procesamiento de información, actualización y respuesta del sistema, estará dada por la cantidad de información que tenga que procesar y por la tecnología que se disponga para esto.

Requisitos de Apariencia o interfaz externa

RNF_4: Diseño sencillo y fácil de usar, permitiendo un rápido aprendizaje para utilizar el sistema.

Requerimientos de hardware

RNF_5: Pentium IV o superior.

RNF_6: 512 Mb de memoria RAM o superior.

RNF_7: Disco duro con capacidad de 80 GB o superior.

Requerimientos de Software

RNF_8: Plataforma de SO: GNU/Linux,

RNF_9: Instalación de las bibliotecas libpoco, Libzmq, libparted, libpqxx, libxmldatamanager.

Requisitos de Soporte

RNF_10: El sistema será modificable, permitiendo la inclusión de nuevos módulos.

RNF_11: Una vez terminado el producto, se instalará por el personal capacitado, esto incluirá pruebas y mantenimiento para verificar el buen funcionamiento de la aplicación.

Requisitos legales

RNF_11: Para el desarrollo de la aplicación se hará uso de herramientas de software libre.

2.3 Diagrama de casos de usos del sistema

Después de identificar los requisitos funcionales del sistema, es necesario modelar el Diagrama de Casos de Uso del Sistema (DCUS). Los casos de uso son un conjunto de escenarios que identifican una línea de utilización para el sistema que va a ser desarrollado, facilitando una descripción de cómo se usará el sistema. Un DCUS muestra la relación entre los actores del sistema y los casos de uso. [25]

A continuación la Figura 2 muestra el diagrama de casos de usos del sistema:

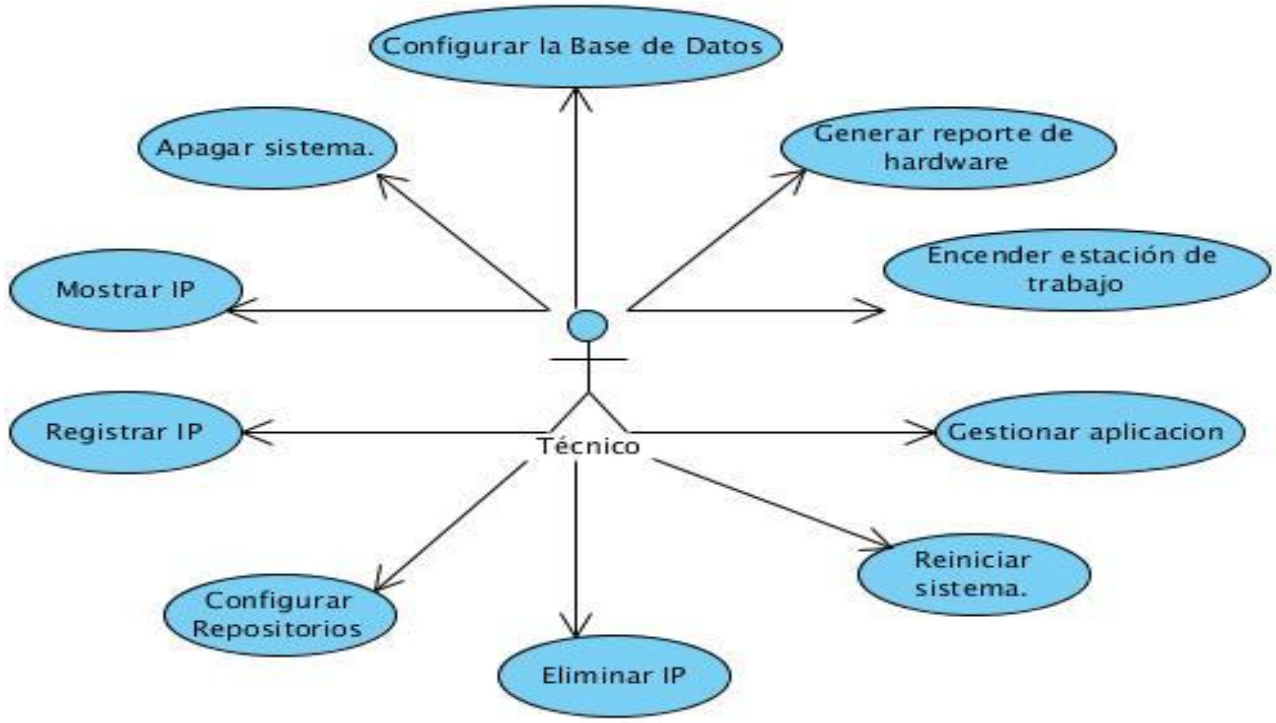

Figura 3. Diagrama del casos de usos del sistema

Descripción del actor:

El actor es la persona, con cierto nivel de informática que interactúa con la aplicación. Para la aplicación, el técnico informático se define como:

Actor	Descripción
Técnico Informático	Es la persona que se encarga de inicializar todos los procesos que puede ejecutar la aplicación.

Tabla 3. Descripción de los actores del sistema.

2.3.1 Descripción de los casos de uso del sistema

La descripción de los casos de uso del sistema, detallan las acciones que tienen lugar durante la interacción actor-sistema, es decir, describe el flujo de actividades que realiza el actor al hacer uso del sistema y las correspondientes respuestas del mismo. A continuación se mostrarán las descripciones de los casos de uso más relevantes. Los demás casos de usos se encuentran en los anexos 1.

Caso de Uso	Generar reporte de hardware
Actor	Técnico
Resumen	El caso de uso se inicia cuando el técnico selecciona la opción hardware, luego escoge una o varias direcciones IP de un listado y selecciona la opción aceptar.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	Deben estar registrados los IP de las estaciones de trabajo.
Referencias	RF_2
Flujo normal de eventos	
Acción del Actor	**Acción del sistema**
1. Selecciona la opción hardware.	2. Muestra el listado de los IP de las estaciones de trabajo. 3. Brinda la opción de: a. Si escoge un IP véase la sección un IP. b. Si escoge varios IP véase la sección de varios IP.
Sección "IP"	
Acción del actor	**Acción del sistema**

1. Seleccionar el botón aceptar.	2. Envía un mensaje con el nombre del componente "INFO_HW" y el comando "lshw -xml> /usr/share/xcs/hard.xml". 3. Recibe la información de hardware. 4. Guarda en la base de dato la información. 5. Muestra los datos del hardware.
Sección "Varios IP"	
Acción del actor	**Acción del sistema**
1. Seleccionar el botón aceptar.	2. Envía un mensaje con el nombre del componente "INFO_HW" y el comando "lshw -xml> /usr/share/xcs/hard.xml", a todas las máquinas seleccionadas. 3. Recibe la información de cada estación. 4. Guarda en la base de dato la información del hardware. 5. Muestra los datos de las estaciones escogidas.

Tabla 4. Descripción del caso de uso Generar reporte de hardware.

Caso de Uso	Configurar la base de datos.
Actor	Técnico
Resumen	El caso de uso se inicia cuando el técnico selecciona la opciónconfigurar la base de datos, luego se le muestra una interfaz para conectarse automáticamente a una base de local o conectarse una externa.
Complejidad	Media
Prioridad	Crítico
Precondiciones	Debe estar funcionando el gestor de la base de datos para establecer conexión.

Referencias	RF_1
Flujo normal de eventos	
Acción del Actor	**Acción del sistema**
1. Selecciona la opción configurar la base de datos.	2. Muestra una interfaz con dos opciones. a. conectarse localmente. b. muestra varios campos para conectarse a una base de datos externa.
3. Selecciona una opción: a. Si selecciona opción localhost véase la sección localhost. b. Si quiere conectarse a una base de datos externa véase la sección externa.	
Sección "Localhost"	
Acción del actor	**Acción del sistema**
1. Seleccionar el botón localhost.	2. Se conecta a la base de datos local. 3. Muestra un mensaje de confirmación.
Sección "Externa"	
Acción del actor	**Acción del sistema**
1. Introduce los datos.	2. Obtiene esos datos y se conecta a la base de datos externa. 3. Muestra un mensaje de confirmación.

Tabla 5. Descripción del caso de uso Configurar la base de dato.

Caso de Uso	Registrar IP
Actor	Técnico
Resumen	El caso de uso se inicia cuando el técnico selecciona la opción Registrar IP, luego se le muestra un formulario para introducir el IP de la estacione de trabajo.
Complejidad	Media
Prioridad	Crítico
Precondiciones	No tiene
Referencias	RF_6

Flujo normal de eventos	
Acción del Actor	**Acción del sistema**
1. Selecciona la opción Registrar IP.	2. Muestra un formulario para introducir los datos.
3. Introduce los datos	4. Obtiene los datos y lo adiciona a la lista de IP. 5. Guarda las direcciones de IP en un archivo de texto. 6. Muestra un mensaje de confirmación.

Tabla 6. Descripción del caso de uso Registrar IP.

2.4 Modelo de diseño

Durante este paso se logra una solución lógica que se funda en el paradigma orientado a objetos. Su esencia es la elaboración de diagramas, que muestran gráficamente como los objetos se comunican entre ellos a fin de cumplir con los requerimientos. [26]

Además, crea un punto de partida para las actividades de implementación que siguen. Es el punto de mayor importancia al final de la fase de elaboración y el comienzo de las iteraciones de construcción.

Diagrama de Clases del diseño

El diagrama de clases del diseño se encarga de representar los métodos y atributos de cada una de las clases del sistema, para mostrar de forma simple la colaboración y las tareas de cada una de ellas en relación al sistema que conforman. A continuación se representa el diagrama de clases del componente Admin, los demás diagramas de clases se encuentran en el anexo 2.

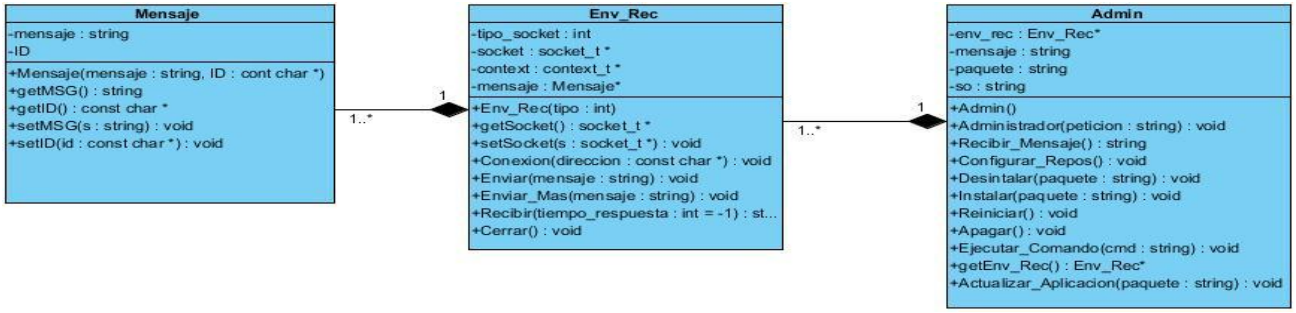

Figura 4. Diagrama de clases del componente Admin

En la Figura 4 se mostraron las clases que intervienen en el componente *Admin*.

Admin: Es la clase principal, la cual se encarga de realizar todas las operaciones, ya sea desde encender la máquina, hasta configurar los repositorios, contiene una instancia de la clase *Env_Rec*, la cual es utilizada para el envío y recibo de mensajes.

2.4.1 Arquitectura de la aplicación

Los patrones arquitectónicos son los encargados de definir la estructura de un sistema, especifican un conjunto predefinido de subsistemas con sus responsabilidades y una serie de recomendaciones para organizar los distintos componentes. [27]

Para la **arquitectura de la aplicación** se hará uso de dos estilos arquitectónicos que son:

El principal

- cliente-servidor

Secundario

- basado en componentes.

Estilo arquitectónico basado en componentes.

El estilo basada en componentes describe una aproximación de ingeniería de software al diseño y desarrollo de un sistema, enfocándose en la descomposición del diseño en componentes funcionales o lógicos que expongan interfaces de comunicación bien definidas. Por otra parte es un estilo para aplicaciones compuestas de componentes individuales. [28]

El principal objetivo de este estilo, es que los componentes sean fáciles de reemplazar, implicando que una nueva implantación de un componente pueda ser utilizada en lugar del anterior sin afectar el funcionamiento del sistema.

Un componente es un objeto de software que expone una interfaz que permite a un programa usar sus funcionalidades.

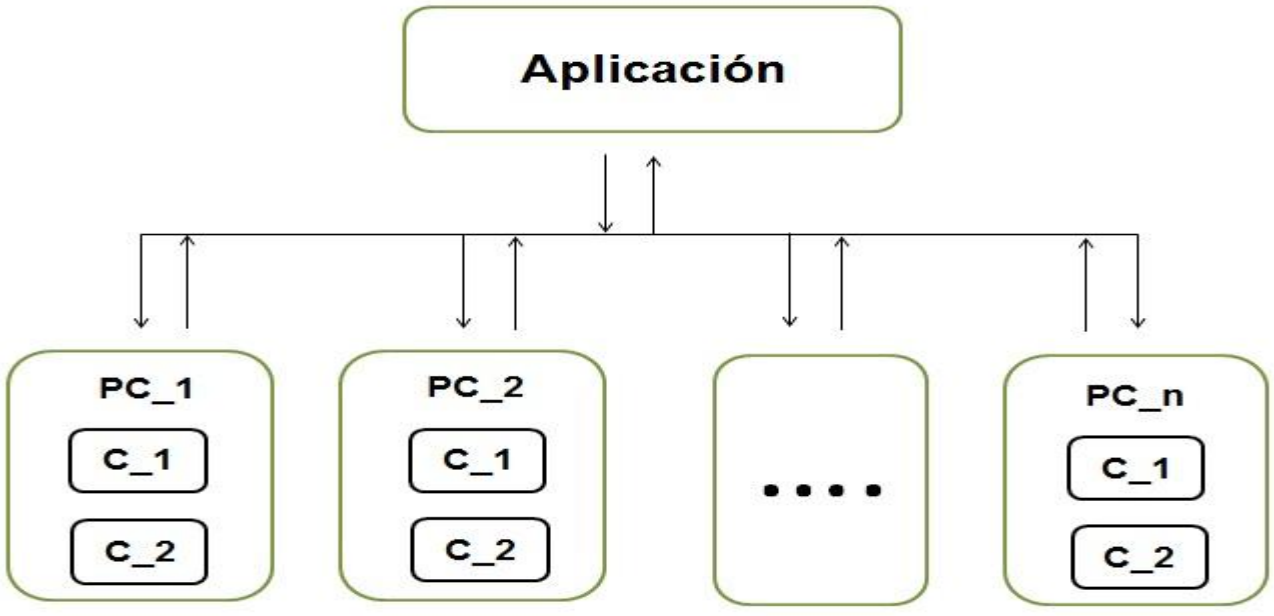

Figura 5. Arquitectura de la aplicación

En la figura anterior se muestra la arquitectura de la aplicación, donde esta herramienta envía parámetros de administración hacia las PC_1, PC_2,…… PC_n, permitiendo controlar la información que proviene desde las PC, y gestiona la información que se desea de las PC a través de los componentes C_1 y C_2.

2.4.2 Patrones de Diseño

Los patrones están concebidos como diseños orientados a objetos, son una forma de formalizar la reusabilidad de código y se plantean como una buena herramienta para el diseño. [29]

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces, brindando una solución ya probada y documentada.

Para el desarrollo de la aplicación se hará uso de los patrones GRASP (*General Responsability Asignment Software Patterns)* para lograr un buen diseño orientado a objetos.

GRASP

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones.

El patrón **experto:** plantea asignar una responsabilidad a la clase que tiene la información necesaria para cumplirla, de este modo se obtiene un diseño con mayor cohesión y así la información se mantiene encapsulada, posibilitando la disminución del acoplamiento.

El patrón **creador:** está diseñado para asignar la responsabilidad de crear una instancia de una clase determinada a otra. Se utiliza este patrón debido a la característica del sistema de ser orientado a objetos. Esto se evidencia en la clase *Env_Rec* (Envió y recibo de mensaje), que crea una instancia de la clase *Mensaje*.

Una **alta cohesión:** caracteriza a las clases con responsabilidades estrechamente relacionadas, que no realicen un trabajo enorme. Una clase con baja cohesión hace muchas cosas no afines o un trabajo excesivo, tales clases no son convenientes pues son difíciles de mantener, de reutilizar y de entender.

Una clase con alta cohesión mejora la claridad y la facilidad de su uso, su mantenimiento se simplifica y es fácil de reutilizar.

Controlador: El patrón controlador asigna la responsabilidad del manejo de los eventos del sistema a una clase. Todas las peticiones de la aplicación son manejadas por un solo controlador, que es el único punto de entrada de toda la aplicación en un entorno determinado. Cuando el controlador recibe una petición, utiliza el sistema de comunicación para enviar el nombre de una acción y el nombre de un módulo. Con el uso de este patrón se puede obtener como beneficio el incremento del potencial de los elementos que pueden ser reutilizados [30].

2.5 Diagramas de secuencia

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso.

A continuación se muestran los diagramas de secuencia para los requisitos funcionales reporte de hardware y configuración de la base datos, que son unos de los requisitos más importantes del sistema. Los diagramas restantes se encuentran en el anexo 3.

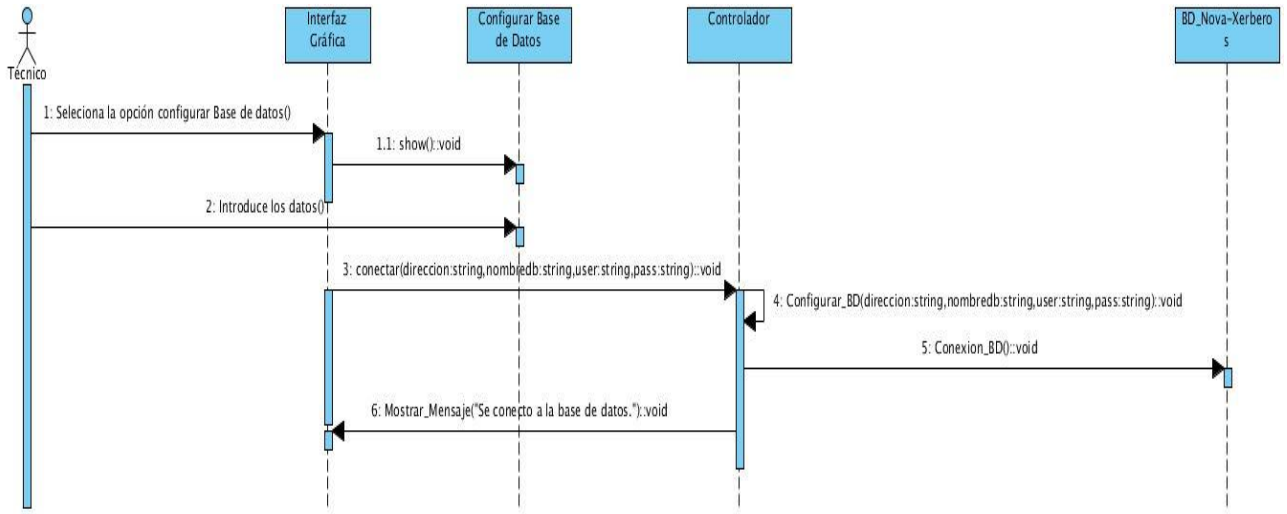

Figura 6. Diagrama de secuencia del caso uso configurar base de datos

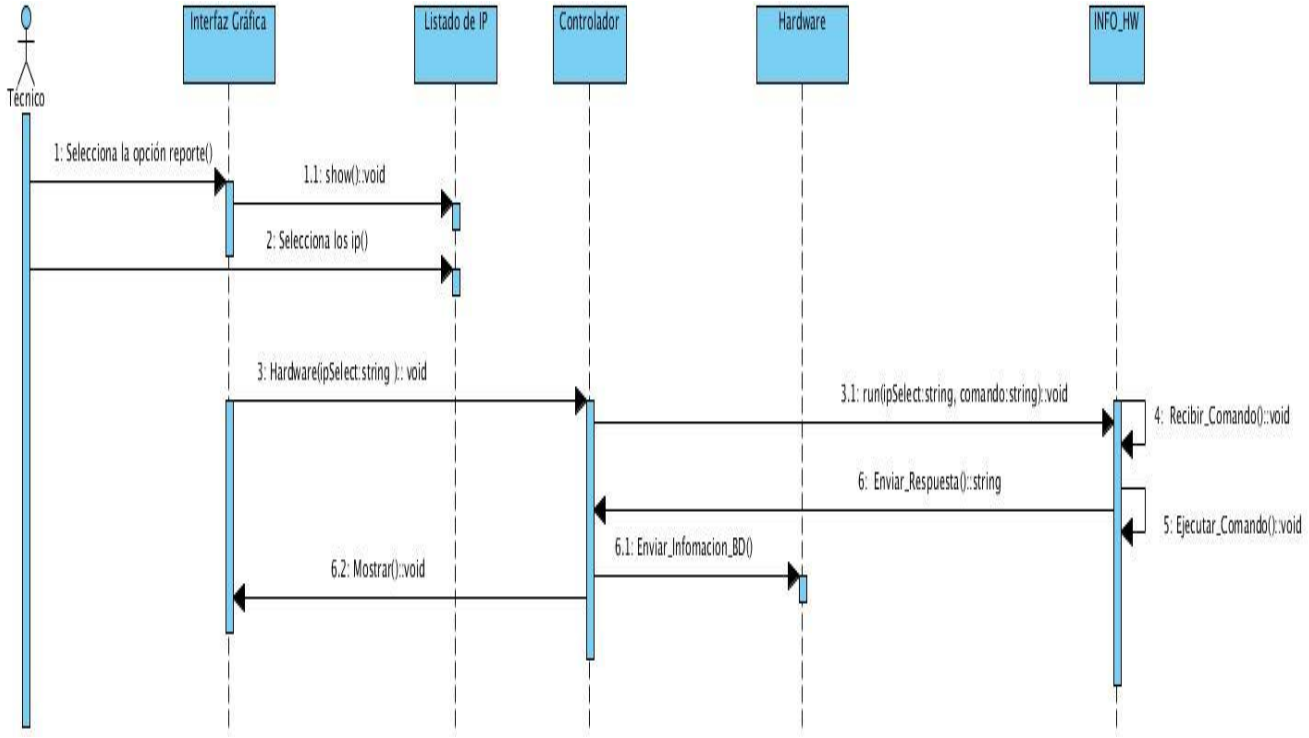

Figura 7. Diagrama de secuencia del caso de uso reporte de hardware

Capítulo 3: Implementación y pruebas

En el siguiente capítulo se describen los elementos relacionados con la implementación y todo lo referente a las pruebas realizadas al sistema. Dentro de los artefactos desarrollados en el capítulo se encuentran: El diagrama de componentes y el modelo de despliegue de la Herramienta.

3.1 Diagrama de componentes

Un diagrama de componentes permite visualizar con más facilidad la estructura general del sistema, además un diagrama de este tipo muestra las dependencias lógicas entre componentes de software, sean estos componentes de código fuente, binarios o ejecutables. [31]

En el diagrama de componentes realizado, se muestra como están distribuidos según el patrón arquitectónico utilizado.

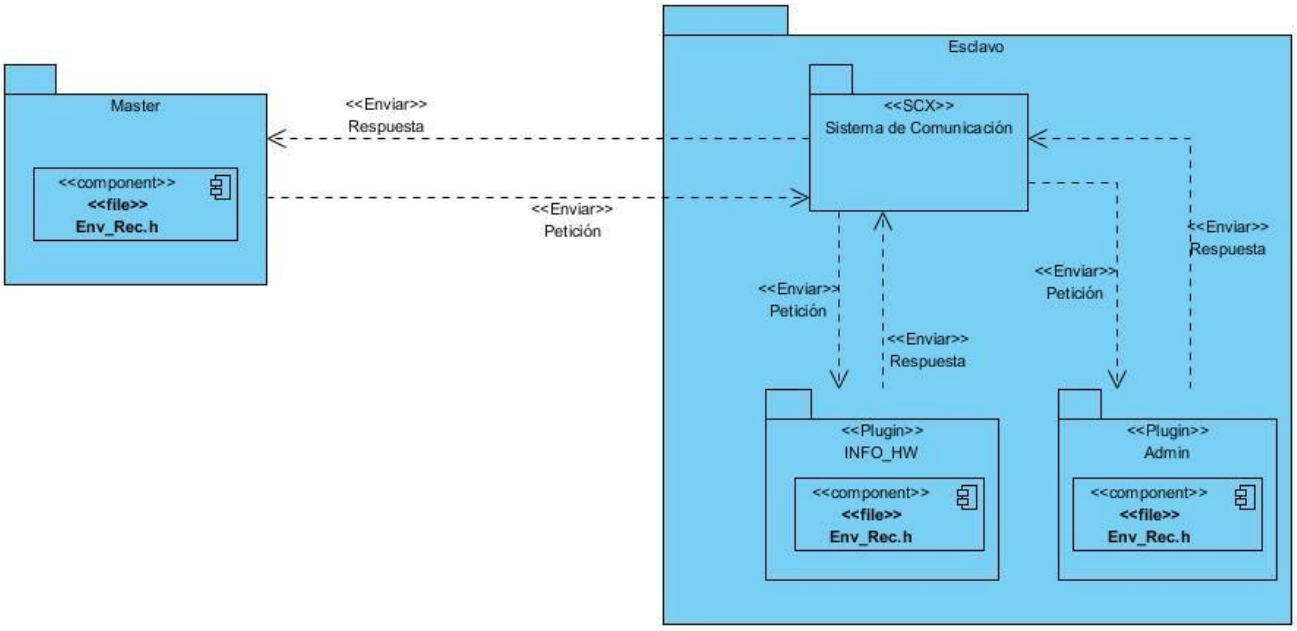

Figura 8. Diagrama de componente del sistema

Para mayor información, de cómo está compuesto el sistema, diríjase al anexo 4.

3.2 Modelo de despliegue

El diagrama de despliegue muestra la disposición física de los distintos nodos que componen un sistema y el reparto de los componentes sobre dicho nodo.

Este diagrama brinda una base para la compresión de la distribución física de un sistema a través de nodos. Es decir, describe los nodos físicos necesarios para la configuración de la plataforma donde se ejecutará el sistema.

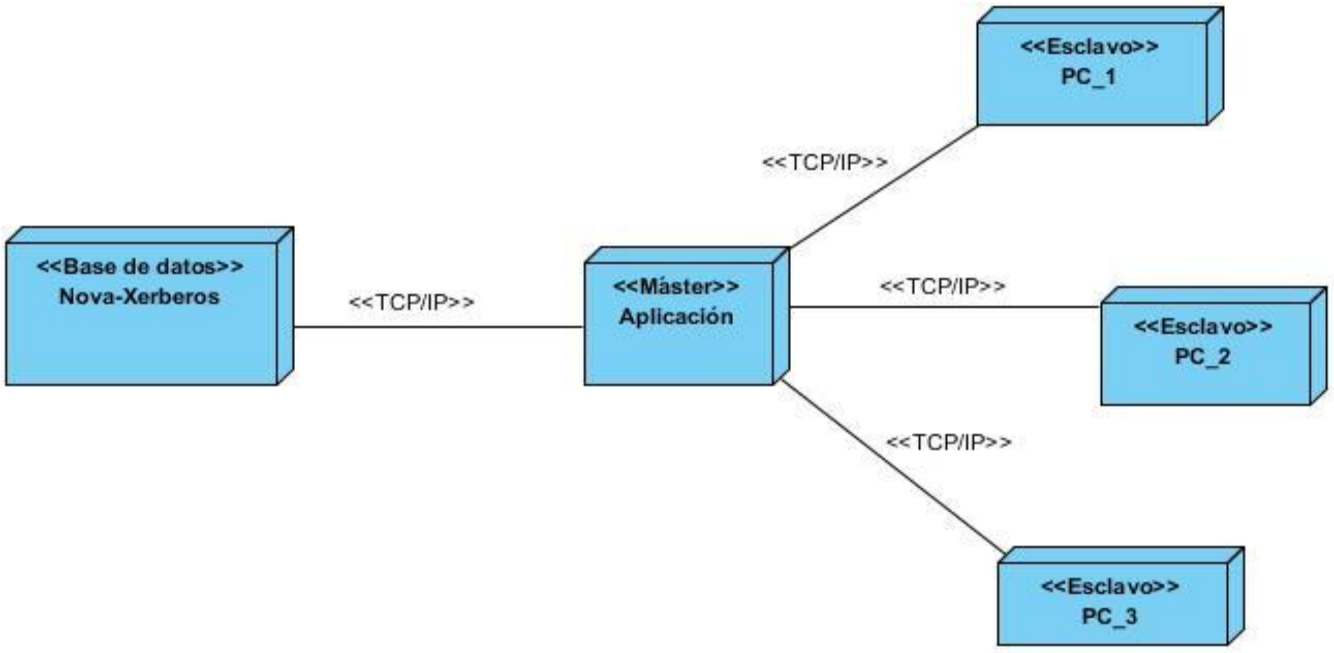

Figura 9. Diagrama de despliegue del sistema

Descripción de los Nodos

Un **nodo** es un objeto físico que representa un recurso computacional, generalmente con memoria y capacidad de procesamiento, puede representarse gráficamente en 3D, la instancia o tipos de nodos que se presenta como un cubo en los diagramas de implementación. [32]

Nodo	Descripción	Componente
Máster	Nodo que controla y se nutre de los servicios de los nodos esclavos, o sea, es el nodo que realiza las peticiones y recibe las respuestas de las mismas.	En este nodo se instala el paquete del Máster.
PC_Esclavo	Nodo que se encarga de dar respuesta a las peticiones del máster.	En esté nodo se instalan varios paquetes, como son: a-) El paquete del sistema de comunicación Xerberos (SCX). b-) El paquete de INFO_HW. c-) El paquete Admin.
BD_Nova-Xerberos	Nodo que representa a la base de datos, lugar donde se guarda la información	

Tabla 7. Descripción de Nodos

3.3 Modelo de Pruebas

Una vez generado el código fuente, es necesario probar el software desarrollado, para detectar y corregir la mayor cantidad de errores posible antes de entregarlo. Su objetivo es diseñar una serie de casos de prueba que tengan una alta probabilidad de encontrar errores. [33]

Para validar la solución propuesta y la correcta integración de los componentes que conforman la herramienta de administración remota de estaciones de trabajo, se hace necesario someter el software desarrollado, a un conjunto de pruebas que garantizarán la calidad y la detección de posibles errores que afecten el funcionamiento de la interfaz. Estas pruebas garantizaron la correcta cohesión entre el diseño y lo que se encuentra implementado, para la validación de esta aplicación se seleccionó la técnica de pruebas de caja negra.

Pruebas de caja negra.

Las pruebas de caja negra son diseñadas para validar los requisitos funcionales sin fijarse en el funcionamiento interno de un programa. [34]

La técnica de pruebas de caja negra se le realizó a las funcionalidades, de registrar IP, generar reporte de hardware, configurar base de datos y la gestión de aplicaciones. Esta técnica se basa más en el entendimiento de que es lo que hace, sin darle importancia a como lo hace, por lo que debe estar bien definida sus entradas y salidas, es decir su interfaz.

Métodos de pruebas de caja negra

- Los métodos de pruebas basados en grafos.
- Análisis de valores equivalentes.
- Adivinando el error.
- Partición equivalente.

Para llevar a cabo las pruebas, se utilizó la técnica de caja negra y el método empleado es la partición equivalente, el cual presenta las siguientes características:

La partición equivalente, se presenta como un método de prueba de caja negra, que divide el campo de entrada de un programa en clases de datos, de los que se pueden derivar casos de prueba. Un caso de prueba ideal detecta de forma inmediata una clase de errores que de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico. La partición equivalente se esfuerza por definir un caso de prueba que descubra ciertas clases de errores, reduciendo así el número de casos de prueba que deben desarrollarse. [34]

Los casos de pruebas realizados se encuentran en el anexo 5, donde se aplicó el método de partición equivalente.

Resultado de las pruebas

Concluida la primera etapa de las pruebas a aplicación Nova-Xerberos v1.0, se pudieron detectar diez no conformidades, las que fueron solucionadas y se sometió al sistema a una segunda etapa de pruebas, donde se detectaron cinco nuevas no conformidades y fueron solucionadas, y finalmente en una tercera etapa de prueba no se detectaron no conformidades, garantizando que todas las funcionalidades de la aplicación se ejecuten de acuerdo con lo previsto.

Conclusiones Generales

Con el desarrollo de la aplicación Nova-Xerberos v1.0, se cumplieron los objetivos específicos del trabajo propuesto, que permite configurar y administrar remotamente las estaciones de trabajo.

- La selección de las herramientas, tecnologías y los lenguajes de programación utilizada en el desarrollo de la aplicación, las que se encuentra QtCreator como IDE de programación, el lenguaje de desarrollo C++ y la tecnología MOM, utilizando la librería de desarrollo Zeromq para la comunicación entre aplicaciones, permitió la implementación de la aplicación.
- Se diseñó una arquitectura usando los estilos arquitectónicos cliente-servidor y basado en componentes, además se definieron los requisitos funcionales y no funcionales de la aplicación, se generaron los diagramas de clases donde muestra la información de referente a las tareas ingenieriles llevadas a cabo, realizando una descripción detallada de los casos de usos del sistema.
- Se implementó la herramienta de administración remota de estaciones de trabajo, la cual permite controlar varias máquinas remotamente, actuando sobre el hardware de las estaciones de trabajo de una subred, reduciendo el tiempo de configuración y administración de las máquinas de un laboratorio, así como también minimiza los recursos humanos.

  Se dieron respuesta a todos los errores generados en las pruebas realizadas a la aplicación después de su implementación en los laboratorios de la facultad, comprobando que la herramienta para la administración de estaciones de trabajo (Nova-Xerberos v1.0), se encuentra funcionando correctamente y lista para usarse en cualquier entorno.

Recomendaciones

Después de la investigación realizada y el desarrollo de la herramienta Nova-Xerberos versión 1.0, se recomienda lo siguiente:

- El desarrollo de futuras versiones de esta aplicación de administración.
- Continuar con el perfeccionamiento de los componentes realizados y agregar nuevos componentes que contribuyan a mejorar la administración remota de las estaciones de trabajo.

Referencias bibliográfica

[1] Dr. Sosa, Víctor J. 2011. MIDDLEWARE: Arquitectura para Aplicaciones Distribuidas. Disponible en http://www.tamps.cinvestav.mx/~vjsosa/clases/sd/Middleware_Recorrido.pdf

[2] Transmisión síncrona y asíncrona. Disponible en http://www.angelfire.com/linux/redes2/Trans_Sin_Asin.htm

[3] Remote Procedure Call. Disponible en http://www.slideshare.net/crack_708/rpc

[4] Yrityksen tietojärjestelmien integrointi, Petri Maaranen, MOM, 2006. Disponible en http://www.cs.jyu.fi/el/tjtse54_06/Luennot/6_MOM_s.ppt.

[5] Message-Oriented Middleware (MOM). Disponible en http://docs.oracle.com/cd/E18930_01/html/821-2443/aeraq.html

[6] Yurisleidy Hernández Moya, Daileny Hernández Barreiro, Luis Enrique Sánchez Arce1, Juan Carlos Lobaina Guzmán, Dovier Antonio Ripoll Méndez, Detección de software malicioso para el filtro de contenido Smart Keeper, publicado el 20 de diciembre de 2012. Disponible en http://rcci.uci.cu/index.php/rcci/article/view/236/182/236-838-1-PB.pdf

[7] ActiveMQ in Action. Disponible en http://www.manning.com/snyder/

[8] ØMQ Community. Disponible en http://www.zeromq.org/community

[9] Capítulo 5. Cliente-Servidor. Disponible en catarina.udlap.mx/u_dl_a/tales/documentos/lis/...a.../capitulo5.pdf

[10] Mikel Jiménez Fernández, Gestión centralizada de equipos, Irontec – Internet y Sistemas sobre GNU/Linux. Disponible en http://www.sc.ehu.es/scwebci/VC/streaming/1.pdf, página 19

[11] Mikel Jiménez Fernández, Gestión centralizada de equipos, Irontec – Internet y Sistemas sobre GNU/Linux. Disponible en http://www.sc.ehu.es/scwebci/VC/streaming/1.pdf, página 24

[12] Mikel Jiménez Fernández, Gestión centralizada de equipos, Irontec – Internet y Sistemas sobre GNU/Linux. Disponible en http://www.sc.ehu.es/scwebci/VC/streaming/1.pdf, página 15

[13] Manual de usuario Sistema de Clonación y distribución de Imágenes de Sistemas Operativos. Disponible en https://repositorio.geitel.prod.uci.cu/svn/xerberos/Xerberos/

[14] Hyena - Red y Monitoreo. Disponible en http://www.software.com.ar/hyena.html.

[15] Julián Gómez, Control total de los PC de tu red. Disponible en http://desktop-orbiter.softonic.com/.

[16] Desktop Orbiter. Disponible en http://www.aol-soft.com/es/desktop-orbiter/download.

[17] Elena Santos. Diseña, vigila y administra cualquier tipo de red. Disponible en http://netsupervisor.softonic.com/.

[18] Programación en C++/Introducción. Disponible en

http://es.wikibooks.org/wiki/Programaci%C3%B3n_en_C%2B%2B/Introducci%C3%B3n.

[19] Introducción a Qt, 25 de septiembre del 2010. Disponible en http://www.zonaqt.com/tutoriales/tutorial-b%C3%A1sico-de-qt-4

[20] Grady Booch, JimRumbaugh e Ivar Jacobson, Lenguaje unificado modelado booch. Disponible en http://www.librospdf.net/lenguaje-unificado-modelado-booch/1/.

[21] Visual Paradigm, "Visual Paradigm for UML," 2010. Disponible en http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p.

[22] PostGreSQL. Disponible en http://www.ecured.cu/index.php/PostGreSQL

[23] OpenUp. Disponible en http://www.ecured.cu/index.php/OpenUp.

[24] MARTINEZ, Ivet Carolina. Modelo Conceptual/ Modelo de dominio. Universidad Simón Bolívar. Disponible en http://lds.usb.ve/martinez/cursos/ci3715/clases6_AJ2010.pdf

[25] **Pressman, R. 2002**. Ingeniería del Software. Un enfoque práctico. Quinta edición. Disponible en la página 180.

[26] Parte_IV_Fase_del_Diseno_1. Disponible en http://eva.uci.cu/mod/resource/view.php?id=9400&subdir=/UML_y_Patrones

[27] Escuela técnica superior de ingeniería informática, Departamento de Lenguajes y Sistemas Informáticos, Tema 1: Patrones Arquitectónico. Disponible en http://www.lsi.us.es/docencia/get.php?id=1130

[28] Carlos Reynoso – Nicolás Kiccillof, UNIVERSIDAD DE BUENOS AIRES. Estilos y Patrones en la Estrategia de Arquitectura de Microsoft. Disponible en http://www.willydev.net/descargas/prev/Estiloypatron.pdf

[29] Patrones de Diseño. Definición - Conceptos de Programación. Disponible en https://sites.google.com/site/conceptoprogramacion/Home/patrones

[30] Veloso Hernández, Pedro. Uso de patrones de arquitectura. 1996.

[31] Diagrama de Componentes. Disponible en http://tvdi.det.uvigo.es/~avilas/UML/node49.html

[32] Marca Huallpara Hugo Michael, QuisbertLimachi Nancy Susana.ANALISIS Y DISEÑO DE SISTEMAS II. Diagrama de Despliegue. Disponible en es.scribd.com/doc/19317161/59/Diagrama-de-Despliegue

[33] Pressman. Técnicas de Prueba. Cap14_Parte_1. Disponible en la Página 1

[34] Pressman, R. 2002. Ingeniería del Software. Un enfoque práctico. Quinta edición. Disponible en la página de 200

[35] Pressman. Técnicas de Prueba. Cap14_ Parte_1. Disponible en la página 296

Bibliografía

RPC, Remote Procedure Call. www.slideshare.net/dianapaolalozano/rpc-llamadas-remotas

Arquitectura MOM (Message-Oriented Middleware) – Oracle. Disponible en www.oracle.com/.../arquitectura-mom-integracion-server-426198-esa.html

Introducción a activemq. Disponible en www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=ActiveMQ

Activemq. Disponible enhttp://activemq.apache.org/configuring-transports.html

ActiveMQ o RabbitMQ o ZeroMQ. Disponible en http://es.softuses.com/79563

Rabbitmq. Disponible en http://www.rabbitmq.com/

¿What is Puppet? Disponible en https://puppetlabs.com/puppet/what-is-puppet/

Arquitectura basada en Componentes - Blog de Juan Pelíez en Geeks.ms. Disponible en http://geeks.ms/blogs/jkpelaez/archive/2009/04/18/arquitectura-basada-en-componentes.aspx

Anexos

Anexo 1. Descripción de los caso de uso

Caso de Uso	Configurar repositorio
Actor	Técnico
Resumen	El caso de uso se inicia cuando el técnico selecciona la opción Configurar repositorio, luego escoge una o varias direcciones IP de un listado y selecciona la opción aceptar.
Complejidad	Media
Prioridad	Secundario
Precondiciones	Deben estar registrados los IP de las estaciones de trabajo.
Referencias	RF_10

Flujo normal de eventos	
Acción del Actor	**Acción del sistema**
1. Selecciona opción configurar repositorio	2. Muestra el listado de los ip de las estaciones de trabajo. Brinda la opción de: a. Si escoge un IP véase la sección un IP. b. Si escoge varios IP véase la sección de Varios IP.
Sección “”	
Acción del actor	**Acción del sistema**
1. Selecciona el botón aceptar.	2. Envía un mensaje (“Configurar repositorio”) al IP seleccionado. 3. Muestra un mensaje de confirmación.
Sección “Varios ip”	
Acción del actor	**Acción del sistema**
1. Selecciona el botón aceptar.	2. Envía un mensaje (“Configurar repositorio”) a todos los IP seleccionados. 3. Muestra un mensaje de confirmación.

Tabla 8. Descripción de caso de uso configurar repositorios

Caso de Uso	Reiniciar Sistema
Actor	Técnico
Resumen	El caso de uso se inicia cuando el técnico selecciona la opción Reiniciar Sistema, luego se le muestra el listado de estaciones de trabajo para seleccionar una estación o todas para reiniciar.
Complejidad	Media
Prioridad	Secundario
Precondiciones	Debe estar funcionando el sistema de comunicación en la estación de trabajo y el componente de Admin.
Referencias	RF_3

Flujo normal de eventos	
Acción del Actor	**Acción del sistema**
1. Selecciona la opción Reiniciar Sistema.	2. Muestra el listado de los IP de las estaciones de trabajo. Brinda la opción de: a. Si escoge un IP véase la sección un IP. b. Si escoge varios IP véase la sección de Varios IP.
Sección "IP"	
Acción del actor	**Acción del sistema**
1. Seleccionar el botón aceptar.	2. Envía un mensaje "Reiniciar" al IP seleccionado. 3. Muestra un mensaje de verificación.
Sección "Varios IP"	
Acción del actor	**Acción del sistema**
1. Seleccionar el botón aceptar.	2. Envía un mensaje ("Reiniciar") a todos los IP seleccionados. 3. Muestra un mensaje de verificación.

Tabla 9. Descripción de caso de uso Reiniciar

Caso de Uso	Apagar Sistema
Actor	Técnico
Resumen	El caso de uso se inicia cuando el técnico selecciona la opción Apagar Sistema, luego se le muestra el listado de estaciones de trabajo para seleccionar una estación o todas para apagar.
Complejidad	Media
Prioridad	Secundario
Precondiciones	Debe estar funcionando el sistema de comunicación en la estación de trabajo y el componente de Admin.
Referencia	RF_4

Flujo normal de eventos	
Acción del Actor	**Acción del sistema**
1. Selecciona la opción Apagar Sistema.	2. Muestra el listado de IP. Brinda la opción de: a. Si escoge un IP véase la sección un IP. b. Si escoge varios IP véase la sección de Varios IP.
Sección "IP"	
Acción del actor	**Acción del sistema**
1. Seleccionar el botón aceptar.	2. Envía el mensaje "Apagar" al IP seleccionado. 3. Muestra un mensaje de verificación.
Sección "Varios IP"	
Acción del actor	**Acción del sistema**
1. Seleccionar el botón aceptar.	2. Envía un mensaje ("Apagar") a todos los IP seleccionados. 3. Muestra un mensaje de verificación.

Tabla 10. Descripción del caso de uso apagar Sistema

Caso de Uso	Gestionar aplicaciones
Actor	Técnico
Resumen	El caso de uso se inicia cuando el técnico selecciona la opción Aplicación, luego se le muestra el listado de estaciones de trabajo, para seleccionar una estación o todas para instalar, desinstalar y actualizar.
Complejidad	Media
Prioridad	Secundario
Precondiciones	Debe estar funcionando el sistema de comunicación en la estación de trabajo, el componente de Admin y los repositorios actualizados.
Referencias	RF_9, RF_9.1, RF_9.2, RF_9.3

Flujo normal de eventos

Acción del Actor	Acción del sistema
1. Selecciona la opción Gestionar Aplicación.	2. Muestra varias opciones. a. Si escoge la opción de instalar, véase sección instalar aplicación. b. Si escoge la opción de desinstalar, véase sección desinstalar aplicación. c. Si escoge la opción de actualizar, véase sección actualizar aplicaciones.

Sección "Instalar aplicación"

Acción del Actor	Acción del sistema
2. Introduce el nombre de la aplicación.	1. Muestra un formulario, para introducir el nombre de la aplicación que desea instalar. 3. Muestra el listado de los IP de las estaciones de trabajo. Brinda la opción de: a. Si escoge un IP véase la sección un IP. b. Si escoge varios IP véase la sección de Varios IP.

Sección "IP"

Acción del Actor	Acción del sistema
1. Seleccionar el botón aceptar.	2. Envía un mensaje con el nombre de la aplicación que desea instalar al IP seleccionado. 3. Muestra un mensaje de verificación.

Sección "Varios IP"	
Acción del Actor	**Acción del sistema**
1. Seleccionar el botón aceptar.	2. Envía un mensaje con el nombre de la aplicación que desea instalar a todos los IP seleccionado. 3. Muestra un mensaje de verificación.
Sección "Desinstalar aplicación"	
Acción del Actor	**Acción del sistema**
2. Introduce el nombre de la aplicación.	1. Muestra un formulario, para introducir el nombre de la aplicación que desea instalar. 3. Muestra el listado de los IP de las estaciones de trabajo. Brinda la opción de: a. Si escoge un IP véase la sección un IP. b. Si escoge varios IP véase la sección de Varios IP.
Sección "IP"	
Acción del Actor	**Acción del sistema**
1. Seleccionar el botón aceptar.	2. Envía un mensaje con el nombre de la aplicación que desea desinstalar al IP seleccionado. 3. Muestra un mensaje de verificación.
Sección "Varios IP"	
Acción del Actor	**Acción del sistema**
4. Seleccionar el botón aceptar.	5. Envía un mensaje con el nombre de la aplicación que desea desinstalar a todos los IP seleccionados. 6. Muestra un mensaje de verificación.
Sección "Actualizar aplicación"	
Acción del Actor	**Acción del sistema**
	1. Muestra el listado de los IP de las estaciones de trabajo. Brinda la opción de: a. Si escoge un IP véase la sección un IP. b. Si escoge varios IP véase la sección de Varios IP.
Sección "IP"	

Acción del Actor	Acción del sistema
1. Seleccionar el botón aceptar.	2. Envía el mensaje “Actualizar” al IP seleccionado. 3. Muestra un mensaje de verificación.
Sección “Actualizar Subred”	
Acción del Actor	**Acción del sistema**
1. Seleccionar el botón Apagar.	2. Envía un mensaje “Actualizar” a todos los IP seleccionados. 3. Muestra un mensaje de verificación.

Tabla 11. Descripción del caso de uso Gestionar aplicaciones

Anexo 2. Diagramas de clases

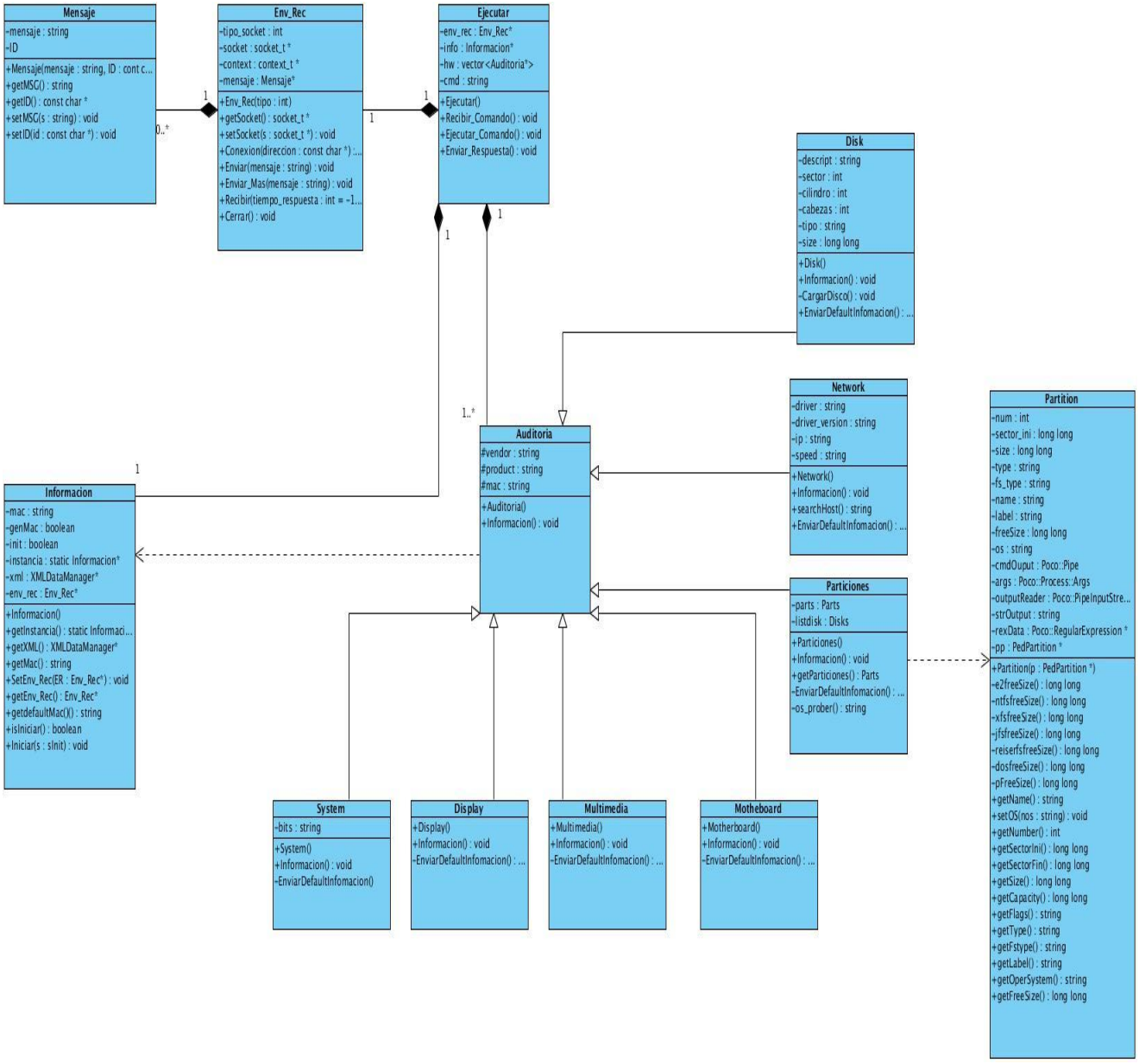

Figura 10. Diagrama de clases del componente de hardware

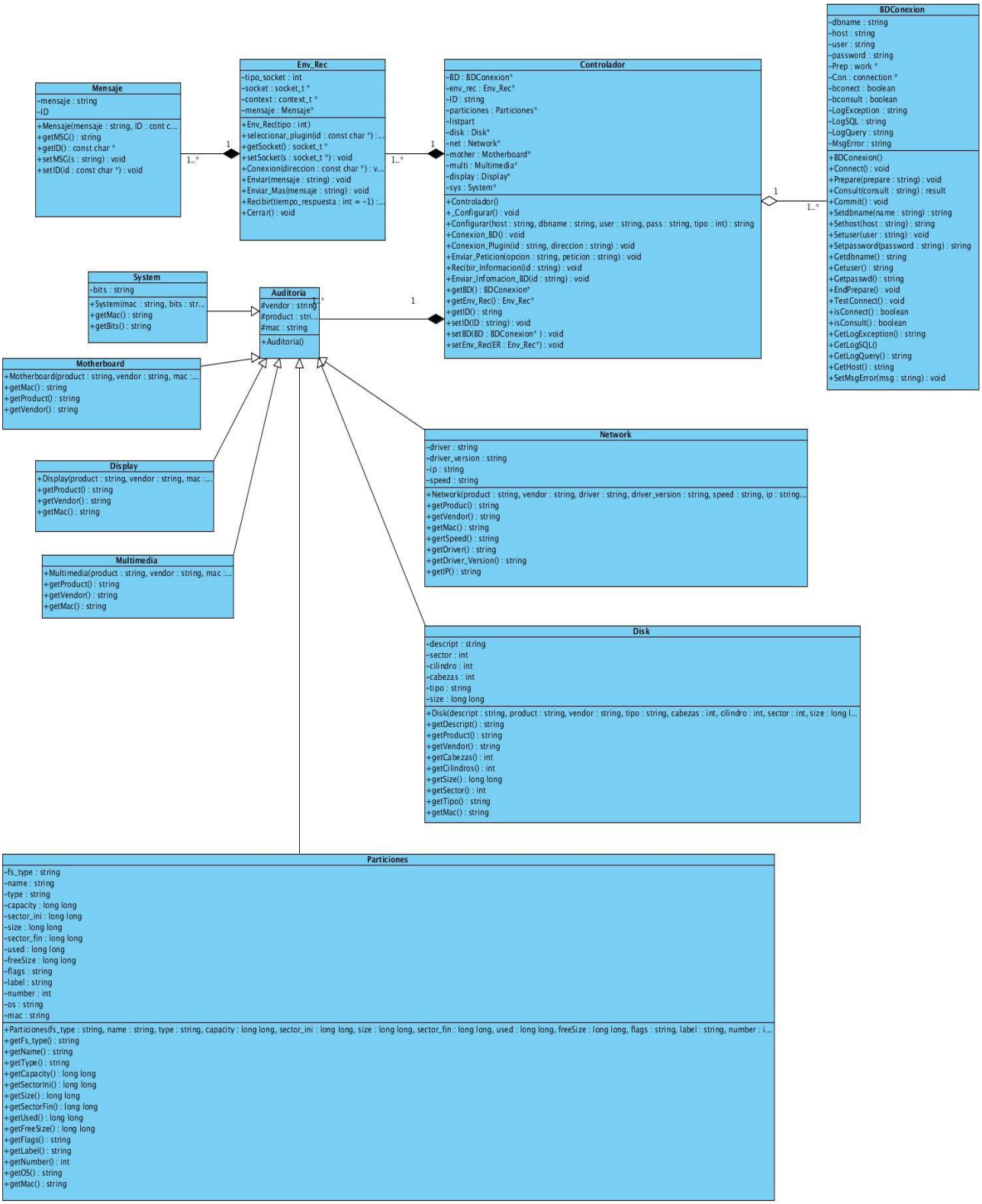

Figura 11. Diagrama de clases Controlador.

Anexo 3. Diagramas de secuencia

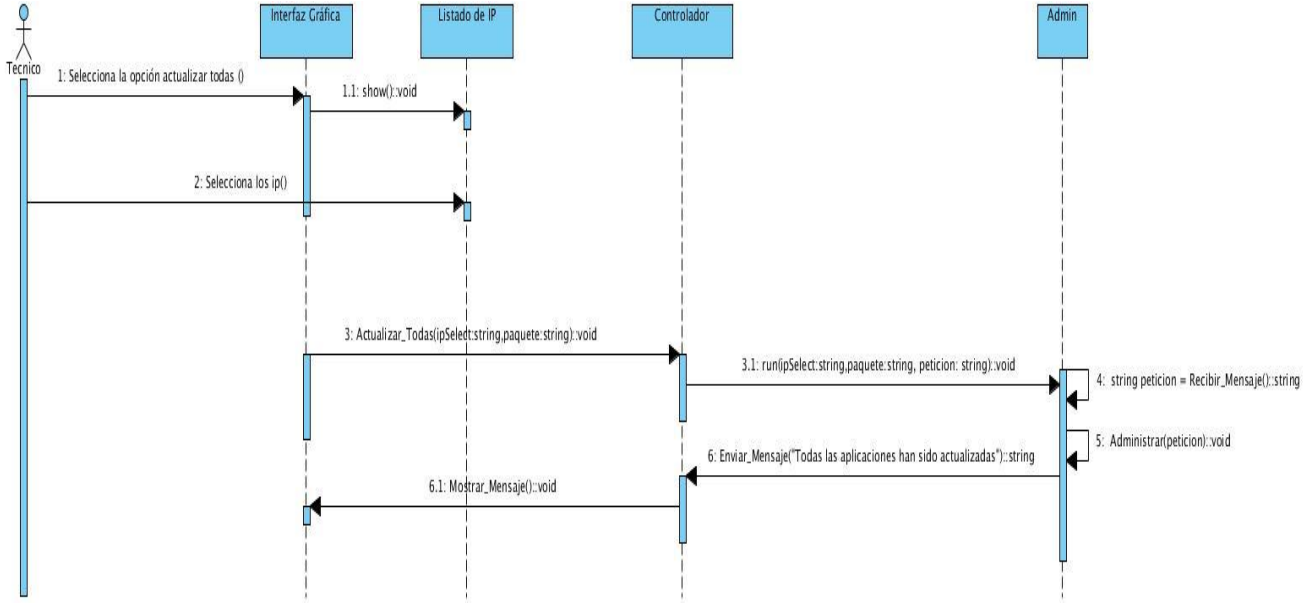

Figura 12. Diagrama de secuencia del caso de uso Actualizar todas las aplicaciones

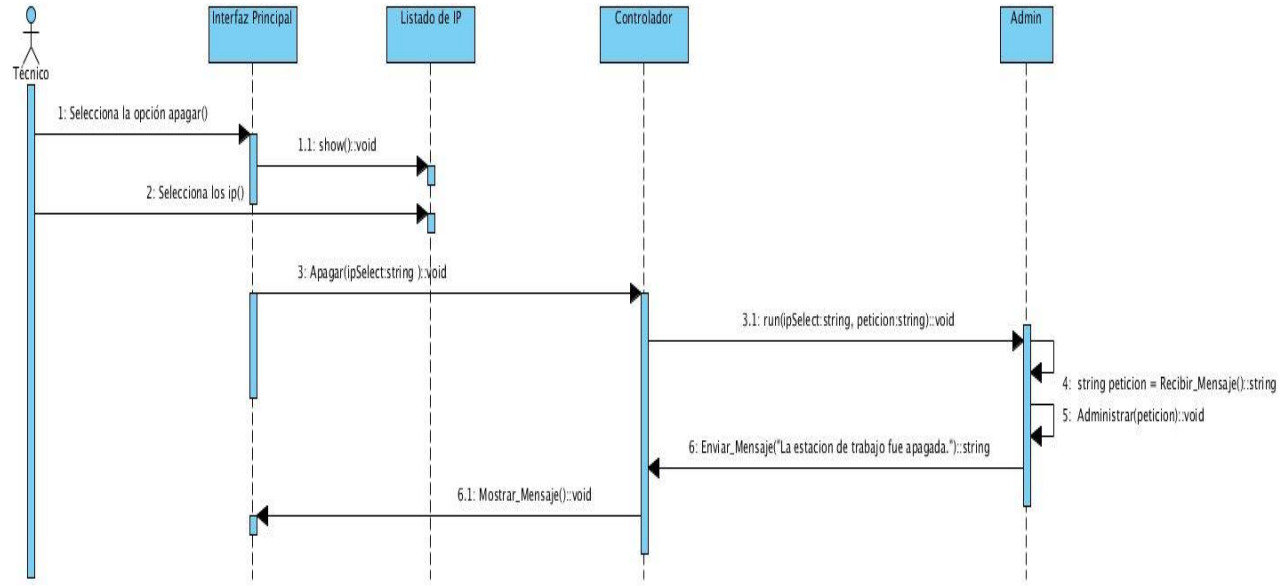

Figura 13. Diagrama de secuencia del caso de uso apagar sistema

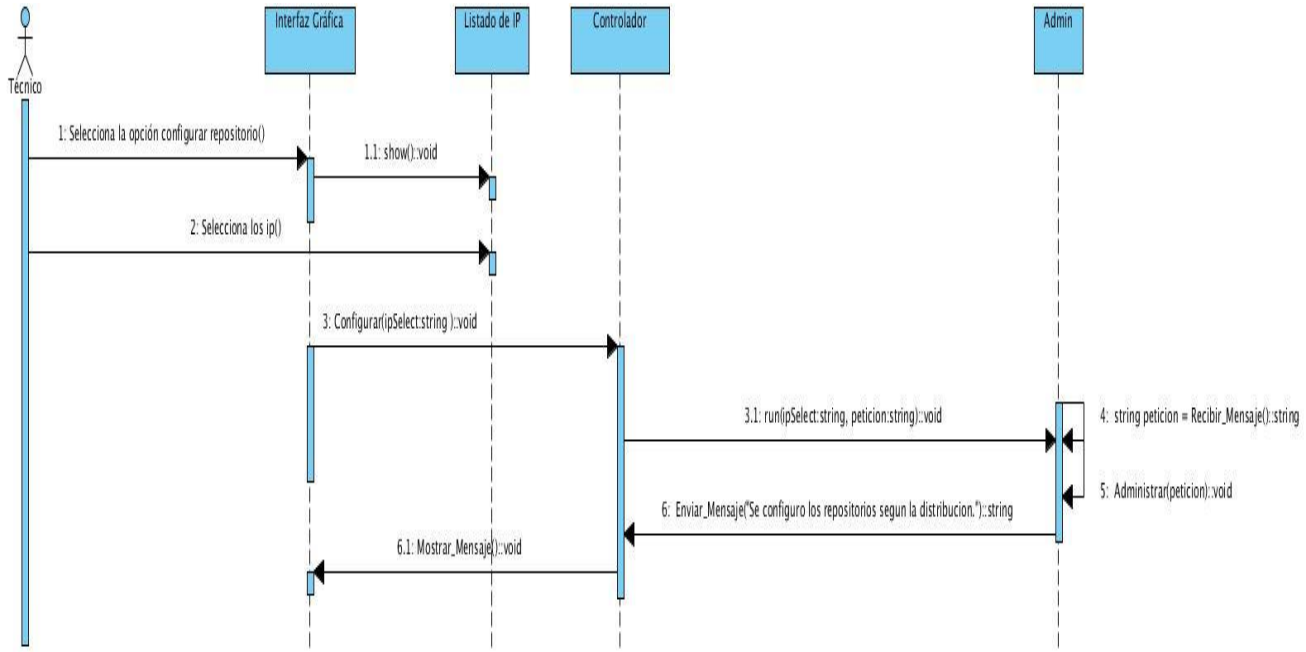

Figura 14. Diagrama de secuencia del caso de uso configurar repositorio

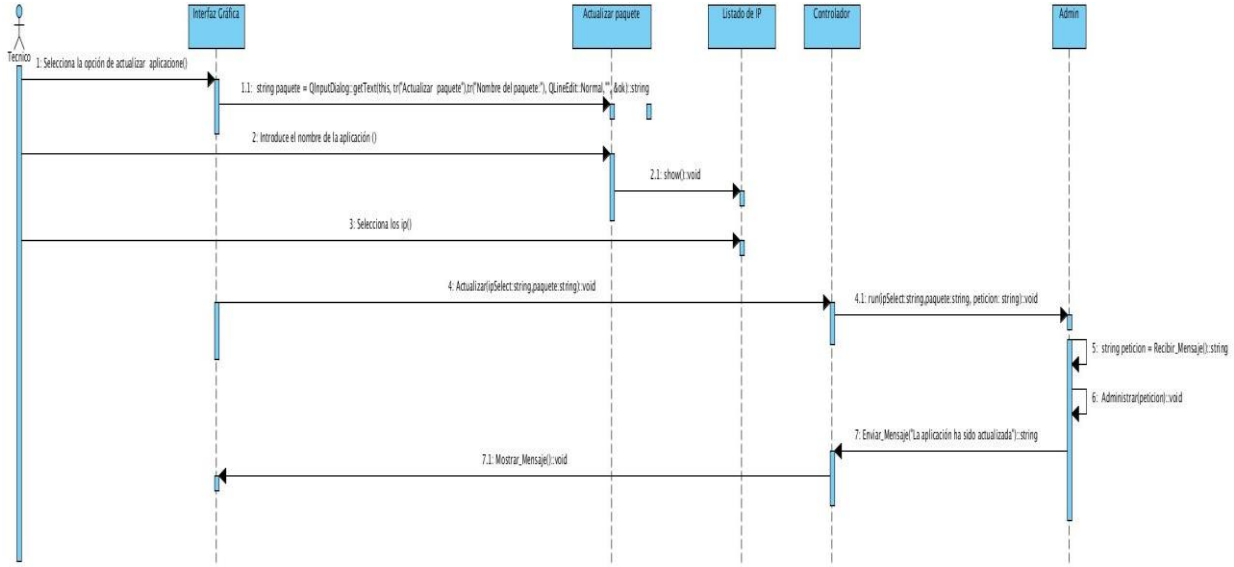

Figura 15. Diagrama de secuencia del caso de uso Actualizar aplicación

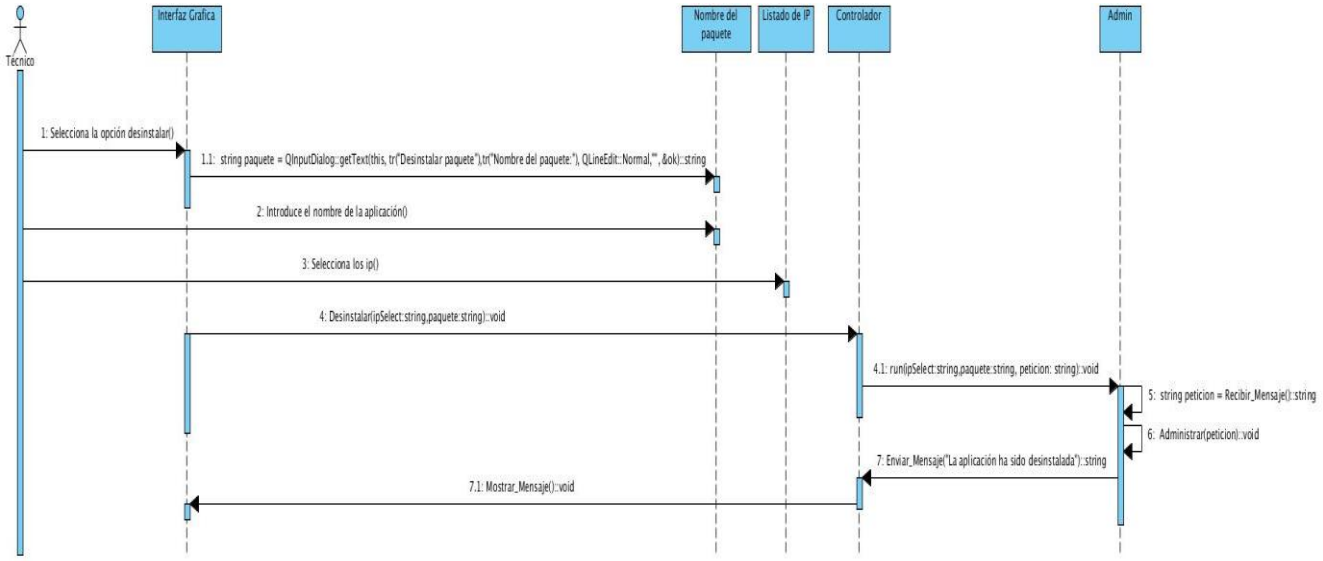

Figura 16. Diagrama de secuencia del caso de uso desinstalar aplicación

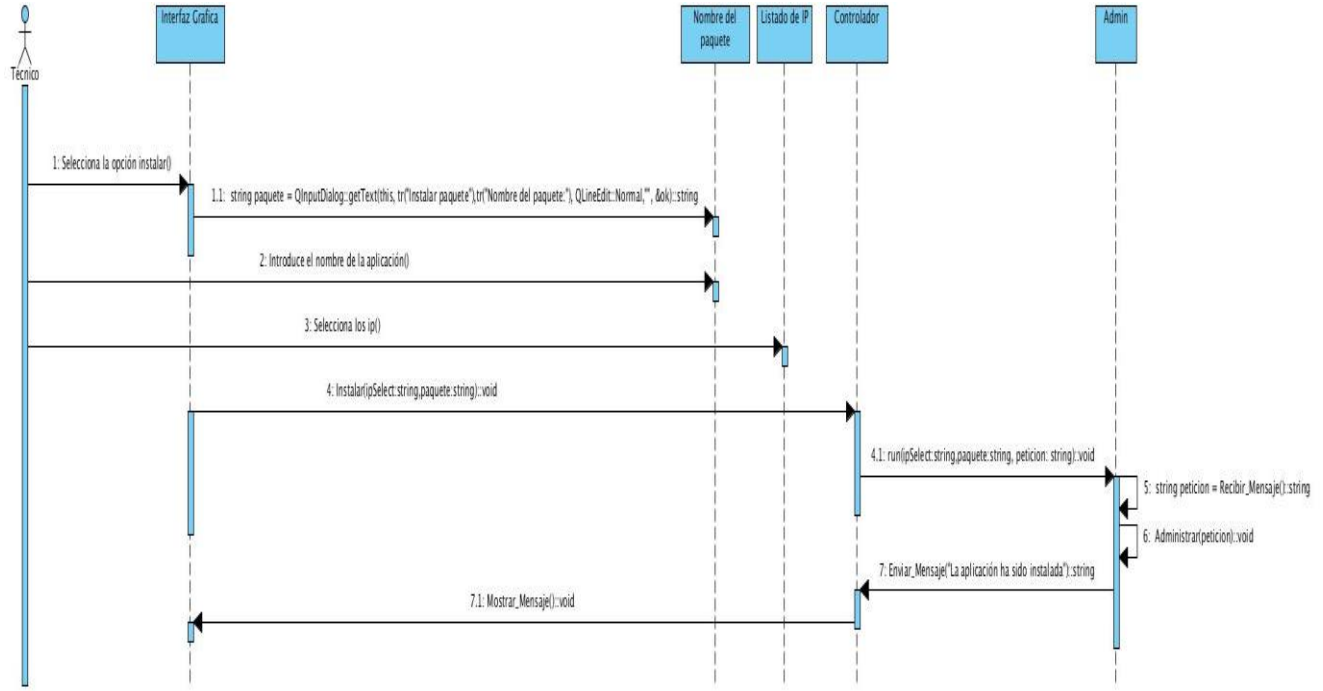

Figura 17. Diagrama de secuencia del caso de uso instalar aplicación

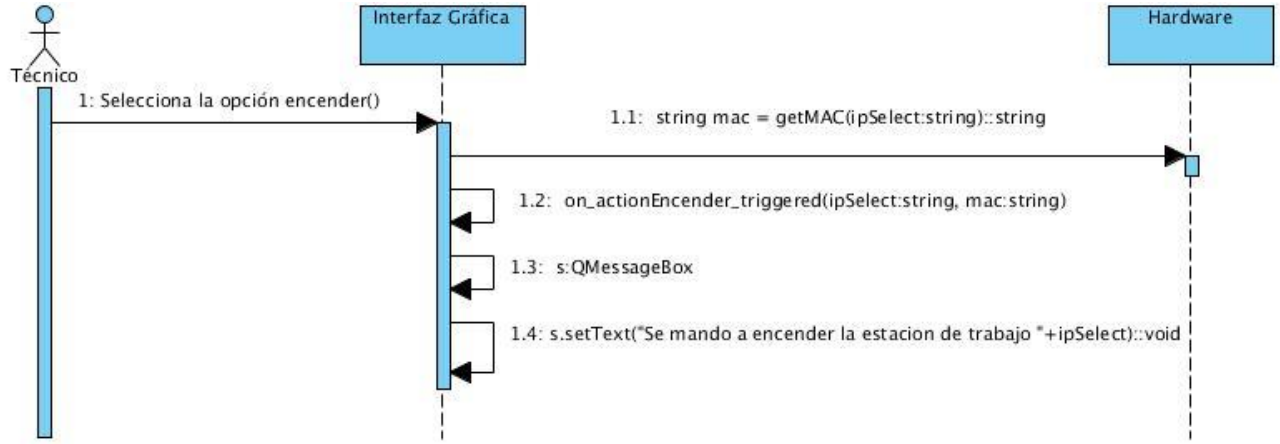

Figura 18. Diagrama de secuencia del caso de uso encender sistema

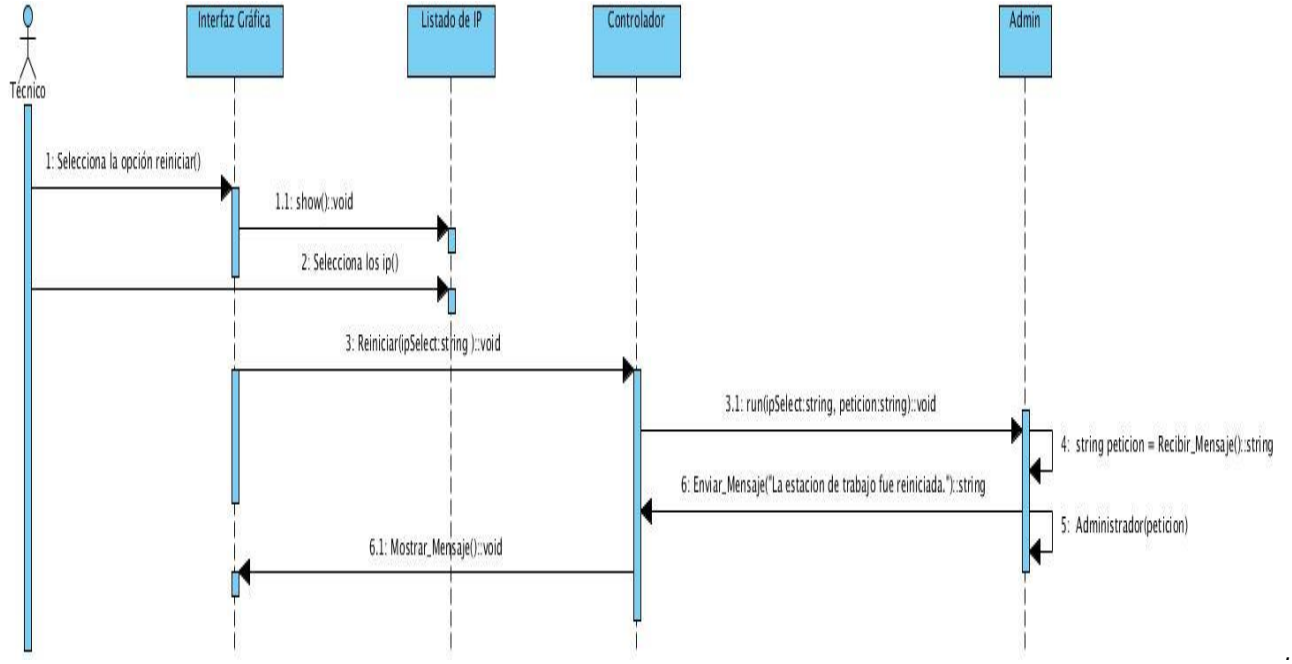

Figura 19. Diagrama de secuencia del caso de uso reiniciar sistema

Anexo 4. Diagramas de componentes

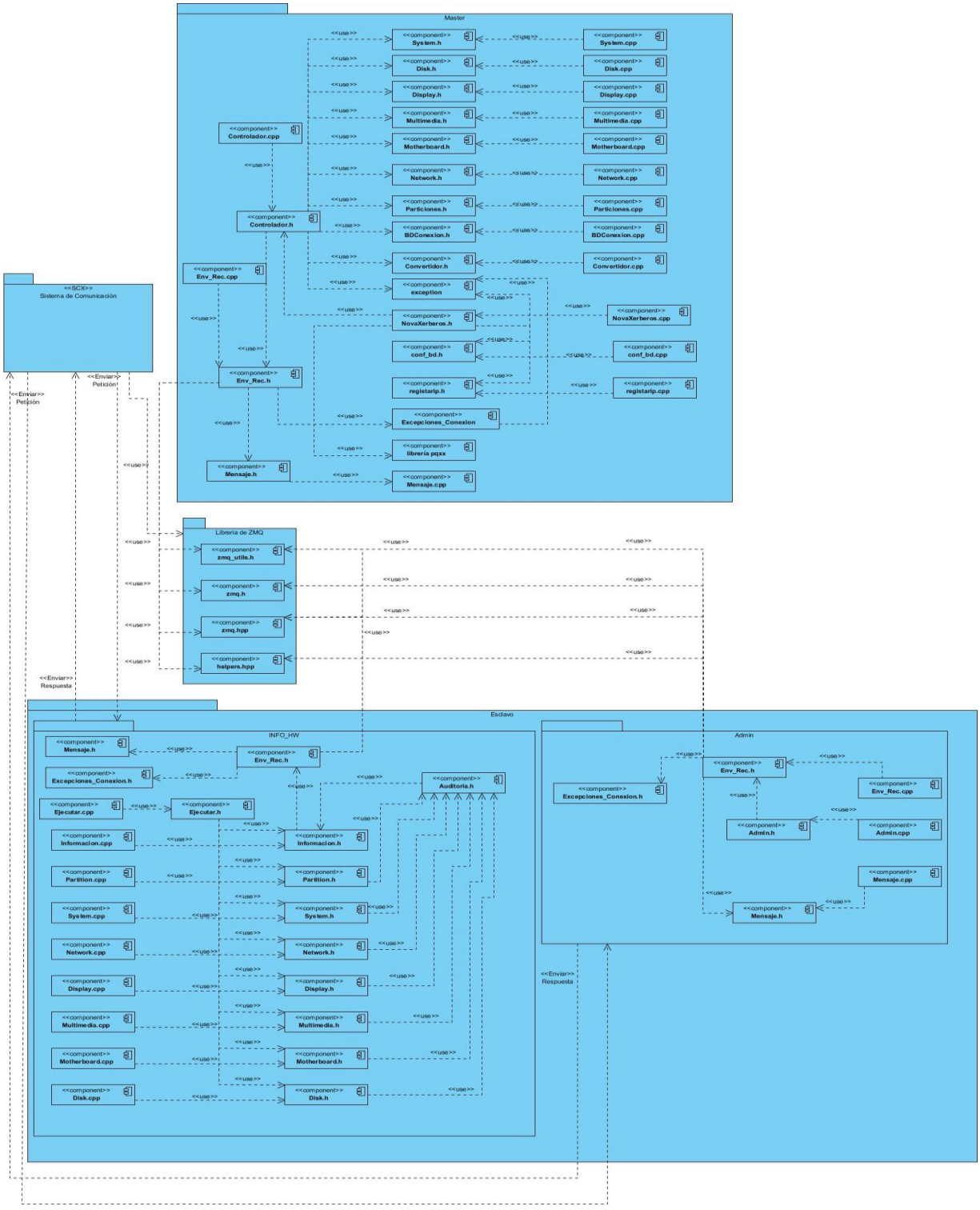

Figura 20. Diagrama de componentes del sistema.

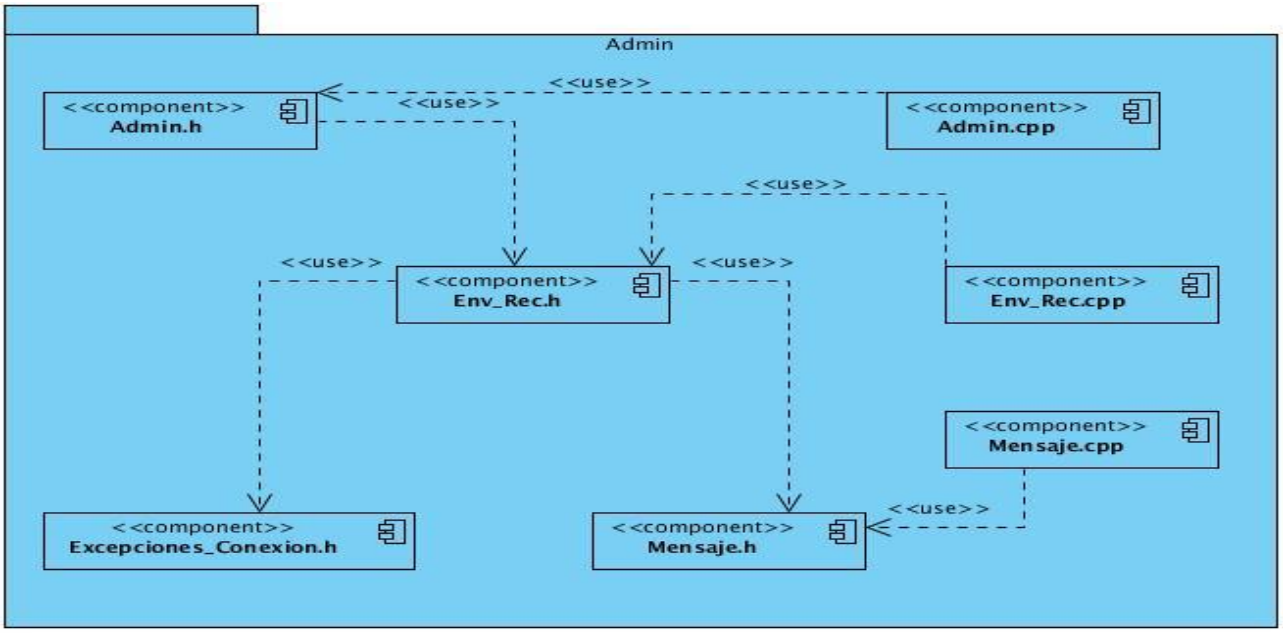

Figura 21. Diagrama del componente Admin

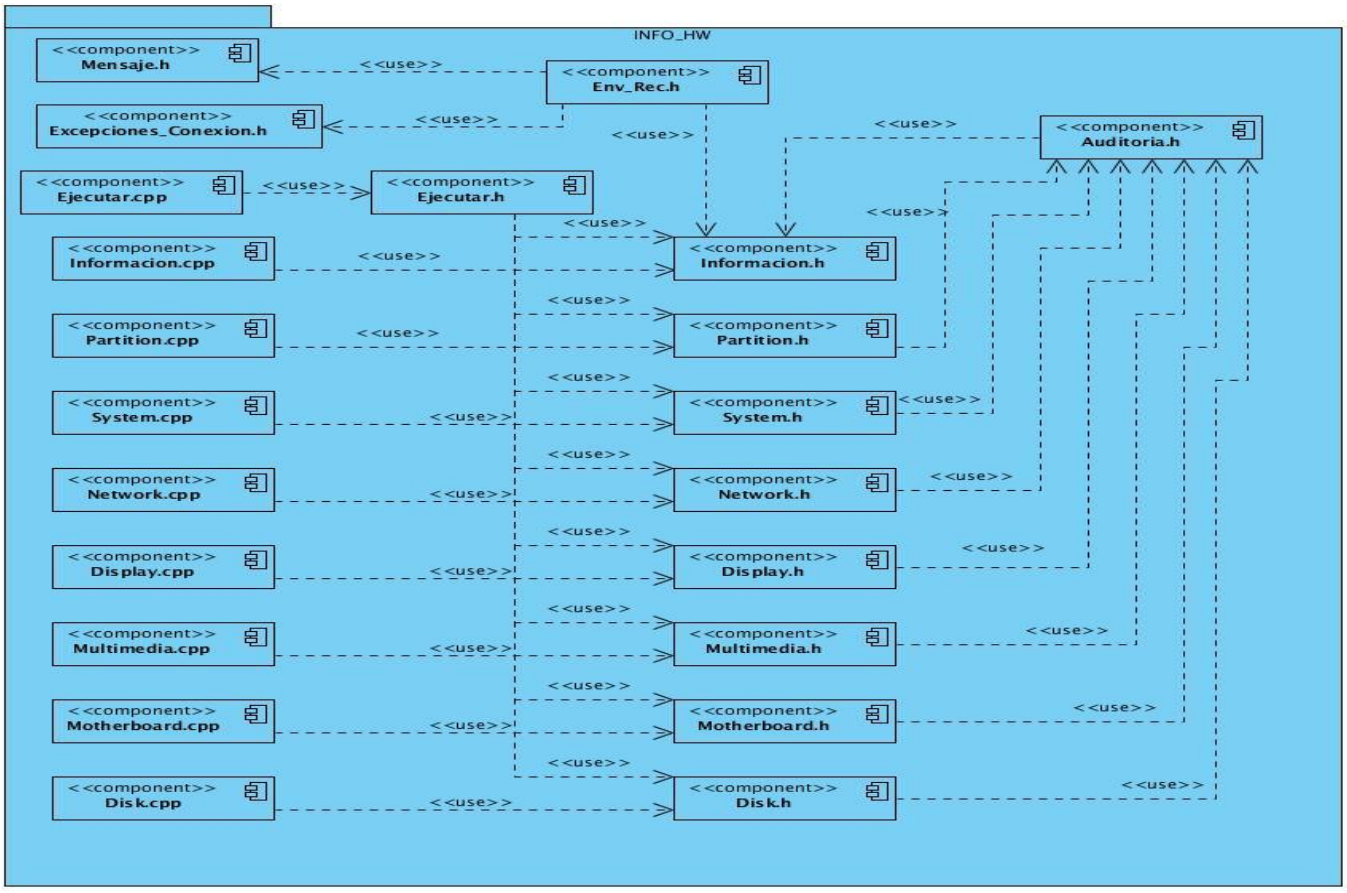

Figura 22. Diagrama del componente Info_HW

Anexo 5. Pruebas de caja negra.

Pre-condiciones: La base de datos debe existir para poder conectarse a ella.

Sección	Escenarios para esta sección	Descripción del escenario
SC: 1 Configurar Base de Dato.	EC: 1.1. Configurar Base de Datos externa.	El usuario llena los campos y se conecta a la base de datos.
	EC: 1.2. Configurar Base de Datos local.	El usuario selecciona la opción localhost y se conecta a la base de datos.
	EC: 1.3. Introduce incorrectamente los datos, para configurar la base datos externa.	El sistema emite el mensaje: "Error en la conexión a la base de datos, inténtelo nuevamente" cuando se introducen datos erróneos y da la posibilidad de volver a entrar los datos.
	EC: 1.4. Campos vacíos	El sistema emite el mensaje: "Existen campos vacío, por favor entre de nuevo los datos" cuando hay campos vacío y da la posibilidad de volver a entrar los datos.
	EC: 1.5. Configurar Base de Datos local cancelar.	Se cancela la operación y se cierra la ventana para realizar otras acciones.
	EC: 1.6 Configurar Base de Datos cerrar.	Se cancela la operación y se cierra la ventana para realizar otras acciones.

Tabla 12. Descripción de las variables del CU Configurar Base de Datos.

No.	Nombre del campo	Clasificación	Valores nulos	Descripción
1	Dirección	Campo de texto.	No	Permite solamente letras y números con puntos para IP versión 4.
2	Nombre de la Base de Datos	Campo de texto.	No	Permite solamente letras.
3	Usuario	Campo de texto.	No	Permite solamente letras
4	Contraseña	Campo de texto.	No	Permite letras y números.

Tabla 13. Descripción de las variables del CU Configurar Base de Datos.

Escenarios	Valor 1	Valor 2	Valor 3	Valor 4	Respuesta	Resultado	Flujo central
EC: 1.1Configurar Base de Dato externa.	(10 .53 .4. 29)	Xerberos	postgres	postgres	El sistema se conecta a la base de datos.	Satisfactorio.	El usuario accede al sistema para Configurar la base de datos. El sistema muestra un formulario para introducir los datos. El usuario introduce los datos. 2.1. Se conecta a la base de datos.
EC: 1.2 Configurar Base de Dato local.	vacío	vacío	vacío	vacío	El sistema se conecta a la base de datos	Satisfactorio.	El usuario accede al sistema para Configurar la base de datos. El sistema muestra un formulario y botón con el nombre de “loclahost” . El usuario selecciona ese botón. 2.1. Se conecta a la base de datos local.

EC: 1.3 Introduce incorrectamente los datos, para configurar la base datos externa.	10.p.p.10	Xerberos	postgres	postgres	El sistema emite el mensaje: “Error en la conexión a la base de datos, inténtelo nuevamente”	Satisfactorio.	El usuario accede al sistema para Configurar la base de datos. El sistema muestra un formulario para introducir los datos. 2 El usuario introduce los datos erróneos. 2.1 Muestra un mensaje de error.
EC: 1.4 Campos vacíos	vacío	vacío	vacío	vacío	El sistema emite el mensaje: “Existen campos vacío, por favor entre de nuevo los datos”	Satisfactorio.	El usuario accede al sistema para Configurar la base de datos. El sistema muestra un formulario para introducir los datos. 2 El usuario deja los campos vacíos. 2.1 Muestra un mensaje de error.
EC: 1.5 Configurar Base de Dato local cancelar.	(10 .53 .4. 29)	Xerberos	postgres	postgres	El sistema cancela la operación y cierra la ventana para realizar otras operaciones.	Satisfactorio.	
EC: 1.6 Configurar Base de Dato cerrar.	Vacío	Vacío	Vacío	Vacío	El sistema cancela la operación y cierra la ventana para realizar otras operaciones	Satisfactorio.	

Tabla 14. Matriz de datos del CU Configurar Base de Datos.

Pre-condiciones: el IP a registrar debe de estar disponible físicamente en la red que se va a administrar.

Sección	Escenarios para esta sección	Descripción del escenario
SC: 1 Registrar IP.	EC: 1.1 Registrar IP correctamente.	El usuario llena los campos y registra la dirección de IP. El mismo se muestra en la interfaz principal y lo almacena en un archivo de texto.
	EC: 1.2 Introduce incorrectamente los datos de la dirección IP.	El sistema emite el mensaje: "Esa dirección IP no es correcta" cuando se introducen datos erróneos y da la posibilidad de volver a entrar los datos.
	EC: 1.3 Registrar IP cancelar.	Se cancela la operación y se cierra la ventana para realizar otras acciones.
	EC: 1.4 Registrar IP cerrar.	Se cancela la operación y se cierra la ventana para realizar otras acciones.

Tabla 15. Escenarios del CU Registrar IP

No.	Nombre del campo	Clasificación	Valores nulos	Descripción
1	Dirección IP.	Campo de texto.	No	Permite solamente números y puntos para IP versión 4.

Tabla 16. Descripción de las variables del CU Registrar IP.

Escenarios	Valor 1	Respuesta	Resultado	Flujo central
EC: 1.1 Registrar IP correctamente.	(10.18.5.20)	El sistema muestra en la interfaz principal el IP y lo almacena en un archivo de texto.	Satisfactorio.	1. El usuario accede al sistema para registrar un IP. 1.1. El sistema muestra un formulario para introducir los datos del IP. 2. El usuario introduce el IP. 2.1. Se almacene la información en un archivo de texto y se visualiza en la interfaz principal.
EC: 1.2 Introduce incorrectamente los datos de la dirección IP.	(10.18.p.p)	El sistema emite el mensaje: "Esa dirección IP no es correcta".	Satisfactorio.	

EC: 1.3 Registrar IP cancelar.	(10.18.5.20)	Se cancela la operación y se cierra la ventana para realizar otras acciones.	Satisfactorio.	
EC: 1.4 Registrar IP cerrar.	(Vacío)	Se cancela la operación y se cierra la ventana para realizar otras acciones.	Satisfactorio.	

Tabla 17. Matriz de datos del CU Registrar IP.