Universidad de las Ciencias Informáticas

Facultad 3

“VIRTUALNET. Herramienta de Simulación de Redes de Computadoras”

Trabajo de Diploma para Optar por el Título de

Ingeniero en Ciencias Informáticas

Autor: Yadira Annabel Frómeta LLarena.

Tutor: Ing. Alberto Limia Navarro

Cotutor: Ing. Yohandri Ril Gil.

Ciudad de la Habana

Junio del 2009

Lo que tenemos que aprender lo aprendemos haciéndolo.

Aristóteles

Declaratoria 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 firmo la presente a los _____ días del mes de _____ del año _____.

Yadira Annabel Frómeta LLarena

_____

Firma del Autor

Alberto Limia Navarro

_____

Firma del Tutor

Datos de Contacto

Nombre y Apellidos: Alberto Limia Navarro.

Edad: 26

Ciudadanía: Cubano

Institución: Universidad de las Ciencias Informáticas (UCI).

Título: Ingeniero en Ciencias Informáticas

Categoría Docente: Instructor Recién Graduado

E – mail: alimia@uci.cu

Fundador de la UCI y graduado en el 2007 con Título de Oro, ha impartido las asignaturas de Teleinformática I y Seguridad Informática. Pertenece al Centro de Tecnologías de Almacenamiento y Análisis de Datos (CENTALAD) y ejerce como especialista en la Línea de Soluciones Integrales y BI de dicho centro.

Nombre y Apellidos: Yohandri Ril Gil.

Edad: 28

Ciudadanía: Cubano

Institución: Universidad de las Ciencias Informáticas (UCI).

Título: Ingeniero en Telecomunicaciones y Electrónica.

Categoría Docente: Profesor Asistente

E – mail: rilltt@uci.cu

Ingeniero en Telecomunicaciones y Electrónica, graduado en el Instituto Superior Politécnico José Antonio Echeverría (ISPJAE) en el 2004. Actualmente trabaja como Asesor Técnico docente del departamento de Sistemas Digitales y Aseguramiento básico de Programación. Posee la categoría docente de Profesor Asistente.

Agradecimientos

A mis padres, Margarita y Luis, por ser las personas que me dieron vida, por quererme, educarme, enseñarme, apoyarme y darme las fuerzas para llegar a ser alguien en la vida, por ser las personas que más quiero y estar siempre al pendiente de mí, por ser los padres más maravillosos del mundo.

A Yailen, mi hermanita a la que tanto quiero y a la que le deseo el mejor de los futuros, por los regaños y consejos que siempre me ha dado, por poder contar con ella para cualquier cosa, por ser cada día tan buena y aplicada, por ser mi amiga y mi confidente.

A Roxana, mi primita que tanto adoro por ser tan buena conmigo, por querer siempre seguir mi ejemplo y compartir conmigo los buenos y malos momentos de mi vida.

A Ñaña, mi tía que es como una madre para mí, por velar por mí desde pequeña, apoyarme y estar ahí en los buenos y malos momentos, por sus consejos y ser tan incondicional conmigo. Te quiero con todo mi corazón.

A Nancy y a Karla, por formar parte también de mi familia, por ser tan leales y buenas con las personas que más quiero en la vida, por ser desinteresadas y quererme tanto. A las dos las quiero y les deseo lo mejor de la vida.

A mi novio Osvaldo, por ser tan paciente conmigo, por apoyarme y sensibilizarse con todo lo que me concierne, al que le agradezco infinitamente esta investigación porque sin el hubiese sido más difícil, por su confianza y admiración por mí.

A mis tías Nena y Deysi, que siempre llevo en mi corazón y pensamiento aunque no estén presentes físicamente.

A toda mi familia, los seres a los que le estoy eternamente agradecida.

A mis tutores Limia y Ril por haberme ayudado y por el tiempo que me dedicaron sin importar el día ni el horario.

A mis nuevos amigos de la universidad, que no quiero que se sientan dolidos si me faltara alguno por mencionar, por eso el agradecimiento va para todos, por hacer mi estancia en la escuela mucho más agradable, amena y alegre.

A la revolución y a nuestro comandante en jefe Fidel Castro Ruz por haber creado y fundado la Universidad de las Ciencias Informáticas.

A la Universidad por darme la posibilidad de ser lo que hoy soy.

A todas las personas que de una forma u otra me han ayudado a terminar mis estudios.

A todos Gracias.

Yadira

Dedicatoria

A mis padres y a mi hermana por quererme tanto y ser el motivo de mi felicidad y mi inspiración en este trabajo.

Resumen

En las redes de computadoras, el direccionamiento IP se refiere a la asignación de un único identificador a los dispositivos que están enlazados a ella. Este identificador, llamado usualmente dirección IP, permite que otras máquinas envíen información a otras subredes destinos. Esta comunicación entre subredes diferentes se logra mediante el enrutamiento estático y el enrutamiento dinámico utilizando para ellos protocolos de enrutamiento. El enrutamiento permite el tráfico de la red buscando el camino más óptimo a una subred destino pasando por varias redes. Para esto las rutas deben estar correctamente configuradas en las tablas de enrutamiento de cada enrutador de la red.

El principal objetivo de esta investigación es la creación de un simulador que represente los procesos de direccionamiento IP y de enrutamiento a partir del diseño de una red de área amplia (WAN), así como el diseño lógico de redes de área local (LAN). Para el desarrollo del mismo se realizó un estudio de las principales metodologías y tecnologías actuales en el desarrollo de este tipo de herramienta, a partir del cual se seleccionó la plataforma Java con su IDE de desarrollo NetBeans 6.0, RUP como metodología de desarrollo, Visual Paradigm como herramienta CASE y UML como lenguaje de modelado.

Esta herramienta será utilizada con fines didácticos en la impartición de la asignatura de Teleinformática II correspondiente a la disciplina de Sistemas Digitales de la Universidad de las Ciencias Informáticas (UCI).

PALABRAS CLAVE

Simulador, Direccionar, Enrutar, Diseñar Redes, LAN, WAN, Topologías.

Introducción

Una de las misiones de las universidades es lograr una buena formación en sus estudiantes. En esta formación, la práctica de los conocimientos teóricos adquiridos juega un papel imprescindible. Es a través de ella que los estudiantes fijan los conceptos teóricos y adquieren los conocimientos fundamentales en el área que están estudiando.

Pero no todos los centros cuentan con suficientes recursos informáticos para que los estudiantes trabajen con comodidad, lo que provoca que estos se asocien en grupos para ejercitar la teoría con la práctica. Si bien este trabajo en grupo es el adecuado en determinadas materias, en otras un trabajo individual resulta indispensable para un buen aprovechamiento.

Debido a esto, el auge de las Tecnologías de la Información y las Comunicaciones (TIC) juegan un papel significativo. En la actualidad se encuentran, en muchos centros educativos, laboratorios de computadoras, donde los estudiantes pueden asistir para aplicar sus conocimientos en la práctica. Pero hay otros centros donde no todos los laboratorios están destinados directamente a estas actividades, lo que provoca un déficit de recursos materiales destinados a la docencia y un menor aprovechamiento de los estudiantes. Sin embargo, además de este uso de los laboratorios, se podrían utilizar para que los estudiantes trabajen con herramientas de simulación. Así, el problema de disponer de máquinas y locales, muchos de ellos costosos, se transformaría en el problema de desarrollar software adecuado que simulen los procesos prácticos posibles a automatizar y contar con suficientes recursos informáticos para el aprovechamiento individual de cada cual.

Los simuladores, dentro del mundo de las redes de la información, son programas computacionales diseñados para crear ambientes de comunicaciones con características muy similares a la realidad. [1]

Estos pueden ser un excelente recurso didáctico. Se han realizado diversas experiencias sobre el uso de los mismos y su influencia en el aprendizaje de los estudiantes. Casi todas se basan en un ámbito de conocimiento muy concreto. Ellos no solo suponen una mejora en el proceso educativo debido a que se adaptan a las necesidades prácticas de los estudiantes, sino también están enfocados eficazmente en su superación.

Los simuladores brindan grandes ventajas para los profesores y alumnos entre las que se pueden indicar las siguientes: Ofrecen una forma más accesible a los mismos de trabajar en la práctica con procesos y procedimientos que hasta entonces solo conocían la teoría, los involucra en su aprendizaje, porque es él el que tendrá que manejar el simulador, observar los resultados y actuar en consecuencia, es una herramienta motivadora, coloca a las personas ante situaciones próximas a la realidad y se pueden trabajar situaciones que en la realidad son difíciles de asimilar.

En la Universidad de las Ciencias Informáticas (UCI) existen varias asignaturas que por las características técnicas de la carrera necesitan de la actividad práctica para poder verificar el nivel en que los estudiantes han adquirido el fundamento teórico tratado en los temas de estas asignaturas. Muchas de éstas, de naturaleza teórico-práctico y de laboratorios, pertenecen al Departamento de Sistemas Digitales (DSD) de la universidad; que es el encargado de planificar las actividades docentes-educativas de las mismas. Ejemplo de estas es la asignatura de Teleinformática en la cual se imparte el contenido de enrutamiento, direccionamiento y diseño de redes de computadoras, temas que necesitan una comprobación práctica, debido a que la misma pretende vincular al estudiante con el desarrollo del país en el campo de las redes de computadoras, en especial las de área local.

Una de las causas por la que le es imposible a la universidad realizar estas actividades prácticas, que utiliza como medios de estudios las computadoras reales, es que; dada las características de la universidad de vincular la producción con la docencia le es imposible destinar un laboratorio a estas actividades debido a que existen diez facultades en la sede central y tres facultades regionales que cuentan con varios grupos por año y esto trae como consecuencias que, como mínimo, se necesitaría equipar un laboratorio por cada asignatura y la universidad ni el país cuentan con el presupuesto para hacer esta inversión, teniendo en cuenta que Cuba es un país subdesarrollado y bloqueado por más de cincuenta años; por lo que se han adquirido varias herramientas que simulan las prácticas experimentales e implementan algunas de las funcionalidades dentro del diseño de una red informática. Ejemplo de ello es el ROUTERSIM el cual permite simular conexiones de redes de mediana complejidad. Se basa en la configuración global de los enrutadores de Cisco, de sus interfaces y en la configuración de las rutas estáticas y dinámicas que utilizan los enrutadores para el envió de los paquetes por la red. Además permite configurar conmutadores y computadoras individuales.

En su versión de evaluación no todos los comandos de los enrutadores están disponibles y solo un número limitado de laboratorios se pueden llevar a cabo con dicha versión. Todas estas restricciones desaparecen en la versión comercial, que ofrece todos los comandos de los enrutadores y la posibilidad de experimental con el entorno libremente. Sin embargo aunque es uno de los mejores, resulta costoso el precio de su licencia individual.

Otro ejemplo es el NETSIM el cual es uno de los simuladores de redes más completos, fue desarrollado con propósitos de aprendizaje, como complemento a los cursos que brinda la empresa Cisco para sus diferentes certificaciones en manejo de dispositivos de redes de datos. Aunque tiene diversas funcionalidades y sus características permiten que se analicen redes de datos casi como si se estuviera trabajando en la realidad, no es un simulador gratuito.

Estos simuladores son muy buenos y presentan grandes ventajas en el campo de las redes de computadoras, pero ninguno de ellos integra de manera factible los contenidos de direccionamiento, enrutamiento y diseño de redes que se imparten en la asignatura de teleinformática porque no fueron implementados para este fin, debido a que en ocasiones los estudiantes de la UCI necesitan realizar el direccionamiento y enrutamiento de manera rápida y factible, sin necesidad de configurar los dispositivos y componentes de la red a través de una serie de comandos que son empleados para estas configuraciones, de esta manera se agiliza el trabajo en los laboratorios y ellos se sienten más motivados para estudiar y comprobar los ejercicios prácticos que se les orientan en las clases prácticas y laboratorios. También necesitan conocer y ver el procedimiento paso a paso de cómo funciona el direccionamiento y el enrutamiento con el objetivo de que asimilen los conocimientos teóricos adquiridos.

Dada la **situación** anteriormente expuesta, se plantea como **problema:** ¿Cómo contribuir al autoaprendizaje de los estudiantes mediante la ejercitación práctica de los contenidos de direccionamiento, enrutamiento y diseño de redes de computadoras impartidos en la asignatura de teleinformática?

A raíz de esto, el trabajo toma como **objeto de investigación:** los simuladores como software educativo. Y dentro de esta extensa rama se propone como **campo de acción:** los simuladores de redes de computadoras.

El **objetivo** de esta investigación es desarrollar una herramienta de simulación de redes de computadoras que integre los contenidos de direccionamiento, enrutamiento y diseño de redes impartidos en la asignatura de teleinformática.

Se plantea como **Hipótesis** del presente trabajo que, el desarrollo de una herramienta de simulación de redes de computadoras que integre los contenidos de direccionamiento, enrutamiento y diseño de redes impartidos en la asignatura de teleinformática contribuirá al autoaprendizaje de los estudiantes mediante la ejercitación práctica de dichos contenidos.

Para dar cumplimiento al objetivo planteado en este trabajo se necesita un grupo de **tareas investigativas** a las cuáles se harán referencia:

1. Identificar y Seleccionar los contenidos de la asignatura de teleinformática relacionados con redes de computadoras posibles a automatizar en una herramienta de simulación de redes.
2. Identificar y Caracterizar los componentes y sus relaciones en una red de computadoras.
3. Caracterizar las herramientas de simulación de redes existentes en la actualidad.
4. Desarrollar una herramienta de simulación de redes de computadoras que integre los contenidos de: direccionamiento, enrutamiento y diseño de redes impartidos en la asignatura de teleinformática para que los estudiantes los ejerciten de forma práctica.

Además para todo el proceso de investigación y elaboración de este trabajo se tomará en cuenta la utilización de varios **métodos científicos de investigación** como:

Histórico – Lógico: método teórico mediante el cual se constatará como ha sido la trayectoria histórica real, la evolución y desarrollo de los aspectos principales de la investigación como: el direccionamiento, el enrutamiento, el diseño de redes y la simulación de redes de computadoras.

Analítico – Sintético: este método teórico será utilizado en la investigación para buscar la esencia de lo que existe en el mundo acerca de los algoritmos de ruteo estático y dinámico y de los métodos y algoritmos de direccionamiento de direcciones IP.

Modelación: método teórico que se utilizará para representar parte del conocimiento que vayamos acumulando, así como los diferentes diagramas que apoyarán el proceso de elaboración de la herramienta, como los diagramas de casos de uso, diagramas de clases etc.

Entrevista: método empírico para obtener información acerca de las necesidades que tiene el departamento de sistemas digitales de la UCI y los contenidos que requieren automatización, para saber que se va a hacer y sobre que se va a investigar.

La entrevista puede ser estructurada o no estructurada.

Para la realización de esta investigación se utilizó la entrevista no estructurada que es más abierta que la estructurada, prevé el tema pero no lleva un cuestionario rígido y puede variar de una persona a otra, es más flexible. Se aplica a especialistas en el tema, es una forma de obtener criterios de expertos.

Posibles Resultados:

Una herramienta de simulación de redes de computadoras que integre los contenidos de direccionamiento, enrutamiento y diseño de redes para la asignatura de Teleinformática.

El documento está estructurado por: un resumen, introducción, cuatro capítulos que constituyen el cuerpo fundamental de la tesis, conclusiones generales y bibliografía. Para tener un conocimiento de manera general de lo que se abordará se describen a continuación una síntesis de cada capítulo.

Capitulo 1: Fundamentación Teórica.

En este capítulo se presentan las bases teóricas fundamentales de las diferentes topologías de redes de computadoras que existen, de los principales dispositivos de interconexión activos y de los elementos que forman parte del diseño de una red, así como de la estructura y formato de una dirección IPv4. También se analizaron las diferentes metodologías y herramientas de desarrollo de software más utilizadas en la actualidad para el desarrollo de la herramienta.

Capítulo 2: Características del Sistema.

En esta sección se presenta el funcionamiento del negocio. En este caso se desarrolla un modelo de dominio donde se analizan los conceptos y las entidades presentes en el negocio.

Además se exponen los elementos imprescindibles para obtener una solución exitosa como son: los requisitos funcionales y no funcionales, así como la descripción de los casos de uso del sistema.

Capítulo 3: Diseño del Sistema.

Se describe el diseño del sistema propuesto a través de la organización de su arquitectura y la realización de los diagramas de clases y de secuencia.

Capítulo 4: Implementación y Prueba.

El desarrollo de este capítulo comienza con los resultados del diseño implementándose el sistema en términos de componentes, así como una revisión final de las especificaciones del diseño y de la codificación mediante la realización de prueba de caja blanca y pruebas de caja negra, garantizando la calidad del software.

Capítulo 1. Fundamentación Teórica

En el desarrollo del presente capítulo se hará un análisis detallado de las principales topologías de redes de computadoras así como de los elementos y componentes que la integran. También se analizarán todos los aspectos que posibilitan el proceso de direccionamiento y enrutamiento de redes para una mejor comprensión de cómo se realizan estos procesos. Además se efectuará un estudio del estado del arte de las herramientas de simulación de redes que existen en la actualidad, de las tecnologías y metodologías de desarrollo de software más utilizadas a nivel internacional, así como de los lenguajes de programación y de los Entornos de Desarrollo Integrado (IDE) que lo soportan.

1.1. Redes de Computadoras

Las redes de computadoras representan un grupo de equipos interconectados mediante uno o varios medios de transmisión con el objetivo de intercambiar información y permitir el empleo en conjunto de estos recursos.

La clasificación y la caracterización de estas se han distinguido por diferentes puntos de vista, siendo el más tradicional el del tamaño que diferencia a las redes de área local con distancias inferiores a los 10 km y a las redes de área amplia con distancias superiores. De igual manera se han distinguido por su velocidad, siendo las redes de área local más rápidas que las redes de área amplia. [2]

Las redes de área local (LAN, Local Área Network) son redes de comunicación que interconectan varios dispositivos y proporcionan un medio para el intercambio de información entre ellos. [3]

Estas redes son pequeñas y rápidas, es decir tienen velocidades altas de transmisión, además tienen capacidad de difusión, mientras una estación transmite todas aquellas que se encuentran conectadas reciben. [2]

Las redes de área amplia (WAN) son un tipo de redes de computadoras capaz de cubrir distancias desde unos 100 hasta unos 1000 km. Su función fundamental está orientada a la interconexión de redes o equipos que se encuentran ubicados a grandes distancias entre sí. En

la construcción de estos tipos de redes, son usadas diversas topologías como son: Anillo, Bus, Estrella, Árbol Jerárquico y Malla. [4]

1.2. Topologías de Redes

Las Topologías de Redes son la forma de situar los nodos de una red y de conectarlos entre sí con el objetivo de encontrar la disposición más económica y efectiva de conectar a todos los usuarios y recursos de la red.

1.2.1. Topología en Anillo

Es una topología de red en la que las estaciones se conectan formando un anillo. Cada estación está conectada a la siguiente y la última está conectada a la primera. Cada estación tiene un receptor y un transmisor que hace la función de repetidor, pasando la señal a la siguiente estación del anillo como se observa en la **Figura 1**.

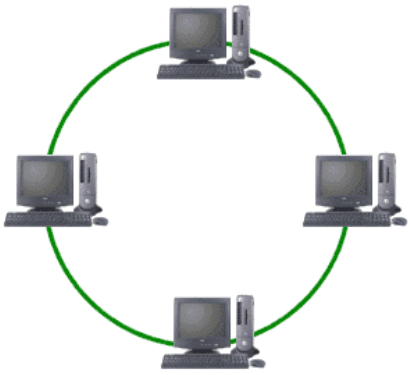


Figura 1 Topología de Red en Anillo

En esta topología si algún nodo de la red se cae, es decir no funciona, la comunicación en todo el anillo se pierde. [5]

1.2.2. Topología en Bus

En esta topología todas las estaciones están conectadas a un único canal de comunicaciones. Las estaciones utilizan este canal para comunicarse con el resto. La topología de bus tiene todos sus nodos conectados directamente a un enlace y no tiene ninguna otra conexión entre nodos. Como se puede observar en la **Figura 2**.

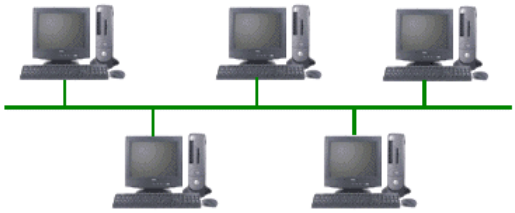


Figura 2 Topología de Red en Bus

Físicamente cada host está conectado a un cable común, por lo que se pueden comunicar directamente, aunque la ruptura del cable hace que los hosts queden desconectados. [5]

1.2.3. Topología en Estrella

En esta topología las estaciones están conectadas directamente al servidor o computadora y todas las comunicaciones se han de hacer necesariamente a través de él. Todas las estaciones están conectadas por separado a un centro de comunicaciones, concentrador o nodo central, pero no están conectadas entre sí. Su punto débil consta en el concentrador debido a que es el que sostiene el funcionamiento de toda la red como se muestra en la **Figura 3**. [5]

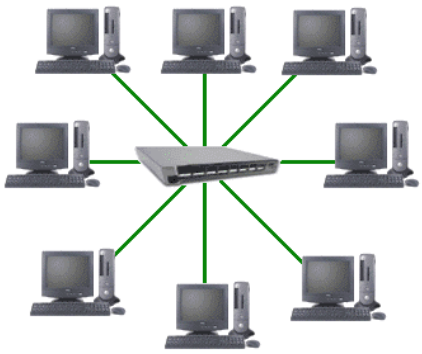


Figura 3 Topología de Red en Estrella

1.2.4. Topología en Malla

En esta topología cada nodo está conectado a uno o más nodos. De esta manera es posible llevar los mensajes de un nodo a otro por diferentes caminos.

Si la red de malla está completamente conectada no puede existir absolutamente ninguna interrupción en las comunicaciones. Cada servidor tiene sus propias conexiones con todos los demás servidores. [5]

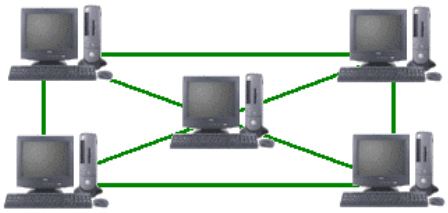


Figura 4 Topología de Red en Malla

1.2.5. Topología en Árbol Jerárquico

En esta topología los nodos están colocados en forma de árbol. Desde una visión topológica, la conexión en árbol es parecida a una serie de redes en estrella interconectadas. Es una variación de la red en bus, la falla de un nodo no implica interrupción en las comunicaciones. Se comparte el mismo canal de comunicaciones.

Cuenta con un cable principal al que hay conectadas redes individuales en bus. Como se observa en la **Figura 5**. [5]

Figura 5 Topología de Red en Árbol Jerárquico

1.3. Componentes y Dispositivos de una Red de Computadoras

Las redes de computadoras están formadas por diferentes componentes y dispositivos de interconexión que permiten el intercambio de la información entre dichos componentes. A continuación se presentan los componentes básicos que conforman una red.

1.3.1. Dispositivos de Interconexión

Los dispositivos de interconexión tienen, entre sus principales objetivos, unir las redes entre sí y permitir el intercambio de información entre las mismas.

Concentrador ó Hub

Un Concentrador o Hub es un dispositivo de interconexión con varios puertos utilizado para concentrar y organizar el cableado en una red de área local. [6]

Repetidor ó Repeater

El Repetidor o Repeater es un dispositivo de interconexión que se instala para amplificar la señal del cable, de forma que se pueda extender la longitud de la red, es decir que regenera la señal de la red para que llegue más lejos. [6]

Puente ó Bridge

Un Puente o Bridge conecta dos segmentos de redes iguales o distintas, es decir se pueden utilizar para dividir una red amplia en dos o más pequeñas o para conectar distintos tipos de redes. [6]

Enrutador ó Router

Los Enrutadores o Routers mantienen el tráfico fluyendo eficientemente sobre caminos predefinidos en una interconexión de redes complejas. Los enrutadores examinan la información de encaminamiento de los paquetes y los dirige al segmento de red adecuado, procesan los paquetes que van dirigidos a él, lo que incluye los paquetes enviados desde otros enrutadores con los que esté conectado. Ellos envían los paquetes por la mejor ruta hacia su destino, mantienen tablas de redes locales y enrutadores adyacentes en la red. Cuando uno de

estos dispositivos recibe un paquete, consulta estas tablas para ver si puede enviar directamente el paquete a su destino. Si no es así, determina la posición de otro enrutador que lo pueda hacer. [6]

Pasarelas ó Gateway

Las Pasarelas o Gateway son programas o dispositivos de comunicaciones que transfieren datos entre redes que tienen funciones similares pero implantaciones diferentes, es decir de naturalezas distintas. [6]

Conmutador ó Switch

Los Conmutadores o Switchs son otros tipos de dispositivo con múltiples puertos, cada uno de los cuáles puede soportar una simple estación de trabajo o toda una red, son utilizados para enlazar redes LAN separadas y proveer un filtrado de paquetes entre ellas. [6]

Punto de Acceso ó Access Point

Los Puntos de Accesos inalámbricos (AP por sus siglas en inglés Access Point) son dispositivos parecidos a un Conmutador o Switch que le da acceso a todos los nodos conectados a él. [6]

Servidor ó Server

Los servidores de red son computadoras que ofrecen servicios y recursos a sus clientes, estaciones de trabajo y a otros servidores a través de una red informática. Los servidores proporcionan la gestión centralizada de los recursos, la seguridad y mayor acceso a los recursos de la red. [7]

Estación de Trabajo

Las estaciones de trabajo son computadoras que pueden funcionar de forma independiente a la red y gestionar sus propios archivos y procesamiento. En una red típica, la mayoría de las computadoras son estaciones de trabajo que pueden trabajar sin estar conectados a una red. La posibilidad de usar medidas de seguridad adicionales y de incorporar la gestión centralizada

que es parte del uso de recursos en red son las dos principales razones por las que las estaciones de trabajo se unen a una red. [7]

1.3.2. Medios de Transmisión

Los medios de transmisión constituyen el camino a través del cuál el emisor y el receptor pueden comunicarse en un sistema de transmisión de datos. Se distinguen dos tipos de medios: guiados o por cables y no guiados o inalámbricos. [8]

Los medios de transmisión guiados o por cables guían las ondas a través de un camino físico. Ejemplos de estos medios son: el cable Par Trenzado (STP, por sus siglas en inglés (Shielded Twisted Pair) y UTP, por sus siglas en inglés (Unshielded Twisted Pair)), el Coaxial y la Fibra Óptica. El Par Trenzado es adecuado para una red LAN que tenga pocos nodos, un presupuesto limitado y una conectividad simple. Sin embargo, en distancias largas y a altas velocidades, no garantiza la integridad de los datos, es decir, que no haya pérdida en la transmisión de los datos. Sin embrago el cable Coaxial, por la protección, se puede utilizar para cubrir grandes distancias y a altas velocidades. La Fibra Óptica, no es apropiada para conexiones de redes LAN debido a que es muy difícil de instalar y además es muy costoso. Por lo que, se prefieren para conexiones cortas, Pares Trenzados o cables Coaxiales. [8]

Los medios de transmisión no guiados son los que no confinan las señales mediante ningún tipo de cable, sino que las señales se propagan libremente a través del medio. Entre los medios más importantes se encuentran el aire y el vacío. Tanto la transmisión como la recepción de información se llevan a cabo mediante antenas. Según el rango de frecuencias de trabajo, las transmisiones no guiadas se pueden clasificar en tres tipos: radio, microondas y luz (infrarrojos/láser). [8]

Estos componentes y dispositivos de interconexión mencionados anteriormente tienen asociados una dirección IP que los identifican en todo momento en Internet.

1.4. Dirección IPv4

Las direcciones de IPv4 (Protocolo de Internet versión 4) representan un número binario de 32 bits separados por punto en 4 grupos de 8 bits. Estas direcciones se utilizan en todas las conexiones desde y hacia la computadora a la cuál identifica. No existen en Internet dos

computadoras con la misma dirección, aunque una misma computadora puede tener más de una dirección IP en el caso de una computadora que tenga más de una tarjeta de red.

Bits	0	1	2	3	4	5	6	7		8	9	10	11	12	13	14	15		16	17	18	19	20	21	22	23		24	25	26	27	28	29	30	31
	Octeto 1								.	Octeto 2								.	Octeto 3								.	Octeto 4							
Ejemplo	0	1	1	1	0	0	0	1	.	1	0	0	1	0	1	0	1	.	0	0	1	1	1	0	1	1	.	1	0	0	1	1	0	0	0
	113								.	149								.	59								.	152							

Figura 6 Formato de la Dirección IPv4

Las direcciones constan de dos partes: la primera parte de la dirección identifica a la red y la segunda identifica al host dentro de la red en cuestión. Para mayor sencillez se utiliza la notación decimal de puntos. Cada bloque de 8 bits puede contener un número que varía entre 0 y 255.

1.5. Direcciones de Internet

La dirección de Internet es la dirección que identifica a cada computadora en Internet, es decir la dirección que tiene asignada cada máquina, a diferencia de la dirección de red que se refiere a un rango de direcciones hacia donde se deberá redireccionar o enrutar los paquetes de información.

Internet se podría definir como una gran red de redes de computadoras, con la finalidad de intercambiar y compartir información entre todos los usuarios. Para que todas las computadoras y dispositivos conectados a la red puedan realizar este proceso, es necesario que cada uno de ellos tenga asociado una dirección de internet que los identifique en todo momento. [9]

Direcciones Especiales

Las direcciones especiales son aquellas direcciones reservadas para el funcionamiento propio de Internet, como por ejemplo los números 0 y 255 (dirección de red y dirección de difusión) que permiten reenviar los paquetes de información a una red específica o a un grupo de computadoras conectadas a determinada red. [9]

Direcciones Privadas

Las direcciones privadas son direcciones reservadas para las redes que no se van a conectar a Internet y que no van a necesitar conectividad con otra organización. Estas direcciones son: [9]

- 10.0.0.0 - 10.255.255.255 (Clase A)
- 172.16.0.0 - 172.31.255.255 (Clase B)
- 192.168.0.0 - 192.168.255.255 (Clase C)

Dirección de Difusión ó Broadcast

La dirección de difusión dirigida es aquella que permite enviar un datagrama IP a todos los host de una red remota dada. Se hace poniendo la parte local de la dirección completa a uno. [9]

1.5.1. Clases de Direcciones de Internet

Existen cinco clases de redes según sus direcciones IP. Las direcciones de clase D y E, están reservadas para aplicaciones especificas. A continuación aparecen las clases de direcciones de uso común. [10]

Clase A

- El primer bit de la dirección es 0.
- Los 7 bits siguientes identifican la red (netid).
- Los últimos 24 al computador (hostid).
- El número de direcciones es igual a 2^{24} = 16 777 216.

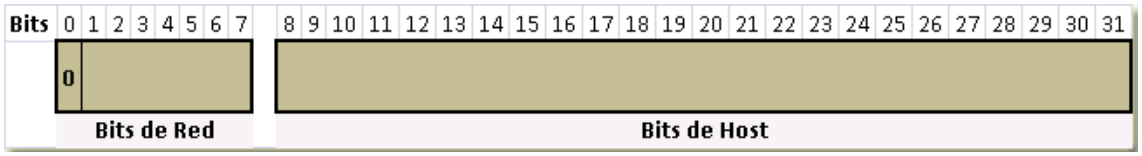


Figura 7 Formato de las Direcciones IPv4 de Clase A

Clase B

- Los dos primeros bits son 10.

- Los 14 bits siguientes identifican la red.
- Los 16 siguientes, a las computadoras.
- El número de direcciones es igual a 2^{16} = 65 536.

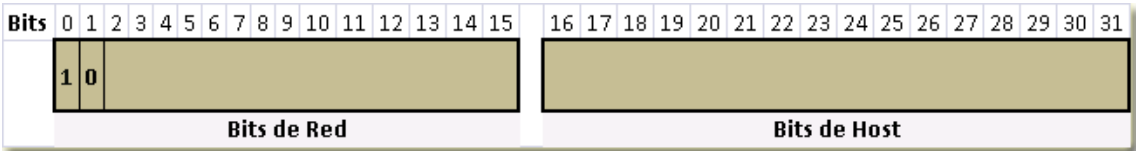


Figura 8 Formato de las Direcciones IPv4 de Clase B

Clase C

- Los tres primeros bits son 110.
- Los siguientes 21 bits identifican la red.
- Los últimos 8 a las computadoras.
- El número de direcciones es igual a 2^{8} = 256.

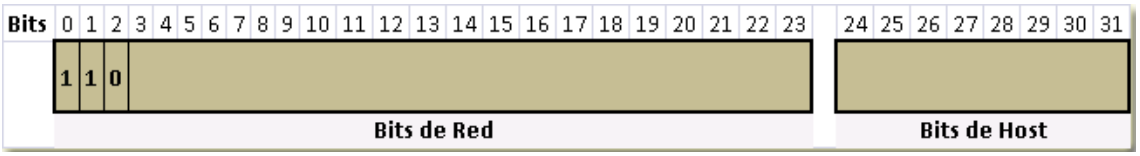


Figura 9 Formato de las Direcciones IPv4 de Clase C

1.5.2. Máscara de Subred.

La máscara de subred es una secuencia de 32 bits. Los bits que corresponden a los campos de red y subred de una dirección se ponen a 1 y los bits para el campo del sistema se ponen a 0.

Cada computadora necesita ser configurada con su dirección de Internet y con su máscara para que pueda comunicarse en su red. La función de la máscara de red es decirle a los dispositivos de la red que parte de una dirección es el número de la red, incluyendo la subred, y que parte es la correspondiente al host.

Una máscara de red por defecto se usa en las redes TCP/IP cuando éstas no están divididas en subredes. Todos los hosts requieren ésta máscara aunque estén en un solo segmento de red. La máscara por defecto que se puede utilizar, depende de la clase de dirección. [10]

1.6. Direccionamiento IP.

El direccionamiento IP consiste en asignarle a cada elemento de la red una dirección IP, que los identifique en todo momento en la red.

En el direccionamiento IP no existen reglas para asignar direcciones IP. Por tanto se deben seguir ciertos principios para asegurarse que se está asignando un número válido de Identificación de RED y de HOST.

- El id de red no debe ser 127. Esta Identificación está reservada para simular una red dentro de un mismo host y para funciones de diagnóstico.
- La identificación de red y el número de host no pueden estar todos a '1'. Si todos los bits están colocados a '1', la dirección se interpreta como una dirección de difusión en vez de una dirección de un host.
- La identificación de red y el número de host no pueden estar todos a 0. Si todos los bits están colocados a 0, la dirección se interpreta como 'esta red únicamente'.
- El número de host debe ser único para la Identificación de red. [11]

1.6.1. Proceso de Direccionamiento IP.

Una vez que se ha diseñado la topología de la red, se procede a configurar los equipos para que puedan comunicarse entre sí, dicha configuración consiste en asignarle a cada uno una dirección IP para que sean identificados en todo momento en la red. Este proceso se realiza de forma manual o automática.

A partir del par Dirección IP/Máscara que brinda el proveedor de servicios se procede a direccionar:

- Se identifican si existen subredes internas entre los enrutadores. Sólo se toman estas subredes de 2 conexiones para efectos de construcción de la tabla de enrutamiento.
- Se organizan las subredes de mayor a menor teniendo en cuenta las redes internas entre los enrutadores, debido a que, además del número de computadoras dentro de las subredes es necesario asignar una o más direcciones para las interfaces de todos los enrutadores conectados a ella. También es necesario señalar cuantos bits se necesitarían para direccionar las conexiones de cada una de las subredes. Para esto se

debe tener en cuenta que hay dos direcciones que no se pueden usar al efecto que son todo 0 (dirección de red) y todo 1 (dirección de difusión).

- Luego de haberse realizado los dos pasos anteriores correctamente se debe definir si es posible realizar o no el direccionamiento. Para esto se suma la cantidad de conexiones involucradas en el proceso y se determinan cuantos bits harían falta para direccionar esa cantidad. Luego se comparan con los bits que se tiene para direccionar, acorde a lo que brinda el proveedor de servicios. Si el proveedor de servicios brinda una dirección IP con máscara X, los bits que se tienen para direccionar son 32 – X, es decir los bits que restan a completar el espacio de direcciones para IPv4. Si este valor es mayor que el que se necesita para direccionar todas las conexiones involucradas en el proceso entonces se concluye que es posible realizar el direccionamiento.
- Una vez que se comprobó que es posible realizar el direccionamiento se procede a ver que tipo de direccionamiento se efectuaría.
- Se verifica si se puede direccionar de manera homogénea, si no se puede, se procede a verificar si es posible direccionar de manera eficiente. [12]

1.6.2. Método Homogéneo.

El direccionamiento homogéneo debe cumplir que:

Número Bits Necesarios Para Direccionar Todas las Subredes + Número Bits Necesarios Para Direccionar la Mayor de las Subredes ≤ Número Bits Que se Tiene Para Direccionar (acorde a lo que brinda el proveedor de servicios)

Para determinar el número de bits necesarios para direccionar todas las subredes se evita la combinación todo 0 (dirección de Red) debido a que trae confusión en algunos enrutadores, en el momento de enrutar los paquetes.

Para direccionar homogéneo se le daría un indicador de subred diferente a cada una de las subredes y se direccionarían todas de igual manera empezando por 1 para el primer host hasta el número total de conexiones que tenga cada una. En un direccionamiento homogéneo las máscaras de todas las subredes son iguales. [12]

1.6.3. Método Eficiente.

Para direccionar de manera eficiente hay un procedimiento bien sencillo que evita todo tipo de problemas que puedan surgir en el camino. El procedimiento consiste en verificar que para cada grupo de subredes se cumple la expresión:

$$A + B \leq R - A'$$

Se entiende como grupo de subredes todas aquellas que operan con la misma cantidad de bits.

A: Es la cantidad de bits necesarios para direccionar de manera independiente cada una de las subredes que operan con una misma cantidad de bits.

El valor de A esta determinado por la siguiente ecuación, para todos los grupos de subredes menos para el último.

$$2^A - 1 \geq N$$

Y por la siguiente ecuación para el último grupo de subredes.

$$2^A \geq N$$

Dónde:

N: Número total de subredes que tiene el grupo de subredes.

B: Cantidad de bits necesarios para direccionar los host dentro de las subredes.

R: Cantidad de bits útiles que brinda el proveedor de servicios.

A': Sumatoria de los valores de A, para el primer grupo de subredes A' toma valor 0. [12]

Este procedimiento se debe realizar antes de comenzar a direccionar.

Si es posible hacer el direccionamiento eficiente, se comienza a direccionar por la mayor de las subredes.

- Se plantea como quedaría la dirección que es dada por el proveedor de servicios, desglosándose bien los octetos donde se van a trabajar.
- De los octetos que se definieron se seleccionan los bits libre que brinda el proveedor de servicios para direccionar las subredes y se separan los bits de subred y los bits para direccionar los host dentro de esa subred.
- Se le ubica un indicador a la subred en base a la cantidad de bits que la identifica, es decir si el número de bits de la subred es 3 el indicador puede ser 001 o cualquier otra combinación posible menos 000 (dirección de red).
- Posteriormente se determina lo que sería la dirección de subred, la de difusión, la máscara para la subred, la dirección para el primer y último host de la subred.
- La dirección de subred se obtiene llevando todos los bits de host a 0.
- La dirección de difusión o Broadcast como también se le dice, se obtiene llevando todos los bits de host a 1.
- La máscara de subred se obtiene llevando todos los bits que definen la red del proveedor de servicios y los bits que definen la subred a 1 y el resto a 0.
- Las máscaras para todas las subredes que operen con una misma cantidad de bits será igual pero será distinta de las que operen con un número de bits diferentes para direccionar sus conexiones.
- Las direcciones del Primer Host dentro de la subred se determina asignándole el número 1 en binario y para el último host dentro de la subred se determina asignándole el valor de la suma de 1 + cantidad total de host de la subred, en binario.

Y así se procede análogamente con cada una de las subredes involucradas en el proceso. Siempre a partir de la dirección IP asignada a la subred que le antecede en la lista ordenada de mayor a menor. [12]

En el caso que sea una subred que pertenezca al mismo grupo de la subred anterior, se utiliza la misma cantidad de bits de la subred pero con un indicador diferente, es decir, si la subred anterior utiliza el indicador 001, la subred que le sigue utilizaría el indicador 010, o cualquier otra combinación menos 000 (dirección de red).

En el caso de una subred que pertenezca a otro grupo que no sea el de la subred anterior se reorganizan lo que sería los bits disponibles a los host y a la subred para este nuevo valor dejando un prefijo no utilizado del nivel o grupo superior, puede ser cualquier secuencia no

utilizada por las subredes anteriores incluso 000 si se garantiza que después habrá un 1 en la porción de subred. [12]

Una vez que son asignadas éstas direcciones IP las subredes se pueden comunicar e intercambiar información a través de los enrutadores.

1.7. Enrutamiento.

En una red de computadoras, el ruteo es el proceso de selección de un camino sobre el que se mandarán paquetes pasando por varias redes físicas si fuera preciso y el enrutador es el equipo que realiza la selección del camino a seguir. Este ruteo se denomina ruteo IP.

El ruteo IP selecciona un camino por el que se debe enviar un datagrama. El algoritmo de ruteo debe escoger como enviar un datagrama pasando por muchas redes físicas.

Una red de redes se compone de múltiples redes físicas interconectadas por enrutadores. Cada enrutador tiene conexiones hacia dos o más redes físicas, en cambio, generalmente un host se conecta directamente a una red física. Tanto los host como los enrutadores participan en el ruteo de datagramas IP que viajan a su destino.

Para que el proceso de enrutamiento se efectúe de manera correcta, es decir que todos los datagramas que se envíen lleguen al destino deseado, es necesaria una adecuada confección de las tablas de enrutamiento en los enrutadores de la red. Este aspecto es muy importante debido a que constituye la base de una correcta comunicación entre diferentes redes de computadoras. [13]

1.7.1. Tablas de Enrutamiento.

Los protocolos de ruteo IP determinan la forma en que son construidas las tablas que emplean los enrutadores para su funcionamiento. Las tablas de enrutamiento pueden ser:

- **Estáticas**: Construidas manualmente por el administrador, no se adaptan a los cambios de la red.
- **Dinámicas**: Emplean algoritmos para su utilización que permiten detectar cambios de la red y condiciones de carga.

Las rutas estáticas se deben introducir una a una para obtener la tabla final de enrutamiento mientras que con las rutas dinámicas se obtienen automáticamente dichas tablas de enrutamiento. [13]

1.7.2. Campos de las Tablas de Enrutamiento.

Las tablas de enrutamiento IP están formadas por las siguientes columnas: destino de red, máscara de red, puerta de enlace, interfaz y métrica. [9]

Destino de red

El destino de red se utiliza junto con la máscara de red para la coincidencia con la dirección IP de destino. El destino de red puede encontrarse entre 0.0.0.0, para la ruta predeterminada, y 255.255.255.255, para la difusión limitada, que es una dirección de difusión especial para todos los hosts del mismo segmento de red. [9]

Máscara de red

La máscara de red es la máscara de subred que se aplica a la dirección IP de destino cuando se produce la coincidencia con el valor del destino de red. [9]

Puerta de enlace

La puerta de enlace es la dirección IP del siguiente enrutador al que se debe enviar un paquete. [9]

Interfaz

La interfaz es la dirección IP configurada en el equipo local para el adaptador de red local que se utiliza cuando se reenvía un datagrama IP en la red. [9]

Métrica

La métrica indica el costo del uso de una ruta, que suele ser el número de saltos al destino IP. Cualquier destino en la subred local está a un salto de distancia y cada enrutador que se atraviesa en la ruta es un salto adicional. Si existen varias rutas al mismo destino con diferentes métricas, se selecciona la ruta con menor métrica. [9]

1.7.3. Enrutamiento Estático.

Para definir una ruta estática se debe conocer el rango IP y la máscara de la subred de destino, la puerta de enlace que generalmente es la dirección IP del salto siguiente es decir el enrutador siguiente y como parámetro opcional la distancia administrativa, que puede servir para dar mayor prioridad a una ruta que a otra.

Este tipo de ruta se utiliza en redes muy pequeñas debido a que el coste de administración de la red es muy elevado. Además resulta más complicado enfrentarse a una falla de red debido a que se deben modificar las rutas estáticas de todas las tablas de enrutamiento manualmente por el administrador para que se produzca el enrutamiento de paquetes a una red destino.

Las operaciones con rutas estáticas pueden dividirse en tres partes, como sigue:

- El administrador de la red configura la ruta.
- El enrutador instala la ruta en la tabla de enrutamiento.
- Los paquetes se enrutan de acuerdo a la ruta estática. [14]

1.7.4. Enrutamiento Dinámico.

Las redes con más de una posible ruta al mismo destino podrían usar enrutamiento dinámico. Las rutas dinámicas son asimiladas por los enrutadores mediante protocolos de enrutamiento como por ejemplo el Protocolo de Información de Enrutamiento (RIP) y el Protocolo del Camino más Corto Primero (OSPF). Estos protocolos implementan distintos tipos de algoritmos: [13]

- Algoritmo Vector – Distancia (VdD).
- Algoritmo Estado – Enlace ó Primero el Camino más Corto (SPF, por sus siglas en inglés Shortest Path First).

1.7.5. Algoritmo Vector – Distancia (VdD).

Este algoritmo es implementado en algunos protocolos de enrutamiento como por ejemplo en el protocolo RIP, con el objetivo de que los enrutadores conozcan la ruta a seguir para el envío de los paquetes de información. El mismo funciona como sigue:

- Cada nodo posee un identificador distinto.

- Cada nodo conoce la distancia para alcanzar a sus vecinos.
- Como distancia puede ser empleado un número que represente la cantidad de saltos, la demora en milisegundo, el número total de paquetes en cola en la trayectoria, o algo similar.
- Cada ruteador mantiene una tabla de ruteo indexada conteniendo una entrada para cada red destino en la red.
- Cada entrada contiene la línea preferida para alcanzar ese destino (vector) y la distancia.
- Al inicio, el vector de distancias de un nodo contiene solamente las redes conectadas directamente, es decir, distancia 0 hacia si mismo.
- Cada nodo transmite su VdD hacia sus vecinos (periódicamente o cuando haya cambios).
- Cada nodo guarda el VdD más reciente recibido de cada vecino.
- Cada nodo recalcula su propio VdD en función de lo que le informan sus vecinos. Lo realiza cuando:
    - Recibe un VdD de un vecino distinto que el tiene almacenado.
    - Se cae un enlace o cambia la distancia.

Las ventajas que presenta este algoritmo es que es muy sencillo, robusto, y utiliza tablas pequeñas de enrutamiento y las desventajas que presenta es que es de convergencia lenta, pueden aparecer bucles, y las tablas de ruteo de cada enrutador puede tener un crecimiento difícil por el gran volumen de información a transmitir. [13]

1.7.6. Algoritmo Estado de Enlace.

El algoritmo Estado de Enlace es implementado en varios protocolos de enrutamiento como por ejemplo en OSPF y tiene el mismo objetivo que el algoritmo de VdD, aunque presenta características diferentes. El algoritmo conocido por SPF (Shortest Path First - primero el camino más corto) funciona como sigue:

- Cada nodo construye un paquete denominado Link State Packet (LSP) que contiene la lista de sus vecinos y el costo de alcanzarlos.
- Los LSP de cada nodo se distribuyen mediante un mecanismo de difusión, al resto de nodos de la red.

- No se especifican rutas específicas sino, estados de los enlaces.
- Cada nodo recibe los LSP del resto de nodos y con ellos construye un mapa global de la topología de la red.
- Sobre el mapa global de la red se calculan mejores las rutas mediante el algoritmo de Dijikstra o cualquier otro.

Las ventajas que presenta este algoritmo es que es de convergencia rápida, debido a que los cálculos son locales, crecimiento fácil, es decir la información que se intercambia no es tabla de ruteo sino la información del estado de las conexiones directas de cada enrutador, por lo que su tamaño no depende del número de redes. Sin embargo presenta las siguientes desventajas: la difusión del estado de los enlaces puede ser complicada y cada nodo debe conocer la topología completa de la red. [13]

1.7.7. Algoritmo de Distribución LSP.

Este algoritmo es la parte más crítica del encaminamiento basado en estado de enlaces. Cada LSP (por sus siglas den inglés Link State Packet) se envía al resto mediante un algoritmo de inundación. [13]

- Un nodo reenvía cada LSP recibida a través de todos sus interfaces salvo por el que lo recibió.
- Para evitar duplicados, se guarda una copia del último LSP recibido de cada nodo. Si se recibe uno idéntico al almacenado, no se reenvía.
- Cada LSP lleva un número de secuencia para saber su orden.
- Cada LSP lleva un campo de tiempo de vida, que indica el tiempo que lleva en la red.

1.7.8. Algoritmo de Dijikstra.

Una versión del algoritmo de Dijikstra es utilizado por los algoritmos VdD y Estado de Enlaces analizados anteriormente. El mismo determina el camino más corto para llegar a cualquier nodo desde un nodo fuente. A continuación se muestra como funciona el algoritmo. [15]

- Estructuras de datos:
    - D arreglo para distancia mínima a cada nodo.
    - R arreglo para próximo tramo a seguir en la ruta.

- Entrada: Grafo con arcos con peso reflejando distancia entre nodos. Nodo fuente.
- Salida: D[i] conteniendo distancia más corta hasta i. R[i] siguiente tramo para llegar a nodo i.

El peso puede ser el número de conmutadores en el camino, reflejar la capacidad de la conexión, o una política de administración.

Estos procesos de direccionamiento IP y de enrutamiento se simularán en una herramienta de simulación de redes de computadoras, con el objetivo de que los usuarios apliquen estos conocimientos teóricos de manera virtual.

1.8. Simulación de Redes de Computadoras.

La simulación se usa como una herramienta que permite al usuario una mejor comprensión de la situación estudiada. En el caso de las redes de computadoras, existe una gran variedad de programas que permiten su simulación, obteniendo un resultado que debe ser interpretado y analizado.

En la actualidad existen varias herramientas de simulación, las cuáles tiene muchas facilidades de uso, pero también son muy costosas. Estas herramientas requieren de una serie de datos de entrada para su funcionamiento y análisis de los resultados.

1.8.1. FLAN (F – Links And Nodes)

Es un software desarrollado en Java y se distribuye con licencia pública GNU. Se considera que pertenece al grupo de los simuladores de propósito general, debido a que por medio de Java se pueden crear y configurar nuevos dispositivos, aplicaciones o protocolos de red, aún si no están incluidos dentro de las librerías del programa, incluso se pueden realizar modificaciones al código fuente de esta herramienta. [Ver Anexo 1]

FLAN es una herramienta de simulación que permite el diseño, la construcción, y la prueba de una red de comunicaciones en un ambiente simulado. El programa hace el análisis de las redes asociando su estructura basada en nodos y enlaces, con bloques simples, por medio de los cuales se puede entender el funcionamiento especialmente de los protocolos de enrutamiento que maneja la capa de red. [16]

1.8.2. PACKET TRACER™

Es un simulador gráfico de redes desarrollado y utilizado por Cisco como herramienta de entrenamiento para obtener la certificación CCNA14. Es un simulador de entorno de redes de comunicaciones de fidelidad media, que permite crear topologías de red mediante la selección de los dispositivos y su respectiva ubicación en un área de trabajo, utilizando una interfaz gráfica. [Ver Anexo 2]

PACKET TRACER™ es un simulador que permite realizar el diseño de topologías, la configuración de dispositivos de red, así como la detección y corrección de errores en sistemas de comunicaciones. Ofrece como ventaja adicional el análisis de cada proceso que se ejecuta en el programa de acuerdo a la capa de Modelo OSI que interviene en dicho proceso; razón por la cual es una herramienta de gran ayuda en el estudio y aprendizaje del funcionamiento y configuración de redes de comunicaciones y aplicaciones telemáticas. [16]

1.8.3. KIVA

Es un simulador de redes basado en Java que permite especificar diferentes esquemas de redes de datos y simular el enrutamiento de paquetes a través de dichas redes. [Ver Anexo 3]

KIVA es un software orientado principalmente a simular el comportamiento del protocolo IP, y especialmente para el estudio del tratamiento de los datagramas y el enrutamiento de los mismos por la red. Con esta herramienta se puede diseñar una topología de red con la interfaz gráfica, configurar el direccionamiento y las tablas de enrutamiento para los dispositivos y simular el envío de paquetes de un equipo a otro. La principal aplicación del programa es en la enseñanza de los fundamentos sobre el funcionamiento de redes de datos; pero este entorno, también puede ser muy útil para el diseño y comprobación del enrutamiento en redes de datos a nivel comercial. [16]

1.8.4. NETWORK VISUALIZER

Network Visualizer es una aplicación comercial con pretensiones mucho más ambiciosas y profesionales. Su objetivo es servir de ayuda, sustituyendo a un laboratorio de enrutadores reales, en la obtención de la certificación CCNA de Cisco.

En cuanto a su valor didáctico de la versión de evaluación, está limitado en gran medida, debido a que no todos los comandos de los enrutadores están disponibles y sólo un número limitado de laboratorios se pueden llevar a cabo con dicha versión.

Por supuesto, todas estas restricciones desaparecen en la versión comercial, que ofrecen todos los comandos de los enrutadores y la posibilidad de experimentar con el entorno libremente.

Se puede decir que es un paso más en la formación de los alumnos, ofreciéndoles una visión más completa de los enrutadores y comando disponibles. Sin duda, su precio de $189.00 por licencia individual echará para atrás a más de una institución educativa, aunque es uno de los mejores simuladores en este campo. [16]

1.8.5. NETSIM

El NetSim es uno de los simuladores de redes más completos, fue desarrollado con propósitos de aprendizaje, como complemento a los cursos que brinda la empresa CISCO para sus diferentes certificaciones en manejo de dispositivos de redes de datos. Este no es un simulador gratuito, pero su funcionalidad es lo más destacado y sus características permiten que se analicen redes de datos casi como si se estuviera trabajando en la realidad. El NetSim presenta una consola similar a la de los enrutadores CISCO y toda su configuración se la realiza de la misma manera como si se estuviera trabajando con equipos reales. [17]

Estas herramientas de simulación de redes de computadoras presentan grandes ventajas desde el punto de vista de la simulación pero no son herramientas de código abierto, son muy costosas y no cumplen las características metodológicas para llegar a los estudiantes de la UCI, debido a que no explican de manera factible el funcionamiento de los procesos de Direccionamiento IP y de Enrutamiento. Por lo que se decidió realizar una Herramienta de Simulación de Redes que sea de código abierto, libre, multiplataforma, gratis, propia de la UCI y que cumpla las características metodológicas necesarias para que los estudiantes entiendan el funcionamiento del direccionamiento IP y el enrutamiento.

A continuación se presentan los tópicos más importantes que se tuvieron en cuenta para emprender el desarrollo de dicha herramienta y garantizar la calidad en el proceso de desarrollo así como la del producto final.

1.9. Metodologías de Desarrollo de Software

Las metodologías de desarrollo de software tienen como objetivo dar solución a los problemas existentes en la producción de software, que cada vez son más complejos. Éstas engloban procedimientos, técnicas, documentación y herramientas que se utilizan en la creación de un producto de software de gran valor y alta calidad. El éxito del producto depende, en gran medida, de la selección de la metodología acogida, sea tradicional o ágil.

1.9.1. Metodologías Tradicionales

Las metodologías tradicionales imponen una disciplina de trabajo sobre el proceso de desarrollo del software, con el objetivo de conseguir un software más eficiente y predecible. Para ello, se hace un especial hincapié en la planificación total de todo el trabajo a realizar y una vez que está todo detallado, comienza el ciclo de desarrollo del producto software. Este planteamiento está basado en el resto de disciplinas de ingeniería, a pesar de que el software no pueda considerarse como la construcción de una obra clásica de la misma. A continuación se detallan unos de los procesos más usados dentro de los metodologías tradicionales: RUP y MSF. [18]

Metodología RUP (Proceso Unificado de Software)

La metodología RUP es un proceso de desarrollo de software que tiene como objetivo asegurar un producto de alta calidad que satisfaga los requerimientos de los usuarios finales. Como lenguaje de modelado utiliza el UML y entre sus principales características se encuentran las siguientes:

- *Dirigido por Casos de Uso:* Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, constituyen la guía fundamental establecida para las actividades a realizar durante todo el proceso de desarrollo del sistema.
- *Centrado en la arquitectura:* La arquitectura muestra la visión común del sistema completo.
- *Iterativo e Incremental:* RUP divide el proceso en cuatro fases de desarrollo, propone además que cada una de ellas se desarrolle en iteraciones, las cuáles aportan un incremento en el proceso de desarrollo y terminan con el cumplimiento del punto de control trazado en la fase.

RUP divide el proceso de desarrollo en cuatro fases dentro de las cuáles de realizan varias iteraciones de las actividades que son agrupadas en 9 flujos de trabajo.

Fases:

- *Inicio***:** El objetivo en esta fase es determinar la Visión del Proyecto.
- *Elaboración***:** En esta fase el objetivo es determinar la arquitectura óptima.
- *Construcción***:** En esta etapa el objetivo es llegar a obtener la capacidad operacional inicial del producto.
- *Transición***:** El objetivo es llegar a obtener la liberación del proyecto.

Flujos de Trabajo: Los flujos de trabajo se dividen en dos disciplinas: disciplina de desarrollo y disciplina de soporte.

Disciplina de Desarrollo

- *Modelamiento del Negocio*: Entendiendo las necesidades del negocio.
- *Requerimientos*: Trasladando las necesidades del negocio a un sistema automatizado.
- *Análisis y Diseño*: Trasladando los requerimientos dentro de la arquitectura de software.
- *Implementación*: Creando software que se ajuste a la arquitectura y que tenga el comportamiento deseado.
- *Pruebas*: Asegurándose que el comportamiento requerido es el correcto y que todo lo solicitado está presente.

Disciplina de Soporte

- *Configuración y administración del cambio*: Guardando todas las versiones del proyecto.
- *Administrando el proyecto*: Administrando horarios y recursos.
- *Ambiente*: Administrando el ambiente de desarrollo.
- *Distribución*: Hacer todo lo necesario para la salida del proyecto.

Los elementos de RUP son:

- *Trabajadores***:** Son las personas o entes involucrados en cada proceso.
- *Actividades***:** Son los procesos que se llegan a determinar en cada iteración.

- *Artefactos*: Un artefacto puede ser un documento, un modelo, o un elemento de modelo.
- *Flujos de Actividades*: Es la secuencia de actividades realizadas por los trabajadores que producen un resultado observable.

Una particularidad de esta metodología es que, en cada ciclo de iteración, se hace exigente el uso de artefactos, siendo por este motivo, una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo del software. [19]

MSF (Microsoft Solution Framework)

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. [19]

Características de MSF:

- *Adaptable*: es parecido a un compás, usado en cualquier parte como un mapa, del cual su uso es limitado a un lugar específico.
- *Escalable*: puede organizar equipos tan pequeños entre 3 o 4 personas, así como también, proyectos que requieren 50 personas a 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.

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.

Modelo de Arquitectura del Proyecto: Diseñado para acortar la planificación del ciclo de vida. Este modelo define las pautas para construir proyectos empresariales a través del lanzamiento de versiones.

Modelo de Equipo: Este modelo ha sido diseñado para mejorar el rendimiento del equipo de desarrollo. Proporciona una estructura flexible para organizar los equipos de un proyecto. Puede ser escalado dependiendo del tamaño del proyecto y del equipo de personas disponibles.

Modelo de Proceso: Diseñado para mejorar el control del proyecto, minimizando el riesgo, y aumentar la calidad acortando el tiempo de entrega. Proporciona una estructura de pautas a seguir en el ciclo de vida del proyecto, describiendo las fases, las actividades, la liberación de versiones y explicando su relación con el Modelo de equipo.

Modelo de Gestión del Riesgo: Diseñado para ayudar al equipo a identificar las prioridades, tomar las decisiones estratégicas correctas y controlar las emergencias que puedan surgir. Este modelo proporciona un entorno estructurado para la toma de decisiones y acciones valorando los riesgos que puedan provocar.

Modelo de Diseño del Proceso: Diseñado para distinguir entre los objetivos empresariales y las necesidades del usuario. Proporciona un modelo centrado en el usuario para obtener un diseño eficiente y flexible a través de un enfoque iterativo. Las fases de diseño conceptual, lógico y físico proveen tres perspectivas diferentes para los tres tipos de roles: los usuarios, el equipo y los desarrolladores.

Modelo de Aplicación: Diseñado para mejorar el desarrollo, el mantenimiento y el soporte, proporciona un modelo de tres niveles para diseñar y desarrollar aplicaciones software. Los servicios utilizados en este modelo son escalables, y pueden ser usados en un solo ordenador o incluso en varios servidores. [19]

1.9.2. Metodologías Ágiles

En contraposición a estas metodologías clásicas, en los últimos años ha aparecido un nuevo grupo de metodologías, que se identifica como metodologías ágiles. Aportan como novedad, nuevos métodos de trabajo que apuestan por una cantidad apropiada de proceso. Es decir, ni se pierden en una excesiva cantidad de cuestiones burocráticas, ni defienden tampoco la falta total de procesos. Buscan el equilibrio en la relación proceso/esfuerzo. Algunos de las metodologías ágiles son. [18]

Programación Extrema (XP)

XP (de sus siglas en ingles Extreme Programming) es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. [20]

SCRUM

SCRUM define un marco para la gestión de proyectos, que se ha utilizado con éxito durante los últimos 10 años. Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos. El desarrollo de software se realiza mediante iteraciones, denominadas sprint, con una duración de 30 días. El resultado de cada sprint es un incremento ejecutable que se muestra al cliente. La segunda característica importante son las reuniones a lo largo proyecto, entre ellas destaca la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración. [20]

Feature – Driven Development (FDD)

FDD define un proceso iterativo que consta de 5 pasos. Las iteraciones son cortas (hasta 2 semanas). Se centra en las fases de diseño e implementación del sistema partiendo de una lista de características que debe reunir el software. [20]

Luego de este análisis de las metodologías ágiles y tradicionales se determinó que se utilizará la metodología tradicional RUP (por sus siglas en inglés Rational Unified Process) porque tiene como objetivo asegurar un producto de alta calidad que satisfaga los requerimientos de los usuarios finales, utiliza el UML como lenguaje de modelado, entre sus principales características que lo hacen una metodología robusta y poderosa está: dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental. Además provee una documentación detallada, a diferencia de las metodologías ágiles que son menos precisas en este sentido.

1.10. Lenguaje Unificado de Modelado (UML)

Como su nombre lo indica, UML es un lenguaje de modelado visual que permite modelar la complejidad de los sistemas a analizar o diseñar, se centra en la representación gráfica de un sistema. Este lenguaje indica cómo crear y leer los modelos, pero no dice cómo crearlos. Permite modelar sistemas utilizando tecnologías orientada a objetos (OO).

Las funciones fundamentales de UML son:

- **Visualizar**: UML permite expresar de una forma gráfica un sistema de forma que otro lo puede entender.
- **Especificar**: UML permite especificar cuáles son las características de un sistema antes de su construcción.
- **Construir**: A partir de los modelos especificados se pueden construir los sistemas diseñados.
- **Documentar**: Los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para su futura revisión. [21]

Para el modelado se utilizará el UML como lenguaje representativo porque permite la representación gráfica de un sistema con tecnologías orientada a objetos, especifica cuáles son las características de un sistema antes de su construcción, a partir de los modelos especificados se pueden construir los sistemas diseñados y los propios elementos gráficos se utilizan como documentación del sistema desarrollado que pueden servir para su futura revisión.

Una vez seleccionado el lenguaje de modelado visual se procede a seleccionar el lenguaje de programación para el desarrollo de la solución propuesta.

1.11. Lenguaje de Programación para Aplicaciones de Escritorio

El término aplicaciones de escritorio se refiere a aplicaciones que corren en una PC local, es decir, que no son hechas principalmente para correr en un servidor. Dependiendo del tipo de programa que se desee realizar hay diferentes lenguajes en los que se puede programar: Java, C++, Delphi, VisualBasic, Visual C# .NET, y otros. Es importante tener en cuenta sus características para elegir el correcto. Por ejemplo, para realizar una aplicación que no

dependa de la plataforma o Sistema Operativo, siempre se piensa en Java primeramente, pues Java es Multiplataforma, puede correr en Unix, Windows y otros sistemas con sólo instalar la Virtual – Machine ó Máquina Virtual.

1.11.1. JAVA

Java es una tecnología orientada al desarrollo de software con la cuál se puede realizar cualquier tipo de programa. Hoy en día, ésta tecnología ha cobrado mucha importancia en el ámbito de Internet gracias a su plataforma J2EE. La tecnología Java está compuesta básicamente por 2 elementos: el lenguaje Java y la máquina virtual (Java Virtual Machine).

Una de las principales características que favoreció el crecimiento y difusión del lenguaje Java es su capacidad de que el código funcione sobre cualquier plataforma de software y hardware. Esto significa que el programa escrito para Linux puede ser ejecutado en Windows sin ningún problema. Además es un lenguaje orientado a objetos que resuelve los problemas en la complejidad de los sistemas, entre otras. [22]

El lenguaje de programación Java no es simplemente otro lenguaje más, el mismo incluye una combinación de características que lo hacen único y está siendo adoptado por múltiples fabricantes como herramienta básica para el desarrollo de aplicaciones comerciales de gran repercusión. Java es un lenguaje de desarrollo de propósito general, y como tal es válido para realizar todo tipo de aplicaciones profesionales. Java presenta múltiples ventajas: es multiplataforma, robusto, de alto rendimiento y posee gran cantidad de herramientas gratuitas, es fácil de aprender para los programadores que conocen la programación orientada a objetos. Es un lenguaje bien estructurado, sin punteros y sin necesidad de tener que controlar la asignación de memoria a estructuras de datos u objetos. Para los programadores en C++ también es sencillo el cambio, debido a que la sintaxis es prácticamente la misma que en este lenguaje. [23]

Dentro de los entornos de desarrollo integrado (IDE) para el desarrollo de aplicaciones usando como lenguaje de programación Java se encuentran NetBeans y Eclipse.

NetBeans es un entorno de programación para varios lenguajes, incluyendo a Java y C++. Este desarrollo es de fuente abierto, es decir, se proporciona el código fuente del entorno para

que se pueda modificar de acuerdo a ciertos parámetros de licencia, es un producto libre y gratuito sin restricciones de uso. [24]

Eclipse es un IDE para todo y para nada en particular, es únicamente un armazón sobre el que se pueden montar herramientas de desarrollo para cualquier lenguaje, mediante la implementación de los componentes conectados (plugins) adecuados. La arquitectura de plugins de Eclipse permite, además de integrar diversos lenguajes sobre un mismo IDE, introducir otras aplicaciones accesorias que pueden resultar útiles durante el proceso de desarrollo como: herramientas UML, editores visuales de interfaces, ayuda en línea para librerías, etc. El entorno de desarrollo Eclipse, incluyendo sus plugins, está desarrollado por completo en el lenguaje Java.

Un problema habitual en herramientas Java como NetBeans y Eclipse es que son demasiado "pesadas". Es decir, necesitan una máquina muy potente para poder ejecutarse de forma satisfactoria. En gran medida, estas necesidades vienen determinadas por el uso de la librería Swing para su interfaz gráfico. [25]

1.11.2. C#

C# (leído en inglés "C Sharp" y en español "C Almohadilla") es el nuevo lenguaje de programación introducido por Microsoft en la plataforma .NET. Aunque es posible escribir código para la plataforma .NET en muchos otros lenguajes, C# es el único que ha sido diseñado específicamente para ser utilizado en ella, por lo que programar en ella usando C# es mucho más sencillo e intuitivo que hacerlo con cualquiera de los otros lenguajes, debido a que C# carece de elementos heredados innecesarios en .NET. Por esta razón, se suele decir que C# es el lenguaje nativo de .NET. [26]

La sintaxis y estructuración de C# es muy parecida a la de C++ o Java, debido a que la intención de Microsoft es facilitar la migración de códigos escritos en estos lenguajes a C# y facilitar su aprendizaje a los desarrolladores habituados a ellos. Sin embargo, su sencillez y el alto nivel de productividad son comparables con los de Visual Basic.

C# es un lenguaje de programación que toma las mejores características de lenguajes preexistentes como Visual Basic, Java o C++ y las combina en uno solo. Este lenguaje presenta diversas características: es sencillo, orientado a objetos, C# soporta todas las

características propias del paradigma de programación orientada a objetos: encapsulación, herencia y polimorfismo, es un lenguaje orientado a componentes, es decir que la sintaxis de C# permite definir cómodamente propiedades, eventos o atributos. También proporciona la gestión automática de memoria, todo lenguaje de .NET tiene a su disposición el recolector de basura del CLR. Esto tiene el efecto en el lenguaje de que no es necesario incluir instrucciones de destrucción de objetos. Sin embargo, dado que la destrucción de los objetos a través del recolector de basura es indeterminista y sólo se realiza cuando éste se active –ya sea por falta de memoria, finalización de la aplicación o solicitud explícita en el fuente-, C# también proporciona un mecanismo de liberación de recursos determinista a través de la instrucción using. [26]

Luego del análisis de los lenguajes de programación para aplicaciones de este tipo, se decidió utilizar el lenguaje Java con su IDE de desarrollo NetBeans 6.0 por las múltiples ventajas que éste trae consigo, tales como: es un producto libre, gratuito y sin restricciones de uso, multiplataforma. Además es robusto, de alto rendimiento y posee una gran cantidad de herramientas gratuitas.

A partir de la selección de la metodología de desarrollo de software, del lenguaje de modelado visual y del lenguaje de desarrollo para aplicaciones de escritorio se realizará un análisis de las herramientas CASE que le dan soporte a la selección anterior.

1.12. Herramientas CASE

Se puede definir a las Herramientas CASE (Ingeniería de Software Asistida por Ordenador de sus siglas en ingles Computer Aided Software Engineering) 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.

La realización de un software requiere que las tareas sean organizadas y completadas en forma correcta y eficiente. Las Herramientas CASE fueron desarrolladas para automatizar esos procesos y facilitar las tareas de coordinación de los eventos que necesitan ser mejorados en el ciclo de desarrollo de software. A continuación se mencionarán algunas de estas herramientas. [27]

Visual Paradigm

Visual Paradigm para UML a pesar de que no es software libre, posee una versión freeware (programa de libre acceso ó programa con derechos de autor que se puede utilizar sin pago alguno) llamada Visual Paradigm for UML Comunity Edition. Es una herramienta CASE profesional que soporta el ciclo completa de desarrollo de software: análisis y diseño orientado a objetos, construcción, pruebas y despliegue. Permite dibujar todos los diagramas de clases, código inverso, generar código desde diagramas y generar documentación automática de informes en formato pdf, word y html. [28]

Rational Rose Enterprise Edition

Es una herramienta software que utiliza el UML como lenguaje de modelado visual. La misma permite: especificar, analizar y diseñar el sistema antes de codificarlo, mantiene la consistencia de los modelos del sistema software, chequeo de la sintaxis UML, generación de la documentación automáticamente y generación de código a partir de los modelos, e ingeniería inversa es decir crear los modelos a partir del código. [27]

Enterprise Architect

Enterprise Architect es una herramienta comprensible de análisis y diseño UML, cubriendo el desarrollo de software desde el paso de los requerimientos a través de las etapas del análisis, modelos de diseño, pruebas y mantenimiento. Es una herramienta multi – usuario, basada en Windows, diseñada para ayudar a construir software robusto y fácil de mantener. Ofrece salida de documentación flexible y de alta calidad. El Lenguaje Unificado de Modelado provee beneficios significativos para ayudar a construir modelos de sistemas de software rigurosos y donde es posible mantener la trazabilidad de manera consistente. Enterprise Architect soporta este proceso en un ambiente fácil de usar, rápido y flexible, permite ingeniería de Código Directa e Inversa (ediciones Corporativa y Profesional solamente) y Soporte para Java, C#, C++, Delphi, Visual Basic, Python y PHP, entre otros. [29]

Como herramienta CASE en la solución propuesta de este trabajo será utilizado Visual Paradigm porque es una herramienta que utiliza UML como lenguaje de modelado, ayuda a una construcción más rápida de aplicaciones con calidad, soporta el ciclo de vida completo del

desarrollo de software, permite además generar código inverso para Java y para otros lenguajes de programación.

Conclusiones Parciales

Para el desarrollo de la herramienta de simulación de redes de computadoras que se propone en este trabajo de diploma se seleccionaron:

- RUP como Metodología de Desarrollo de Software, debido a que posibilita desarrollar el sistema dirigido por Casos de Uso, centrado en la Arquitectura e Iterativo e Incremental, siendo cada iteración superior a la anterior.
- UML como Lenguaje de Modelado porque permite la representación gráfica de un sistema con tecnologías orientada a objetos.
- Visual Paradigm como herramienta CASE, soporta UML, el ciclo completo de desarrollo que propone RUP, posibilita realizar ingeniería inversa, generar documentación correspondiente en cada flujo de trabajo.
- Lenguaje de programación Java con su IDE de desarrollo Netbeans 6.0. Por ser un lenguaje multiplataforma y las facilidades gráficas que brinda este IDE.

Capítulo 2. Características del Sistema

En este capítulo se tratan los conceptos más importantes del área donde se desarrolla el problema planteado mediante el modelo de dominio. Se describirán las capacidades que deberá cumplir el sistema a través de los requisitos funcionales y las características del mismo expresadas por los requisitos no funcionales. Además se definirán los actores y casos de uso del sistema así como la relación entre ellos mediante el diagrama de casos de uso para un mejor entendimiento de las acciones que realiza el actor en el sistema.

2.1. Modelo de Dominio

El Modelo de Dominio ó Modelo Conceptual es una representación visual de los conceptos u objetos del mundo real significativos para un problema o área de interés. Representa clases conceptuales del dominio del problema y conceptos del mundo real, no de los componentes de software. Una clase conceptual puede ser una idea o un objeto físico (símbolo, definición y extensión). [30]

El modelo de Dominio tiene como objetivo fundamental la comprensión y descripción de las clases más importantes en el sistema.

A continuación se representa un acercamiento de la solución propuesta donde se modelan los principales conceptos con los que se trabajarán en el desarrollo de la solución. Así como las relaciones existentes entre ellos.

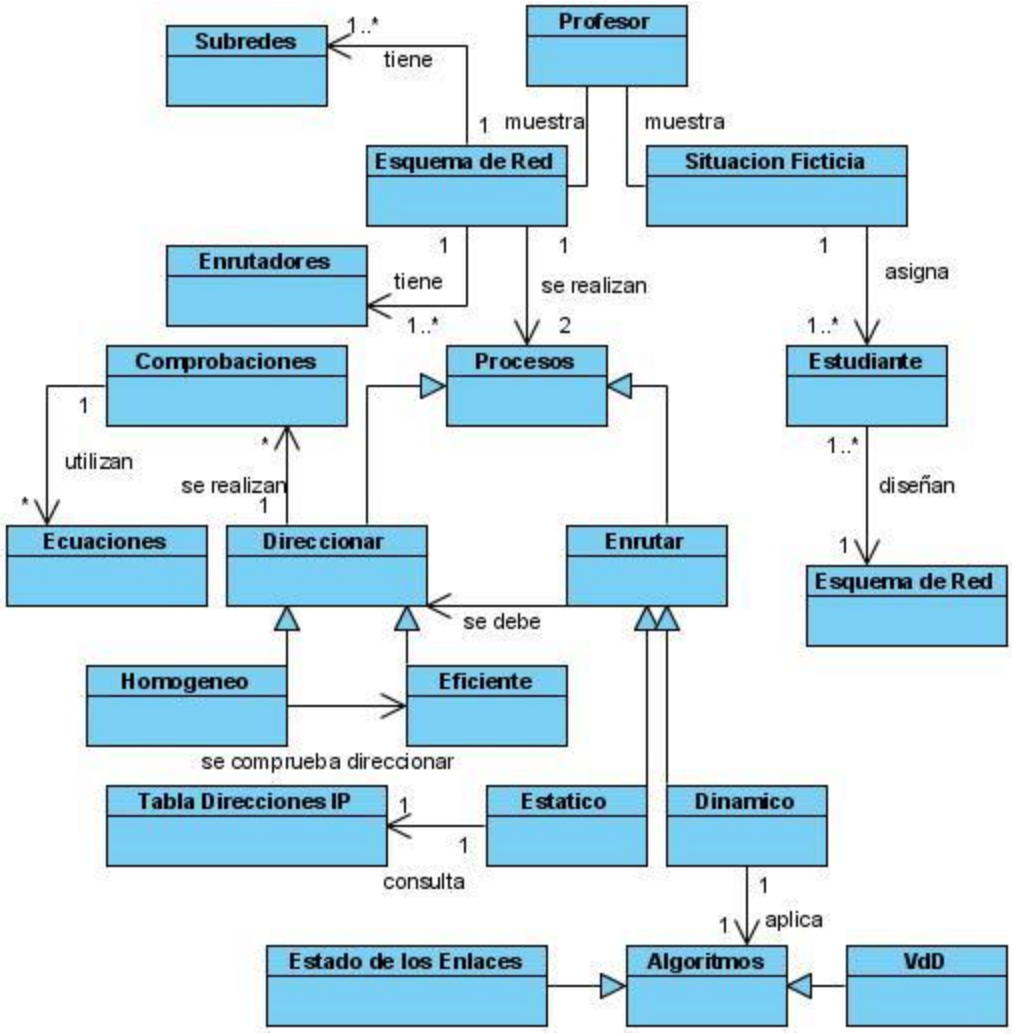


Figura 10 Modelo de Dominio

2.1.1. Descripción Textual del Modelo de Dominio

El profesor muestra un esquema de red y una situación ficticia sobre las necesidades de interconexión de una empresa que es asignada a un grupo de estudiantes para que realicen el diseño de la red. El esquema de la red está formado por subredes y enrutadores. Como dato tiene el espacio de direcciones asignadas por el proveedor de servicios. En el esquema se realizan varios procesos como son: Direccionar y Enrutar. Para direccionar se realizan una serie de comprobaciones mediante las ecuaciones correspondientes a ver si es posible realizar el proceso, el direccionamiento puede ser Homogéneo o Eficiente. Si no se puede direccionar homogéneo se comprueba direccionar eficiente. Para enrutar se debe haber direccionado previamente. En este proceso hay dos tipos de enrutamiento: estático y dinámico. Si se va a

enrutar de manera estática, se obtienen los datos de la tabla de direcciones IP asignadas a las subredes del esquema de red, sino, se aplican algoritmos como VdD o Estado de Enlaces.

2.1.2. Glosario de Términos del Dominio

Profesor: Es la persona que enseña, de la mejor manera posible, determinada materia a los estudiantes.

Esquema de Red: Es el conjunto de equipos y componentes de la red interconectados entre sí.

Situación Ficticia: Es un documento, un ejercicio ficticio que muestra el profesor sobre las necesidades de interconexión de una empresa, el presupuesto virtual y el listado de precios de empresas comercializadoras de equipamiento.

Subred: Es una red dentro de otra, es decir, equipos y componentes interconectados entre sí que forman parte de una red más grande.

Enrutador: Es un dispositivo de interconexión que permite el intercambio de paquetes de información entre varias subredes, enrutando los paquetes por la ruta más corta.

Procesos: Son operaciones que se realizan a partir del diseño de la red.

Direccionar: Es el proceso que se realiza para asignar direcciones IP a las subredes que forman parte del diseño de la red.

Homogéneo: Es un método empleado para direccionar. Este método direcciona todas las subredes con una misma dirección IP asignada por el proveedor de servicios y la máscara de todas las subredes involucradas en el proceso es la misma.

Eficiente: Es un método empleado para direccionar. Este método direcciona todas las subredes con diferentes direcciones IP a partir de la dirección asignada por el proveedor de servicios y la máscara de todas las subredes que operan con la misma cantidad de bits es igual pero diferente para las otras que operan con un número de bits distinto.

Comprobaciones: Son pruebas que se realizan para determinar si es posible o no realizar el proceso de direccionamiento IP.

Ecuaciones: Son cálculos que se realizan en las comprobaciones.

Enrutar: Es el proceso de actualización de las tablas de enrutamiento de los enrutadores para que puedan enviar los paquetes de información de una subred a otra por la ruta más corta.

Estático: Es un tipo de enrutamiento en el que el usuario debe actualizar las entradas de las tablas de enrutamiento de los enrutadores manualmente cuando ocurre algún cambio e irregularidad en la red.

Dinámico: Es un tipo de enrutamiento que utiliza algoritmos de enrutamiento (VdD o Estado de Enlaces) para actualizar las tablas de los enrutadores automáticamente cuando ocurre algún cambio e irregularidad en la red.

Tabla de Direcciones IP: Es la tabla resumen que se genera una vez que el usuario ha realizado el proceso de direccionamiento IP. En ella se encuentran las direcciones correspondientes a cada subred.

Algoritmos: Son técnicas empleadas por los enrutadores para actualizar automáticamente las entradas de sus tablas de enrutamiento.

VdD: Vector de Distancia, es un algoritmo empleado por los enrutadores para actualizar automáticamente sus tablas de enrutamiento.

Estado de Enlace: Es un algoritmo empleado por los enrutadores para actualizar automáticamente sus tablas de enrutamiento. A diferencia del anterior cada nodo de la red debe conocer la topología completa de la red.

Estudiante: Es el que realiza las tareas asignadas por el profesor para lograr vencer determinado objetivo.

2.2. Captura de Requisitos

Los requisitos constituyen capacidades o condiciones que el sistema debe cumplir. Estos facilitan el entendimiento entre usuarios y desarrolladores del sistema a elaborar. A continuación se representan los requisitos funcionales por los que se regirá el sistema y los no funcionales que exponen las características de la aplicación.

2.2.1. Requisitos Funcionales

Los siguientes requisitos establecen las funcionalidades que el sistema debe cumplir en su implementación.

El sistema debe permitir:

R1. Diseñar Redes.

R1.1. Diseñar redes LAN.

R1.2. Diseñar redes WAN.

R2. Direccionar.

R2.1. Direccionar por el método Homogéneo.

R2.2. Direccionar por el método Eficiente.

R2.3. Direccionar paso a paso.

R3. Enrutar.

R3.1. Enrutar de manera estática.

R3.2. Enrutar de manera dinámica.

R3.3. Enrutar paso a paso.

R3.3.1. Seleccionar el algoritmo de enrutamiento

R3.4. Insertar ruta estática en la tabla de ruteo.

R3.5. Eliminar ruta estática de la tabla de ruteo.

R3.6. Modificar ruta estática de la tabla de ruteo.

R4. Salvar los datos de la red en un fichero.

R5. Cargar los datos de la red desde un fichero.

R6. Mostrar Ayuda.

2.2.2. Requisitos No Funcionales

Los siguientes requisitos son propiedades o cualidades que el sistema debe tener, estableciendo de esta forma aspectos que regulan el comportamiento del mismo.

Requerimientos de Apariencia o Interfaz Externa

La aplicación debe ser diseñada con una interfaz sencilla y amigable para que el usuario navegue por ella sin dificultad alguna.

Requerimientos de Portabilidad

El sistema podrá ser ejecutado sobre los Sistema Operativos: Linux y Windows, de ahí su característica de ser multiplataforma.

Requerimientos de Rendimiento

Para lograr un mayor rendimiento es necesario un tiempo de respuesta inmediato para que los usuarios no se sientan desmotivados por la espera y cierren la aplicación.

Requerimientos de Usabilidad

El sistema será utilizado por cualquier persona que posea conocimientos de computación y de redes de computadoras.

Requerimientos de Diseño e Implementación

El sistema será implementado en Java, usando el IDE NetBeans 6.0, se utilizará como herramienta de modelado Visual Paradigm, y un paradigma de Programación Orientado a Objetos.

Requerimientos de Software

Se debe disponer de la maquina virtual de Java versión 1.3.

Requerimientos de Hardware

Se requiere, como mínimo, de una computadora con 512 Mb de RAM y 50 Mb de capacidad del disco duro.

Requerimientos de Seguridad

El usuario tendrá control total de la aplicación, sin restricción alguna.

2.3. Modelo de Casos de Uso del Sistema

En esta sección se identifican los actores del sistema a desarrollar, y se definen los casos de uso del sistema, así como la descripción textual de los mismos.

2.3.1. Actores del Sistema

Los actores de un sistema son agentes externos, roles que las personas (usuarios) o dispositivos juegan cuando interactúan con el software.

Actores	Descripción
Usuario	Representa a las personas que van a interactuar con el sistema, ya sea el profesor, el estudiante, o cualquier persona que requiera de su uso.

Tabla 1 Actor del Sistema

2.3.2. Definición de los Casos de Uso del Sistema

Los casos de uso especifican una secuencia de acciones que el sistema puede llevar a cabo interactuando con sus actores, incluyendo alternativas dentro de la secuencia.

Los casos de uso definidos son los siguientes:

- Diseñar Redes
- Direccionar
- Enrutar
- Gestionar Ficheros
- Mostrar Ayuda

2.3.3. Diagrama de los Casos de Uso del Sistema

Los diagramas de casos de uso se crean para visualizar las relaciones entre los actores y los casos de uso.

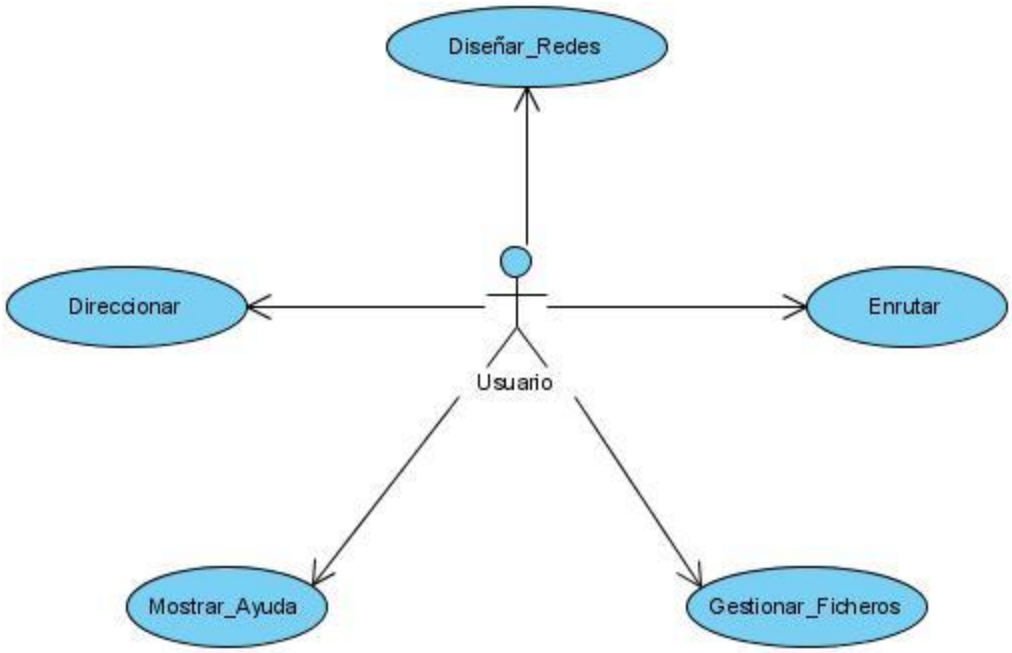


Figura 11 Diagrama de los Casos de Uso del Sistema

2.3.4. Descripción de los Casos de Uso del Sistema

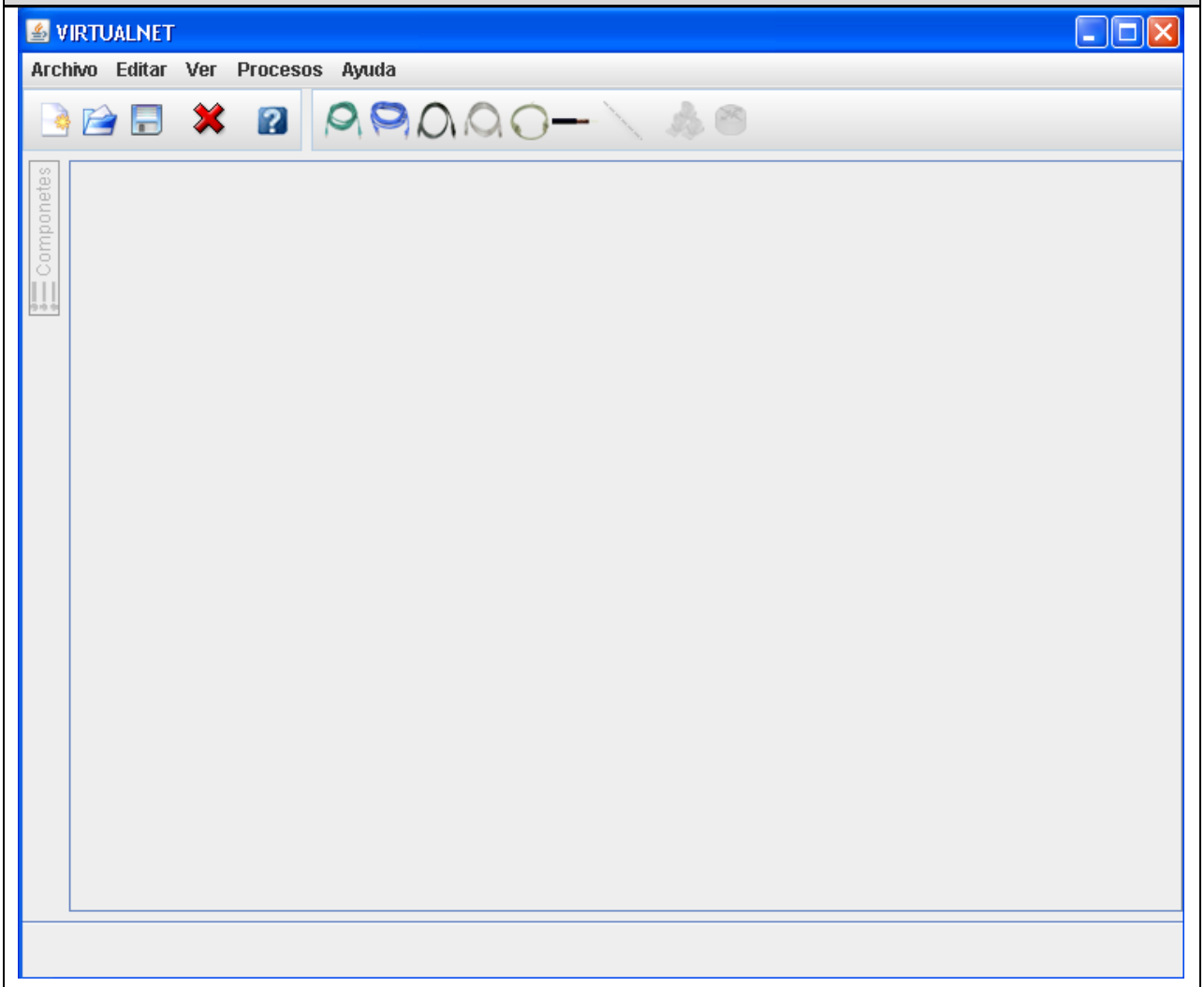
Cada caso de uso tiene una descripción de las funcionalidades que ejecutará el sistema propuesto como respuesta a las acciones del usuario. Las tablas presentadas a continuación argumentan los flujos operacionales de cada uno de ellos.

Caso de Uso Diseñar Redes

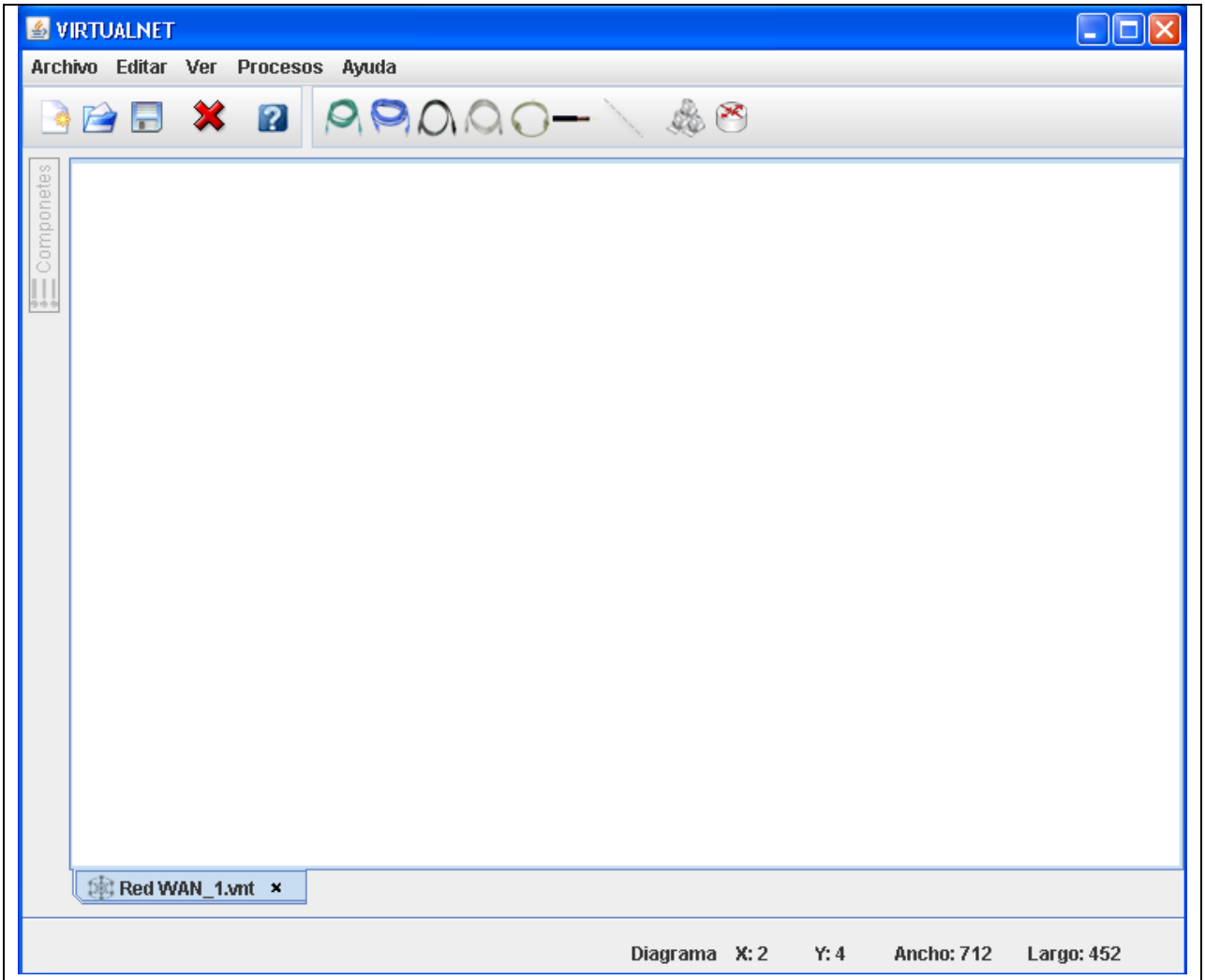
Caso de Uso	**Diseñar Redes**
Actores	Usuario
Propósito	Diseñar la red.
Resumen	El caso de uso se inicia cuando el usuario accede al sistema y selecciona la opción que desea realizar ya sea Diseñar LAN o WAN y termina cuando el usuario selecciona los componentes de la red a diseñar y establece todas las conexiones pertinentes entre los mismos.
Referencia	R1.1, R1.2
Precondición	Crear un nuevo diagrama.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario selecciona la opción del menú Diseñar Redes.	1.1. El sistema muestra un submenú con los tipos de redes que puede diseñar: • Red LAN. • Red WAN.
2. Elige la opción a realizar. Si elige la opción Red LAN. Ir a sección "**Diseñar Redes LAN**". Si elige la opción Red WAN. Ir sección "**Diseñar Redes WAN**".	

	3. Finaliza el Caso de Uso
Sección: Diseñar Redes LAN	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra un menú con varios componentes de redes como son: • Equipos. • Dispositivos de Interconexión Activos. • Y en el menú de acceso rápido los tipos de cables.
2. Selecciona los componentes que formarán parte de la red.	2.1. Muestra una breve descripción de los componentes seleccionados.
3. Selecciona el cableado a utilizar.	
4. Establece las conexiones entre los componentes.	4.1. Verifica la conexión. 4.2. Muestra una línea con un color determinado en dependencia del cableado seleccionado entre los componentes de la conexión.
Sección: Diseñar Redes WAN	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra un menú con los siguientes componentes de redes: • Enrutadores. • Subredes.
2. Selecciona los componentes que formarán parte de la red.	
3. Establece las conexiones entre los componentes seleccionados.	3.1. Verifica la conexión. 3.2. Muestra una línea con un color determinado en dependencia del cableado seleccionado entre los

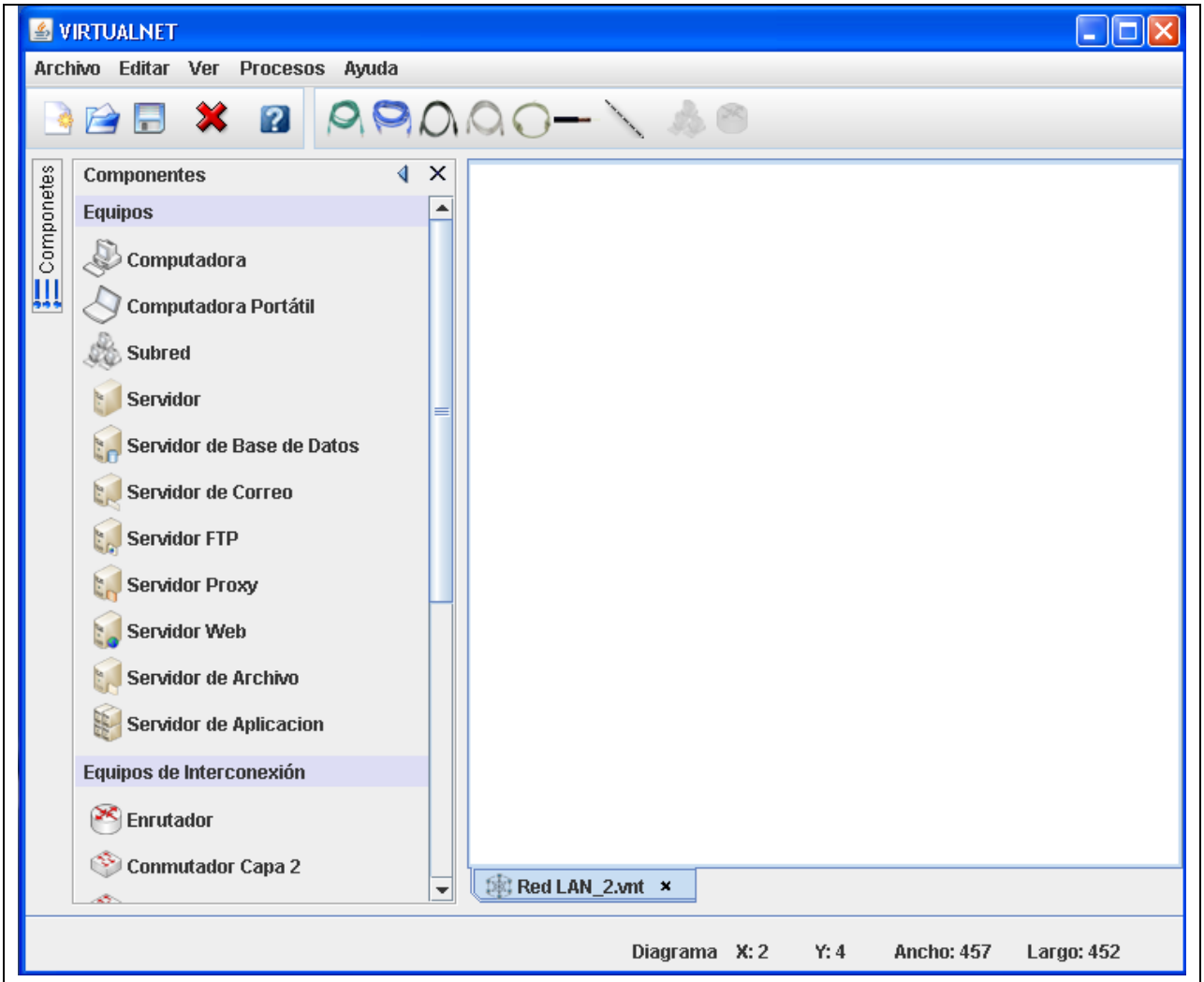
	componentes de la conexión.
Flujos Alternos	
Sección: Diseñar Redes WAN 3.2. Si la conexión que se establece entre los componentes de la red es incorrecta, el sistema no muestra la conexión entre los mismos.	
Poscondiciones	Se obtiene el diseño de la red seleccionada.
Prioridad	Crítica
Interfaces	



Interfaz Principal



Interfaz Diseñar Red WAN



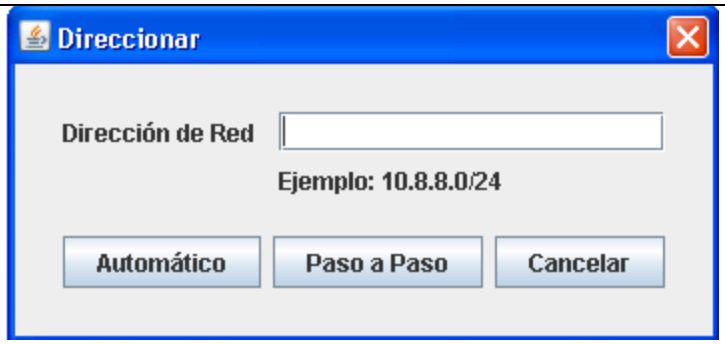
Interfaz Diseñar Red LAN

Tabla 2 Descripción del Caso de Uso Diseñar Redes

Caso de Uso Direccionar

Caso de Uso	Direccionar
Actores	Usuario
Propósito	Direccionar los componentes de la red.
Resumen	El caso de uso se inicia cuando el usuario accede al sistema y selecciona la opción Direccionar del menú Procesos y finaliza con la asignación de una dirección IP a cada

	componente de la red.
Referencia	R2.1, R2.2, R2.3
Precondición	Se debe haber diseñado una red WAN.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario selecciona la opción Direccionar del menú Procesos.	1.1. El sistema muestra una interfaz con el siguiente campo: • Dirección de Red. Y las opciones: • Automáticamente. • Paso a Paso. • Cancelar.
2. Selecciona la opción que desea realizar.	2.1. Si selecciona la opción Automáticamente o Paso a Paso se direcciona por el método homogéneo o eficiente en dependencia de la dirección de red entrada.
	3. Finaliza el Caso de Uso.
Flujos Alternos	
Si selecciona la opción Cancelar se cierra la interfaz Direccionar.	
Poscondiciones	Se le asigna una dirección IP a cada componente de la red.
Prioridad	Crítica
Interfaces	



Interfaz Direccionar

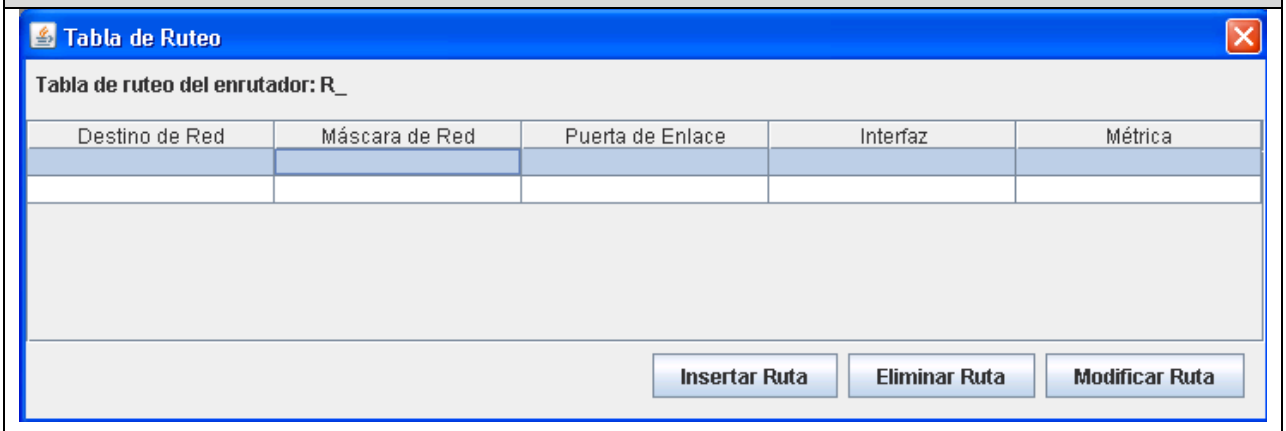
Tabla 3 Descripción del Caso de Uso Direccionar

Caso de Uso Enrutar

Caso de Uso	**Enrutar**
Actores	**Usuario**
Propósito	Confeccionar las tablas de enrutamiento de los enrutadores para que determinen la ruta más corta por donde debe viajar un paquete de información de una subred a otra.
Resumen	El caso de uso se inicia cuando el usuario accede al sistema y selecciona la opción Ver Tabla de Ruteo del popupMenu del Enrutador y finaliza cuando se han actualizado las tablas de enrutamiento de los enrutadores.
Referencia	R3.1, R3.2, R3.2.1, R3.3
Precondición	Se debe haber direccionado.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario selecciona la opción Ver Tabla de Ruteo del popupMenu del Enrutador.	1.1. El sistema confecciona la tabla de ruteo del Enrutador seleccionado y muestra una interfaz con los valores de dicha tabla. Y las opciones: • Insertar Ruta.

	• Eliminar Ruta. • Modificar Ruta.
2. Selecciona la opción que desea realizar. Si selecciona Insertar Ruta. Ir a la sección "**Insertar Ruta**". Si selecciona Eliminar Ruta. Ir a la sección "**Eliminar Ruta**". Si selecciona Modificar Ruta. Ir a la sección "**Modificar Ruta**".	
	3. Finaliza el Caso de Uso.
Sección: Insertar Ruta	
Acción del Actor	**Respuesta del Sistema**
1. El usuario entra manualmente los valores de la tabla de enrutamiento de los enrutadores.	1.1. El sistema actualiza esos valores en la tabla de enrutamiento modificada, chequeando que la entrada de la ruta es correcta. 1.2 Muestra al usuario si la ruta entrada es correcta o incorrecta.
Sección: Eliminar Ruta	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la ruta que desea eliminar de la tabla de ruteo.	1.1. Verifica que la ruta seleccionada es una ruta estática de la tabla de ruteo. 1.2. Si es una ruta estática, elimina la ruta de la tabla de ruteo del Enrutador y muestra nuevamente los valores de la nueva tabla.
Sección: Modificar Ruta	
Acción del Actor	**Respuesta del Sistema**

1. El usuario selecciona la fila de la tabla de ruteo que desea modificar.	1.1. Verifica que la fila seleccionada represente una ruta estática. 1.2. Si es una ruta estática, modifica la ruta de la tabla de ruteo del Enrutador y muestra nuevamente los valores de la nueva tabla.
Flujos Alternos	
Sección: Eliminar Ruta. 1.2. Si la ruta seleccionada no es una ruta estática, no elimina la fila y sigue mostrando la misma tabla de ruteo del Enrutador. **Sección: Modificar Ruta.** 1.2. Si la ruta seleccionada no es una ruta estática, no modifica la fila y sigue mostrando la misma tabla de ruteo del Enrutador.	
Poscondiciones	Se configuran las tablas de enrutamiento de los enrutadores.
Prioridad	Crítica
Interfaces	



Interfaz Tabla de Ruteo

Tabla 4 Descripción del Caso de Uso Enrutar

Caso de Uso Gestionar Ficheros

Caso de Uso	**Gestionar Ficheros**
Actores	**Usuario**
Propósito	Salvar y Cargar los datos de la red.
Resumen	El caso de uso se inicia cuando el usuario accede al sistema y selecciona la opción Abrir o Guardar en el menú Archivo o en el menú de acceso rápido y finaliza con la ejecución de la opción seleccionada.
Referencia	R4.1, R4.2
Precondición	En caso que se desee abrir un fichero, el mismo debe existir, y además debe estar en el formato con el que la aplicación lee los ficheros, es decir con extensión ".vnt" En caso que se desee salvar los datos de la red, debe existir previamente dicha red en el sistema.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra en el menú Archivo y en el menú de acceso rápido las opciones: • Abrir • Guardar
2. El usuario selecciona la opción deseada. Si selecciona la opción Abrir. Ir a la sección "**Abrir Fichero**". Si selecciona la opción Guardar. Ir a la sección "**Guardar Fichero**".	

	3. Finaliza el Caso de Uso
Sección: Abrir Fichero	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema muestra una ventana con varios directorios.
2. El usuario busca en los directorios y selecciona el fichero que desea abrir.	2.1. Abre el fichero seleccionado por el usuario y visualiza los datos en el panel de edición principal.
Sección: Guardar Fichero	
Acción del Actor	**Respuesta del Sistema**
	1. El sistema verifica que exista la red, en caso afirmativo, muestra una ventana con varios directorios.
2. El usuario busca el directorio donde desea guardar el archivo y especifica el nombre del mismo.	2.1. Guarda los datos en el fichero creado.
Flujos Alternos	
Sección: Abrir Fichero 2.1. Si el usuario selecciona un fichero sin el formato correcto, el sistema muestra un mensaje de error informando que no es posible efectuar dicha acción. **Sección: Guardar Fichero** 2.1. Si no existe la red en el sistema, se muestra un mensaje de error informando que no existen redes en el sistema.	
Poscondiciones	Se abren los datos del fichero seleccionado o se guardan los datos existentes en un fichero con extensión ".vnt".
Prioridad	Secundario

Tabla 5 Descripción del Caso de Uso Gestionar Ficheros

Caso de Uso Mostrar Ayuda

Caso de Uso	**Mostrar Ayuda**
Actores	**Usuario**
Propósito	Mostrar información de las topologías y dispositivos de la red así como del funcionamiento del sistema de manera general.
Resumen	El caso de uso se inicia cuando el usuario accede al sistema a la opción Ayuda en el menú principal y selecciona el tema que desea obtener información: sobre topologías de redes, sobre dispositivos o sobre VIRTUALNET y finaliza con la ejecución de la opción especificada.
Referencia	-
Precondición	-
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario selecciona la opción Ayuda del menú principal.	1.1. Muestra 3 opciones: • Topologías de Redes • Dispositivos de Redes • VIRTUALNET
2. Selecciona la opción que desea realizar.	2.1. Muestra la información concerniente por el usuario.
	3. Finaliza el Caso de Uso.
Poscondiciones	Se muestra la información que el usuario solicitó.
Prioridad	Secundario

Tabla 6 Descripción del Caso de Uso Mostrar Ayuda

Conclusiones Parciales

En este capítulo se crearon las bases en cuanto a la realización del sistema a construir definiéndose:

- Los principales conceptos del área donde trabajará el sistema mediante el modelo de dominio.
- Las funcionalidades que deberá hacer el mismo a través de los requisitos funcionales así como sus características mediante los requisitos no funcionales.
- Los actores, casos de uso del sistema y sus relaciones mediante el diagrama de casos de uso del sistema para una mejor comprensión de sus funcionalidades.

Capítulo 3. Diseño del Sistema

Según Pressman, el diseño es una representación significativa de ingeniería de algo que se va a construir. Se puede hacer el seguimiento basándose en los requisitos del cliente, y al mismo tiempo la calidad se puede evaluar y cotejar con el conjunto de criterios predefinidos para obtener un buen diseño. En el contexto de la ingeniería del software, el diseño se centra en cuatro áreas importantes de interés: datos, arquitectura, interfaces y componentes.

El diseño del software es un proceso iterativo mediante el cual los requisitos se traducen en un plano para construir el software y es tanto un proceso como un modelo, es una secuencia de pasos que hacen posible al diseñador describir como se va a construir el software.

Durante el diseño orientado a objetos**,** se presta especial atención a la definición de los objetos software y en cómo colaboran para satisfacer los requisitos. [32]

Según la Ayuda del Rational, se puede definir como propósitos del diseño:

- Transformar los requerimientos en un diseño de cómo el sistema debe ser.
- Desarrollar la arquitectura del sistema.
- Adaptar el diseño para que se corresponda con el entorno de implementación, diseñando sus funcionalidades.

El diseño no solo se basa en la representación de su modelo de diseño sino también en el desarrollo de la arquitectura del sistema a construir, donde se llevan a cabo actividades más detalladas, como el empleo de un conjunto de estilos arquitectónicos.

3.1. Arquitectura de Software

La Arquitectura del Software, según la IEEE Std 1471-2000, es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución.

En el libro dedicado a la arquitectura de software, Bass y sus colegas identifican tres razones fundamentales por las cuáles es importante la arquitectura:

- Las representaciones de la arquitectura facilitan la comunicación entre todas las partes interesadas en el desarrollo de un sistema basado en computadoras.
- Destaca decisiones tempranas de diseño, teniendo profundo impacto en todo el trabajo de la ingeniería.
- Constituye un modelo relativamente pequeño comprensible de como está estructurado el sistema y como trabajan juntos sus componentes. [33]

Para la realización de la arquitectura del sistema propuesto se seleccionaron los estilos arquitectónicos, los patrones de diseño y se diseñaron los componentes que brindan información a cerca de cómo debe ser desarrollado el mismo. A continuación se explicarán en detalle cada uno de estos aspectos.

3.1.1. Estilos Arquitectónicos

Los estilos arquitectónicos definen un conjunto de reglas de diseño que identifica las clases de componentes y conectores que se pueden utilizar para componer en sistema o subsistema, junto con las restricciones locales o globales de la forma en que la composición se lleva a cabo.

Los estilos arquitectónicos sirven para sintetizar y tener un lenguaje que describa la estructura de las soluciones. Pocos estilos abstractos encapsulan una enorme variedad de configuraciones concretas. También definen los patrones posibles de las aplicaciones y permiten evaluar arquitecturas alternativas con ventajas y desventajas conocidas ante diferentes conjuntos de requerimientos no funcionales. [33]

Estilos Arquitectónicos Empleados en la Solución

Arquitectura Orientada a Objetos

Los componentes de este estilo son los objetos, o más bien instancias de los tipos de dato abstractos. En la caracterización clásica de David Garlan y Mary Shaw, los objetos representan una clase de componentes que ellos llaman managers, debido a que son responsables de preservar la integridad de su propia representación.

Los componentes del estilo se basan en principios OO: encapsulamiento, herencia y polimorfismo. Son asimismo las unidades de modelado, diseño e implementación, y los objetos

y sus interacciones son el centro de las incumbencias en el diseño de la arquitectura y en la estructura de la aplicación.

La principal ventaja de esta arquitectura es que se puede modificar la implementación de un objeto sin afectar a sus clientes. Entre las limitaciones, el principal problema del estilo se manifiesta en el hecho de que para poder interactuar con otro objeto a través de una invocación de procedimiento, se debe conocer su identidad. [33]

Se decidió utilizar este estilo porque cumple con el paradigma orientado a objetos, donde los componentes de un sistema encapsulan los datos y funciones, que se encargan de realizar operaciones sobre ellos, además de la comunicación entre los componentes a través de mensajes.

Arquitectura en Capas

Específicamente el sistema que se propone contendrá 2 capas: una lógica de presentación y otra lógica del negocio. Se decidió utilizar este estilo debido a que permite organizar la estructura lógica de un sistema en capas separadas de responsabilidades distintas y relacionadas, con una separación clara y cohesiva de intereses como que las capas "más bajas" son servicios generales de bajo nivel, y las capas más altas son más específicas de la aplicación. La colaboración y el acoplamiento es desde las capas más altas hacia las más bajas; evitándose el acoplamiento de las capas más bajas a las más altas. [32]

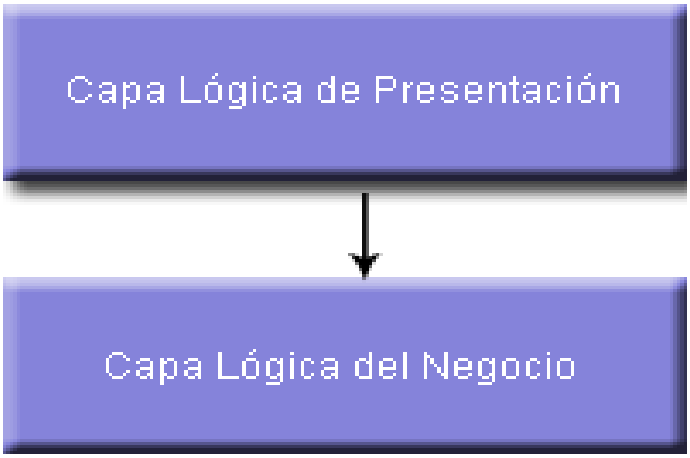


Figura 12 Estructura de la Arquitectura en 2 Capas

Capa Lógica de Presentación

Esta capa es la responsable de todos los aspectos relacionados con la interfaz de usuario de la aplicación. Se comunica únicamente con la capa de negocio y en ella se resuelven cuestiones como:

- Navegabilidad del sistema.
- Formato de los datos de salida: Resolución del formato más adecuado para la presentación de resultados.
- Validación de los datos de entrada, en cuanto a formatos y longitudes máximas.
- Interfaz gráfica con el usuario.

La Capa de Presentación contiene las funcionalidades necesarias para que el usuario intercambie información con la aplicación. Sus principales componentes son los desarrollados a partir del Framework Swing, que posibilita una interfaz más amigable para la interacción con el usuario.

Framework Swing

La JFC (Java Foundation Classes), comprende un grupo de características para ayudar a construir interfaces gráficos de usuario (GUIs). Incluye prácticamente todo tipo de elementos gráficos como botones, paneles, menús y ventanas, con muchas ventajas sobre el AWT.

Swing es una parte de las JFC que permite incorporar en las aplicaciones elementos gráficos de una forma mucho más versátil y con más capacidades que utilizando el AWT básico de Java. [34]

Características de Swing

Amplia variedad de componentes: En general las clases que comiencen por "J" son componentes que se pueden añadir a la aplicación. Por ejemplo: JButton.

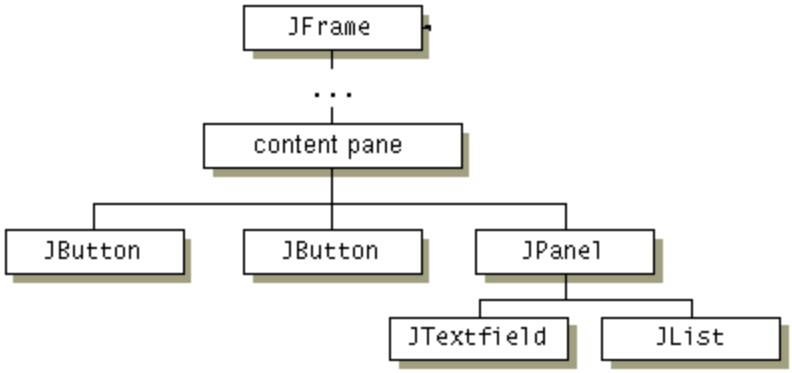


Figura 13 Ejemplo de jerarquía de Clases Swing

Aspecto modificable (look and feel): Se puede personalizar el aspecto de las interfaces o utilizar varios aspectos que existen por defecto (Metal Max, Basic Motif, Window Win32).

Gestión mejorada de la entrada del usuario: Se pueden gestionar combinaciones de teclas en un objeto KeyStroke y registrarlo como componente. El evento se activará cuando se pulse dicha combinación si está siendo utilizado el componente, la ventana en que se encuentra o algún hijo del componente.

Contenedores anidados: Cualquier componente puede estar anidado en otro. Por ejemplo, un gráfico se puede anidar en una lista.

Todas las clases componentes de Swing (clases hijas de JComponent), son hijas de la clase Component de AWT. [35]

Capa Lógica del Negocio

En ella se encuentran todas las clases que implementan las funcionalidades del sistema. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados a través de las clases controladoras de información del sistema. Estas clases permiten disminuir las dependencias entre las clases de la capa de presentación y las de la capa del negocio, lográndose un bajo acoplamiento entre ellas para que, en caso de que se produzca una modificación en alguna, tenga la mínima repercusión posible en el resto.

El uso de este estilo arquitectónico también tiene sus inconvenientes y es que la capa lógica de la aplicación se entrelaza con la interfaz de usuario, entonces no se puede reutilizar con una interfaz diferente, ni distribuirse a otro nodo de proceso. Otro de los problemas es que la lógica más específica de la aplicación se entrelaza con los servicios técnicos o la lógica del negocio, por lo que no se puede reutilizar, distribuir a otro nodo o reemplazar fácilmente con una implementación diferente.

3.2. Patrones de Diseño

Los patrones de diseño son "Soluciones ya probadas y eficaces de los problemas de diseño que pueden expresarse como un conjunto de principios". Una de las principales razones de las ciencias de la computación en cuanto a uso de los patrones de diseño, es la necesidad de obtener soluciones elegantes, simples y reutilizables. [32]

En el diseño existen varios patrones como son los patrones GOF (patrones de la "pandilla de los cuatro", de sus siglas en ingles "Gang of Four") y los patrones GRASP (Patrones Generales del Software para Asignar Responsabilidades).

3.2.1. Patrones GRASP

Los patrones **GRASP** describen principios fundamentales de diseño de objetos para la asignación de responsabilidades que están relacionadas con las obligaciones de un objeto en cuanto a su comportamiento.

En el sistema propuesto se utilizaron algunos de estos patrones los cuales se listan a continuación:

Patrón Experto

Este patrón permite que las responsabilidades sean acorde a la información con que cuenta cada clase. Su uso esta presente en todas las clases del sistema y una de ellas es la clase *TablaRuteo*. A esta clase se le asigna la responsabilidad de eliminar una ruta estática de la tabla de ruteo. Para realizar esta operación se necesitan conocer todas las instancias de *Entradas* estáticas de la tabla y una instancia de *TablaRuteo* las contiene, por lo que, según la

guía del patrón Experto en información, la clase de objeto *TablaRuteo* es adecuada para esta responsabilidad debido a que es la experta en la información.

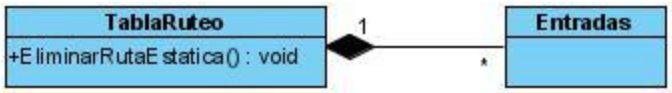


Figura 14 Representación del Patrón Experto en la Solución

Patrón Polimorfismo

Este patrón permite manejar diferentes comportamientos en dependencia del tipo de Objeto. Uno de sus usos esta presente en la clase *ElementoRed* y en las clases que heredan de ella como son: *Conexión*, *Enrutador*, *Subred* entre otras. El número de host de los elementos de la red toma valor diferente de acuerdo al tipo de elemento que sea. Por lo que, según el polimorfismo, se debe asignar esta responsabilidad de determinar el número de host a los diferentes objetos elementos de red, mediante la implementación de una operación polimórfica en la clase padre *ElementoRed* e implementándose de manera diferente en sus clases hijas. De esta manera se garantiza que el comportamiento sea según el tipo, y se evita programar un método que implique sentencias condicionales o el uso de switch {case}.

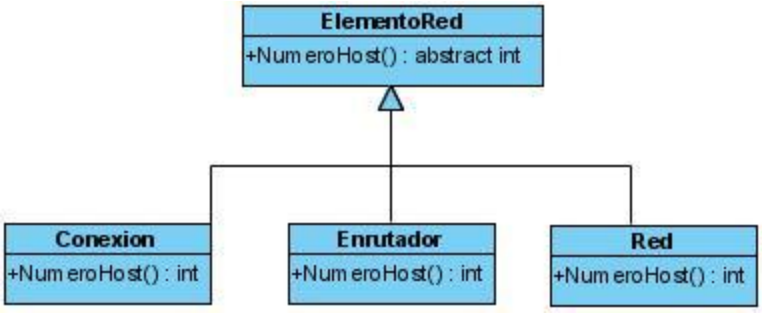


Figura 15 Representación del Patrón Polimorfismo en la Solución

3.2.2. Patrones GOF

Los patrones **GOF** se separan en patrones de creación, estructurales y de comportamiento. Los de creación son los que se encargan de crear los objetos, mientras que los estructurales se dedican al planteamiento de las relaciones entre las clases combinándolas para la formación de

estructuras, finalmente los de comportamiento se dedican a la interacción y cooperación entre las clases, estudiando las relaciones entre los mensajes que ocurren entre los objetos. [36]

Patrón Singleton

Singleton (Instancia única, es un patrón creacional) garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a ella. Su uso está presente en las clases *ListenerElementoRed, ListenerPanelPrincipal* entre otras, para tener acceso desde cualquier parte del código a los métodos de la clase Diagrama y de la Red en general. La clase *ListenerPanelPrincipal,* contiene el diagrama de la red y el panel que se encuentra activo en el sistema, por lo que, se crea una instancia única de la clase para tener acceso a todas las funcionalidades del diagrama, de la red y controlar los eventos de los componentes que la integran.

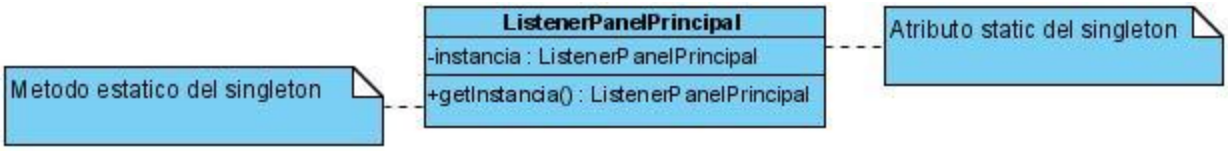


Figura 16 Representación del Patrón Singleton en la Solución

Patrón Strategy

Strategy (Estrategia, es un patrón de Comportamiento) el mismo permite disponer de varios métodos para resolver un problema y elegir cuál utilizar en tiempo de ejecución. Su uso esta presente en la clase Red cuando se va a realizar el proceso de direccionamiento IP. En el direccionamiento se usan dos métodos: el método homogéneo y el método eficiente. Estos métodos dependen del diseño de la red que realice el usuario y de la dirección IP que entre el mismo. En dependencia de estos valores el sistema en tiempo de ejecución debe determinar por cual método direccionar, y mostrarle al usuario la información relacionada con el método utilizado para el direccionamiento.

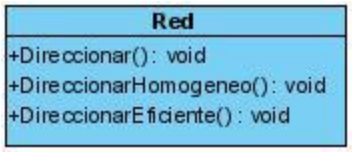


Figura 17 Representación del Patrón Strategy en la Solución

En este flujo de trabajo, el Proceso Unificado define, entre sus artefactos el Modelo de Diseño que contiene varios tipos de diagramas, que incluye los diagramas de interacción, de paquetes, y los de clases.

3.3. Modelo de Diseño

El modelo de diseño es el conjunto de diagramas que describen el diseño lógico. Su propósito es lograr una abstracción de la implementación del sistema. Es usado para concebir un documento del diseño del sistema de software. Es abarcador, compuesto por artefactos que engloban todas las clases del diseño, subsistemas, paquetes, colaboraciones, y las relaciones entre ellos. [32]

3.3.1. Paquetes del Diseño

Los paquetes son una colección de clases, relaciones, realizaciones de casos de usos, diagramas y otros paquetes que son empleados para dividir en partes más pequeñas al modelo de diseño. Estos son utilizados para agrupar un conjunto cohesivo de responsabilidades, elementos del modelo de diseño que se relacionen y como herramienta organizacional. [32]

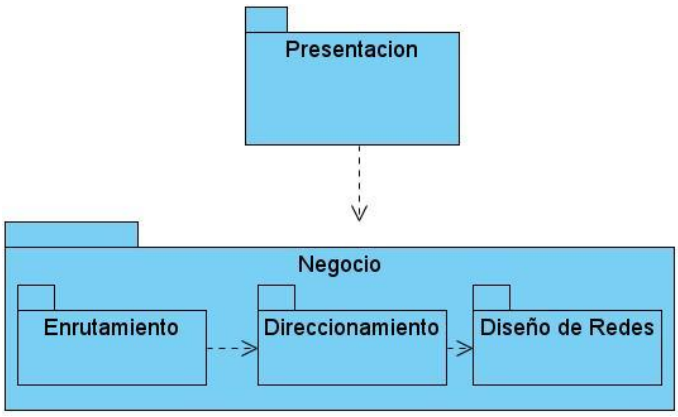


Figura 18 Diagrama de Paquetes del Diseño

Paquete Presentación: Contiene todas las clases y formularios pertenecientes a la capa de presentación.

Paquete Enrutamiento: Contiene todas las clases y sus relaciones que intervienen en el proceso de enrutamiento.

Paquete Direccionamiento: Contiene todas las clases y sus relaciones que intervienen en el proceso de direccionamiento.

Paquete Diseño de Redes: Contiene todas las clases y sus relaciones que intervienen en el proceso del diseño de la red.

3.3.2. Diagramas de Clases del Diseño

Los diagramas de clases del diseño (DCD) representan las especificaciones de las clases e interfaces software en una aplicación. Entre la información general se encuentran: [32]

- Clases, asociaciones y atributos.
- Interfaces, con sus operaciones y constantes.
- Métodos.
- Información acerca del tipo de los atributos.
- Navegabilidad.
- Dependencias.

Diagrama de Clases

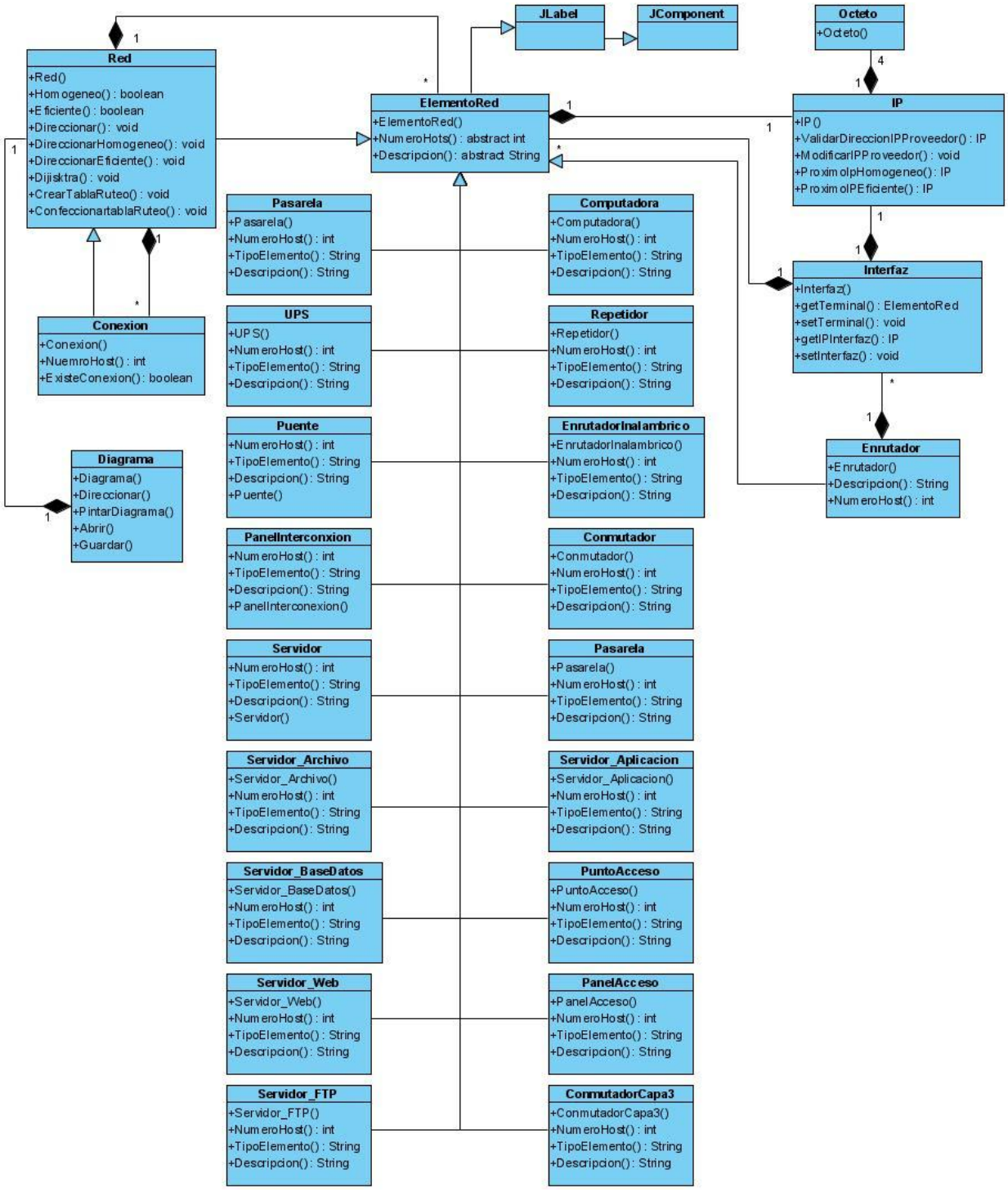


Figura 19 Diagrama de Clases del Paquete Diseño de Redes

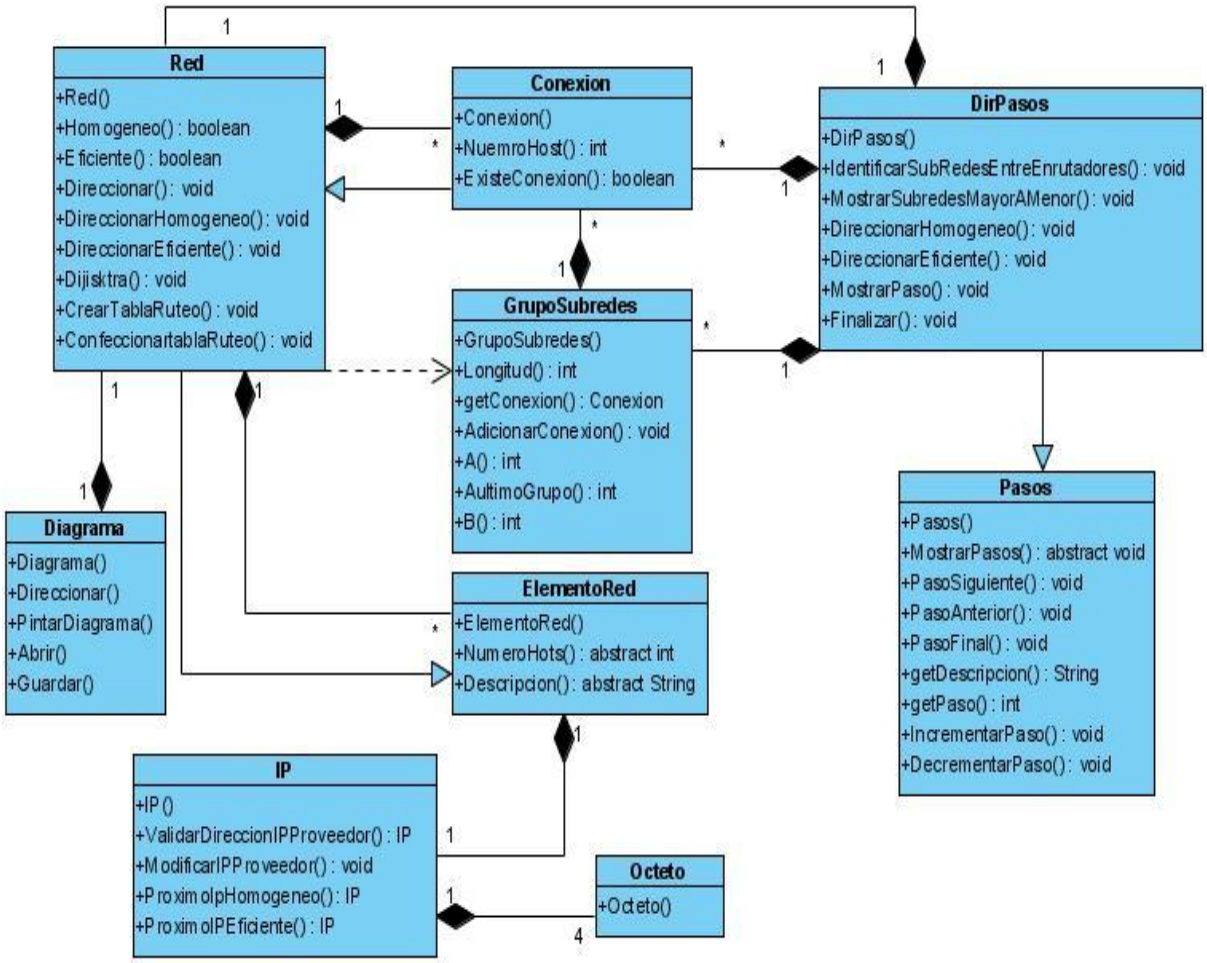


Figura 20 Diagrama de Clases del Paquete Direccionamiento

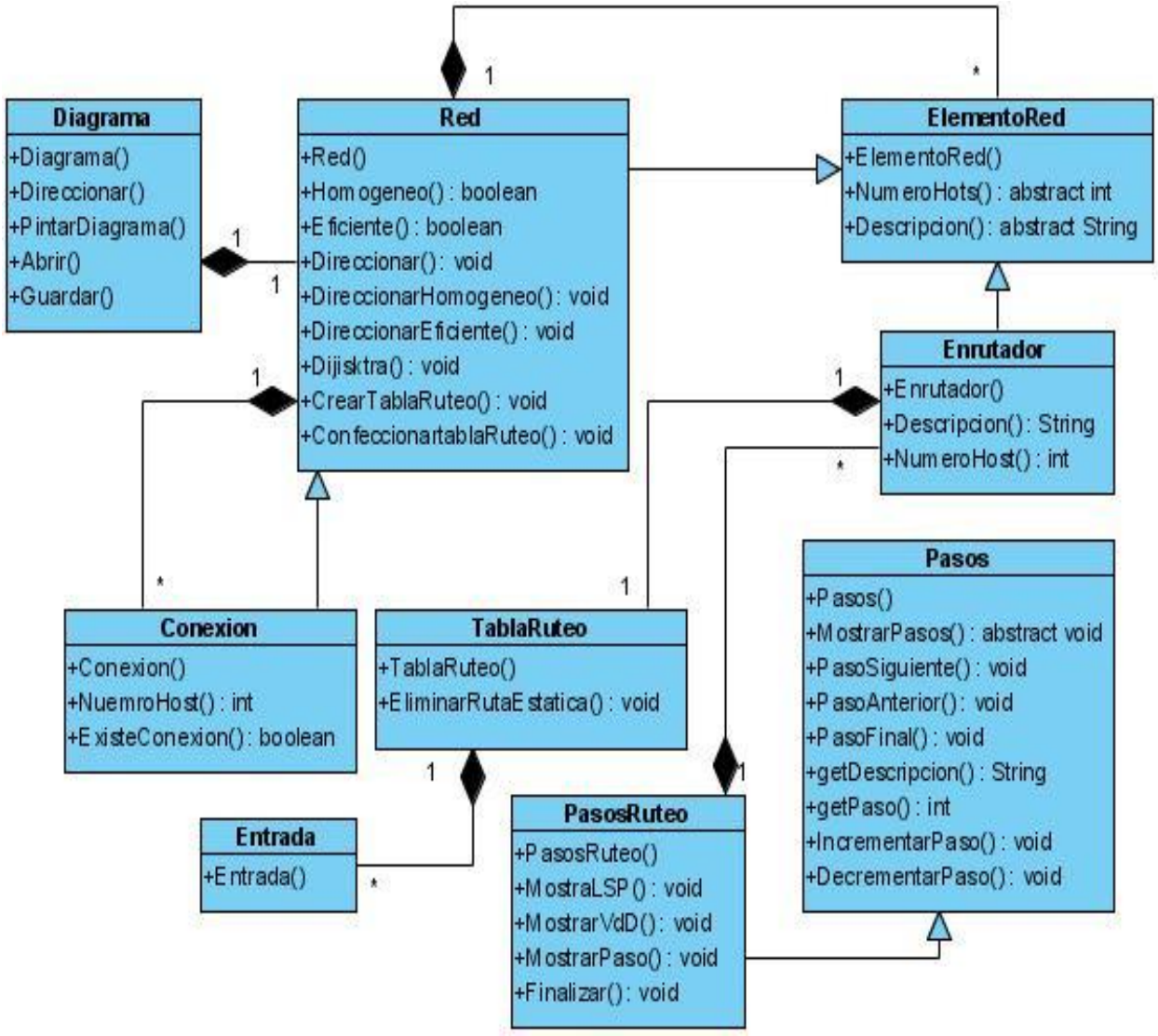


Figura 21 Diagrama de Clases del Paquete Enrutamiento

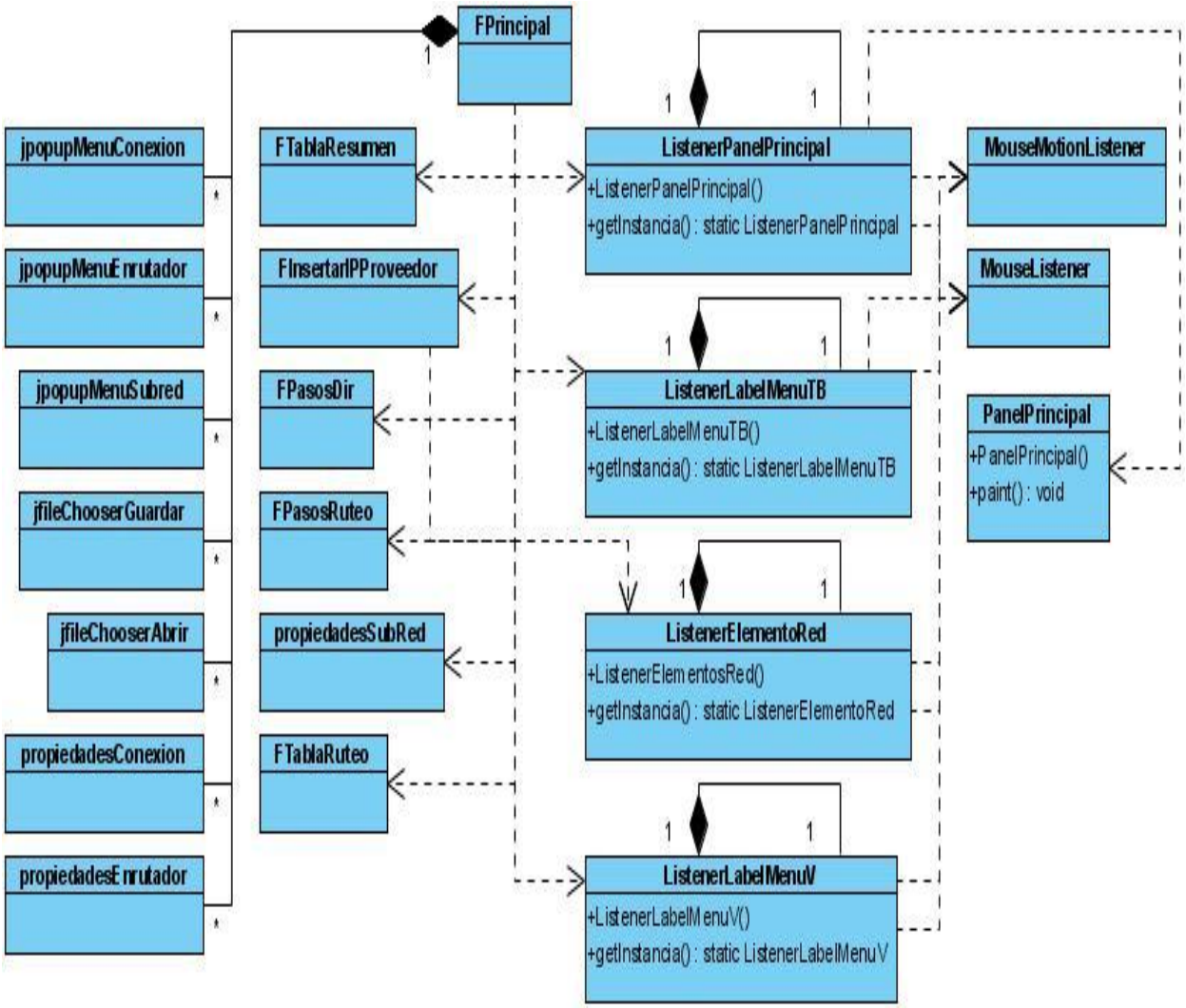


Figura 22 Diagrama de Clases del Paquete Presentación

Clases del Diseño

Las clases del diseño, a diferencia de las clases conceptuales del Modelo del Dominio, muestran las definiciones de las clases software en lugar de los conceptos del mundo real.

Descripción de las Clases del Diseño más Significativas

ListenerPanelPrincipal: Esta clase es la encargada de controlar todos los eventos del mouse que se ejecutan sobre los paneles de los diagramas creados por el usuario.

ListenerElementoRed: Esta clase es la encargada de controlar todos los eventos del mouse que se ejecutan sobre los elementos de la red seleccionados por el usuario en el diagrama de la red.

PanelPrincipal: Esta clase es una clase controladora que hereda de la clase JPanel de swing. Tiene como atributo un Diagrama y se utiliza para tener acceso a todas las clases del dominio. En ella se realizan todas las operaciones que se llevan a cabo sobre el diagrama de la red y es la encargada de mantener actualizada el pintado del diagrama de la red.

Diagrama: Esta clase es una clase controladora. Tiene como atributo una Red y se utiliza para tener acceso a los elementos de la red creada. La misma es la encargada de pintar las conexiones y componentes de la red.

Red: Esta clase es una clase controladora e hija de la clase *ElementoRed*. Tiene como atributo todos los elementos de la red y las conexiones entre dichos elementos. En ella se realizan el proceso de direccionamiento IP y de enrutamiento.

ElementoRed: Esta clase entidad es abstracta, debido a que en ella se encuentran métodos virtuales puros, es decir métodos que no tienen implementación en la clase padre y se comportan de manera diferente en las clases hijas de acuerdo al tipo de Objeto. Contiene además todos los atributos comunes de las clases que heredan de ella como son: *Enrutador*, *Red*, *Conmutador*, etc. reimplementandose en estas últimas los métodos abstractos de la clase padre.

GrupoSubredes: Esta clase entidad contiene todas las conexiones con igual número de host. La misma es utilizada por la clase Red en el direccionamiento IP para separar los grupos de subredes y seguir con el procedimiento. En ella se calculan todas las funcionalidades que se necesitan conocer de las subredes con igual número de host brindándole la información a la clase Red.

TablaRuteo: Esta clase entidad contiene las entradas estáticas y dinámicas del enrutador y realiza todas las funcionalidades que se llevan a cabo sobre la tabla de ruteo de los enrutadores, como modificar, eliminar una ruta etc.

Entrada: Esta clase entidad contiene los campos de la tabla de ruteo como son: ip de la subred destino, la máscara de red correspondiente, el ip de la puerta de enlace del enrutador siguiente o de la subred a la que se le va a hacer la entrega, el ip de la interfaz del enrutador y la métrica. Una instancia de esta clase me representa una ruta de la tabla de ruteo.

Pasos: Esta clase entidad es abstracta, debido a que tiene el método abstracto *MostrarPaso* y es usada en el paso a paso de los procesos de direccionamiento IP y de enrutamiento. De ella heredan las clases *DirPasos* y *RuteoPasos*. En ella se controla el estado actual, el estado final y la descripción del paso que se está realizando, así como el paso siguiente y el paso anterior del proceso que se este efectuando.

DirPasos: Esta clase entidad controla todas las variables que se muestran en el proceso de direccionamiento IP. Y es la encargada de mantener el control de los pasos que se están realizando en el proceso de direccionamiento.

RuteoPasos: Esta clase entidad controla todas las variables que se muestran en el proceso de enrutamiento. Y es la encargada de mantener el control de los pasos que se están realizando en dicho proceso.

3.3.3. Realización de los Casos de Uso – Diseño

La realización de los casos de uso describe cómo se realizan los casos de uso particularmente en el modelo de diseño, en función de los objetos que colaboran.

De manera más precisa, un diseñador puede describir el diseño de uno o más escenarios de un caso de uso; cada uno de estos se denomina una realización del caso de uso. La realización del caso de uso es un término o concepto del Proceso Unificado que se utiliza para la conexión entre los requisitos expresados como casos de uso, y el diseño de objetos que satisface los requisitos. [32]

Diagramas de Interacción

Los diagramas de interacción UML (secuencia y colaboración), son un lenguaje común para ilustrar las realizaciones de los casos de uso.

Los diagramas de interacción comprenden la interacción de mensajes entre objetos software cuyos nombres se inspiran algunas veces en los nombres de las clases conceptuales del Modelo del Dominio, además de otras clases de objetos. [32]

Los diagramas de colaboración y secuencia están basados en la misma información, pero cada uno enfatizando un aspecto particular. Son isomorfos, es decir, se puede convertir de uno a otro sin pérdida de información. Un diagrama de secuencia es un diagrama de interacción que destaca la ordenación temporal de los mensajes; un diagrama de colaboración es un diagrama de interacción que destaca la organización estructural entre los objetos que interactúan. [37]

Diagramas de Secuencia

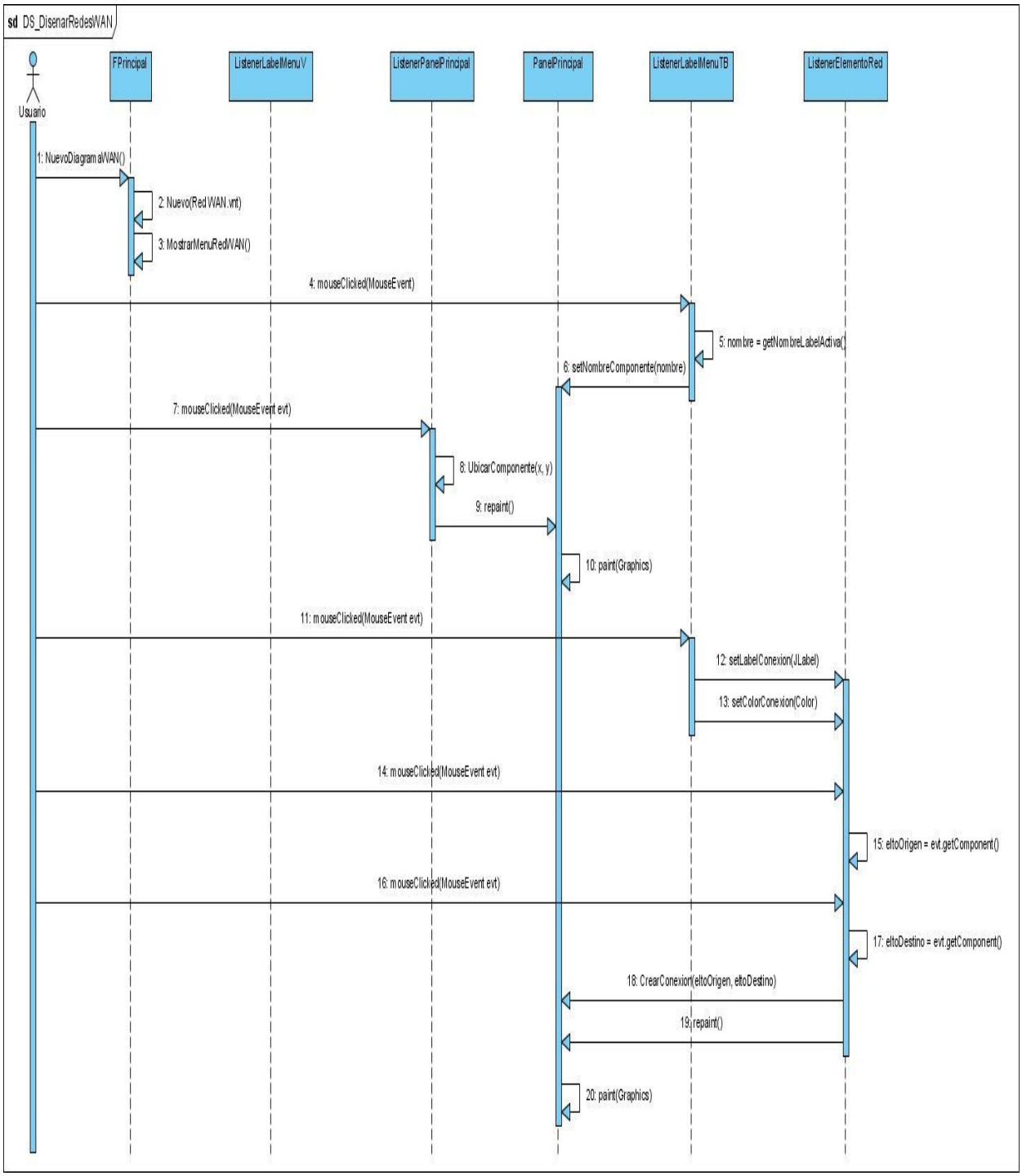


Figura 23 Diagrama de Secuencia del CU Diseñar Redes WAN

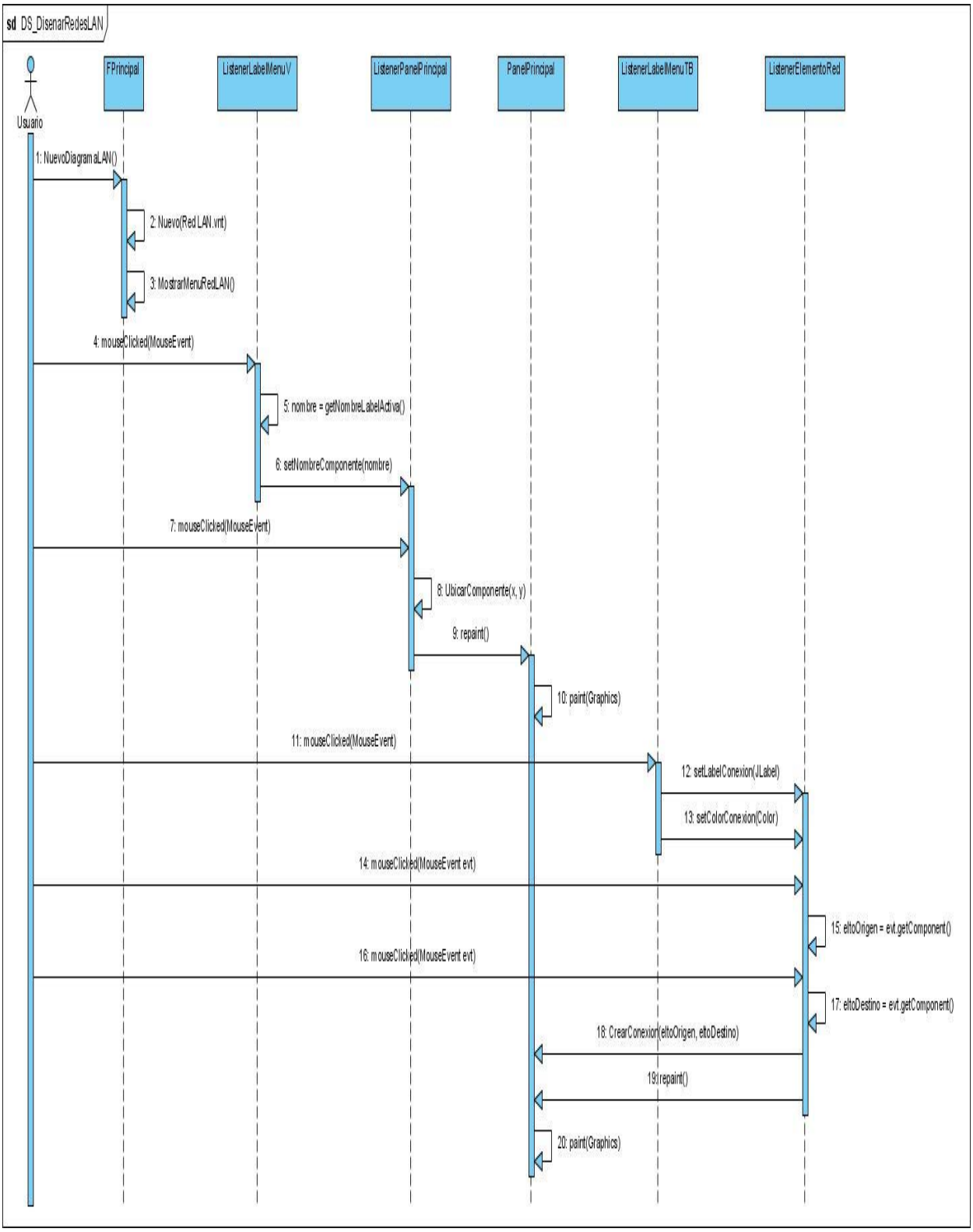


Figura 24 Diagrama de Secuencia del CU Diseñar Redes LAN

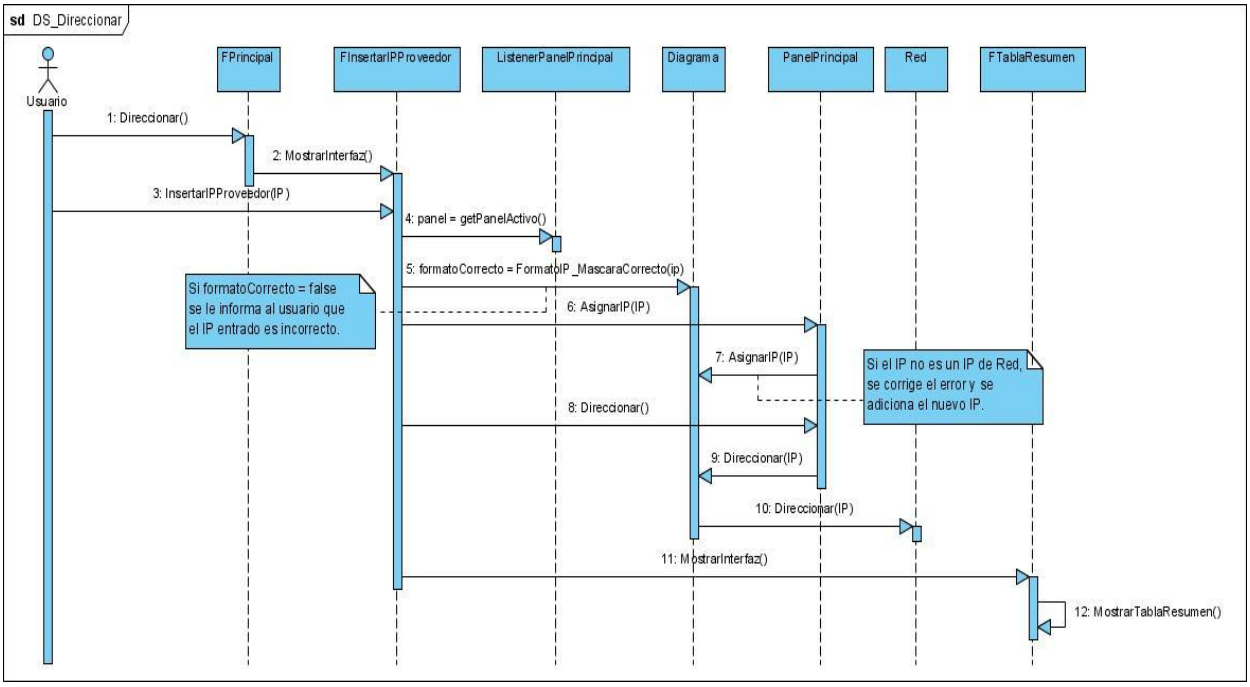


Figura 25 Diagrama de Secuencia del CU Direccionar

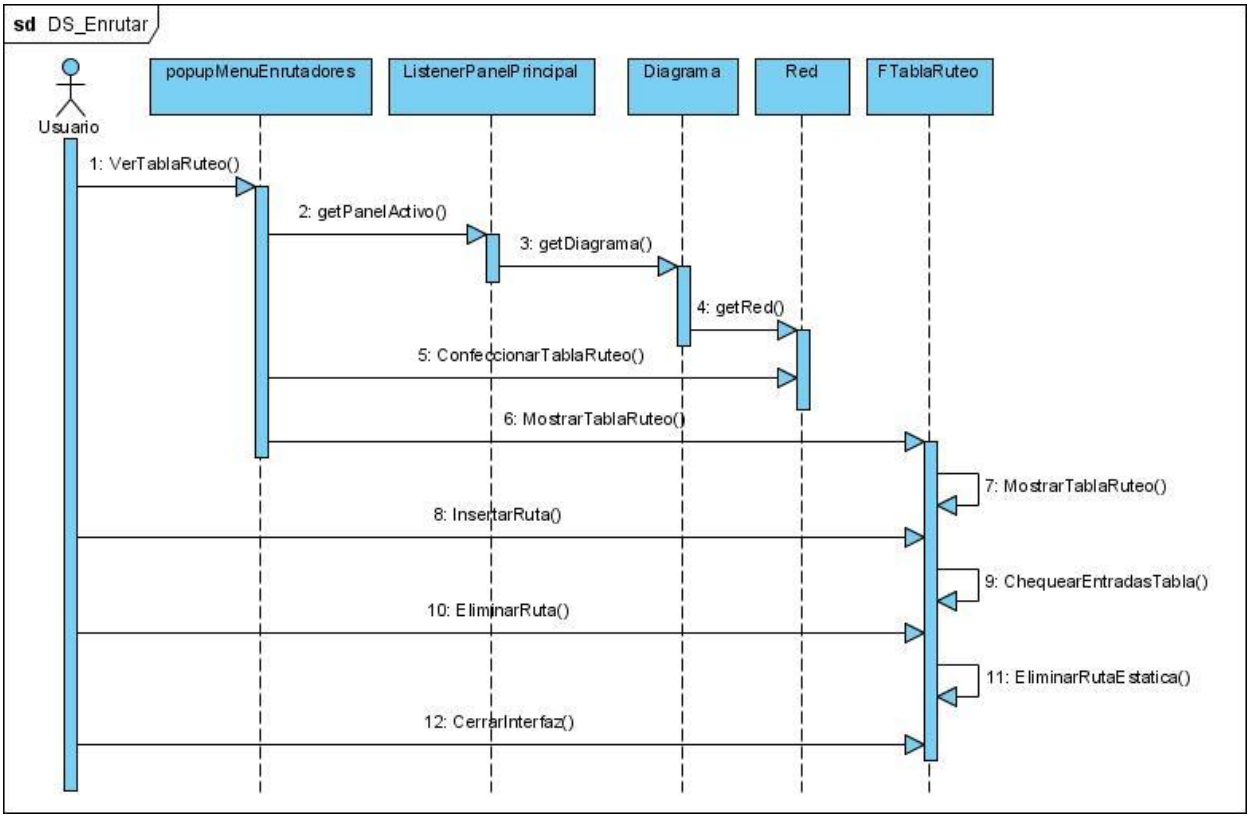


Figura 26 Diagrama de Secuencia del CU Enrutar

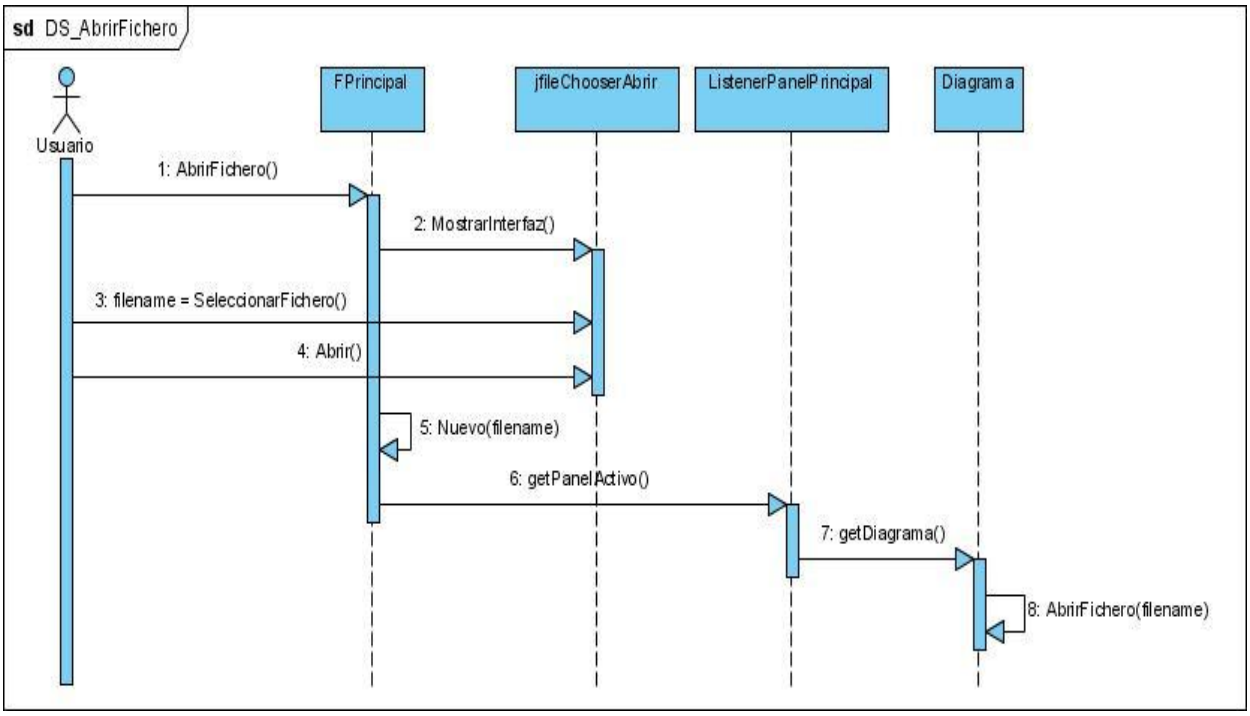


Figura 27 Diagrama de Secuencia del CU Abrir Fichero

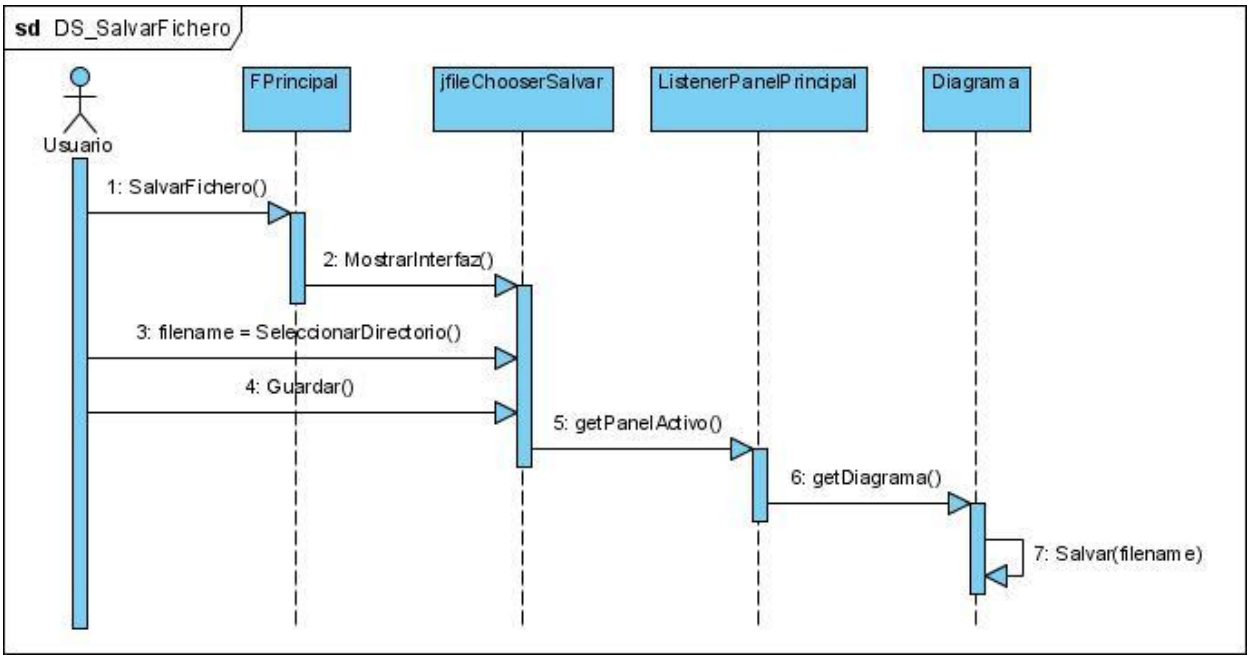


Figura 28 Diagrama de Secuencia del CU Salvar Fichero

3.3.4. Diagrama de Despliegue

Los Diagramas de Despliegue muestran la disposición física de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Para aplicaciones como la que se propone, que se ejecuta en una sola máquina y todos los dispositivos con que se relaciona son los estándares (teclado, mouse), el diagrama de despliegue va a estar constituido por un nodo, por lo que no se considera relevante incluirlo en la investigación.

Conclusiones Parciales

En este capítulo, se definieron los principales artefactos de la etapa del diseño, con el objetivo de dar a conocer como se debe desarrollar el sistema.

- Se definió la arquitectura orientada a objetos y en dos capas, abarcando todas las necesidades de la aplicación, con el objetivo de lograr la menor dependencia posible entre las clases del sistema.
- Se definieron los patrones de diseño que permiten obtener una solución simple y reutilizable.
- Se generaron los artefactos correspondientes al diseño como: clases del diseño, diagrama de clases, los diagramas de interacción (secuencia) sirviendo de entrada a la etapa de implementación.

Capítulo 4. Implementación y Prueba

Este capítulo comienza con el resultado del diseño, implementando el sistema en términos de componentes. Además se hace una revisión final de las especificaciones del diseño y de la codificación mediante la realización de pruebas que garanticen la calidad del software.

4.1. Implementación

El flujo de trabajo de implementación describe cómo los elementos del modelo del diseño se implementan en términos de componentes y cómo estos se organizan de acuerdo a los nodos específicos en el modelo de despliegue.

Los diagramas de despliegue y componentes**,** que son artefactos generados en este flujo de trabajo conforman lo que se conoce como un modelo de implementación al describir los componentes a construir y su organización y dependencia entre nodos físicos en los que funcionará la aplicación. [38]

4.1.1. Diagrama de Componentes

Los diagramas de componentes representan un conjunto de componentes que se relacionan entre si mediante dependencias, estos pueden ser ejecutables, librerías, ficheros de código, una tabla de base de datos, un documento etc. [32]

Los componentes que conforman el sistema propuesto están agrupados por paquetes como se muestra a continuación.

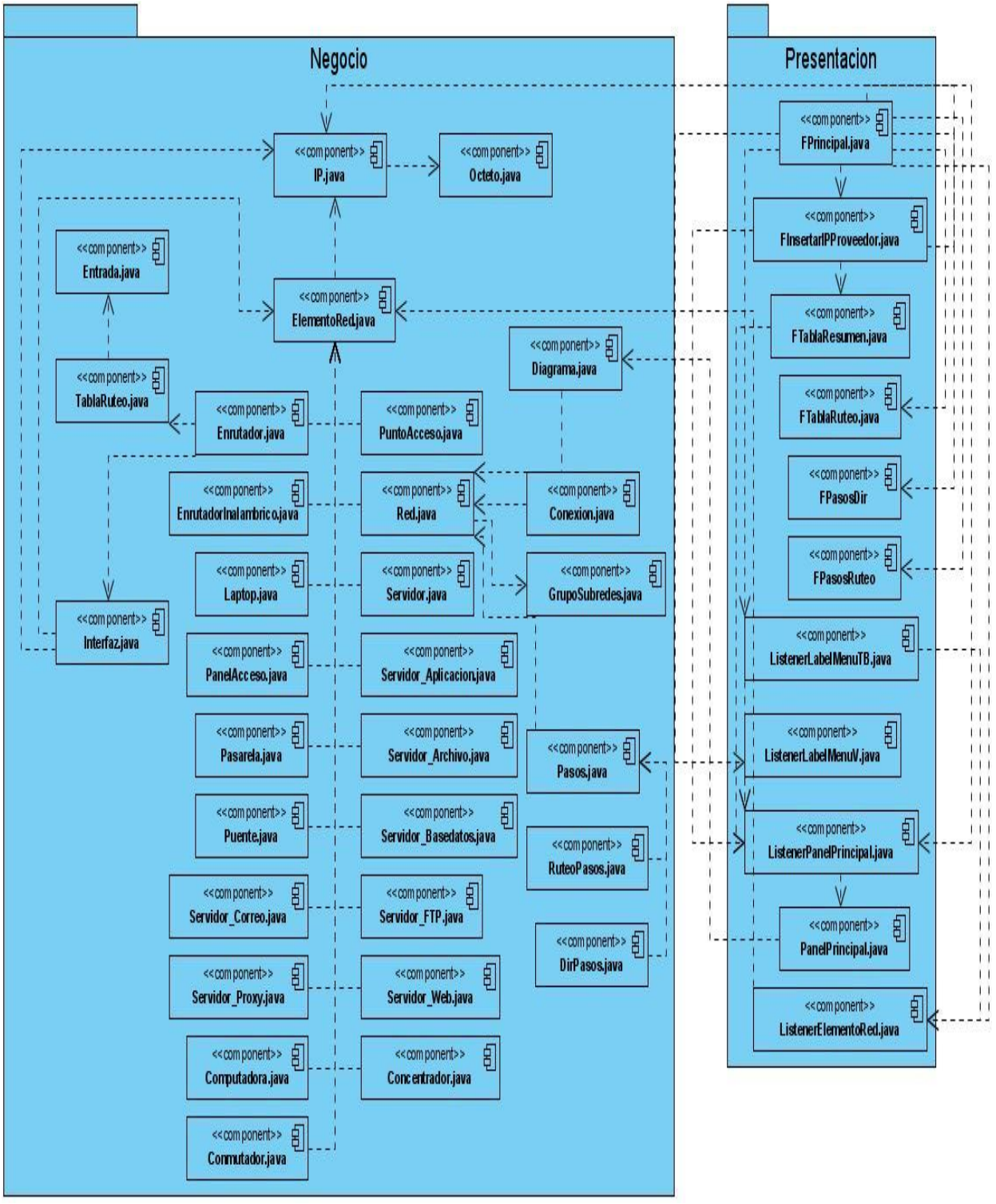


Figura 29 Diagrama de Componentes

4.2. Estándar de Codificación

Los estándares de codificación son las reglas, pautas de cómo va a quedar estructurado el código fuente. El uso de un estándar de codificación trae grandes ventajas desde el punto de vista de la programación, entre ellas se encuentran las siguientes: [31]

- Asegurar la legibilidad del código entre distintos programadores, facilitando la depuración del mismo.
- Proveer una guía para el encargado de mantenimiento/actualización del sistema, con código claro y bien documentado.
- Facilitar la portabilidad entre plataformas y aplicaciones.
- Facilidad a la hora de entender el código de otro desarrollador que aplica el mismo estándar.

Debido a esto, en la implementación del sistema propuesto, se ha seguido una serie de requisitos detallados a continuación.

Operadores Binarios:

Deben de incluirse espacios en ambos lados de los operadores.

Ejemplo: A = d + 5

Operadores Unitarios:

Los operadores unarios (++, --, etc.) deben ponerse junto a su operando.

Ejemplo: v++, m --

Nombre de Variables:

Los nombres de las variables locales, como los iteradores o los contadores, pueden especificarse en minúscula. Si el nombre de la variable contiene más de una cadena, comienza con minúscula y luego la próxima cadena con la primera letra en mayúscula y las demás en minúsculas. Siempre y cuando su lectura sea legible.

Ejemplo: **int** cont, **int** i, **int** posElemento.

Identación:

Utilizar el formato que contiene el IDE empleado, en este caso el Netbeans 6.0.

```
public class NewClass {
    public void Metodo(){
    }
}
```

```
for(int i = 0;i < cant;i++){
    for(int j = 0;j < cant;j++){
    }
}
```

```
if(cant == 0){
}
else{
}
```

Comentarios:

Utilizar el estándar // sólo para comentarios de una sola línea en caso de que sean varias utilizar el estándar /* */.

Declaración de Clases:

La declaración de las clases debe tener el formato del IDE empleado, en este caso Netbeans 6.0. El nombre de la clase debe comenzar con una letra mayúscula. Si el nombre está compuesto por varias cadenas, cada una debe comenzar con letra mayúscula.

Declaración de Atributos:

Los atributos de las clases deben de comenzar con letra inicial minúscula y reflejar verdaderamente la propiedad a la que representa. Si está compuesto por varias cadenas, la primera comienza con letra minúscula y la segunda con letra mayúscula.

Ejemplo: nombrePersona, edad, sexo, color

Declaración de Funciones:

Las funciones pueden o no comenzar con letra inicial mayúscula.

Ejemplo: Insertar() o insertar().

Interfaces Visuales:

En cuanto a la interfaces visuales, las que interactúan con el usuario, nombrarlas comenzando por la palabra F seguida de un nombre. Los demás componentes visuales se llamarán normalmente por el nombre que le asigna el IDE empleado comenzando por la letra j.

Ejemplo:

Para los componentes JFrame que son los formularios: FPrincipal, FTablaRuteo, etc. Para los restantes componentes dejar que el IDE les asigne el estándar que lleva incluido.

4.3. Prueba

La IEEE [IEEE90] define las pruebas como una actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, se observan o almacenan los resultados y se realiza una evaluación de algún aspecto del sistema o componente.

Cuando se habla de condiciones específicas en la definición anterior, se puede suponer la presencia de una especie de ambiente de operación de la prueba, para el cual deben existir determinados valores para las entradas y las salidas, así como también ciertas condiciones que delimitan a dicho ambiente de operación.

Formalmente esto es conocido como caso de prueba. De acuerdo a la IEEE [IEEE90] un caso de prueba es un conjunto de entradas, condiciones de ejecución y resultados esperados diseñados para un objetivo particular.

La prueba de software es un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones del diseño y de la codificación.

Existen distintas técnicas de pruebas que proporcionan criterios para generar casos de pruebas que provoquen fallos en los programas, estas técnicas se agrupan en: **técnicas de caja blanca o estructurales** y **técnicas de caja negra o funcionales,** la primera se basa en un minucioso examen de los detalles procedimentales a evaluar, por lo que es necesario conocer la lógica del programa, sin embargo la segunda se basa en la realización de pruebas sobre la interfaz del programa, entiéndase por interfaz las entradas y salidas de dicho programa. No es necesario conocer la lógica del programa, únicamente la funcionalidad que debe realizar. [40]

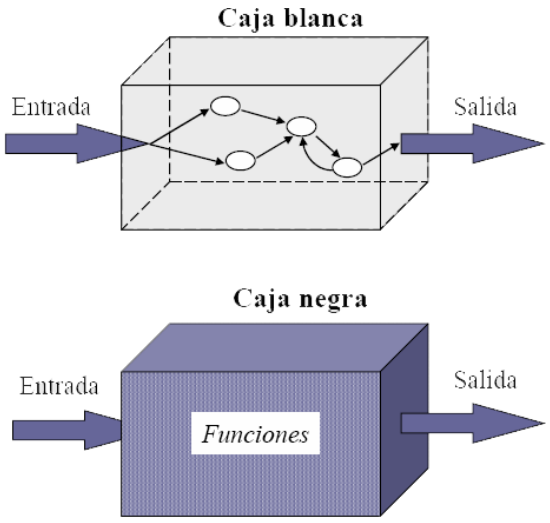


Figura 30 Representación de Pruebas de Caja Blanca y Pruebas de Caja Negra

Al sistema propuesto se le realizaron pruebas de caja negra y pruebas de caja blanca con el objetivo de medir la funcionalidad operativa y los caminos lógicos del software.

4.4. Prueba de Caja Negra

Las pruebas de caja negra, también denominadas de comportamiento, se centran en los requisitos funcionales del software. Estas pruebas le permiten al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. La prueba de caja negra intenta encontrar errores de las siguientes categorías: [40]

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a bases de dato externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

A diferencia de las pruebas de caja blanca, que se basan en la lógica interna del software, las pruebas de caja negra se concentran en su funcionalidad, por lo que mucho del trabajo se realiza interactuando con la interfaz del software. Los casos de prueba generados en este enfoque, se diseñan a partir de valores de entrada y salida. De esta forma, se puede determinar la validez de una salida para un conjunto de entradas proporcionadas.

4.4.1. Pruebas de Caja Negra efectuadas al Sistema Propuesto

Nombre del Caso de Uso	**Diseñar Redes**	
Nombre del Caso de Prueba	**Diseñar Redes LAN**	
Entrada	**Resultados**	**Condiciones**
Se seleccionó la opción crear un nuevo diagrama LAN.	Se mostró el menú de elementos de la red correspondiente.	Se debe seleccionar la opción nuevo diagrama LAN.
Se seleccionaron los elementos que formarán parte de la red.	Se mostró el elemento de red en el panel principal, mostrándose una descripción del elemento pasándole el mouse por encima al elemento.	
Se seleccionaron los tipos de cableados para establecer las conexiones entre los componentes seleccionados.	Pintó las conexiones con el color del cable seleccionado entre los componentes de la red.	
Nombre del Caso de Prueba	**Diseñar Redes WAN**	
Se seleccionó la opción crear un nuevo diagrama WAN.	Se mostró el menú de elementos de la red correspondiente.	Se debe seleccionar la opción nuevo diagrama WAN.
Se seleccionaron los elementos que formarán parte de la red.	Se mostró el elemento de red en el panel principal, mostrándose una descripción del elemento pasándole el mouse por encima al elemento.	
Se seleccionaron los tipos de cableados para establecer las conexiones entre los componentes seleccionados.	Pintó las conexiones con el color del cable seleccionado entre los componentes de la red.	

Tabla 7 Caso de Prueba del Caso de Uso Diseñar Redes

Nombre del Caso de Uso	**Direccionar**
Nombre del Caso de Prueba	**Direccionar por el método Homogéneo**

Entrada	Resultados	Condiciones
Se diseñó una red WAN y se seleccionó la opción Direccionar Automáticamente.	Se mostró una interfaz para entrar la dirección IP con la que se va a realizar el direccionamiento de la red. Se tuvo en cuenta que la dirección IP fuese acorde con la red para que diera el resultado esperado.	Se debe diseñar una red WAN.
Se seleccionó la opción Aceptar.	Se mostró una tabla con las direcciones IP de cada una de las subredes, la dirección IP a partir de la cual se direccionó y el método de direccionamiento utilizado. En este caso se mostro “Método Homogéneo”.	
Nombre de Caso de Prueba	**Direccionar por el método Eficiente**	
Se diseñó una red WAN y se seleccionó la opción Direccionar Automáticamente.	Se mostró una interfaz para entrar la dirección IP con la que se va a realizar el direccionamiento de la red. Se tuvo en cuenta que la dirección IP fuese acorde con la red para que diera el resultado esperado.	Se debe diseñar una red WAN.
Se seleccionó la opción Aceptar.	Se mostró una tabla con las direcciones IP de cada una de las subredes, la dirección IP a partir de la cual se direcciono y el método de direccionamiento utilizado. En este caso se mostro “Método Eficiente”.	

Nombre del Caso de Prueba	**Direccionar Paso a Paso**	
Se seleccionó la opción Direccionar Paso a Paso.	Se mostró una interfaz para entrar la dirección IP con la que se va a realizar el direccionamiento de la red. Se tuvo en cuenta que la dirección IP fuese acorde con la red para que diera el resultado esperado.	Se debe diseñar una red WAN.
Se seleccionó la opción Aceptar.	Se mostró en cada paso el valor esperado, tanto por el método homogéneo como por el eficiente.	

Tabla 8 Caso de Prueba del Caso de Uso Direccionar

Nombre del Caso de Uso	**Enrutar**	
Nombre del Caso de Prueba	**Enrutar Dinámicamente**	
Entrada	**Resultados**	**Condiciones**
Se diseñó una red WAN y se seleccionó la opción Ver Tabla de Ruteo en el popupMenu del Enrutador al que se le quería configurar la tabla de ruteo.	Se mostró la tabla de ruteo del enrutador seleccionado con las menores rutas para llegar desde el a cada una de las subredes de la red.	Se debe direccionar previamente y los enrutadores deben estar conectados a la red.
Nombre del Caso de Prueba	**Eliminar Ruta de la Tabla de Ruteo**	
Se seleccionó una ruta estática de la tabla de ruteo y la opción Eliminar Ruta.	Se elimino la ruta de la interfaz visual y de la tabla de ruteo del enrutador.	Debe haber rutas estáticas en la tabla de ruteo.
Se seleccionó una ruta dinámica de la tabla de ruteo y la opción Eliminar Ruta.	No se eliminó la ruta de la interfaz, ni de la tabla de ruteo del enrutador.	Se debe confeccionar la tabla de ruteo de forma dinámica.
Nombre de Caso de Prueba	**Enrutar Paso a Paso por el algoritmo VdD**	
Se diseñó una red WAN y se seleccionó la opción Enrutar	Se mostró una interfaz con la imagen del diseño de la red y el	Se debe direccionar previamente y los

paso a paso por el algoritmo de VdD.	intercambio de información de las entradas del VdD de cada enrutador de la red.	enrutadores deben estar conectados a la red.
Nombre del Caso de Prueba	**Enrutar Paso a Paso por el algoritmo Estado de Enlaces**	
Se diseñó una red WAN y se seleccionó la opción Enrutar paso a paso por el algoritmo Estado de Enlaces.	Se mostró una interfaz con todos los enrutadores para que el usuario seleccione a cual de ellos desea crear el mapa global de la red. En cada paso del proceso se obtuvieron los resultados esperados.	Se debe direccionar previamente y los enrutadores deben estar conectados a la red.

Tabla 9 Caso de Prueba del Caso de Uso Enrutar

Nombre del Caso de Uso	**Gestionar Ficheros**	
Nombre del Caso de Prueba	**Abrir Fichero**	
Entrada	**Resultados**	**Condiciones**
Se seleccionó la opción Abrir Fichero.	Se mostró la interfaz en la cual se buscó el fichero que se quería abrir. Se seleccionó el fichero y se presionó la opción Abrir. Se cargaron los datos del fichero seleccionado en el sistema.	El fichero debe ser valido, debe tener extensión .vnt
Nombre del Caso de Prueba	**Salvar Fichero**	
Se seleccionó la opción Guardar Fichero.	Se mostró una interfaz en la cual se buscó el directorio donde se querían guardar los datos de la red que se encontraban en el sistema.	Debe haberse diseñado la red en el sistema.

	Se seleccionó la opción Guardar y se guardaron los datos de la red en el directorio especificado satisfactoriamente.	

Tabla 10 Caso de Prueba del Caso de Uso Gestionar Fichero

4.5. Pruebas de Caja Blanca

La prueba de caja blanca, es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para obtener los casos de prueba. Mediante los métodos de prueba de caja blanca, el ingeniero del software puede obtener casos de prueba que garanticen que se ejerciten por lo menos una vez todos los caminos independientes de cada módulo, todas las decisiones lógicas en sus vertientes verdaderas y falsas; que se ejecuten todos los ciclos en sus límites y con sus límites operacionales, y las estructuras internas de datos para asegurar su validez. Con este método se determinan cuáles son los casos de prueba a partir del código fuente del software y se utilizan las especificaciones para determinar el resultado esperado del caso.

Una de las técnicas de pruebas de caja blanca es el método del camino básico. El método del camino básico permite al diseñador de casos de prueba obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución. Se basa en construir un caso de prueba por camino básico que se encuentre en el grafo del programa asociado al método de la clase que se desea someter a pruebas. Para aplicar la técnica del camino básico se debe:

- Enumerar las instrucciones del algoritmo al que se le realizará la prueba.
- Presentar el grafo de flujo del algoritmo.
- Determinar la Complejidad Ciclomática y los caminos independientes del grafo de flujo.
- Diseñar los casos de pruebas para cada camino.

Un **Grafo de Flujo** está formado por 3 componentes fundamentales que ayudan a su elaboración, comprensión y brinda información para confirmar que el trabajo se está haciendo adecuadamente. Los componentes son:

- Nodos

- Aristas
- Regiones

La **Complejidad Ciclomática** es una métrica del software que proporciona una medición de la complejidad lógica de un programa.

Hay tres formas fundamentales de calcular la complejidad:

1. V (G) = número de regiones del grafo de flujo.

2. V (G) = A - N + 2

Donde: A es el número de aristas del grafo y N es el número de nodos.

3. V (G) = P + 1

Donde: P es el número de nodos predicado (nodo del cual salen varias aristas) contenidos en el grafo G. [41]

4.5.1. Pruebas de Caja Blanca efectuadas al sistema propuesto utilizando la técnica del Camino Básico

```
    public void Direccionar(IP ipProveedor){
        setIP(ipProveedor);
        pasosDir.proveedor = ipProveedor;
1       boolean esPosibleDireccionar = EsPosibleDireccionar();
        pasosDir.esPosibleDireccionar = esPosibleDireccionar;

2       if(esPosibleDireccionar){
3           if(Homogeneo()){
                tipoDirecionamiento = 0;
4               pasosDir.setPasoFinal(2);
                pasosDir.homogeneo = true;
                DireccionaHomogeneo();
            }
5           else if(Eficiente()){
                tipoDirecionamiento = 1;
6               pasosDir.setPasoFinal(3);
                pasosDir.eficiente = true;
                DireccionaEficiente();
            }
        }
7       else {
            tipoDirecionamiento = -1;
8           InicializarDireccionesIP();
            seDirecciono = false;
            pasosDir.setPasoFinal(1);
            pasosDir.homogeneo = false;
            pasosDir.eficiente = false;
        }

9       pasosDir.conexiones = conexiones;
10
```

Figura 31 Algoritmo Direccionar

Grafo de Flujo del método Direccionar

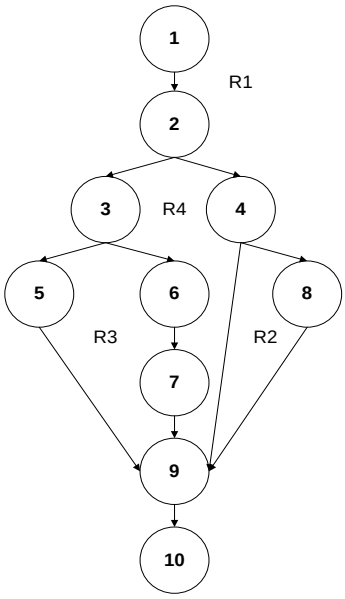


Figura 32 Grafo de Flujo del método Direccionar del CU Direccionar

Complejidad Ciclomática y Caminos Independientes.

V (G) = 4, V (G) = 12 – 10 + 2 = 4, V (G) = 3 + 1 = 4

Caminos Independientes: 1-2-3-4-9-10, 1-2-3-5-6-9-10, 1-2-7-9-10, 1-2-7-8-9-10

Casos de Prueba

Camino: 1-2-3-4-9-10

- **Caso de Uso**: Direccionar.
- **Caso de Prueba**: Direccionar Homogéneo.
- **Entrada**: Seleccionar la opción Direccionar. El sistema muestra una interfaz para entrar la dirección IP a partir de la cual se va a direccionar y se selecciona la opción Aceptar.
- **Resultados**: Se direccionó por el método homogéneo, mostrando una tabla resumen indicando las direcciones IP de cada subred, la dirección IP a partir de la cuál se realizó el direccionamiento y el método por el cuál se direccionó, en este caso muestra “Método Homogéneo”.

- **Condiciones**: Se debe diseñar una red WAN y entrar una dirección IP con la cuál se pueda direccionar por el método homogéneo.

  Camino: 1-2-3-5-6-9-10

- **Caso de Uso**: Direccionar
- **Caso de Prueba**: Direccionar Eficiente.
- **Entrada**: Seleccionar la opción Direccionar. El sistema muestra una interfaz para entrar la dirección IP a partir de la cual se va a direccionar y se selecciona la opción Aceptar.
- **Resultados**: Se direccionó por el método eficiente, mostrando una tabla resumen indicando las direcciones IP de cada subred, la dirección IP a partir de la cuál se realizó el direccionamiento y el método por el cuál se direccionó, en este caso muestra "Método Eficiente".
- **Condiciones**: Se debe diseñar una red WAN y entrar una dirección IP con la cual se pueda direccionar por el método Eficiente.

  Camino: 1-2-7-9-10

- **Caso de Uso**: Direccionar.
- **Caso de Prueba**: Verificar si es posible Direccionar.
- **Entrada**: Se seleccionó la opción Direccionar y se entró una dirección IP que no se correspondía con el diseño de la red, no alcanzaban los bits de host para realizar el direccionamiento.
- **Resultados**: No se puedo direccionar a partir de la dirección IP de entrada, mostrando un mensaje de error diciendo que no se pudo realizar el direccionamiento.
- **Condiciones**: Se debe diseñar una red WAN y entrar una dirección IP con la cual no alcancen los bits de host para realizar el direccionamiento de la red.

  Camino: 1-2-7-8-9-10

- **Caso de Uso**: Direccionar
- **Caso de Prueba**: Inicializar direcciones IP de los elementos de la red.
- **Entrada**: Se seleccionó la opción Direccionar y se entró una dirección IP.
- **Resultados**: No se pudo direccionar y se inicializaron las direcciones IP de los elementos de la red.

- **Condiciones**: Se debe haber direccionado al menos una vez y la nueva dirección IP entrada por el usuario para direccionar nuevamente debe indicar que no es posible realizar el direccionamiento.

```
1  public void ConfeccionarTablaRuteo(){
       for(int i = 0;i < CantidadElementos();i++){
           ElementoRed e = getElementoRed(i);
2          if(e instanceof Enrutador){
3              pasosRuteo.setEnrutador((Enrutador)e);
               Dijikstra((Enrutador) e);
           }
4      }
   }
```

Figura 33 Algoritmo Confeccionar Tabla Ruteo.

Grafo de Flujo del método Confeccionar Tabla de Ruteo

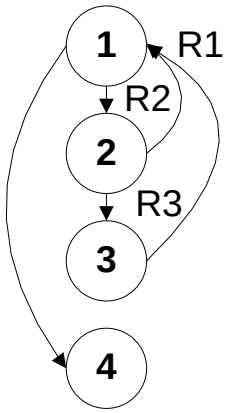


Figura 34 Grafo de Flujo del método Confeccionar tabla de Ruteo del CU Enrutar

Complejidad Ciclomática y Caminos Independientes

V (G) = 3, V (G) = 5 – 4 + 2 = 3, V (G) = 2 + 1 = 3

Caminos Independientes: 1-2-3-1-4, 1-2-1-4, 1-4

Casos de Prueba

Camino: 1-2-3-1-4

- **Caso de Uso**: Enrutar

- **Caso de Prueba**: Confeccionar Rutas Dinámicas.
- **Entrada**: Seleccionar la opción ver tabla de ruteo en el enrutador. En el sistema esta la red completamente creada y direccionada.
- **Resultados**: Se crean las tablas de ruteo de todos los enrutadores conectados a la red y se muestra la tabla del enrutador con las rutas creadas de manera dinámica.
- **Condiciones**: Se debe direccionar previamente.

  Camino: 1-2-1-4

- **Caso de Uso**: Enrutar
- **Caso de Prueba**: Enrutar sin haber enrutadores en el sistema.
- **Entrada**: Seleccionar opción enrutar por el algoritmo Estado de Enlaces o Vector – Distancia paso a paso.
- **Resultados**: No se crearon las tablas de ruteo.
- **Condiciones**: No existir enrutadores en el sistema.

  Camino: 1-4

- **Caso de Uso**: Enrutar
- **Caso de Prueba**: Seleccionar la opción enrutar sin haber elementos de red en el sistema.
- **Entrada**: Seleccionar la opción enrutar por el algoritmo Estado de Enlaces o Vector – Distancia paso a paso.
- **Resultados**: En el sistema no existen elementos de redes. Por lo que no se pueden crear las tablas de ruteo, mostrándose un mensaje de error informando al usuario que no existe la red en el sistema para realizar esa operación.
- **Condiciones**: Se debe haber creado un diagrama WAN.

Conclusiones Parciales

En este capítulo se abordaron los temas correspondientes a la implementación y pruebas:

- Se definió la estructura del diagrama de componentes para conformar el modelo de implementación.

- Se propone un estándar de codificación que posee la misma estructura del IDE seleccionado, facilitando la actualización y mejoramiento del sistema en años posteriores.
- Se realizaron pruebas de caja blanca y de caja negra de los casos de uso más críticos del sistema a partir de los ejercicios de las clases prácticas, trabajos de control, obteniéndose resultados favorables y demostrando el cumplimiento de los requisitos funcionales.

Conclusiones

Con la realización de este trabajo se logró realizar una herramienta de simulación de redes de computadoras que integra los contenidos de diseño de redes, direccionamiento IP y de enrutamiento ejecutándose estos últimos procesos a partir del diseño de una red WAN. Lográndose así el cumplimiento de los objetivos y tareas planteadas debido a que:

- Se realizó un estudio previo de los principales conceptos impartidos en Teleinformática correspondientes al diseño, direccionamiento y enrutamiento de la red. Se seleccionó la metodología de desarrollo de software, el lenguaje de programación y el IDE, quedando todo listo para comenzar el desarrollo.
- Se definieron las características que debe cumplir el sistema mediante los requisitos funcionales y no funcionales, así como los casos de uso que guiarán el proceso de desarrollo.
- Se implementó una herramienta de simulación de redes de computadoras siguiendo la arquitectura en dos capas, haciendo uso de patrones de diseño, y cumpliendo con el objetivo planteado.
- Se le realizaron pruebas a la herramienta, tomándose como juegos de datos los ejercicios de las clases prácticas correspondientes al contenido de direccionamiento IP y enrutamiento. Las mismas fueron satisfactorias, lográndose de esta manera el cumplimiento de los requisitos exigidos por el cliente y la solución automatizada.

Recomendaciones

Para darle continuidad e incrementar las prestaciones de la solución propuesta se recomienda:

- Incorporar nuevos elementos de redes y tipos de cableados para el diseño lógico de redes LAN.
- Realizar el proceso de direccionamiento IP a partir de más de una dirección IP.
- Extender el proceso de direccionamiento IP al diseño de redes LAN.
- Incorporar la funcionalidad de permitir conocer las características de la dirección IP entrada por el proveedor de servicios para el direccionamiento IP.

Bibliografía

1. **Martínez, Eduardo Zornoza.** Aprendizaje con Simuladores. Aplicación a las Redes de Comunicaciones.

2. Introducción a las Redes de Area Local. **Universidad de las Ciencias Informáticas.** Cuba : s.n., 2007. Conferencia de Redes LAN. págs. 1-14.

3. **Stalling, William.** Comunicaciones y Redes de Computadores.

4. Redes WAN. **Universidad de las Ciencias Informáticas.** Cuba : s.n., 2006. Conferencia de Redes WAN. págs. 1-15.

5. Pozo, Javier Espín del, José Luis Ruiz Ludeña. Topologias de Redes.

6. Tecnologias de Redes de Computadoras. Equipos de Interconexión. **Universidad de las Ciencias Informáticas.** Cuba : s.n., 2006. Conferencia de Equipos de Interconexión. págs. 1-9.

7. **Richard A. McMahon, Sr.** Introducción a las Redes. Huston : s.n.

8. **Vozmediano, Rafael Moreno.** El Nivel Físico. Medios de Transmisión. [En línea] http://www.scribd.com/doc/6694450/Medios-de-Transmision.

9. **Microsoft Corporation.** Microsoft Technet. Microsoft Technet. [En línea] Microsoft Corporation, 2009. [Citado el: 5 de Marzo de 2009.] http://technet.microsoft.com.

10. Direcciones Internet. Extensiones de Direccionamiento. **Universidad de las Ciencias Informáticas.** Cuba : Departamento de Sistemas Digitales, 2006.

11. **LLop, José Manuel Tella.** ABCDatos. ABCDatos. [En línea] 7 de Agosto de 2008. [Citado el: 10 de marzo de 2009.] http://www.multingles.net/.

12. Ejercicios de extensión de direccionamiento. **Informaticas, Universidad de las Ciencias.** Cuba : s.n., 2007.

13. Ruteo. Protocolos RIP y OSPF. **Informáticas, Universidad de las Ciencias.** Cuba : s.n., 2007.

14. Direccionamiento IP y Configuración de las Tablas de Enrutamiento.

15. **González, Agustín J.** Tecnologias WAN y Enrutamiento. Redes de Computadoras.

16. Simulación de Redes TCP/IP. Herramientas. **Informáticas, Universidad de las Ciencias.** Cuba : s.n., 2007.

17. Máquinas Virtuales y Simuladores. 2007.

18. Paloma Cáceres, Esperanza Marcos, Grupo Kybele, Departamento de Ciencias Experimentales e Ingeniería Universidad Rey Juan Carlos. Procesos Ágiles para el desarrollo de Aplicaciones Web. [En línea] 2001.

http://www.dlsi.ua.es/webe01/articulos/s112.pdf.

19. **Mendoza Sánchez, María A. informatizable.** [En línea] 17 de junio de 2004. http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

20. **José H. Canós, Patricio Letelier y Mª Carmen Penadés.** Metodologías Ágiles en el Proceso de Desarrollo de Software. [En línea] 2003.

http://www.willydev.net/descargas/prev/TodoAgil.pdf.

21. El lenguaje Unificado de Modelado UML. **Orallo, Enrique Hernández.** 2001.

22. **Navarra, Universidad de.** unav. unav. [En línea] [Citado el: 10 de Febrero de 2009.] http://www.unav.es/SI/manuales/Java/indice.html#1.2.

23. El lenguaje de Programación Java™.

24. **Universidad Carlos III de Madrid.** El entorno de programación de fuente abierto Netbeans versión 3.3.2. Madrid. España : s.n., 2002.

25. Mi Primera Hora con Eclipse.

26. La plataforma .NET: el furuto de la Web? Unai Extremo Baigorri, Borja Sotomayor Basilio.

27. Selección de Herramientas CASE. **Cámara, D. Arantzazu.** 2005.

28. **Lana de la Torre Quintana, Michel Cruz González.** Análisis y Diseño del Sistema de Gestión de Inquietudes de la FEU en la UCI. Cuba : Universidad de las Ciencias Informáticas, 2008.

29. **© 2000-2009, 7 Curtis Street, Creswick, VIC 3363 AUSTRALIA.** Sparx Systems. Sparx Systems. [En línea] http://sparxsystems.com.ar/products/ea.html.

30. Expendedora: Modelo de Dominio.

31. **Carbonell, Carlos Matos, Valle, Arnolis Rodríguez del.** Creación de un sistema virtual de Transmisión de Datos para el apoyo a la docencia en la asignatura teleinformatica I. Cuba : Universidad de las Ciencias Informáticas, 2008.

32. **Larman, Craig.** UML y Patrones.

33. **Reynoso, Carlos Billy.** Introducción a la Arquitectura de Software. Buenos Aires : s.n., 2004.

34. Manual Java Swing.

35. **Gómez, S. P.** Swing. Universidad Politécnica de Madrid : s.n., 2006.

36. **Cooper, J.W.** Introduction to Design Patterns in C#. 2002.

37. Modelo de Diseño. **Universidad de las Ciencias Informáticas.** Cuba : s.n., 2007.

38. **Sánchez, Yamilka González.** Simulador de Algoritmos de planificación de Procesos para la Asignatura Sistema Operativos. Cuba : Universidad de las Ciencias Informáticas, 2008.

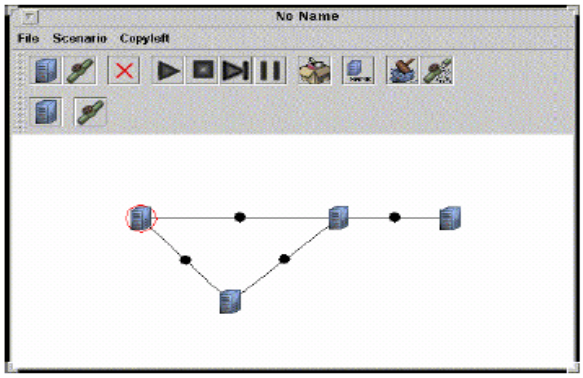
39. Flujo de Implementación. **Universidad de las Ciencias Informáticas.** Cuba : s.n., 2007.

40. **Natalia Juristo, Ana M. Moreno, Sira Vegas.** Técnicas de Evaluación de Software. 2006.

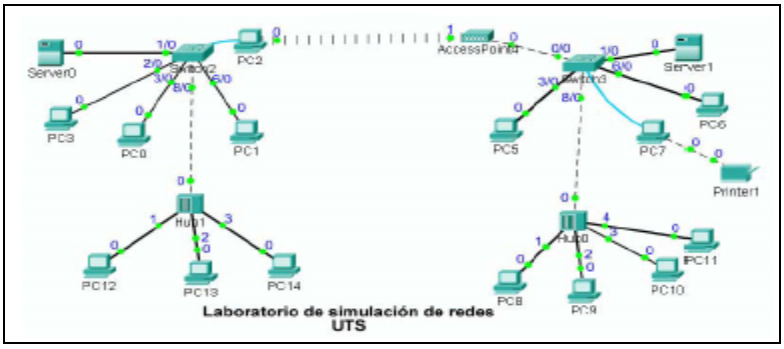
41. *Flujo de Trabajo de Prueba.* **Universidad de las Ciencias Informaticas.** Cuba : s.n., 2007.

Anexos

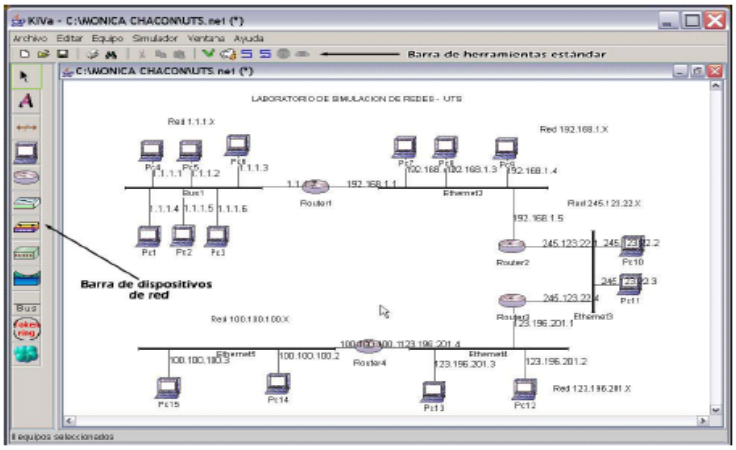
Anexo 1: Herramienta de Simulación de Redes FLAN.



Anexo 2: Herramienta de Simulación de Redes PACKET TRACER™



Anexo 2: Herramienta de Simulación de Redes KIVA



Glosario de Términos

CASE: Computer Aided Software Engineering o Ingeniería de software Asistida por Computadora. Es un conjunto de métodos, utilidades y técnicas que facilitan la automatización del ciclo de vida del desarrollo de sistemas de información.

CLR: Common Language Runtime o Lenguaje común en tiempo de ejecución es el motor de todo .NET. Este CLR recompila el código generado en .NET para generar código nativo, es decir optimizado para el sistema operativo y el hardware actual. Esta compilación la realiza el compilador llamado JIT (Just In Time).

Framework: Marco de Trabajo. Es un esquema (un esqueleto, un patrón) para el desarrollo y/o la implementación de una aplicación.

GUI: Graphical User Interface o Interfaz Gráfica de Usuario. Es un sistema de interacción entre el ordenador y el usuario, caracterizado por la utilización de iconos y elementos gráficos en su concepción.

IDE: Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI.

IEEE: The Institute of Electrical and Electronics Engineers o Instituto de Ingenieros Eléctricos y Electrónicos, es una asociación técnico – profesional internacional dedicada a la estandarización, entre otras cosas.

J2EE: Java 2 Enterprise Edition. Define un estándar para el desarrollo de aplicaciones empresariales multicapa diseñado por Sun Microsystems.

Open Source: Código abierto. Es el término con el que se conoce el software distribuido y desarrollado libremente.

Plug-ins: Significa “enchufar”. Es una aplicación informática que interactuando con otra aplicación le aporta una función específica a ésta última.