

Universidad de las Ciencias Informáticas

Facultad 4



Título: Laboratorio Virtual de Redes

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

Autor(es): Lilier Pérez Pérez.

Janet Bárbara Puerto Viera.

Yordan Danilo Grao Rodríguez.

Tutor(es): Ing. Ruth Yurina Vega Cutiño.

Ing. Yoandro Hechevarría Toranzo.

La Habana, Junio del 2007

DECLARACIÓN DE AUTORÍA

Declaramos que Lilier Pérez Pérez, Yordan Danilo Grao Rodríguez y Janet Bárbara Puerto Viera somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste se firma la presente a los ____ días del mes de ____ de 2007.

Lilier Pérez Pérez

Firma del Autor

Yordan Danilo Grao Rodríguez

Firma del Autor

Janet B. Puerto Viera.

Firma del Autor

Ruth Yurina Vega Cutiño.

Firma del Tutor

Yoandro Hechavarría Toranzo

Firma del Tutor

DATOS DE CONTACTO

Tutora: Ruth Yurina Vega Cutiño.

Área: Departamento de Sistemas Digitales y Aseguramiento básico de Programación.

Correo: ruth@uci.cu

Apartamento: 21101

Teléfono: 8358798

Tutor: Yoandro Hechavarría Toranzo

Área: Facultad #4

Correo: yoandro@uci.cu

Apartamento: 49303

Teléfono: -

Datos: Janet Bárbara Puerto Viera

Correo: jpuerto@estudiantes.uci.cu

Datos: Yordan Danilo Grao Rodríguez

Correo: ygrao@estudiantes.uci.cu

Datos: Lilier Pérez Pérez

Correo: lperez@estudiantes.uci.cu

AGRADECIMIENTOS

Queremos agradecerle a la Revolución Cubana por darnos la posibilidad de graduarnos como Ingenieros en Ciencias Informáticas. A Fidel, por darnos esta oportunidad. A nuestros tutores, por haber asumido esta tarea. A nuestros profesores, que nos proporcionaron la mayor parte de los conocimientos que tenemos. A aquellos que nos tendieron la mano cuando los necesitamos, a todos, gracias.

A mis padres, por darme tanto amor, y prepararme para la vida. A mi abuela, porque este su regalo. A Popy , gracias por existir, sin ti simplemente no podría Vivir! A tata, por los buenos momentos juntas. A Lianna, por la música, por su amistad, y porque simplemente si! A mis familiares y amigos....

Lilier

Porque aunque crean que es un granito de arena lo que me dieron, quiero que sepan que hoy no estaría logrando graduarme si no hubieran estado para mi. Porque son especiales, porque estoy agradecida por lo que me han apoyado, y porque son mi ejército ante las dificultades, quiero agradecerles en primer lugar a mi familia, a mi madre porque por ella soy todo lo mejor posible, mi guía, mi apoyo. A mi padre, porque fue uno de mis más grandes amores, eternamente te quiero papi. A Yelena, mi hermana querida, de la que vivo orgullosa y mi ejemplo a seguir. Al pedazo de mi corazón y la eterna sonrisa, Irma. A mi único Anyely, porque en poco tiempo has llegado a ser mi complemento, te amo. A mis abuelos Goyos, por ser otros grandes amores para mí. Dios me ha dado muchas madres y tantos hermanos, Isora sabes que mi corazón es tuyo también, quiero agradecerte junto con tía Mirian e Irmita, porque su amor me ha levantado cuando me he caído, a mis tías Mirna, Mirta, Ana María y Marlen porque más que un soporte para mi, lo han sido para mi madre, a mis tíos Mongui, Enrique, Frank, Chago y Felipe, les agradezco cada apoyo, porque son padres para mi. Sacra no naciste de mi sangre pero sabes que lo eres, tu presencia es un regalo para nosotros. Viannys y Danay, que sería yo sin su amor, ahí tienen mi corazón para lo que necesiten siempre. A Mabel, por haber sido mi madre estos años lejos de mi casa, junto con Ana Mary, Ernesto y Angelito. No podemos temer nunca cuando tenemos amistades, como las que tengo yo, Anacelia, tú y tu familia, me dieron fuerzas y amor estos 5 años, eres mi sangre donde estés. Quiero agradecerles a mis mosqueteras, todas para una y una para todas: Elizabeth, Maylé, Yaima, Annia ¿por qué no nacieron en mi casa, o al lado? si Dios sabe que tienen mi amor desde que las conocí, cuenten con un pedazo de mi siempre, donde estén. A Nelly y Daniel porque ya no puedo dejar de quererlos. A

Belquis y a Gabriel les quiero agradecer por el amor que me tienen. A mis primos, A Kiki, porque aunque estés lejos, te debemos demasiado. A todos, miles de agradecimientos, aquí está el fruto de su amor.

Janet

A mis abuelos Eva y Pedro, por su cariño amor y dedicación, lo que soy y seré se lo debo a ustedes.

A mis padres y hermanos, Marta, Carlos, Yuniór y Yordanis, por siempre estar y ser especiales.

A mi familia en general y a aquellos que estuvieron pendientes de mí en todo momento, mis tíos Anita, Jorge e Idania.

A mis amigos de siempre, Regino y Yeny y a mi primo Jorge, gracias por su apoyo.

A mis amistades de la UCI, que siempre me han brindado su ayuda y apoyo incondicional, especialmente a Ariesky, Eduardo, Yudith, Sianny, Giraltné, Marita, Yanet, Heidy, Dunia, Cuan, Julito, Yasmany, Yorker, Reinier, Yadira y Yise.

A los que a pesar de estar lejos aún se acuerdan de mí, y me muestran su apoyo, Tami, Rosme y Barby.

A Jose Luis, por su paciencia.

En fin a todos los que me día a día me preguntaban cómo está tu tesis.

Yordan

DEDICATORIA

A Rolando Puerto Armesto.

A Gilda, Maria y Orlando.

A Eva, Pedro, Marta y Carlos.

RESUMEN

Este trabajo se propone estudiar la propuesta de un sistema que automatice las actividades prácticas que actualmente no cuentan con un soporte informático en la asignatura Teleinformática, perteneciente al Departamento de Sistemas Digitales de la UCI.

Actualmente solo se encuentran programadas pocas prácticas del diseño de una red antes de presentar un proyecto evaluativo final en el curso, por no contar con la tecnología ni las herramientas necesarias para mejorar la calidad de estas. Entre los procesos más prioritarios desde el punto de vista de los intereses del usuario, se encuentran la orientación de ejercicios prácticos de diseño de redes de computadoras, la respuesta de estos ejercicios por parte del profesor y los estudiantes, la actualización de los datos de los elementos de red, con datos reales de precios propuestos por los proveedores de equipamiento informático en Cuba y la simulación del funcionamiento de algunos dispositivos de red.

Por tanto el objetivo concreto de este trabajo consiste en desarrollar un laboratorio virtual que permita al usuario dar respuesta a ejercicios de diseño de red por entrada de parámetros, y a su vez observar el funcionamiento de algunos dispositivos de red, y que además brinde facilidades para el trabajo en general de la asignatura Teleinformática.

En este documento se recogen los resultados de todo el trabajo investigativo realizado. Se identifican y describen los procesos actuales de las prácticas de Teleinformática, especialmente aquellos que se van a automatizar. Posteriormente se hace un análisis comparativo acerca de las tecnologías existentes y se seleccionan las más apropiadas. Se muestran los resultados del diseño de la propuesta del sistema y finalmente se logra definir la línea base ejecutable de la arquitectura de la aplicación.

TABLA DE CONTENIDOS

INTRODUCCIÓN	1
CAPITULO 1: FUNDAMENTACIÓN TEÓRICA	4
1.1. Introducción	4
1.2. Herramientas de simulación en el área de diseño de redes informáticas.	5
1.2.1. FLAN	5
1.2.2. PACKET TRACER™	6
1.2.3. Kiva	6
1.2.4. NS	7
1.2.5. COMNET III™	7
1.2.6. OPNET MODELER™	8
1.2.7. OMNET ++	8
1.3. ¿Por qué una Aplicación Web?	9
1.4. Arquitectura de software.....	10
1.4.1. Arquitectura Cliente-Servidor.	11
1.4.2. Modelo Vista Controlador (MVC)	13
1.5. Lenguajes de programación.	14
1.5.1. Lenguajes del lado del cliente.	14
1.5.1.1. HTML	15
1.5.1.2. Java Script	15
1.5.2. Lenguajes del lado del servidor.....	16
1.5.2.1. Perl.....	16
1.5.2.2. ASP	17
1.5.2.3. JSP.....	17
1.5.2.4. PHP.....	17
1.5.2.5. C++	19
1.6. CSS	20
1.7. Herramientas de programación.	20
1.7.1. Macromedia Dreamweaver	20
1.7.2. Builder	21
1.7.2.1. OpenGL.....	22

1.8.	Zend Development Enviroment	23
1.9.	Sistema de gestión de base de datos.	24
1.9.1.	Oracle.....	25
1.9.2.	SQL Server.....	25
1.9.3.	PostgreSQL.....	26
1.9.4.	MySQL	26
1.10.	Servidor de la aplicación.....	27
1.10.1.	Apache.....	27
1.11.	Características de la tecnología WAMP	28
1.12.	Metodología a utilizar.....	28
1.12.1.	Rational Rose Enterprise	29
1.12.2.	El Proceso Unificado del Rational. RUP	30
1.13.	UML	31
1.14.	Propuesta.....	31
CAPITULO 2: CARACTERÍSTICAS DEL SISTEMA.....		33
2.1.	Objeto de estudio	33
2.1.1.	Problema y situación problemática	33
2.1.2.	Objeto de automatización.....	34
2.1.3.	Información que se maneja	35
2.1.4.	Propuesta de sistema.....	36
2.2.	Modelo de dominio.	36
2.3.	Especificación de los requisitos de software	38
2.3.1.	Requisitos funcionales	39
2.3.2.	Requisitos no funcionales	41
2.3.2.1.	Apariencia o interfaz externa	42
2.3.2.2.	Rendimiento	42
2.3.2.3.	Soporte.....	42
2.3.2.4.	Portabilidad	42
2.3.2.5.	Requerimientos de Hardware	43
2.3.2.6.	Requerimientos de Software	43
2.3.2.7.	Seguridad.....	43

2.3.2.8.	Disponibilidad.....	43
2.3.2.9.	Confiabilidad	43
2.3.2.10.	Integridad.....	44
2.3.2.11.	Usabilidad	44
2.4.	Definición de los casos de uso	44
2.4.1.	Definición de los actores	44
2.4.2.	Listado de casos de uso.....	45
2.4.3.	Diagrama de casos de uso.....	49
2.4.4.	Casos de uso por ciclo	51
2.4.5.	Casos de uso expandidos	52
CAPITULO 3:	ANÁLISIS Y DISEÑO DEL SISTEMA	61
3.1.	Análisis.....	61
3.1.1.	Modelo de clases de análisis.....	61
3.2.	Diseño	65
3.2.1.	Diagramas de interacción.....	65
3.2.1.1.	Diagramas de Secuencia	66
3.2.2.	Diagrama de clases del diseño.....	66
3.2.2.1.	Descripción de las clases.....	70
3.3.	Diseño de la base de datos (BD).....	70
3.3.1.1.	Modelo Entidad–Relación (ER).....	71
3.3.1.2.	Descripción de las tablas.....	73
3.4.	Definiciones de diseño que se apliquen.....	73
3.5.	Tratamiento de errores	73
3.6.	Seguridad	74
3.7.	Interfaz.....	74
3.8.	Concepción de la ayuda	75
CAPITULO 4:	IMPLEMENTACIÓN Y PRUEBA.....	76
4.1.	Implementación	76
4.1.1.	Modelo de despliegue	76
4.1.2.	Modelo de implementación	78
4.2.	Modelo de prueba.....	79

CONCLUSIONES	81
RECOMENDACIONES.....	82
BIBLIOGRAFÍA.....	83
ANEXOS	85
Anexo 1: Expansión de Casos de Uso restantes.....	85
Anexo 2: Diagramas de clases del análisis restantes.....	97
Anexo 3: Diagramas de Secuencia para cada Caso de Uso.....	103
Anexo 4: Diagramas de clases del diseño restantes.....	121
Anexo 5: Descripción detallada de las clases, sus atributos y operaciones.....	127
Anexo 6: Descripción de las tablas del modelo ER.....	155
Anexo 7: Modelo de implementación.....	161
Anexo 8: Descripción de los Casos de Prueba.....	164
GLOSARIO	173

INTRODUCCIÓN

Perteneciente a la Dirección de Formación Profesional, el Departamento de Sistemas Digitales imparte la asignatura Teleinformática I y II, a estudiantes de tercer año de la carrera Ingeniería Informática en la Universidad de las Ciencias Informáticas (UCI), como parte de la formación de pregrado. Teleinformática o telemática es la ciencia que trata de la conectividad a distancia entre procesos: las temáticas del acceso y procesamiento de datos a distancia entre computadoras. El programa de esta asignatura incluye los temas: fundamentos de la teoría de la información y transmisión de datos, tecnologías de redes de computadoras, arquitectura TCP/IP, simulación de redes y administración de redes. La asignatura está organizada en conferencias y clases prácticas, que generalmente se imparten en los laboratorios, como complemento para cumplir los objetivos.

Los estudiantes necesitan practicar el diseño de redes, y poder observar el funcionamiento de algunos dispositivos de red para apropiarse de los conocimientos básicos acerca del funcionamiento de las redes de computadoras. Para realizar el diseño de una red se siguen pasos como: determinar el tipo de cable y la topología física (cableado) a utilizar, decidir cuál de las distintas topologías de Ethernet utilizar, diseñar un esquema óptimo de cableado estructurado, entre otros. Para la realización de estas actividades se requiere de un laboratorio con determinadas exigencias de hardware con las que la universidad no cuenta. Por lo costosa que es esta tecnología, la cantidad de estudiantes que necesitan de estos laboratorios y la necesidad de laboratorios en la universidad estas prácticas no son realizadas con éxito. Los estudiantes actualmente no cuentan con las herramientas necesarias que les muestren las soluciones óptimas a sus ejercicios, quedando el nivel de enseñanza muy pobre.

Las actividades prácticas se encuentran actualmente sin la mejor cobertura informática, lo que evidentemente introduce ineficiencia en el proceso educativo. Los estudiantes no pueden aplicar correctamente los conocimientos que se les imparten por no contar con un espacio virtual donde se les muestren los errores que cometen en las prácticas; y al profesor le es difícil saber el estado en que se encuentran sus estudiantes, o grupos docentes. También se imposibilita la atención diferenciada por parte de los profesores.

Por tanto **el problema** a resolver queda formulado de la siguiente forma: ¿cómo lograr que los estudiantes de la carrera de ingeniería informática de la UCI puedan realizar las prácticas de la asignatura de Teleinformática con eficiencia?

El objeto de estudio lo constituyen los simuladores de redes existentes de algunos fabricantes, y el funcionamiento del equipamiento de red de diferentes tecnologías necesario para las prácticas de la asignatura Teleinformática en la UCI.

Del Objeto de estudio antes expuesto se deriva que **el campo de investigación** abarque las herramientas simuladoras de algunos dispositivos de red, y todas las mejoras posibles para el desarrollo de un sistema informático en la UCI para la asignatura Teleinformática, que sea parte de los servicios automatizados que brinda la universidad.

El objetivo general es desarrollar un laboratorio virtual que permita al usuario dar respuesta a ejercicios de diseño de red por entrada de parámetros, y a su vez observar el funcionamiento de algunos dispositivos de red.

De acuerdo con este objetivo general, se derivan los siguientes objetivos específicos:

- Obtener información sobre softwares existentes vinculados al diseño de redes.
- Analizar y determinar los requisitos del sistema.
- Analizar y diseñar una aplicación web que permita al usuario dar respuesta a ejercicios de diseño de red, y observar el funcionamiento de algunos dispositivos de red.
- Diseñar y construir la base de datos del sistema, que garantice la integridad de la información contenida.
- Implementar la aplicación.
- Realizar las pruebas pertinentes para el correcto funcionamiento de la aplicación.

Para complementar estos objetivos se proponen las siguientes tareas de investigación:

1. Selección y revisión bibliográfica de temas relacionados con diseño de redes informáticas, así como las herramientas simuladoras existentes.
2. Realizar una valoración de las tendencias actuales de las tecnologías que se utilizan para cumplimentar sistemas como el que se pretende desarrollar.
3. Estudio exhaustivo de la asignatura Teleinformática para hacer un levantamiento de Requisitos.
4. Identificación y selección de la metodología de Análisis y Diseño de sistemas informáticos, que facilite la creación y garantice la calidad del sistema.
5. Seleccionar y fundamentar las herramientas y la plataforma para llevar a cabo el proyecto.
6. Diseño de una base de datos que soporte las funcionalidades del sistema.
7. Implementación de una aplicación web que permita al usuario dar respuesta a ejercicios de diseño de red, y observar el funcionamiento de algunos dispositivos de red.

8. Identificar el tipo y los métodos de pruebas a utilizar.

Este documento está dividido en 4 capítulos:

Capítulo 1. Fundamentación Teórica.

Este capítulo hace referencia a algunos simuladores existentes en el mundo, dejándose explicada por qué la necesidad de desarrollar una aplicación en nuestra institución, se presentan las tendencias y tecnologías existentes en la actualidad que se deben considerar para hacer la selección de aquellas que se van a utilizar en el proyecto, y fundamentar la selección.

Capítulo 2. Características del sistema.

Este capítulo describe el objeto de estudio, el entorno de trabajo en que se desarrolla la aplicación, la propuesta del sistema, requerimientos funcionales y no funcionales, así como la especificación de los casos de uso del sistema.

Capítulo 3. Análisis y diseño del sistema.

En este capítulo se plantean los detalles relacionados con el análisis y diseño del sistema que se propone, se utilizan para su modelado los diagramas de clases del análisis, de interacción, el diseño de clases y el diseño de la base de datos que se necesita para el almacenamiento de la información persistente.

Capítulo 4. Implementación y prueba.

Este capítulo presenta la correspondencia entre la arquitectura del software y la del hardware de la aplicación mediante el modelo de despliegue. Mediante el modelo de implementación se indica la implementación actual del sistema en términos de componentes y subsistemas; y finalmente, se presentan en el modelo de pruebas, algunos casos de pruebas para probar los componentes ejecutables en el modelo de implementación.

CAPITULO 1: FUNDAMENTACIÓN TEÓRICA

En este capítulo se aborda el estado actual de las herramientas simuladoras utilizadas en el ámbito mundial para planificar y dimensionar los recursos de las redes informáticas previamente a su instalación. Se presenta un estudio de las tecnologías en las que se apoya el desarrollo del sistema en función de un análisis de las tendencias actuales, de manera que permita situar el marco de referencia de las herramientas que se emplearán. La idea de la cual se parte, es la de integrar múltiples herramientas, que unidas contribuyan al adecuado desempeño de los procesos que se quieren automatizar.

1.1. Introducción

Con el surgimiento de Internet se dio lugar a que los medios para publicar y compartir información, y dentro de ellos, los Servicios Web, evolucionaran hasta el punto de constituir un factor importante en la economía mundial. La Web fue pensada como un medio para desplegar información, esta reposa de manera estática en los servidores y es accedida a través de una consulta hecha por un navegador valiéndose del protocolo HTTP. Por lo que se considera una aplicación Web como un Sitio Web donde la navegación a través de él y la entrada de datos por parte del usuario, afectan el estado de la lógica del negocio. En esencia, una aplicación Web usa un sitio Web como entrada a una aplicación típica. Si no existe lógica del negocio en el servidor, el sistema no puede ser llamado aplicación Web.

Hoy en día, las empresas, organismos y disímiles entidades usan generalmente la Internet como medio para lograr que el uso compartido de la información sea más efectivo. La utilización de las aplicaciones Web, logran una mayor rentabilidad y eficiencia, pues gracias a la interconexión que permite la red, se logran optimizar esfuerzos, recursos, tiempo, y dinero, así como la eliminación de las barreras de las distancias, para cualquier intercambio de datos o enseñanza. La forma de presentar esa información, así de cómo comunicárselo a la entidad y a toda persona involucrada en ella, es un paradigma hoy en día; debido a que con los rápidos avances de las tecnologías de la información se dejan viejas reglas de competencia y el duradero entendimiento de la relación cliente-proveedor obsoleto, por lo que se va creando la necesidad de un entorno de información mas dinámico en el cual los temas y procesos pueden cambiar semanalmente y en algunos casos hasta diariamente.

El siglo XXI ofrece a los educadores brillantes oportunidades para que puedan promover los cambios en la educación. Como las transiciones entre el modelo educativo tradicional (presencial o a distancia) y aquel que utiliza las nuevas tecnologías de la informática y la comunicación (NTIC), se van produciendo paulatinamente, han surgido varios sistemas para el aprendizaje académico en un entorno sencillo,

intuitivo y amigable que brinda oportunidades de comunicación y colaboración para la construcción de conocimientos en un ambiente rico en información. Los sitios y las aplicaciones Web han surgido como soporte de estos sistemas de información que buscan facilitar la organización y el acceso a los contenidos y servicios disponibles.

Cuando nos referimos a la relación entre software y el aprendizaje académico nos tenemos que referir a la Informática Educativa, que es el área vinculada con la psicología cognitiva, la informática y la Psicopedagogía, que se ocupa de la investigación, el desarrollo de planes y software educativos y la implementación de la informática al servicio del aprendizaje. La asignatura Teleinformática que se imparte en la carrera de Ingeniería Informática utiliza algunas herramientas y software con propósitos educativos para mejorar la calidad del proceso docente.

1.2. Herramientas de simulación en el área de diseño de redes informáticas.

La simulación es la imitación del funcionamiento de un sistema real durante un intervalo de tiempo. La simulación, actualmente tiene una gran cantidad de usos prácticos, los cuales pueden resumirse en: predicción, entrenamiento, entretenimiento, mejor comprensión de la situación estudiada y apoyo a la toma de decisiones. En el entrenamiento desempeña un papel importante como una herramienta mas para la capacitación de alumnos en circunstancias que la práctica real puede ser muy costosa. La simulación se usa como herramienta que permite al usuario una mejor comprensión de la situación estudiada. En el caso de las redes de computadoras, debido a la dificultad para diseñarlas, han surgido varias herramientas simuladoras con características particulares que ayudan a los investigadores y diseñadores a planificar y dimensionar los recursos de una red como fase previa a su instalación. Existe una gran variedad de programas que nos permiten simular una red de computadoras, controlando muchas de sus variables, y obteniendo un resultado que debe ser interpretado y analizado.

1.2.1. FLAN

Es un software desarrollado con el lenguaje de programación Java y se distribuye con licencia pública. 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. El FLAN es un software multiplataforma, es decir que puede ser implementado sobre cualquier sistema operativo que soporte la máquina virtual de Java. El programa contiene unas herramientas llamadas manejadores, que

son protocolos específicos que ayudan a determinar cómo es recibida la información, cómo procesarla y además cómo dirigir la simulación. Este simulador tiene varias desventajas que nos impiden utilizarlo, por ejemplo: el mismo está diseñado para la prueba de protocolos en redes pequeñas, y es necesario tener conocimientos básicos sobre el lenguaje de programación Java para poder hacer más configuraciones con el software y dar solución a problemas que se presenten al momento de definir características o parámetros de los dispositivos, protocolos o aplicaciones.

1.2.2. PACKET TRACER™

Es un simulador gráfico de redes desarrollado y utilizado por Cisco como herramienta de entrenamiento. 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. Packet Tracer tiene tres modos de operación: topology (topología), simulation (simulación), al cual se accede cuando se ha creado el modelo de la red; finalmente aparece el modo realtime (tiempo real), en donde se pueden programar mensajes para detectar los dispositivos que están activos en la red y si existe algún problema de direccionamiento o tamaño de tramas entre las conexiones. El enfoque pedagógico de este simulador, hace que sea una herramienta muy útil como complemento de los fundamentos teóricos sobre redes de comunicaciones. Permite ver el desarrollo por capas del proceso de transmisión y recepción de paquetes de datos de acuerdo con el modelo de referencia OSI. Pero como desventaja tenemos que es un software propietario, y por ende se debe pagar una licencia para instalarlo, solo permite modelar redes en términos de filtrado y retransmisión de paquetes y no permite crear topologías de red que involucren la implementación de tecnologías diferentes a Ethernet.

1.2.3. Kiva

Es un simulador de redes basado en Java que permite especificar diferentes esquemas de redes de datos y simular el encaminamiento de paquetes a través de dichas redes. Es una herramienta software orientada principalmente a simular el comportamiento del protocolo IP, y especialmente para el estudio del tratamiento de los datagramas y el encaminamiento de los mismos por una red. También al utilizarlo, se puede estudiar el funcionamiento de los protocolos auxiliares ARP e ICMP y emular el funcionamiento básico de tecnologías de enlace como Ethernet. Con esta herramienta, se puede diseñar una topología de red con la interfaz gráfica, configurar el direccionamiento y las tablas de encaminamiento para los dispositivos y simular el envío de paquetes de un equipo a otro. El objetivo principal de este programa, es

ayudar a diseñar y comprender el funcionamiento de redes de datos y en especial el encaminamiento de paquetes en la arquitectura TCP/IP, sin necesidad de una infraestructura real y de herramientas de análisis de tráfico; éste programa, también es capaz de simular distintos tipos de errores en el funcionamiento de las redes, como la pérdida de paquetes o fallos en tablas de enrutamiento. El programa se distribuye con software libre y además es multiplataforma. Su orientación académica, hacen que sirva de ayuda para el diseño y comprensión del funcionamiento de redes de datos. Para el diseño y comprobación del encaminamiento en redes de datos a nivel comercial o para fines de investigación y desarrollo se debe hacer programación en Java.

1.2.4. NS

El Network Simulator, más conocido como NS, es un software orientado a simular eventos discretos; ha sido diseñado especialmente para el área de la investigación de redes telemáticas. NS es una herramienta con un amplio rango de uso y que continuamente sirve como base para el desarrollo de otros programas de simulación; este software soporta una gran cantidad de protocolos de las capas de aplicación y transporte, además de otros utilizados para el enrutamiento de los datos, entre los cuales están: HTTP, FTP, TCP, UDP, RTP entre otros; los cuales pueden ser implementados tanto en redes cableadas, como inalámbricas locales o vía satélite; y que son aplicables a grandes redes con topologías complejas y con un gran número de generadores de tráfico. Para visualizar los resultados es necesario instalar el Network Animador (NAM), que es una herramienta de interfaz gráfica muy sencilla de utilizar. NS requiere varios componentes adicionales instalados para su correcto funcionamiento como: Tcl/Tk, Otcl, TclCL20 que hacen parte del compilador para Linux. Contempla mecanismos referentes a la capa de enlace de datos en redes locales, como protocolos MAC (Control de Acceso al Medio) del tipo CSMA/CD (Acceso Múltiple por Censado de Portadora y Detección de Colisiones). La configuración de las simulaciones a través de código hace que sea mayor el tiempo de desarrollo; además también se incrementa el tiempo necesario para el aprendizaje del software. NS es universalmente aceptado como simulador de prueba para investigación en redes, pero resulta poco apto para la enseñanza.

1.2.5. COMNET III™

Es un software gráfico que permite analizar y predecir el funcionamiento de redes informáticas, desde topologías básicas de interconexión hasta esquemas mucho más complejos de simulación con múltiples redes interconectadas con diversos protocolos y tecnologías de red. Dentro del área de trabajo del programa, se hace la descripción gráfica del modelo de red, se asocian las fuentes generadoras de tráfico

en la red, se configuran los parámetros y las características de los dispositivos de acuerdo a la aplicación que se desea implementar; luego se pone en marcha la simulación y finalmente, se analizan los resultados estadísticos sobre el desempeño de la red. COMNET III es un software propietario, y por ende se debe pagar una licencia para instalarlo. Además de los conocimientos sobre el manejo y el diseño de redes de comunicaciones, se requieren conocimientos en otras áreas como por ejemplo la estadística.

1.2.6. OPNET MODELER™

OPNET Modeler™ es un programa ampliamente utilizado en la industria para modelar y simular sistemas de comunicaciones; permite diseñar y estudiar redes, dispositivos, protocolos y aplicaciones, brindando escalabilidad y flexibilidad, cualidades que le permiten ofrecer a sus usuarios, trabajar en procesos de investigación y desarrollo. Originalmente fue introducido al mercado en 1987 como el primer simulador comercial. Esta herramienta se utiliza para el modelado y simulación; está basada en la teoría de redes de colas e incorpora las librerías para facilitar el modelado de las topologías de red. Ofrece múltiples capacidades de simulación, animación y análisis. Permite mostrar el tráfico por la red a través de una animación, durante y después de la simulación. Los resultados se exhiben mediante gráficos estadísticos. Tiene interfaces para visualización del modelo en 3D. Los modelos pueden ser transferidos sin modificación entre plataformas Windows 2000, XP, Linux y Sun Solaris. Maneja topologías de red complejas con subredes anidadas ilimitadas. Es un software propietario, lo cual lo hace costoso para ambientes universitarios, es necesario obtener la licencia para poder utilizar el software, ya que no existen versiones académicas o de prueba. Otra desventaja es que el tiempo de aprendizaje para poder utilizarlo es elevado.

1.2.7. OMNET ++

Es una herramienta eficiente, enfocada al área académica y desarrollada para modelar y simular eventos discretos en redes de comunicaciones. Básicamente este simulador de redes recrea dichos eventos discretos por medio de módulos orientados a objetos; puede ser utilizado para modelar el tráfico de información sobre las redes, los protocolos de red, las redes de colas, multiprocesadores y otros sistemas de hardware distribuido; además para validar arquitecturas de hardware y evaluar el rendimiento de sistemas complejos. Las simulaciones en OMNET++ pueden utilizar varias interfaces de usuario, dependiendo del propósito. Este software es gratuito solamente para propósitos académicos, lo que facilita su utilización en universidades y grupos de investigación. Es multiplataforma. Por ser un software

de aplicación en áreas comerciales y para efectos de investigación y desarrollo, tiene un alto grado de complejidad en su manejo.

Estos simuladores no sirven como herramientas eficientes para las prácticas de la asignatura Teleinformática, porque no están enfocadas al área académica planificada en nuestra universidad, no cumplen las condiciones del simulador que se necesita, ya sea por sus requerimientos del sistema, por razones técnicas, por restricciones de licencia o los costos involucrados. Algunos están diseñados para la prueba de protocolos, el diseño de topologías, la configuración de dispositivos de red, así como la detección y corrección de errores en sistemas de comunicaciones, otros permiten ver el desarrollo por capas del proceso de transmisión y recepción de paquetes de datos o simular el comportamiento de protocolos. La mayoría son propietarios, lo cual los hace costosos para ambientes universitarios. Otra desventaja es el tiempo de aprendizaje que conllevaría utilizar alguno de ellos, que se puede hacer muy largo por la complejidad del software.

La asignatura Teleinformática necesita un ambiente sencillo y de fácil uso que les permita a sus usuarios observar las simulaciones del funcionamiento de algunos dispositivos de redes y realizar algunos ejercicios orientados por su profesor. Esto originó el proyecto a desarrollar: Laboratorio Virtual de Redes. Para desarrollar este laboratorio virtual se tuvo en cuenta hacerlo en una multimedia, en un simulador o una aplicación Web. Se selecciona desarrollarlo mediante una aplicación Web por las ventajas que nos ofrecen.

1.3. ¿Por qué una Aplicación Web?

Una aplicación Web, fruto del desarrollo de Internet, no es más que la unión de varias páginas Web, estas a su vez son documentos que van a presentar la información de una forma agradable y sencilla al usuario. Consideraremos una aplicación Web a un sitio Web donde la navegación a través de él y la entrada de datos por parte de un usuario, afectan el estado de la lógica del negocio. En esencia, una aplicación Web usa un sitio Web como entrada a una aplicación típica. Si no existe lógica del negocio en el servidor, el sistema no puede ser llamado aplicación Web. Bajo este concepto las aplicaciones Web no solo se encargan de desplegar información, sino que también, deben contener una lógica asociada que permita apoyar algún proceso propio del negocio para el cual fue diseñada, y manejar los datos con la ayuda de algún sistema gestor de base de datos. Las aplicaciones Web constituyen un avance en el desarrollo de aplicaciones, pero a la vez un reto para desarrolladores ya que cada vez se necesitan aplicaciones más robustas, rápidas y eficaces.

Los usuarios, a la vez que llaman a las páginas de una determinada aplicación Web, pueden observar la información a través del browser o sea del explorador, su manejo es muy intuitivo por estar acostumbrados al sistema de hipervínculos de Internet. El protocolo que se utiliza para la comunicación Web es el HTTP. El formato que se utiliza en la transferencia es el HTML, lenguaje hipertexto de marcas. La finalidad del comportamiento Servicio Web es proporcionar una manera sencilla de utilizar y aplicar SOAP sin requerir un conocimiento profundo de su implementación. El Servicio Web admite el uso de una amplia variedad de tipos de datos, incluidos tipos de datos SOAP intrínsecos, matrices y datos XML. Este componente flexible permite a Internet Explorer recuperar información de servicios Web de XML y actualizar una página de forma dinámica mediante el uso de DHTML y secuencias de comandos, sin requerir el desplazamiento ni una actualización de página completa.

La Plataforma Web, posee una arquitectura Cliente/Servidor donde el proceso cliente solicita al proceso servidor la ejecución de alguna acción en particular. Esta arquitectura se refiere a la forma en la que es diseñado el software tanto física como lógicamente. En el diseño físico se especifica exactamente donde se encontrarán las piezas de la aplicación (discos, ejecutables, cable de red y computadoras) y en el diseño lógico o conceptual se especifica la estructura de la aplicación y sus componentes sin tomar en cuenta dónde se localizará el software, hardware e infraestructura. La Plataforma Web esta compuesta por un servidor Web, una red física y un navegador o cliente. Las actualizaciones, mejoras o cambios en la aplicación son puestas en servicio de manera inmediata para todos los usuarios y para todos los ordenadores de la red sin intervención sobre los equipos ya que se realizan exclusivamente sobre el servidor.

Las aplicaciones Web no requieren instalación en los puestos de cliente por lo que se evitan costes de intervención al ampliar el número de computadoras o realizar cambios en las mismas. La utilización de ésta tecnología conlleva a reducir costos y complicaciones, y proporciona mayor libertad a la hora de realizar cualquier tipo de cambios.

1.4. Arquitectura de software.

Una arquitectura es un entramado de componentes funcionales que aprovechando diferentes estándares, convenciones, reglas y procesos, permite integrar una amplia gama de productos y servicios informáticos, de manera que pueden ser utilizados eficazmente dentro de la organización. La arquitectura de software es el conjunto de decisiones significativas sobre la organización de un sistema, la selección de los elementos estructurales y sus interfaces, junto con su comportamiento.

Los tipos de arquitecturas que existen son: la arquitectura Centralizada, la Arquitectura Servidor de Archivos y la Arquitectura Cliente-Servidor. La primera se basa que se basa en la existencia de una máquina servidora que almacena los datos y las aplicaciones que los procesan. Los clientes se comportan como terminales y solo sirven para introducir datos desde teclados. Los datos se envían al servidor, quien procesa las solicitudes y devuelve los resultados. Sus ventajas son el gran nivel de seguridad y la facilidad de administrar. Tiene como desventajas un alto costo y una máquina servidora muy cargada. La Arquitectura Servidor de Archivos, se basa en la existencia de una o varias máquinas servidoras que almacenan datos y estaciones de trabajo que ejecutan aplicaciones que los procesan. La base de datos consiste en sólo un archivo de base de datos. En esta arquitectura el motor de bases de datos puede ejecutarse en la máquina cliente o en el servidor, muy estrechamente unida a la aplicación cliente. Tiene como ventaja un bajo coste y que es escalable. Su desventaja es que necesita clientes potentes y una fuerte dependencia de las comunicaciones. Para desarrollar el Laboratorio virtual de redes se utiliza la Arquitectura Cliente-Servidor.

1.4.1. Arquitectura Cliente-Servidor.

Este término, se usa para describir una aplicación en la cual dos o más procesos separados trabajan juntos para completar una tarea. Un usuario utilizando un navegador realiza una petición de una página Web al servidor, este a su vez receptiona la petición, la procesa y envía la respuesta al cliente en formato HTML a su navegador, el cual receptiona la respuesta y se desconecta. La aplicación está ubicada, generalmente, en un único servidor por lo que los trabajos de copia de seguridad y mantenimiento son mucho más eficientes.

El uso de esta arquitectura tiene muchas ventajas:

1. Existe una centralización del control; los accesos, recursos y la integridad de los datos son controlados por el servidor de forma que un programa cliente defectuoso o no autorizado no pueda dañar el sistema.
2. La aplicación es servida simultáneamente a múltiples usuarios de una o varias redes sin que se den errores por escrituras al mismo tiempo en las bases de datos, por ejecución simultánea de determinados procesos, evitándose, de esta forma, duplicidades de datos.
3. El proceso servidor activo devuelve sólo la información solicitada en la red, de tal modo que el tráfico en la red es sustancialmente reducido.
4. Un proceso servidor activo puede asegurar más eficazmente la integridad de los datos.

5. La escalabilidad permite que se pueda aumentar la capacidad de clientes y servidores por separado.

Las operaciones que realiza un sistema pueden agruparse formando capas. Podemos diferenciar tres tipos de arquitectura: monolíticas (1 capa), de 2 capas, de 3 o más capas.

En el Laboratorio Virtual de Redes se utiliza la arquitectura en 3 capas. Los diseños en tres capas son ampliamente utilizados en el mercado, y a lo largo del tiempo han probado sus ventajas. Estas aplicaciones típicamente tienen mayor capacidad de crecimiento y son más sencillas de mantener, dada su naturaleza altamente modular. La arquitectura de 3 capas es una guía, no un requerimiento. Este es un modelo lógico, no físico, que describe cómo se diseña la aplicación, no cómo se despliega.

La primera capa es la de presentación o interfaz de usuario, está formada por los formularios, informes, respuestas al usuario y los controles que se encuentran en los formularios, se compone de aplicaciones cliente que se crean a partir de componentes de aplicaciones cliente, incluye todo aquello con lo que el usuario puede interactuar.

La capa de negocio está formada por las entidades empresariales, que representan objetos que van a ser manejados o consumidos por toda la aplicación, reglas del negocio, validaciones y cálculos. La capa intermedia (lógica de aplicación) es básicamente el código al que recurre la capa de presentación para recuperar los datos deseados. Se compone de servidores de negocios, la cual se crea a partir de componentes de aplicaciones de servidor de negocios. Es aquí donde se modela el comportamiento del sistema, basándose en los datos provistos por la capa de datos, y actualizándolos según sea necesario. La función de esta capa lógica es primordialmente la de hacer valer las políticas del negocio y encapsular un modelo de los negocios así como exponer tal modelo a las aplicaciones cliente. Esta separación entre la lógica de aplicación de la interfaz de usuario añade una enorme flexibilidad al diseño de la aplicación.

La capa de acceso a datos contiene los datos necesarios para la aplicación, base de datos, tablas, procedimientos almacenados y componentes de datos. Los datos consisten en cualquier fuente de información, incluido una base de datos como Oracle o MySQL, un conjunto de documentos XML o incluso un servicio de directorio como el servidor LDAP. Además del tradicional mecanismo de almacenamiento relacional de bases de datos, existen muchas fuentes diferentes de datos a las que pueden acceder las aplicaciones.

De esta manera se puede lograr todo un sistema distribuido lo que permite mejorar la administración del software en términos de seguridad, escalabilidad y aprovechamiento de recursos. Se mejora la administración puesto que las reglas de negocio están instaladas en uno o más servidores y no en cientos

de clientes. La seguridad se ve mejorada ya que se puede centralizar el control de acceso en la capa de negocios.

1.4.2. Modelo Vista Controlador (MVC)

El Modelo Vista Controlador es un patrón de arquitectura de software utilizado en la aplicación para separar los datos, la interfaz de usuario, y la lógica de control en tres componentes distintos. Este patrón se utiliza frecuentemente en aplicaciones Web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página, el controlador es el sistema gestor de base de datos y el modelo es el modelo de datos.

La lógica de un interfaz de usuario cambia con más frecuencia que los almacenes de datos y la lógica de negocio. Por eso se trata de realizar un diseño que desacople la vista del modelo, con la finalidad de mejorar la reusabilidad. De esta forma las modificaciones en las vistas impactan en menor medida en la lógica de negocio o de datos.

MVC divide una aplicación interactiva en 3 áreas: procesamiento, salida y entrada. Para esto, utiliza las siguientes abstracciones:

1. **Modelo:** Es la representación específica de la información con la cual el sistema opera. La lógica de datos asegura la integridad de estos y permite derivar nuevos datos. El modelo es el responsable de acceder a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento. Además define las reglas de negocio, es decir, la funcionalidad del sistema. El modelo es independiente de cualquier representación de salida y/o comportamiento de entrada.
2. **Vista:** Presenta el modelo en un formato adecuado para interactuar. Es responsable de recibir los datos del modelo y mostrarlos al usuario, por lo que usualmente corresponde a la interfaz de usuario. Pueden existir múltiples vistas del modelo. Cada vista tiene asociado un componente controlador.
3. **Controlador:** Responde a acciones del usuario e invoca cambios en el modelo, y probablemente en la vista. Reciben las entradas, usualmente como eventos que codifican los movimientos o pulsación de botones del ratón, pulsaciones de teclas, etc. Los eventos son traducidos a solicitudes de servicio para el modelo o la vista.

Alguna de las razones por las cuales se emplea el patrón de Modelo-Vista-Controlador son:

1. Permitir desarrollar independientemente el modelo y las capas de interfaz para el usuario.

2. Reducir al mínimo el impacto que los cambios de requerimientos de la interfaz tienen en la capa de dominio.
3. Permitir conectar fácilmente otras vistas a la capa actual de dominio, sin que esto la afecte.
4. Permitir vistas simultáneas y múltiples del mismo objeto modelo.
5. Permitir transportar fácilmente la capa de modelo a otro esquema de interfaz para el usuario.

1.5. Lenguajes de programación.

Uno de los ejes fundamentales que diferencian a Internet de otros medios de comunicación es la interacción y personalización de la información con el usuario. Esto se logra por medio de alguno de los diferentes lenguajes de programación para Web que existen hoy en día; se clasifican en dos partes fundamentales que reconocen la propia arquitectura Cliente/Servidor de esta plataforma de desarrollo: los del lado del Servidor y los del lado del Cliente.

Existen numerosos lenguajes de programación utilizados para el desarrollo de Aplicaciones Web que se caracterizan por la arquitectura Cliente/Servidor, entre los que destacan: PHP, ASP/ASP.NET, JSP, Perl, Ruby y Python, aunque ASP no es un lenguaje de programación, sino una arquitectura de desarrollo Web en la que se pueden usar por debajo distintos lenguajes (por ejemplo VB.NET o C# para ASP.NET, o VBScript/JScript para ASP).

1.5.1. Lenguajes del lado del cliente.

El cliente es un software que se activa en el navegador cuando los usuarios acceden a la información localizada en computadoras remotas o locales a través del WWW (World Wide Web) o simplemente la Web. Es un sistema de hipertexto que funciona sobre Internet, para ello se utiliza una aplicación llamada navegador o browser que extrae elementos de información llamados "documentos" o "páginas Web" que se encuentran en los servidores Web y así mostrarlos en la pantalla del usuario, ejemplos de navegadores tenemos, el Netscape, Internet Explorer, Mozilla, entre otros. La programación en el cliente se utiliza para muchas cosas, ejemplos de ello son efectos diversos en las páginas, presentación del documento, sonidos, videos, menús interactivos, control y respuesta a las acciones de un usuario en la página, control sobre los formularios y validación de estos, entre otras funcionalidades. Los lenguajes que más se conocen que corren sobre el cliente son el HTML y el XHTML, debido a que son formatos estándar de interpretación, también encontramos el Visual Basic Script (VBScript) y el Java Script. Para el control de los diferentes eventos que ocurren en dicha página, de estos dos se escoge el Java Script. Para el Laboratorio Virtual de Redes también se utiliza el HTML.

1.5.1.1. HTML

Es un lenguaje de marcación diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas Web. Ofrece lineamientos para definir el despliegue de información en un Navegador o "Browser". Es un formato no propietario basado sobre el SGML (Lenguaje de Marcación Generalizado), y puede ser creado y ser procesado por una amplia gama de herramientas. El surgimiento de HTML se debió al eterno problema de formatos en sistemas de cómputo.

El hecho que HTML sea un estándar hace que su evolución sea muy conservadora, esto implica que funcionalidades adicionales sean difíciles de integrarse y formen parte del estándar como tal, ante esta problemática muchos navegadores han sido diseñados para ejecutar tareas que se encuentran fuera del estándar HTML, mientras esto permite diseños más versátiles, el uso de elementos HTML propietarios restringen que determinado diseño sea visualizado de la misma manera, o bien, correctamente en distintos Navegadores.

Un documento HTML puede ser generado en un editor de textos simplemente agregándole la extensión .html o .htm al archivo en cuestión, al utilizarse un editor de texto el archivo es manipulado sin ningún tipo de formato implícito. Algunos editores de texto son: Notepad para ambientes Windows, Emacs y Gedit para ambientes Unix/Linux.

El hecho de generar elementos HTML en un editor de textos puede resultar laborioso, por esta razón han surgido herramientas especiales para editar exclusivamente HTML que permiten generar documentos a través de iconos y ventanas, sin embargo, aunque inicialmente estas herramientas pueden acelerar el desarrollo de documentos HTML, en ocasiones su utilización entorpece el mismo desarrollo.

Al ser un lenguaje de marcación este carece de elementos dinámicos o en términos de programación: lógica de ejecución. Esta lógica en un navegador es llevada a cabo mediante un lenguaje de scripting, con el uso de Java Script el cual ya forma parte complementaria de prácticamente todo diseño en HTML.

1.5.1.2. Java Script

La característica principal de Java Script, es la de ser un lenguaje de scripting por excelencia. Es un lenguaje de programación compacto. El código de las aplicaciones Java Script se integra en el mismo archivo con el texto en formato HTML y es interpretado por el cliente (navegador), las características de interfaz como los cuadros de diálogo, formularios y otros elementos GUI (Interfaz Gráfico de Usuario), son gestionados por el navegador y por el código HTML. Permite crear efectos especiales e interactuar con el visitante. Con él se pueden controlar los elementos de la página (validación de formularios, entre otros). Otra función muy importante es la utilización de cookies, que permite reconocer a usuarios que ya

visitaron la página. Java Script no requiere un tiempo de compilación, los scripts se pueden desarrollar en un período de tiempo relativamente corto. Este lenguaje es seguro y fiable porque está en claro y hay que interpretarlo, por lo que puede ser filtrado; para Java Script, la seguridad es casi total. El código Java Script se ejecuta en el cliente por lo que el servidor no es solicitado más de lo debido; un script ejecutado en el servidor, sin embargo, sometería a éste a dura prueba y los servidores de capacidades más limitadas podrían resentir de una continua solicitud por un mayor número de usuarios.

1.5.2. Lenguajes del lado del servidor

Los lenguajes del lado del servidor se caracterizan por desarrollar la lógica de negocio dentro del Servidor, además de ser los encargados del acceso a Bases de Datos y del tratamiento de la información. La utilización de estos proporciona una seguridad muy satisfactoria, permitiendo esconder claves y datos importantes en la seguridad de las bases de datos. Pero lo más importante es la potencia que desarrollan sus herramientas de encriptación. Los lenguajes más utilizados actualmente son Perl, Asp (Active Server Pages), JSP (Java Server Pages), Ruby, Python y PHP (Hypertext Preprocessor Page).

Para el Módulo de Simulación se escogió trabajar con C++, para poder simular el movimiento de los paquetes por la red. Más adelante se detallan las características de este lenguaje.

Es importante seleccionar el lenguaje a utilizar, tanto del lado del servidor como del lado del cliente. A continuación se hace un breve análisis de los lenguajes de uso más común en la actualidad del lado del servidor, con vistas a hacer una apropiada selección.

1.5.2.1. Perl

Perl (Lenguaje Práctico para la Extracción e Informe) es un lenguaje interpretado que está principalmente orientado a la búsqueda, extracción y formateado de ficheros de tipo texto, también es muy usado para manejo y gestión de procesos (estado de procesos, conteo y extracción de parámetros característicos, etc.). Es un lenguaje de programación muy utilizado para construir aplicaciones CGI (common gateway interface), o scripts ejecutados desde páginas de la World Wide Web para el Web; y es considerado el lenguaje perfecto para este fin dadas sus facilidades en cuanto a la manipulación de texto. La forma del lenguaje facilita la programación rápida y sucia, el hacer un programa rápido para que funcione. Esto hace también que se utilice para hacer prototipos rápidos de algún algoritmo que se quiera ver funcionar antes de programarlo en un lenguaje más eficiente, como el C++. Es un lenguaje de libre uso. Antes estaba muy asociado a la plataforma Unix, pero en la actualidad está disponible en otros sistemas operativos como Windows.

1.5.2.2. ASP

Una de las características más importantes de las páginas ASP (Active Server Pages) es la posibilidad de conectar con diferentes tipos de bases de datos, para extraer, agregar o eliminar datos de ellas, y generar páginas con esos datos. Pueden conectarse a motores de bases de datos SQL, Access, Oracle, y a cualquier otro con soporte de conexión ODBC. Se utiliza exclusivamente en los servidores Web de Microsoft (Internet Information Server (IIS) y Personal Web Server) para páginas web generadas dinámicamente, lo cual constituye su principal desventaja. Los scripts ASP se ejecutan, por lo tanto, en el servidor y pueden utilizarse conjuntamente con HTML y Java Script para realizar tareas interactivas y en tiempo real con el cliente. Con ASP se pueden realizar fácilmente páginas de consulta de bases de datos, funciones sencillas como obtener la fecha y la hora actual del sistema servidor, cálculos matemáticos simples, etc. El ASP clásico está siendo reemplazado por ASP.NET desde 2002, que, entre otras cosas, reemplaza los lenguajes interpretados como VBScript o JScript por lenguajes compilados a código intermedio como Visual Basic, C#, o cualquier otro lenguaje que soporte la plataforma .NET. Para implementar ASP.NET es necesario instalar en el Servidor la Plataforma .NET.

1.5.2.3. JSP

Java Server Pages es una tecnología Java que permite a los programadores generar contenido dinámico para web, en forma de documentos HTML, XML, o de otro tipo. Las JSP's permiten al código Java y a algunas acciones predefinidas ser incrustadas en el contenido estático del documento web. La principal ventaja de JSP frente a otros lenguajes es que permite integrarse con clases Java lo que permite separar en niveles las aplicaciones web, almacenando en clases java las partes que consumen más recursos así como las que requieren más seguridad, y dejando la parte encargada de formatear el documento html en el archivo jsp. La idea fundamental detrás de este criterio es el de separar la lógica del negocio de la presentación de la información.

1.5.2.4. PHP

Es un lenguaje creado por una gran comunidad de personas que incluye funciones de correo electrónico, gestión de bases de datos, gestión de archivos, tratamiento de imágenes, tratamiento de cookies, accesos restringidos, comercio electrónico ó para propósito general (funciones matemáticas, explotación de cadenas, de fechas, corrección ortográfica, compresión de archivos). A esta inmensa librería cabe ahora añadir todas las funciones personales que uno ha creado por necesidades propias y que luego son

reutilizadas en otros sitios y todas aquellas intercambiadas u obtenidas en foros o sitios especializados. PHP es un lenguaje de código abierto interpretado, de alto nivel, embebido en páginas HTML, independiente de plataforma y muy rápido.

PHP es software libre, lo que implica menores costes y servidores más baratos que otras alternativas, a la vez que el tiempo entre el hallazgo de un fallo y su resolución es más corto. Funciona en toda máquina que sea capaz de compilar su código, entre ellas diversos sistemas operativos. El código escrito en PHP en cualquier plataforma funciona exactamente igual en cualquier otra.

PHP permite la programación de páginas Web dinámicas, habitualmente en combinación con el motor de base datos MySQL, aunque tiene capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad debido a que el acceso a las bases de datos de PHP es muy heterogéneo, pues dispone de un juego de funciones distinto por cada gestor. Permite leer y manipular datos desde diversas fuentes, incluyendo datos que pueden ingresar los usuarios desde formularios HTML.

PHP cuenta con un motor de plantillas denominado SMARTY que permite separar la lógica de la programación de la presentación, es decir, el código PHP del código HTML. Permite que una misma plantilla sea utilizada por varias páginas PHP que muestren el contenido en el mismo formato, independientemente de que los procedimientos usados para obtener la información a mostrar sean diferentes.

PHP tiene las cuatro grandes características necesarias para ser un potente lenguaje de scripts: velocidad, estabilidad, simplicidad y seguridad:

Velocidad: No requiere demasiados recursos de sistema. Por esta razón no crea demoras en la máquina.

Estabilidad: Con el respaldo de una increíble comunidad de programadores y usuarios es mucho más difícil para los errores sobrevivir. Se utiliza su propio sistema de administración de recursos y dispone de un sofisticado método de manejo de variables, conformando un sistema robusto y estable.

Seguridad: PHP provee diferentes niveles de seguridad.

Simplicidad: PHP permite a los programadores generar código en el menor tiempo posible.

Algunas de las principales ventajas de PHP son:

- Es un lenguaje muy fácil de aprender dada su similitud con Perl y C.
- Soporta la programación orientada a objetos.

- La validación de los parámetros que entra el usuario es realizada en el servidor y no en el cliente. Esto hace que el proceso de chequeo de los datos sea más eficaz y de que se pueda apoyar en toda una serie de funciones de seguridad ya implementadas.

Otra de las ventajas de utilizar PHP, es poder usar la librería ADOdb, una capa de abstracción de base de datos, de alta velocidad, y que tiene características avanzadas, como la gestión de sesiones, generación automática del código SQL, simulación de SELECT LIMIT para todas las bases de datos y monitorización del rendimiento. Esta capa lleva siendo utilizada desde el año 2000, actualmente cuenta con una comunidad amplia de usuarios.

PHP, con todas las ventajas que presenta por su propia esencia (es el más rápido de todos los analizados, es multiplataforma, con una sintaxis familiar a los estudiantes de la universidad, y cuenta con gran disponibilidad de recursos en Internet); y por el hecho de contar con un motor de plantillas como SMARTY y una capa de abstracción de datos como ADOdb, se convierte en la más atractiva de las opciones analizadas. Por ello se propone su uso como lenguaje del lado del servidor para el Laboratorio Virtual de Redes.

1.5.2.5. C++

El C++ es un lenguaje de programación diseñado a mediados de los años 1980, por Bjarne Stroustrup, como extensión del lenguaje de programación C, continuando con las ventajas, flexibilidad y eficacia de este lenguaje. Las principales características del C++ son el soporte para programación orientada a objetos y el soporte de plantillas o programación genérica. Se puede decir que C++ es un lenguaje que abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos.

Además posee una serie de propiedades difíciles de encontrar en otros lenguajes de alto nivel como la sobrecarga de operadores y la identificación de tipos en tiempo de ejecución.

C++ está considerado por muchos como el lenguaje más potente, debido a que permite trabajar tanto a alto como a bajo nivel. Es un lenguaje de programación que permite programar desde sistemas operativos, compiladores, aplicaciones de bases de datos, procesadores de texto o juegos. C++ genera código muy eficiente, es altamente transportable, es muy flexible y muy expresivo, y con él se pueden realizar muchas funciones escribiendo pocas líneas de código.

1.6. CSS

CSS es el lenguaje de hojas de estilo en cascada. En términos de páginas Web se puede establecer una diferencia entre los términos estilo lógico y estilo físico. El estilo lógico se refiere a la lógica del documento: cabeceras, párrafos, etc., no se preocupa de la apariencia final, sino de la estructura del documento. Por el contrario, el estilo físico no se preocupa de la estructura del documento, sino por la apariencia final: párrafos con un cierto tipo de letra, tablas con un determinado color de fondo, etc. La finalidad de las hojas de estilo es crear unos estilos físicos, separados de las etiquetas HTML y aplicarlos en los bloques de texto en los que se quieran aplicar. Estos estilos podrán ser modificados en algunas ocasiones desde Java Script, lo que brinda mayor interactividad.

Las hojas de estilo facilitan muchísimo la creación de páginas, ya que son las encargadas de separar el formato visual de las páginas del contenido. De esta forma, se logran webs más rápidas y accesibles tanto para usuarios como para buscadores. Por la ventaja que representan, se propone su utilización en la propuesta de solución del Laboratorio Virtual de Redes.

1.7. Herramientas de programación.

En el Laboratorio Virtual de Redes se propone utilizar como herramienta de programación Web la Macromedia Dreamweaver, por las funcionalidades que nos ofrece en el soporte de las tecnologías web como CSS, Java Script y PHP, y por otras características que se presentan a continuación. Para el módulo de simulación se propone programar en C++ Builder, utilizando OpenGL como librería gráfica 3D, y además las librerías GLU, GLUT y GLAUX.

1.7.1. Macromedia Dreamweaver

Esta herramienta es un editor de páginas web, creado por Macromedia. Es el programa de este tipo más utilizado en el sector del diseño y la programación web, por sus funcionalidades y su integración con otras herramientas como Macromedia Flash. Tiene soporte tanto para edición de imágenes como para animación a través de su integración con otras herramientas. La superioridad de este editor sobre otros se basa en su gran poder de ampliación y personalización. Esto hace que los archivos del programa no sean instrucciones de C++, sino rutinas Java Script, que hacen que sea un programa muy fluido. Las versiones más recientes soportan otras tecnologías web como CSS, Java Script y algunas plataformas del lado servidor. Esta aplicación está disponible tanto para la plataforma MAC como Windows, aunque también se puede ejecutar en plataformas basadas en UNIX utilizando emuladores.

Dreamweaver permite al usuario utilizar la mayoría de los navegadores Web instalados en su ordenador para visualizar previamente las páginas web. También dispone de herramientas de administración de sitios dirigidas a principiantes como, por ejemplo, la habilidad de encontrar y reemplazar líneas de texto y código por cualquier tipo de parámetro especificado, hasta el sitio web completo. El panel de comportamientos también permite crear Java Script básico sin conocimientos de código.

Un aspecto de alta consideración de Dreamweaver es su arquitectura extensible, es decir, permite el uso de "Extensiones". Las extensiones son pequeños programas, que cualquier desarrollador web puede escribir, normalmente en HTML y Java Script, y que cualquiera puede descargar e instalar, ofreciendo así funcionalidades añadidas a la aplicación. Dreamweaver cuenta con el apoyo de una gran comunidad de desarrolladores de extensiones que hacen posible la disponibilidad de extensiones gratuitas y de pago para la mayoría de las tareas de desarrollo web.

1.7.2. Builder

C++ Builder es una aplicación Windows creada por la compañía de software Borland Software Corporation. Esta herramienta proporciona un entorno de trabajo visual para construir aplicaciones Windows, integrando distintos aspectos de la programación en un entorno unificado o integrado. La integración y facilidad de manejo hace que sea una herramienta indispensable para el desarrollo rápido de aplicaciones. El principal inconveniente del C++ Builder como herramienta de desarrollo, es que no se crea una aplicación multiplataforma.

La primera versión fue lanzada después de Delphi, y con un entorno muy similar a este. Muchos componentes de Delphi pueden utilizarse en C++ Builder. La última versión es C++Builder 2006, de la que existen dos ediciones: Professional y Enterprise.

El IDE (Entorno de Desarrollo Integrado) de C++ Builder es una aplicación Windows 95, y como tal, cumple con los estándares de aspecto, diseño y comportamiento que aconseja Microsoft a los desarrolladores de aplicaciones. Algunas de las características de esta herramienta son:

1. Productividad de Desarrollo:
 - Visor de código, y completamiento de código y de parámetros.
 - IDE personalizable con ventanas de herramienta flotantes. Repositorio de objetos para almacenar y reutilizar formas, módulos de datos y asistentes. Tipos de proyectos predefinidos.
 - Herencia y encadenamiento visual de formas.
2. Depuración avanzada:
 - Depuración de las DLL para un más fácil y completo control de depuración.

- Visor de CPU para la depuración de bajo nivel.
 - Inspector de depuración para supervisar las propiedades de los componentes.
 - Depuración multiproceso.
3. Soporte integrado a bases de datos:
- Actualizaciones en caché, soporte para múltiples bases de datos y API de bajo nivel.
 - Componentes inteligentes de datos para construir poderosas aplicaciones de base de datos.
 - Explorador de Base de Datos, para navegar visualmente y administrar tablas, alias, e índices.
4. Compilación del lenguaje C++:
- Compilador de código nativo de 32-bit, Borland ANSI C++ de alto rendimiento.
 - Arquitectura de componentes orientada a objetos totalmente extensible.
 - Encadenador incremental para desarrollos increíblemente rápidos.

1.7.2.1. OpenGL

OpenGL es una biblioteca estilizada de trazado de gráficos de alto rendimiento, es sin lugar a dudas, la API que prevalece en la industria para desarrollar aplicaciones gráficas 2D y 3D. Los objetos en dos o tres dimensiones son pintados como una secuencia de vértices, que definen los objetos geométricos, o de píxeles, que definen las imágenes. OpenGL realiza varios pasos de procesamiento de estos datos, para convertirlos a los píxeles para formar la imagen deseada.

OpenGL es por diseño independiente de plataformas y sistemas operativos. Además es perceptiva a la red, de manera que es posible separar nuestra aplicación OpenGL en un servidor y un cliente que verdaderamente produzca los gráficos. Gracias a su independencia del sistema operativo, el servidor y el cliente no tienen por qué ejecutarse en el mismo tipo de plataforma, muy a menudo el servidor ejecutará una compleja simulación y el cliente una simple estación de trabajo mayormente dedicada a la visualización gráfica.

La utilización de OpenGL consiste en activar y desactivar opciones, y realizar ciertas acciones, que tendrán como fruto una representación en pantalla (o no) de una serie de datos.

Algunas de las características que OpenGL implementa son:

- Las primitivas geométricas permiten construir descripciones matemáticas de objetos. Las actuales primitivas son: puntos, líneas, polígonos, imágenes y bitmaps.
- La visualización y modelado permite disponer objetos en una escena tridimensional, mover una cámara por el espacio y seleccionar la posición ventajosa deseada para visualizar la escena de composición.

- El mapeado de texturas ayuda a traer realismo a los modelos, por medio del dibujo de superficies realistas en las caras de modelos poligonales.
- La iluminación de materiales es una parte indispensable de cualquier gráfico 3D. OpenGL provee de comandos para calcular el color de cualquier punto, dadas las propiedades del material y las fuentes de luz en la habitación.
- El doble buffering ayuda a eliminar el parpadeo de las animaciones. Cada fotograma consecutivo en una animación se construye en un buffer separado de memoria y mostrado solo cuando está completo.
- El Anti-alisado reduce los bordes escalonados en las líneas dibujadas sobre la pantalla.
- El Z-buffering mantiene registros de la coordenada Z de un objeto 3D. El Z-buffer se usa para registrar la proximidad de un objeto al observador, y es también crucial para el eliminado de superficies ocultas.

La librería principal de OpenGL nos suministra todas las funciones necesarias para mostrar un entorno 3D; aunque hay algunas operaciones que son algo tediosas de realizar utilizando solo esta librería. Para esto se han desarrollado también algunas librerías auxiliares:

- GLU: Esta librería acompaña a la OpenGL principal. Incluye funciones más complejas que GL, por ejemplo las de definir un cilindro o un disco con un solo comando, también contiene operaciones para trabajar con matrices.
- GLUT: Las aplicaciones OpenGL pueden usar la biblioteca GLUT como su manejador de ventanas. Esta biblioteca es dependiente de la plataforma, y ofrece un paradigma común para el sistema de ventanas y dispositivos periféricos.
- GLAUX: Esta librería es muy parecida a GLUT; es la que Microsoft ha desarrollado para Windows. Mantiene prácticamente la misma estructura con el defecto de que solo sirve para Windows, mientras que GLUT sirve para cualquier plataforma.

También hay otras librerías más específicas para el control de entradas, de sonido y la red.

1.8. Zend Development Enviroment

La tecnología Zend es el proveedor líder de productos y servicios para desarrollar y administrar las aplicaciones PHP. Zend Studio 5, concebido con el fin de crear aplicaciones altamente fiables, proporciona una facilidad de uso inigualable, escalabilidad, fiabilidad, y la extensión que los programadores requieren para desarrollar, distribuir, depurar y administrar aplicaciones PHP. Asegura el

desarrollo de software mediante la combinación del IDE líder para PHP, con un entorno de prueba que agiliza la seguridad de la calidad, integración y las etapas de los procesos.

Algunas características que lo hacen el entorno de desarrollo más poderoso para PHP son:

- Integra el uso y completamiento de código personalizado de Zend Framework y vista de la lista de las funciones del framework desde la Visualización de Funciones PHP.
- Aumenta la productividad con el soporte de PHP 5 completo, analizador de código, coloreado de sintaxis, administrador de proyecto, editor de código, depurador de gráficos y asistentes.
- Documenta el código de forma más sencilla.
- Simplifica el despliegue, con la integración FTP y SFTP de forma tal que permita a los programadores en forma segura subir y descargar archivos de proyectos de modo transparente hacia y desde servidores remotos.

Zend Studio 5 permite el desarrollo de aplicaciones de negocios superiores. Permite conectarse directamente con bases de datos profesionales como MySQL, Oracle, Microsoft SQL Server, PostgreSQL y SQLite, escribir y realizar consultas a servidores conectados usando el editor de consultas y visualizar las estructuras de la base de datos y administrar el contenido con el explorador SQL de Zend.

Contiene un entorno de depuración completo para PHP, en forma local y remota. Incluye características de depuración avanzadas como: condiciones límites, visualización de errores, vistas avanzadas, variables y buffer de salida.

1.9. Sistema de gestión de base de datos.

“Los Sistemas de gestión de base de datos son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta.” **[Sis07]**

Un Sistema Gestor o Manejador de Bases de Datos (SGBD) es un conjunto de programas que permite a los usuarios crear y mantener una Base de Datos (BD), por lo tanto, es un software de propósito general que facilita el proceso de definir, construir y manipular la BD de manera clara, sencilla y ordenada, para diversas aplicaciones.

En la actualidad existe una gran variedad de SGBD, tanto de tipo comercial como libre. Algunos ejemplos son: PostgreSQL, MySQL, Firebird, SQLite que son SGBD libres, y otros como Oracle, Progress y Microsoft SQL Server que son comerciales. Para seleccionar el SGBD para la propuesta de solución del Laboratorio Virtual de Redes se hace un breve análisis de los de uso más común en la actualidad con vistas a hacer una apropiada selección. Cada sistema tiene características, ventajas e inconvenientes, la

elección de uno u otro sistema para gestionar nuestra base de datos vendrá definida por nuestras necesidades.

1.9.1. Oracle

Oracle es considerado el SGBD más completo que existe. Sus características más destacadas son el soporte de transacciones, su gran estabilidad y seguridad, su escalabilidad, así como que es un sistema multiplataforma, entre otras ventajas. En sus inicios fue muy revolucionario dado que usaba la filosofía de bases de datos relacionales, algo que por los años 70, fecha en que surge Oracle, era todavía desconocido. Su mayor defecto es su enorme precio. Otro aspecto que ha sido criticado por algunos especialistas es la seguridad de la plataforma, y las políticas de suministro de parches de seguridad. En los parches de actualización provistos durante el primer semestre del 2005 fueron corregidas 22 vulnerabilidades públicamente conocidas, algunas de ellas con una antigüedad de más de 2 años.

Aunque su dominio en el mercado de servidores empresariales ha sido casi total hasta hace poco, recientemente sufre la competencia del Microsoft SQL Server de Microsoft y de la oferta de otros con licencia libre como PostgreSQL, MySql o Firebird. Las últimas versiones de Oracle han sido certificadas para poder trabajar bajo Linux.

1.9.2. SQL Server

Este gestor de bases de datos posee características peculiares que lo clasifican como la base de datos más fácil de utilizar para construir, administrar e implementar aplicaciones de negocios. Tiene a disposición un modelo de programación rápido y sencillo para desarrolladores, eliminando la administración de base de datos para operaciones estándar, y suministrando herramientas sofisticadas para operaciones más complejas.

Es un potente SGBD que está totalmente habilitado para Web. Ostenta marcas de referencia en cuanto a escalabilidad y confiabilidad, que son críticas para el éxito de bases de datos de gran tamaño. Presenta mejoras en lenguaje debido al empleo de triggers, procedimientos almacenados, disparador de eventos antes y después de conexiones, procedimientos almacenados extendidos utilizando C/C++ y sentencias DLL permitidas dentro de transacciones. En cuanto a los aspectos de seguridad se puede resumir que trabaja con un único ID de login tanto para red como para la DB para mejorar la seguridad y facilitar la administración, proporciona password y encriptación de datos en red, encriptación de procedimientos almacenados para la integridad y seguridad de código de aplicación.

El SQL Server permite lograr una gran velocidad en el procesamiento de transacciones, y agilidad en todas sus operaciones. A pesar de todas las ventajas que presenta este SGBD, tiene el inconveniente de que, al igual que Oracle, no es un sistema libre.

1.9.3. PostgreSQL

Por otra parte está PostgreSQL que proporciona un gran número de características que normalmente sólo se encontraban en las bases de datos comerciales de alto calibre, tales como DB2 u Oracle. Es un SGBD objeto-relacional, aproximando los datos a un modelo objeto-relacional. Es capaz de manejar complejas rutinas y reglas. Su avanzada funcionalidad se pone de manifiesto con las consultas SQL declarativas, el control de concurrencia multiversión, soporte multiusuario, transacciones, optimización de consultas, herencia y valores no atómicos. Es altamente extensible: soporta operadores y tipos de datos definidos por el usuario.

PostgreSQL cuenta con una API flexible, lo que ha permitido dar soporte para el desarrollo en diversos lenguajes de programación entre los que se incluyen: Object Pascal, Python, Perl, PHP, Java/JDBC, Ruby, TCL, C/C++. Además tiene soporte para lenguajes procedurales internos. Muchos desarrolladores en sus discusiones a través de la Web en torno al tema de qué es mejor: MySQL o PostgreSQL, recomiendan la utilización de PostgreSQL para la elaboración de un sistema robusto y para lograr mayor escalabilidad. La mayoría coincide en que cada SGBD tiene sus ventajas y desventajas, y que la elección de uno de los dos depende de lo que se quiera construir. Se destaca sobre todo que MySQL ha avanzado vertiginosamente comparado con PostgreSQL que ya lleva alrededor de 15 años de desarrollo.

1.9.4. MySQL

MySQL Database Server es la base de datos de código fuente abierto más usada del mundo. Su ingeniosa arquitectura lo hace extremadamente rápido y fácil de personalizar. MySQL es un servidor de bases de datos relacionales muy rápido, fiable, fácil de usar y robusto, se desarrolló bajo la filosofía de código abierto. Es software libre, publicado bajo la licencia GPL (GNU Public License) y mantenido por la compañía sueca MySQLAB.

Este gestor se creó con la rapidez en mente, de modo que no tiene muchas de las características de los gestores comerciales más importantes, como Oracle, Sybase o SQL Server. No obstante, eso no ha impedido que sea el más indicado para aplicaciones que requieren muchas lecturas y pocas escrituras y no necesiten de características muy avanzadas, como es el caso de las aplicaciones Web. Su bajo consumo lo hacen apto para ser ejecutado en una máquina con escasos recursos sin ningún problema,

además el conjunto de aplicaciones Apache-PHP-MySQL es uno de los más utilizados en aplicaciones en ambiente Web.

Su conectividad, velocidad, seguridad hacen que el Servidor de MySQL sea uno de los más preferidos a la hora de acceder las bases de datos en Internet, este consume muy pocos recursos de CPU y memoria, además de proporcionar muy buen rendimiento, tamaño del registro sin límite, buena integración con PHP, utilidades de administración y un buen control de acceso usuarios-tablas-permisos.

Es uno de los Sistemas Gestores de bases de datos más populares, en la Web, el mismo se puede identificar por:

- Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente.
- Disponible en una gran cantidad de plataformas.
- Diferentes opciones de almacenamiento en dependencia de la velocidad en las operaciones.
- Permite las transacciones y claves foráneas.
- Mantiene una conectividad segura.
- Permite la replicación.
- Implementa la búsqueda e indexación de campos de texto.

Por las características de este SGBD se convierte en la más atractiva de las opciones analizadas. Por ello se propone su uso como lenguaje del lado del servidor para el Laboratorio Virtual de Redes.

1.10. Servidor de la aplicación.

El servidor Web es un programa que corre sobre el servidor que escucha las peticiones HTTP que le llegan de las máquinas clientes y las satisface. Dependiendo del tipo de la petición, el servidor Web buscará una página Web o bien ejecutará un programa en el servidor. De cualquier modo, siempre devolverá algún tipo de resultado HTML al cliente o navegador que realizó la petición. Algunos servidores Web importantes son: Apache, IIS y Cherokee. Para el laboratorio Virtual de Redes se elige Apache.

1.10.1. Apache.

El servidor Apache es el servicio que se encarga de resolver las peticiones de páginas de Internet de los clientes utilizando el protocolo de Internet HTTP. Apache, es totalmente gratuito, y permite la modificación del código. Es además extensible pues se pueden añadir módulos para ampliar sus capacidades. Hay una amplia variedad de módulos, que permiten desde generar contenido dinámico (con PHP, Java, Perl, Python), monitorizar el rendimiento del servidor, atender peticiones encriptadas por SSL, hasta crear servidores virtuales por direcciones IP o por nombre (varias direcciones Web son manejadas en un mismo

servidor) y limitar el ancho de banda para cada uno de ellos. Dichos módulos incluso pueden ser creados por cualquier persona con conocimientos de programación. Apache presenta entre otras características mensajes de error altamente configurables, bases de datos de autenticación, negociado de contenido y una característica fundamental es que es multiplataforma.

1.11. Características de la tecnología WAMP

La plataforma WAMP (Windows, Apache, MySQL, PHP) está considerada como una de las mejores herramientas disponibles para que cualquier organización o individuo pueda emplear un servidor Web versátil y potente. Aunque creados por separado, cada una de las tecnologías que lo forman dispone de una serie de características comunes. Especialmente interesante es el hecho que estos cuatro productos pueden funcionar en una amplia gama de hardware, con requerimientos relativamente pequeños sin perder estabilidad. Esto ha convertido a WAMP en la alternativa más adecuada para pequeñas y medianas empresas.

WAMP es un sistema indicado para los usuarios que no tienen instalado en el sistema ninguno de los programas necesarios para programar en PHP (Apache, PHP y MySQL), ya que realiza una instalación completa y desde cero. Pero también pueden utilizar este programa los usuarios que disponen de Apache, PHP y/o MySQL en su sistema. En cuyo caso, simplemente se realizará otra copia de las aplicaciones en un directorio distinto, que en principio, no tiene por qué interferir con las otras instalaciones alojadas en el equipo.

Se utiliza la versión WAMP5 para el desarrollo del laboratorio virtual de redes.

1.12. Metodología a utilizar.

Uno de los temas más comunes en el mundo de la informática hoy en día es el de las metodologías de desarrollo de software: cómo trabajar eficientemente evitando las catástrofes que conllevan al fracaso de un gran porcentaje de proyectos. En estos últimos años se han desarrollado dos corrientes en lo referente a los procesos de desarrollo, los llamados métodos pesados y los ligeros. La diferencia fundamental entre ambos es que mientras los métodos pesados intentan conseguir el objetivo común por medio de orden y documentación, los ligeros (también llamados métodos ágiles) tratan de mejorar la calidad del software por medio de una comunicación directa e inmediata entre las personas que intervienen en el proceso. Existen ciertas tendencias fundamentadas en la idea de construir sistemas más grandes y complejos. Se

quiere un software que este mejor adaptado a nuestras necesidades lo que a su vez hace que el software sea más complejo, pero no solo eso, sino que además, se requiere de un software lo más rápido posible. En un proyecto de desarrollo de software la metodología define quién debe hacer qué, cuándo y cómo debe hacerlo. No existe una metodología de software universal. Las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigen que el proceso sea configurable. En la actualidad existen varias metodologías orientadas a objetos basadas en Lenguaje Unificado de Modelado como Rational Unified Process (RUP), Metvisual, ADOOSI, OPEN, entre otros.

Para controlar, y planificar la propuesta que presenta este trabajo, se decide utilizar como metodología RUP, por sus características y las facilidades que aporta, teniendo en cuenta de que viene acompañado de una herramienta que soporta cada uno de los procesos que se necesitan: Rational Rose Enterprise Edition 2003.

1.12.1. Rational Rose Enterprise

Rational Rose es una poderosa herramienta que se utiliza para modelar el sistema antes de escribir cualquier código, garantizando que este tenga una arquitectura sana desde el principio. Cubre todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases y entregables. Usando el modelo, se pueden coger defectos de diseño temprano.

Un modelo de Rose es un cuadro de un sistema de varias perspectivas, incluye todos los diagramas de UML, agentes, casos del uso, objetos, clases, componentes, y nodos del despliegue en un sistema. Describe en gran detalle qué el sistema incluirá y cómo trabajará. Rational ofrece un Proceso Unificado (RUP) para el desarrollo de los proyectos de software, desde la etapa de Ingeniería de Requerimientos hasta la de pruebas. Para cada una de estas etapas existe una herramienta de ayuda en la administración de los proyectos, Rose es la herramienta del Rational para la etapa de análisis y diseño de sistemas. Sin un diseño documentado, es difícil estar seguro de que el sistema que se construyó, es realmente el sistema que los usuarios tenían en mente.

Rational Rose es la herramienta de modelación visual que provee el modelado basado en UML. El navegador UML de Rational Rose nos permite establecer una trazabilidad real entre el modelo y el código ejecutable. Facilita el desarrollo de un proceso cooperativo en el que todos los agentes tienen sus propias vistas de información (vista de Casos de Uso, vista Lógica, vista de Componentes y vista de Despliegue), pero utilizan un lenguaje común para comprender y comunicar la estructura y la funcionalidad del sistema en construcción.

1.12.2. El Proceso Unificado del Rational. RUP

Esta es una de las metodologías más generales de las que existen en la actualidad, pues está pensada para adaptarse a cualquier proyecto, no sólo de software. El Proceso Unificado (RUP) para el desarrollo de los proyectos de software, es una propuesta para el desarrollo de software basada en la orientación a objetos, el desarrollo iterativo y el modelamiento visual usando el Lenguaje Unificado de Modelado (UML) para describir un sistema, lo cual permite incorporar al proceso de desarrollo de software un mejor control de los requerimientos y cambios.

RUP es uno de los procesos más generales de los existentes actualmente, es el resultado de varios años de desarrollo y uso práctico en el que se han unificado técnicas de desarrollo y trabajo de muchas metodologías utilizadas por los clientes. La versión que se ha estandarizado vio la luz en 1998 y se conoció en sus inicios como Proceso Unificado de Rational 5.0; de ahí las siglas con las que se identifica. A pesar de ser una metodología desarrollada directamente para el trabajo con clases y objetos brinda amplias posibilidades con el manejo eficiente del tiempo de diseño e implementación de aplicaciones Web.

Un proceso realizado siguiendo RUP se divide en cuatro fases:

- Intercepción (puesta en marcha).
- Elaboración (definición, análisis, diseño).
- Construcción (implementación).
- Transición (fin del proyecto y puesta en producción).

Características de RUP

- Guiado por Casos de Uso.

El software surge para dar servicio a sus usuarios; por tanto, para construir un sistema con éxito debemos conocer lo que sus futuros usuarios necesitan y desean. Un caso de uso es un fragmento de funcionalidad del sistema que proporciona al usuario un resultado importante, representan los requerimientos funcionales. Todos los casos de uso juntos describen la funcionalidad total del sistema. Los casos de uso guían la arquitectura del sistema y la arquitectura del sistema influye en la selección de los casos de uso.

- Centrado en la arquitectura.

La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo,

desarrollarlo y producirlo económicamente. La arquitectura en un sistema software se describe mediante diferentes vistas del sistema en construcción. Este concepto incluye los aspectos estáticos y dinámicos más significativos del sistema. Esta se refleja en los casos de uso pues cada producto tiene tanto una función como una forma, ninguna es suficiente por sí sola. RUP se desarrolla mediante iteraciones, comenzando por los CU relevantes desde el punto de vista de la arquitectura.

- Iterativo e incremental.

RUP propone que cada fase se desarrolle en iteraciones. Resulta práctico dividir el trabajo en partes más pequeñas o mini proyectos, los cuales no son más que iteraciones que resultan en un incremento. Una iteración es una secuencia de actividades de todos los flujos de trabajo con un plan establecido y criterios de evaluación, cuyo resultado es una versión del software. Los beneficios de las iteraciones son que reduce el coste del riesgo al coste de un solo incremento, hay menos riesgo de no sacar el producto al mercado en fecha que acelera el ritmo de desarrollo. Las necesidades del usuario y correspondientes requisitos no pueden definirse completamente al principio, se requieren iteraciones sucesivas.

1.13. UML

UML es el lenguaje de modelado unificado (Unified Modeling Language) con una especificación de notación orientada a objetos. Divide cada proyecto en un número de diagramas que representan las diferentes vistas, estos diagramas en su conjunto son los que definen la arquitectura del proyecto.

UML es un estándar, no existe otra especificación de diseño orientado a objetos, puesto que es el resultado de las tres opciones existentes en el mercado. Su utilización es independiente del lenguaje de programación y de las características de los proyectos. Ha sido diseñado para modelar cualquier tipo de solución.

Uno de los objetivos principales de la creación de UML era posibilitar el intercambio de modelos entre las distintas herramientas CASE orientadas a objetos del mercado. Para ello era necesario definir una notación y semántica común. Hay que tener en cuenta que el estándar UML no define un proceso de desarrollo específico, sino que solo se trata de una notación.

1.14. Propuesta

Tomando como base los elementos antes expuestos de algunas de las herramientas y tecnologías disponibles actualmente, se puede plantear una propuesta que consiste en desarrollar una aplicación sobre plataforma Web, que siga el modelo de un laboratorio virtual, que incluya inicialmente, además de

otras funcionalidades, cómo realizar las prácticas de diseño de redes. Esta aplicación se propone basada en una arquitectura cliente servidor de 3 capas, utilizando el patrón de arquitectura de software Modelo Vista Controlador, para separar los datos, la interfaz de usuario, y la lógica de control. Utilizando como lenguaje de programación del lado del servidor Web el PHP, dada su portabilidad y eficiencia, y el Java Script para lograr una interactividad con el usuario en el navegador. Para el módulo de simulación del funcionamiento de algunos dispositivos de redes, se propone utilizar el lenguaje de programación C++. Se propone también la utilización de MySQL como SGBD, y de Apache como servidor Web. Como metodología de desarrollo se utilizará RUP, con UML como el lenguaje de modelado unificado.

Conclusiones

En este capítulo se hizo un análisis de las tecnologías a utilizar en el desarrollo de la propuesta de solución. Se fundamentó la elección de los lenguajes de programación, el sistema gestor de bases de datos, el servidor Web, y la metodología de desarrollo de software; así como el uso de otras tecnologías. Finalmente se planteó la propuesta que incluye dichos aspectos. A partir de este punto se comenzará el desarrollo de la propuesta del sistema.

CAPITULO 2: CARACTERÍSTICAS DEL SISTEMA

El objetivo de este capítulo es la descripción del problema que da origen al trabajo. Se aborda el objeto de estudio, y conjuntamente se realiza la propuesta de sistema a realizar para el proceso que será automatización. Al no existir un negocio que defina los procesos, para poder entender el contexto en que se enmarca el sistema, se hizo necesario definir conceptos que se pudieran agrupar en un Modelo de Dominio, para capturar correctamente los requisitos y construir la aplicación. Se enumeran los requisitos funcionales y no funcionales, lo que permite entender la concepción general e identificar mediante un Diagrama de Caso de Uso las relaciones de los actores y las secuencias de acciones con las que interactúan.

2.1. Objeto de estudio

2.1.1. Problema y situación problemática

La asignatura Teleinformática, perteneciente al Departamento de Sistemas Digitales, tiene organizado el contenido de clases correspondiente a esta ciencia de un modo diferente a lo que hasta ahora estaba concebido, utilizando un Entorno Virtual de Aprendizaje (EVA). En el EVA implementado en la UCI (Teleformación), se realiza la distribución y gestión de cursos a través de la red a todos los usuarios del dominio; para mejorar la calidad del proceso de enseñanza aprendizaje. Dentro de estos cursos se encuentran Teleinformática I y II, propuestos para todos los estudiantes de tercer año. Así los estudiantes y profesores cuentan con una forma más independiente de investigar, y se aprovecha la posibilidad de interactuar de modo síncrono o asíncrono entre estos.

Una de las misiones de la UCI es proporcionar una buena formación a sus alumnos. En esa formación, indudablemente la parte práctica juega un papel imprescindible. Es a través de ella que el estudiante fijará conceptos teóricos y adquirirá procedimientos fundamentales del área de conocimiento que esté estudiando. Para garantizar una preparación académica de gran calidad, así como el entrenamiento profesional, la asignatura Teleinformática se propone el objetivo de que los estudiantes al conocer los fundamentos de la teoría de la información y transmisión de datos y las tecnologías de redes de computadoras, sean capaces de diseñar una red; además deben conocer el funcionamiento de los dispositivos de red más importantes. Para realizar el diseño de una red se deben seleccionar algunos elementos de redes necesarios como: accesorios, dispositivos, equipamiento, conectores, equipos pasivos, servidores, medios de transmisión y topología física a utilizar, siempre teniendo en cuenta el presupuesto planteado. Toda esta tecnología es muy costosa como para presentarla físicamente en las

clases, y los estudiantes no cuentan con ninguna herramienta virtual que les permita realizar con éxito las prácticas.

Los ejercicios de diseño de redes informáticas están planificados para realizarse sólo una vez en el transcurso de la asignatura por no contar con las herramientas necesarias para practicarlos de forma sistemática. Esto atenta contra el buen desempeño de los profesores o alumnos ayudantes; pues se requiere de un gran esfuerzo laboral para impartir los conocimientos imprescindibles a la hora de diseñar las redes. Esto evidentemente, conlleva a una gestión de las prácticas de la asignatura deficiente y sin la calidad y organización requeridas en un centro de estudios superiores. También los profesores sienten la necesidad de contar con un soporte informático que les ayude a señalar los errores y otorgar una calificación de los trabajos finales, para agilizar el proceso.

En la asignatura se utiliza un simulador para equipamiento Cisco de Routers (encaminadores) y Switches (conmutadores), el RouterSim Network Visualizer 4.0 BSCI, con el objetivo de servir de ayuda a los estudiantes, sustituyendo a un laboratorio de routers reales. Este simulador cuenta con distintos tipos de enlaces y configuraciones a la hora de construir LANs y VLANs, conectando y configurando los routers; pero no muestra el funcionamiento de los encaminadores o de los conmutadores de red. Esto conlleva a que los estudiantes desconozcan las funcionalidades internas de estos dispositivos de red.

Todo lo antes expuesto implica la necesidad inminente de aplicar una solución efectiva al control de la información generada en el proceso de gestión de las prácticas de la asignatura Teleinformática, y simular el funcionamiento de los dispositivos de red más importantes, en aras de contribuir al mejoramiento y calidad del proceso docente.

2.1.2. Objeto de automatización

La Universidad de las Ciencias Informáticas desea realizar un sistema automatizado para la gestión de las prácticas de la asignatura Teleinformática. Las actividades prácticas de diseño de redes presentan ejercicios con pasos a seguir como: determinar el tipo de cable y la topología física (cableado) a utilizar, decidir cuál de las distintas topologías de Ethernet se deberá utilizar, desarrollar una topología de LAN de la Capa 2, desarrollar una topología de la Capa 3, y diseñar un esquema óptimo de cableado estructurado, entre otros.

Para desarrollar el diseño se deben agregar los distintos elementos de red necesarios para cada capa como:

- Los accesorios de red, por ejemplo: conectores, marcadores, argollas de sujeción de cables, chasis, jumper dúplex.

- Los dispositivos de red, por ejemplo: Modems, Routers y Switches.
- El equipamiento de red, como: UPS, computadoras, impresoras.
- Los equipos pasivos, por ejemplo: canaletas, armarios, tomas de red.
- Los medios de transmisión, por ejemplo: cables de fibra óptica, cables coaxiales.
- Los servidores, por ejemplo: Print Server y Proxy.

Los datos de los distintos elementos de red (código, descripción, nombre, precio, proveedor, categoría y unidad de medida) se tienen que mantener actualizados para presentarlos a los estudiantes.

Los profesores orientan un ejercicio de diseño de una red, incluyendo el texto de este, con alguna imagen de algún plano del área donde se desea montar la red; y la respuesta correcta a este ejercicio. A los estudiantes se les presentan los ejercicios propuestos, y ellos proceden a dar respuesta a estos, guardándose en su historial los pasos por los que han pasado en el sitio. Cada respuesta debe tener el estándar y la topología de red que se utiliza.

Existen documentos, o archivos que los profesores pueden publicar, materiales complementarios que necesitan ponerse al alcance de los estudiantes para la superación individual y para el apoyo de la asignatura. Además se necesita generar un reporte con los errores y la calificación obtenida por los estudiantes en cada ejercicio, para mejorar el nivel de enseñanza docente.

Mediante las estadísticas se pueden apreciar los resultados por ejercicio o por estudiante, y ayudar a valorar cómo se comporta el proceso docente mediante gráficos que les facilitan la valoración a los profesores del aprendizaje de sus estudiantes.

2.1.3. Información que se maneja

1. Registro de usuarios.

Es un modelo en el que se recogen los datos característicos de cada usuario.

2. Registro de ejercicios.

En él se recoge la orden del ejercicio, las imágenes de los planos que se utilizan y otros datos de apoyo.

3. Registro de respuestas.

Este modelo recoge la topología, el estándar y los datos de los productos necesarios para diseñar la red propuesta.

4. Registro de productos.

Este modelo recoge los datos de todos los elementos de red necesarios para diseñar una red.

2.1.4. Propuesta de sistema

Con el objetivo de permitir a los profesores de la asignatura Teleinformática la gestión de las prácticas de diseño de redes en un centro de estudios superiores, se propone el Laboratorio Virtual de Redes, el cual contribuirá al mejoramiento y la calidad del proceso docente. La gestión de los ejercicios prácticos de diseño de redes y la simulación de algunos dispositivos van a ser sus funcionalidades más importantes.

Este sistema permitirá el control automatizado de los resultados en las actividades prácticas, por ejercicio o por estudiante, mostrando estadísticas a los profesores. Se podrán generar reportes de los resultados obtenidos por usuario en los ejercicios. Los profesores podrán operar el historial de cada estudiante para saber las acciones que han realizado en el sistema, facilitarán materiales complementarios y podrán mantener actualizado el listado de productos necesarios para diseñar una red. Se permitirá también la simulación del envío de paquetes entre subredes, y la simulación interna de algunos dispositivos.

2.2. Modelo de dominio.

Un modelo del dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las cosas que existen o los eventos que suceden en el entorno en el que trabaja el sistema.

Sin cambiar ninguno de los procesos existentes, se da un nuevo enfoque a la realización de las prácticas de la asignatura Teleinformática, basándose en un modelo de dominio que permita de manera visual mostrar los principales conceptos que se manejan en el contexto del sistema en desarrollo, ayudando a los usuarios a utilizar un vocabulario común. Para capturar correctamente los requisitos y poder construir un sistema correcto se necesita tener un firme conocimiento del funcionamiento del objeto de estudio. Por tanto, dicho modelo, va a contribuir posteriormente a identificar algunas clases que se utilizarán en el sistema.

Las formas típicas de las clases del dominio son:

1. Objetos del entorno que representan cosas que se manipulan.
2. Objetos del mundo real y conceptos de los que el futuro sistema debe de hacer seguimiento.
3. Sucesos que ocurrirán o han ocurrido.

El primer paso es identificar todos los conceptos que se utilizan en el diagrama, mediante un glosario de términos sobre los nombres:

Se llamará **Universidad** al centro de estudios superiores UCI y **Persona UCI** a cualquier persona que estudie o trabaje en la Universidad, sin importar su categoría o lugar de residencia.

Se llamará **Profesor** a los profesores de la Universidad, sin importar el departamento docente al que pertenezcan.

Se llamará **Estudiante** a cualquier alumno de la Universidad.

Se llamará **Teleformación** al sistema de formación docente, que garantiza el soporte parcial a todo el sistema de pregrado y postgrado de la Universidad de Ciencias Informáticas.

Se llamarán **Cursos_asignatura** a los cursos asociados a todas las asignaturas del plan de estudio de pregrado y asignaturas optativas y demandadas por la formación desde la producción, que se proponen en el sistema Teleformación.

Se llamará **Teleinformática** a los cursos Teleinformática I y II, asociados a la asignatura de pregrado del plan de estudio de tercer año de la carrera Ingenieros en Ciencias Informáticas.

Se llamará **Clase Práctica** a una de las formas de organización de la asignatura Teleinformática donde se orientan los ejercicios prácticos a desarrollar por los estudiantes.

Se llamará **Ejercicios** a la orientación de ejercicios prácticos en las clases prácticas.

Se llamará **Conferencia** a una de las formas de organización de la asignatura Teleinformática donde se imparten los nuevos conocimientos sobre la asignatura.

Se llamará **Laboratorio** a una de las formas de organización de la asignatura Teleinformática donde los estudiantes realizan las prácticas de los ejercicios orientados.

El Modelo de Dominio se describe mediante diagramas UML, específicamente con un diagrama de clases conceptuales significativas en el dominio del problema.

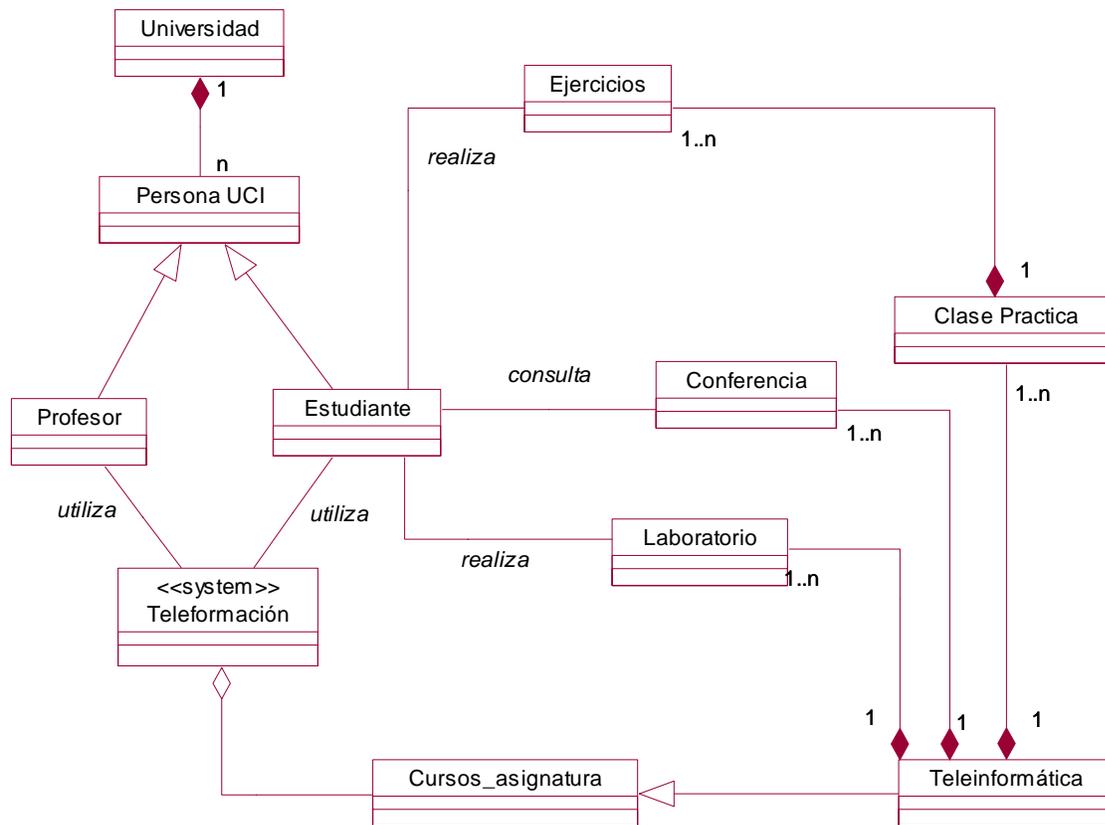


Figura 1 Modelo de dominio.

Más adelante, los objetos del dominio ayudan a identificar algunas de las clases del análisis y diseño. Las clases del dominio y el glosario de términos también se utilizan en el desarrollo de los modelos de casos de uso.

2.3. Especificación de los requisitos de software

El propósito fundamental del flujo de trabajo de los requisitos es guiar el desarrollo hacia el sistema correcto. Esto se consigue mediante una descripción de los requisitos del sistema, las condiciones y propiedades del sistema, suficientemente buena como para que pueda llegarse a un acuerdo sobre qué debe y qué no debe hacer el sistema.

2.3.1. Requisitos funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. De acuerdo con los objetivos planteados el sistema debe ser capaz de:

RF1. Autenticar usuario.

1.1 Mostrar la opción de autenticarse.

1.1.1 Introducir usuario y contraseña.

1.1.2 Verificar si el usuario pertenece al Idap.

1.1.3 Verificar si existe en el sistema.

RF2. Insertar producto.

2.1 Se especifica el proveedor, categoría, nombre, descripción, precio y unidad de medida del producto.

RF3. Actualizar producto.

3.1 Modificar producto.

3.1.1 Se especifican los nuevos datos del proveedor, categoría, nombre, descripción, precio o unidad de medida del producto.

3.2 Eliminar producto.

3.2.1 Se indica el producto a eliminar.

RF4. Mostrar producto.

4.1 Imprimir reporte de producto.

RF5. Buscar todos los productos de una categoría.

5.1 Se especifica la categoría y se muestran los productos que tienen esa categoría.

RF6. Insertar proveedor.

6.1 Se especifica el nuevo proveedor.

RF7. Eliminar proveedor.

7.1 Se especifica el proveedor a eliminar.

RF8. Mostrar proveedores.

RF9. Insertar categoría.

9.1 Se especifica la nueva categoría.

RF10. Eliminar categoría.

10.1 Se especifica la categoría a eliminar.

RF11. Mostrar categoría.

RF12. Insertar unidad de medida.

12.1 Se especifica la nueva unidad de medida.

RF13. Eliminar unidad de medida.

13.1 Se especifica la unidad de medida a eliminar.

RF14. Mostrar unidad de medida

RF15. Adicionar un nuevo puerto.

15.1 Se especifica el tipo de conexión y la subred a la que se va a conectar.

RF16. Adicionar una PC.

16.1 Se especifica el IP de la PC y la subred a la que se va a conectar.

RF17. Adicionar un servidor.

17.1 Se especifica el IP del servidor y la subred a la que se va a conectar.

RF18. Establecer conexión.

18.1 Se indica el IP de la PC o servidor, el puerto al que se va a conectar y la subred.

RF19. Eliminar conexión.

19.1 Se especifica la subred y la conexión a eliminar.

RF20. Enviar paquetes.

20.1 Se indica el IP origen y el IP destino del paquete.

RF21. Cancelar envío de paquetes.

21.1 Se selecciona el envío a cancelar.

RF22. Mostrar el interior de un router.

RF23. Insertar ejercicio.

23.1 Se indica el texto, imagen del ejercicio y presupuesto del ejercicio.

RF24. Actualizar ejercicio.

24.1 Modificar ejercicio.

24.1.1 Se especifican los nuevos datos del texto, imagen del ejercicio y/o presupuesto.

24.2 Eliminar ejercicio.

24.2.1 Se especifica el ejercicio y se elimina.

RF25. Mostrar ejercicio.

RF26. Responder ejercicio.

26.1 Elaborar respuesta al ejercicio seleccionado.

26.1.1 Se especifica el texto, topología, estándar y productos de la respuesta.

RF27. Insertar topología.

27.1 Se especifica la nueva topología.

RF28. Eliminar topología.

- 28.1 Se especifica la topología y se elimina.
- RF29. Mostrar topología.
- RF30. Insertar estándar.
 - 30.1 Se especifica el nuevo estándar.
- RF31. Eliminar estándar.
 - 31.1 Se especifica el estándar y se elimina.
- RF32. Mostrar estándar.
- RF33. Mostrar reporte por ejercicio.
 - 33.1 Se especifica el usuario y el ejercicio.
- RF34. Mostrar errores por respuesta.
 - 34.1 Calcular errores por respuesta, según la respuesta correcta.
- RF35. Mostrar reporte de estadísticas.
 - 35.1 Se indican los siguientes criterios: ejercicio y estudiante.
- RF36. Mostrar reporte de historial de usuario.
 - 36.1 Se especifica el usuario para reportear las fechas de las acciones.
- RF37. Insertar nueva documentación.
 - 37.1 Se especifica el tipo y dirección del archivo o documento.
- RF38. Eliminar documentación.
 - 38.1 Se especifica la documentación y se elimina.
- RF39. Mostrar documentación.
- RF40. Modificar rol de usuarios.
 - 40.1 Se especifica el nuevo rol.
- RF41. Mostrar todos los usuarios.
- RF42. Buscar todos los usuarios por rol del sistema.
 - 42.1 Se especifica el rol del sistema y se muestran los usuarios que tienen ese rol.
- RF43. Insertar usuario.
 - 43.1 Se especifica el nombre, rol de dominio, el usuario del dominio y el rol del sistema.

2.3.2. Requisitos no funcionales

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener, como: restricciones del entorno o de la implementación, rendimiento, dependencias de la plataforma, facilidad de

mantenimiento, extensibilidad, factibilidad, usabilidad, y confiabilidad, entre otras. De acuerdo con los objetivos planteados el sistema debe tener las siguientes propiedades:

2.3.2.1. Apariencia o interfaz externa

La interfaz de este sistema debe contar con un diseño sencillo y de fácil uso, que contenga sólo los gráficos necesarios, con vista a acelerar la velocidad de respuesta hacia el usuario. Debe tener colores refrescantes a la vista, colaborando con los requerimientos medio ambientales, contendrá una ayuda para que el usuario que se encuentra trabajando en el sistema no se pierda y sepa lo que debe hacer en cada paso. También se le deben incorporar otros elementos de diseño como: gráficos de encabezamiento, estilos y formatos de texto y paletas de color de los gráficos.

2.3.2.2. Rendimiento

El Laboratorio Virtual de Redes está concebido para un ambiente cliente/servidor así que los tiempos de respuestas deben ser generalmente rápidos al igual que la velocidad de procesamiento de la información. Estará implementado sobre una tecnología Web, facilitando su uso a través de la red. El tiempo de respuesta así como la generación de reportes debe ser en el tiempo más corto pues se deben generar pantallas dinámicas, implicando esto que el acceso a la base de datos debe ser lo más disponible, rápido y consistente posible.

2.3.2.3. Soporte

Para lograr la solidez de los datos se realizará el mantenimiento automatizado en la base de datos, orientados a la actualización y corrección de la información, a horas del día donde haya la menor cantidad de usuarios conectados. Para garantizar el soporte, se realizarán encuentros con los responsables de mantener actualizada la aplicación, con el fin de explicarles como funciona la herramienta. Se requiere de un servidor Apache en vista al trabajo con PHP y con MySQL instalado para un manejo eficiente con la base de datos correspondiente.

2.3.2.4. Portabilidad

El sistema deberá ser independiente de la plataforma, compatible con los Sistemas Operativos: Linux, Windows, Unix, entre otros.

2.3.2.5. Requerimientos de Hardware

Se requiere que las computadoras de los usuarios tengan tarjeta de red y al menos 64 MB de RAM.

2.3.2.6. Requerimientos de Software

El servidor se necesita que esté instalado en cualquier sistema operativo que soporte los software Apache como servidor Web y MySQL como el sistema de gestión de base de datos, preferentemente plataformas Windows 2000 o superior y Linux. En las computadoras de los usuarios solo se requiere de un navegador (Internet Explorer, Netscape, FireFox u otro) para visualizar la página y poder interactuar con el sistema.

2.3.2.7. Seguridad

El sistema debe contar con varios niveles de acceso para permitir el trabajo organizado. Permitir que a la información solo acceda quien está autorizado para ello y dándole el uso al cual está dirigida. Se debe garantizar la permanencia de las acciones realizadas por cada usuario en un historial. Identificar a la persona a la que se autoriza, a quien se le concede permiso para determinadas tareas (son los procesos de identificación, autorización y asignación de roles). Mantener la integridad de la información, es decir que no se perderá durante su almacenamiento o transporte. Permitir que cuando se intente borrar cualquier documento o información pueda existir una opción de advertencia para verificar el hecho.

2.3.2.8. Disponibilidad

A los usuarios autorizados se les garantizará el acceso a la aplicación y que los dispositivos o mecanismos utilizados para lograr la seguridad no ocultarán o retrasarán a los usuarios para realizar las acciones correspondientes que se le asignan en el sistema. Todo esto se logra a través de la creación de grupos de usuarios los cuales tendrán asignados permisos de acción sobre cada información manejada por el sistema, para lo cual se requiere la autenticación del usuario. Si no se autentifica, no es un usuario reconocido en el sistema, lo cual implica que no puede navegar en la aplicación.

2.3.2.9. Confiabilidad

La información manejada por el sistema está protegida de acceso no autorizado. Se debe garantizar un tratamiento adecuado a las excepciones y la validación de las entradas del usuario.

2.3.2.10. Integridad

La información manejada por el sistema será objeto de cuidadosa protección contra la corrupción, de la misma forma será considerada igual a la fuente o autoridad de los datos.

2.3.2.11. Usabilidad

La aplicación se debe documentar bien, y proporcionar materiales de ayuda para hacer mejor cada uno de los servicios que este ofrece.

2.4. Definición de los casos de uso

2.4.1. Definición de los actores

En este caso con el sistema interactúan cuatro actores que se definen a continuación:

Tabla 1 Listado de actores del sistema.

Actores	Justificación
Usuario	Cualquier persona que estudie o trabaje en la Universidad. En el Laboratorio Virtual de Redes pueden observar la simulación de los dispositivos de red publicados, responder los ejercicios de diseño de red y recibir un reporte con los errores que tuvo y la calificación que alcanzó.
Usuario Avanzado	Los profesores de la asignatura Teleinformática, que van a ser los responsables de gestionar los ejercicios, publicar la documentación, podrán revisar el historial de los usuarios, consultar las estadísticas y mantener actualizados los datos de los productos.
Administrador	Controla la administración del sistema. El jefe del Departamento de Sistemas Digitales, que se encarga de otorgar los roles y los permisos a los usuarios.
Directorio Activo	Es el sistema que controla las cuentas de usuarios del dominio, mediante él es que se realiza la autenticación de los usuarios.

2.4.2. Listado de casos de uso

Un sistema software ve a la luz para dar servicio a sus usuarios. Los casos de uso son fragmentos de funcionalidad del sistema que proporcionan al usuario un resultado importante. Los casos de uso representan los requisitos funcionales, en ellos se describe la secuencia determinada de eventos que realiza un actor en interacción con la aplicación. Los casos de uso ayudan a llevar a cabo el desarrollo iterativo del producto, y a idear la arquitectura del sistema.

Tabla 2 Descripción CU Autenticar usuario.

CU-1	Autenticar usuario
Actor	Usuario, Usuario Avanzado y Administrador
Descripción	Los usuarios al conectarse al Laboratorio Virtual de Redes se deben autenticar, el sistema verificar si el usuario pertenece al ldap, verificar si existe en el sistema, si no es así lo inserta. El sistema solo le concede los privilegios que le corresponden a cada usuario. Dependiendo del tipo de usuario que sea, el sistema debe mostrar el área que le está permitido trabajar.
Referencia	RF1, RF43

Tabla 3 Descripción CU Gestionar producto.

CU-2	Gestionar producto
Actor	Usuario Avanzado
Descripción	El usuario avanzado administra los elementos necesarios para el diseño de la red. Actualiza los datos de los productos de red propuestos al usuario.
Referencia	RF2, RF3

Tabla 4 Descripción CU Gestionar proveedor

CU-3	Gestionar proveedor
Actor	Usuario Avanzado
Descripción	El usuario avanzado administra los datos de los proveedores de los distintos productos para el diseño de la red.
Referencia	RF6, RF7, RF8

Tabla 5 Descripción CU Gestionar categoria

CU-4	Gestionar categoria
Actor	Usuario Avanzado

Descripción	El usuario avanzado administra los datos de las categorías de los distintos productos para el diseño de la red.
Referencia	RF9, RF10, RF11

Tabla 6 Descripción CU Gestionar unidad de medida

CU-5	Gestionar unidad de medida
Actor	Usuario Avanzado
Descripción	El usuario avanzado administra las distintas unidades de medida de los productos para el diseño de la red.
Referencia	RF12, RF13, RF14

Tabla 7 Descripción CU Gestionar subred

CU-6	Gestionar subred
Actor	Usuario
Descripción	El usuario puede adicionar un nuevo puerto a los conmutadores, adicionar una PC o un servidor a las subredes.
Referencia	RF15, RF16, RF17

Tabla 8 Descripción CU Establecer conexión

CU-7	Establecer conexión
Actor	Usuario
Descripción	El usuario puede establecer una conexión entre una PC o un servidor a los conmutadores.
Referencia	RF18, RF19

Tabla 9 Descripción CU Enviar paquetes

CU-8	Enviar paquetes
Actor	Usuario
Descripción	El usuario puede simular el envío de paquetes desde un servidor o PC a otro servidor o PC.
Referencia	RF20, RF21

Tabla 10 Descripción CU Simular interior dispositivos

CU-9	Simular interior dispositivos
Actor	Usuario
Descripción	El usuario puede observar el funcionamiento de algunos dispositivos de red.

Referencia	RF22
-------------------	------

Tabla 11 Descripción CU Gestionar ejercicio

CU-10	Gestionar ejercicio
Actor	Usuario Avanzado
Descripción	El usuario avanzado introduce la orientación de un ejercicio propuesto, con su respuesta correcta, puede incluir una imagen de apoyo al ejercicio, y puede modificar algún ejercicio anterior.
Referencia	RF23, RF24

Tabla 12 Descripción CU Responder ejercicio

CU-11	Responder ejercicio
Actor	Usuario
Descripción	Los usuarios seleccionan el ejercicio a responder y redactan la respuesta a guardar.
Referencia	RF4, RF5, RF6, RF25, RF26

Tabla 13 Descripción CU Gestionar topología

CU-12	Gestionar topología
Actor	Usuario Avanzado
Descripción	El usuario avanzado administra las distintas topologías para el diseño de las redes.
Referencia	RF27, RF28, RF29

Tabla 14 Descripción CU Gestionar estándar

CU-13	Gestionar estándar
Actor	Usuario Avanzado
Descripción	El usuario avanzado administra los estándares de red propuestos para el diseño de las redes.
Referencia	RF30, RF31, RF32

Tabla 15 Descripción CU Gestionar reporte de ejercicios.

CU-14	Gestionar reporte ejercicio
Actor	Usuario
Descripción	Los usuarios consultan el reporte de un ejercicio realizado anteriormente, y el sistema le muestra el reporte del ejercicio con la respuesta correcta, los

	errores y la calificación alcanzada.
Referencia	RF33, RF34

Tabla 16 Descripción CU Consultar estadísticas.

CU-15	Consultar estadísticas
Actor	Usuario Avanzado
Descripción	Los usuarios avanzados seleccionan la estadística que quieren consultar, y el sistema las muestra.
Referencia	RF35

Tabla 17 Descripción CU Operar historial de usuario.

CU-16	Operar historial usuario
Actor	Usuario Avanzado
Descripción	Se le muestra el historial de los estudiantes al usuario avanzado para analizar el desarrollo de estos mediante las acciones que han realizado en el sistema.
Referencia	RF36

Tabla 18 Descripción CU Publicar documentación.

CU-17	Publicar documentación
Actor	Usuario Avanzado
Descripción	El usuario avanzado puede publicar cualquier tipo de documentación dependiendo del tamaño de esta.
Referencia	RF37, RF38, RF39

Tabla 19 Descripción CU Administrar usuarios.

CU-18	Administrar usuarios
Actor	Administrador
Descripción	El administrador selecciona cuales usuarios son usuarios avanzados y los permisos por tipo de usuario.
Referencia	RF40, RF41, RF42

2.4.3. Diagrama de casos de uso

El modelo de casos de uso ayuda a los usuarios y a los desarrolladores a llegar a un acuerdo sobre cómo utilizar el sistema. Los diagramas de casos de uso son una representación de grafos con dos tipos de nodos:

1. Actores: representan elementos que intercambian información con el sistema, por lo que están fuera de él. No todos los actores representan a personas, pueden ser otros sistemas o hardware externo que interactuará con el sistema.
2. Casos de uso: especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto. Constituyen una secuencia de intercambios que representan el diálogo entre el sistema y uno o varios actores.

Entre casos de uso las relaciones pueden ser:

- Inclusión (<< include >>): indica que un caso de uso está incluido dentro de otro.
- Extensión (<< extend >>): ofrece una forma de extensión más controlada que la relación de generalización.
- Generalización: indica que un caso de uso es una variante de otro.

Entre actores las relaciones que se establecen son de generalización.

Para una mejor organización del trabajo, los casos de uso se han agrupado en correspondencia con sus responsabilidades. A esta forma de organización se le denomina paquetes.

Para la aplicación se han definido cuatro paquetes: Administración, Ejercicio, Materiales complementarios y Simulación. El paquete Administración contiene todos los casos de uso encargados del acceso al sistema mediante la autenticación de los usuarios; la configuración y mantenimiento del sistema. El paquete de Gestión de Ejercicios contiene los casos de uso que referencian las acciones sobre los ejercicios prácticos de diseño de redes. El paquete Materiales complementarios contiene a los casos de uso que manipulan la documentación y otros materiales de apoyo docente. Y por último el de Simulación, que contiene los casos de uso encargados de gestionar la simulación de los dispositivos de redes.

A continuación se presenta el diagrama de casos de uso del sistema para el Laboratorio Virtual de Redes.

2.4.4. Casos de uso por ciclo

Los ciclos de desarrollo se organizan de acuerdo a los requerimientos de los casos de uso. Se asigna un ciclo para desarrollar uno o más casos, en dependencia de la complejidad que estos posean. Para la selección de los casos que deben iniciar el primer ciclo de desarrollo es necesario identificar cuáles son los que influyen de manera determinante en la arquitectura básica del sistema.

Tabla 20 Casos de uso del primer ciclo de desarrollo.

Código	Nombre de caso de uso	Paquete	Justificación de la selección
CU-1	Autenticar usuario	Administración	Constituyen los casos de uso de mayor complejidad, representan los procesos más importantes para el funcionamiento del sistema e influyen en la arquitectura básica.
CU- 2	Gestionar Producto	Gestión de Ejercicios	
CU- 3	Gestionar proveedor	Gestión de Ejercicios	
CU- 4	Gestionar categoría	Gestión de Ejercicios	
CU- 5	Gestionar unidad de medida	Gestión de Ejercicios	
CU- 6	Gestionar subred	Simulación	
CU- 7	Establecer conexión	Simulación	
CU- 8	Enviar paquete	Simulación	
CU- 9	Simular interior dispositivos	Simulación	
CU- 10	Gestionar ejercicio	Gestión de Ejercicios	
CU- 11	Responder ejercicio	Gestión de Ejercicios	
CU- 12	Gestionar topología	Gestión de Ejercicios	
CU- 13	Gestionar estándar	Gestión de Ejercicios	
CU- 14	Gestionar reporte ejercicio	Gestión de	

		Ejercicios	
CU- 18	Administrar usuarios	Administración	

Tabla 21 Casos de uso del segundo ciclo de desarrollo.

Código	Nombre de caso de uso	Paquete	Justificación de la selección
CU- 15	Consultar estadísticas	Materiales complementarios	Estos casos de uso no son imprescindibles para el funcionamiento inicial del sistema ya que no constituyen procesos básicos.
CU- 16	Operar historial usuario	Administración	
CU- 17	Publicar documentación	Materiales complementarios	

2.4.5. Casos de uso expandidos

Mediante los casos de uso expandidos se describe la secuencia de eventos que los actores utilizan para completar un proceso usando el sistema. A continuación se muestra la expansión de los casos de uso más importantes que se desarrollarán en el Laboratorio Virtual de Redes.

Tabla 22 Autenticar usuario.

Caso de uso	
CU-1	Autenticar usuario
Propósito	Diferenciar los privilegios de cada usuario. Dependiendo del rol de usuario que tenga, el sistema debe mostrar el área que le está permitido trabajar.
Actor(es)	Usuario(Inicia)
Resumen:	Un usuario del dominio se conecta al sistema, introduce su ID de usuario y su contraseña. El sistema verifica si el usuario pertenece al ldap, si existe en el sistema los datos del usuario y en dependencia del rol de este, cargará las funcionalidades que tenga asignadas a ese rol específico, y se dará acceso a las secciones correspondientes.
Referencias	RF1, RF43
Acción del actor	Respuesta del sistema
1. El caso de uso se inicia cuando un usuario del dominio se conecta al sistema.	2. El sistema muestra una ventana donde el usuario debe introducir, identificador de dominio y su clave.
3. El usuario introduce su identificador de	4. El sistema verifica si los datos introducidos por el

dominio y su clave.	usuario pertenecen al ldap.
	5. Si los datos del usuario son correctos se verifica si existe en el sistema. Si no existen en el sistema ir a la sección "Insertar usuario".
	6. Si el usuario se encuentra en el sistema, se busca el rol del sistema que tiene asignado.
	7. El sistema muestra sólo aquellas funcionalidades a las que tiene acceso el usuario dependiendo de su rol.
Flujo alternativo	
CA 4: En caso de que los datos del usuario sean incorrectos el sistema muestra mensaje de error y se solicita que se introduzcan los datos nuevamente.	
Sección1: "Insertar usuario"	
	6. El sistema guarda los datos del usuario que se obtuvieron del ldap, asignando el rol del sistema de menor privilegio.
	7. El sistema muestra sólo aquellas funcionalidades a las que tiene acceso el usuario dependiendo de su rol.
Puntos de extensión	

Tabla 23 CU Gestionar producto

Caso de Uso	
CU-2	Gestionar producto
Propósito:	Permitir registrar, modificar y eliminar los diferentes productos necesarios para el diseño de la red.
Actor(es):	Usuario Avanzado(Inicia)
Resumen:	El caso de uso se inicia cuando el usuario avanzado selecciona la opción Gestionar Productos. El sistema muestra al usuario avanzado la lista con los productos actuales y le permite realizar cambios, o insertar un nuevo producto.
Referencias:	RF2, RF3, CU Gestionar proveedor (extend), CU Gestionar categoría (extend), CU Gestionar unidad de medida (extend).

Acción del Actor	Respuesta del Sistema
1. El caso de uso se inicia cuando el usuario avanzado selecciona la opción que desea realizar.	2. Si el usuario avanzado selecciona: a) Insertar un producto, ir a la sección "Registrar Producto" b) Modificar los datos de un producto, ir a la sección "Modificar Producto". c) Eliminar un producto, ir a la sección "Eliminar Producto".
Sección1: "Registrar Producto"	
	3. El sistema muestra la opción de seleccionar el proveedor, categoría, unidad de medida del producto, y la opción de introducir otros datos del producto.
4. El usuario avanzado busca el proveedor. Si encuentra el proveedor lo selecciona. En caso contrario ver Punto de Extensión 1.	
5. El usuario avanzado busca la categoría. Si encuentra la categoría la selecciona. En caso contrario ver Punto de Extensión 2.	
6. El usuario avanzado busca la unidad de medida. Si encuentra la unidad de medida la selecciona. En caso contrario ver Punto de Extensión 3.	
7. El usuario avanzado entra los datos restantes del producto para realizar su registro en la aplicación.	8. El sistema verifica que los campos obligatorios estén llenos.
	9. El sistema verifica que este producto no exista.
	10. El producto se almacena en el sistema.
	11. Se muestra un mensaje informándole al usuario avanzado que ha sido efectuado el registro del producto y finaliza el caso de uso.
Cursos Alternativos	

CA 8. Se emite un mensaje para que el usuario avanzado llene los campos obligatorios.	
CA 9. Si el producto existe se muestra un mensaje informativo y finaliza el caso de uso.	
Sección2: "Modificar Producto"	
3. El usuario avanzado selecciona el producto a modificar.	4. El sistema brinda la posibilidad de modificar los datos de los productos.
5. El usuario avanzado realiza las modificaciones deseadas.	6. El sistema verifica que los campos obligatorios estén llenos.
	7. Se actualizan los datos del producto y finaliza el caso de uso.
Cursos Alternativos	
CA 4. Se emite un mensaje para que el usuario avanzado llene los campos obligatorios.	
Sección3: "Eliminar Producto".	
3. El usuario avanzado selecciona el producto a eliminar.	4. El sistema pide confirmación.
	5. El sistema elimina el producto.
	6. Se muestra un mensaje de eliminación y finaliza caso de uso.
Puntos de extensión	
PE 1: Ver CU Gestionar proveedor.	
PE 2: Ver CU Gestionar categoría.	
PE 3: Ver CU Gestionar unidad de medida.	

Tabla 24 CU Gestionar subred.

Caso de Uso	
CU-6	Gestionar subred.
Propósito:	Permitir adicionar un nuevo puerto, una PC o un servidor.
Actor(es):	Usuario(Inicia)
Resumen:	El caso de uso se inicia cuando el usuario selecciona la opción hacer simulación. El sistema permite al usuario adicionar un nuevo puerto, una PC o un servidor a cada subred.
Referencias:	RF15, RF16, RF17

Acción del Actor	Respuesta del Sistema
1. El caso de uso se inicia cuando el usuario selecciona la opción que desea realizar.	2. Si el usuario selecciona: a) Adicionar un puerto, ir a la sección “Adicionar puerto” b) Adicionar una PC, ir a la sección “Adicionar PC” c) Adicionar un servidor, ir a la sección “Adicionar servidor”
Sección1: “Adicionar puerto”	
3. El usuario selecciona el tipo de conexión del puerto y la subred.	4. El sistema dibuja el puerto en la subred especificada.
Sección2: “Adicionar servidor”	
3. El usuario introduce el IP del servidor y la subred a la que pertenece.	4. El sistema verifica que no esté asignado ese IP anteriormente.
	5. El sistema dibuja el servidor en la subred.
Cursos Alternativos	
CA 5. Si el IP ya se encuentra asignado a una PC o un servidor se muestra un mensaje informativo y se solicita que se introduzcan los datos nuevamente.	
Sección3: “Adicionar PC”	
3. El usuario introduce el IP de la PC y la subred a la que pertenece.	4. El sistema verifica que no esté asignado ese IP anteriormente.
	5. El sistema dibuja la PC en la subred.
Cursos Alternativos	
CA 5. Si el IP ya se encuentra asignado a una PC o un servidor se muestra un mensaje informativo y se solicita que se introduzcan los datos nuevamente.	
Puntos de extensión	

Tabla 25 CU Enviar paquete.

Caso de Uso	
CU-8	Enviar paquete.
Propósito:	Permitir simular el envío de paquetes de datos en una red.
Actor(es):	Usuario(Inicia)

Resumen:	El caso de uso se inicia cuando el usuario selecciona la opción hacer simulación. El sistema permite al usuario simular el envío de paquetes de datos en una red.	
Referencias:	RF20, RF21	
Acción del Actor	Respuesta del Sistema	
1. El caso de uso se inicia cuando el usuario selecciona la opción que desea realizar.	2. Si el usuario selecciona: a) Enviar paquetes, ir a la sección "Enviar paquetes" b) Cancelar envío de paquetes, ir a la sección "Cancelar envío".	
Sección1: "Enviar paquetes"		
3. El usuario selecciona el IP origen y el IP destino del paquete.	4. El sistema dibuja paquetes y simula en envío hasta el destino.	
Sección2: "Cancelar envío"		
3. El usuario selecciona el envío a cancelar.	4. El sistema pide confirmación.	
	5. El sistema cancela el envío de paquetes.	
Puntos de extensión		

Tabla 26 CU Gestionar ejercicio.

Caso de Uso	
CU-10	Gestionar ejercicio
Propósito:	Permitir registrar, modificar y eliminar los diferentes ejercicios orientados de diseño de una red.
Actor(es):	Usuario Avanzado(Inicia)
Resumen:	El caso de uso se inicia cuando el usuario avanzado selecciona la opción gestionar ejercicio. El sistema permite al usuario avanzado, adicionar o eliminar un ejercicio, así como guardarlo.
Referencias:	RF23, RF24, CU Gestionar topología (extend), CU Gestionar estándar (extend).
Acción del Actor	Respuesta del Sistema

1. El caso de uso se inicia cuando el usuario avanzado selecciona la opción que desea realizar.	2. Si el usuario avanzado selecciona: a) Registrar un ejercicio, ir a la sección "Registrar Ejercicio" b) Modificar los datos de un ejercicio, ir a la sección "Modificar Ejercicio". c) Eliminar un ejercicio, ir a la sección "Eliminar Ejercicio".
Sección1: "Registrar Ejercicio"	
	3. El sistema muestra la opción de seleccionar el estándar y topología de red, y la opción de introducir otros datos del ejercicio.
4. El usuario avanzado busca la topología. Si encuentra la topología la selecciona. En caso contrario ver Punto de Extensión 1.	
5. El usuario avanzado busca el estándar. Si lo encuentra lo selecciona. En caso contrario ver Punto de Extensión 2.	
6. El usuario avanzado entra los datos restantes del ejercicio para realizar su registro en la aplicación.	7. El sistema verifica que los campos obligatorios estén llenos.
	8. El sistema verifica que este ejercicio no exista.
	9. El ejercicio se almacena en el sistema.
	10. Se muestra un mensaje informándole al usuario avanzado que ha sido efectuado el registro del ejercicio y finaliza el caso de uso.
Cursos Alternativos	
CA 7. Se emite un mensaje para que el usuario avanzado llene los campos obligatorios.	
CA 8. Si el ejercicio existe se muestra un mensaje informativo y finaliza el caso de uso.	
Sección2: "Modificar Ejercicio"	
3. El usuario avanzado selecciona el ejercicio	4. El sistema brinda la posibilidad de modificar los

a modificar.	datos.
5. El usuario avanzado realiza las modificaciones deseadas.	6. El sistema verifica que los campos obligatorios estén llenos.
	7. Se actualizan los datos del ejercicio y finaliza el caso de uso.
Cursos Alternativos	
CA 4. Se emite un mensaje para que el usuario avanzado llene los campos obligatorios.	
Sección3: "Eliminar Ejercicio".	
3. El usuario avanzado selecciona el ejercicio a eliminar.	4. El sistema pide confirmación.
	5. El sistema elimina el ejercicio.
	6. Se muestra un mensaje de eliminación y finaliza caso de uso.
Puntos de extensión	
PE 1: Ver CU Gestionar topología.	
PE 2: Ver CU Gestionar estándar.	

Tabla 27 CU Responder ejercicio.

Caso de uso	
CU-11	Responder ejercicio
Propósito	Guardar la respuesta dada por el usuario al ejercicio seleccionado.
Actor(es)	Usuario(Inicia)
Resumen:	El usuario selecciona un ejercicio a realizar, introduce la respuesta del mismo. El sistema guarda la respuesta del usuario.
Referencias	RF4, RF5, RF6, RF25, RF26
Acción del actor	Respuesta del sistema
1. El caso de uso se inicia cuando un usuario selecciona la opción responder ejercicio orientado.	2. El sistema muestra una ventana con la lista de los ejercicios orientados, con la opción de seleccionar uno para responder.
3. El usuario selecciona el ejercicio a realizar.	4. El sistema muestra una ventana donde el usuario debe introducir la respuesta al ejercicio.

5. El usuario introduce el texto, topología, estándar y productos de la respuesta.	6. El sistema guarda la respuesta del usuario.
Puntos de extensión	

En el **Anexo 1** se muestra la expansión de los casos de uso restantes.

Conclusiones

En este capítulo se han abordado las características fundamentales del sistema. Comenzando por el objeto de estudio, donde se incluyeron temas que sirvieron de base para definir el objeto de automatización. La propuesta del sistema a automatizar nos da una visión general de la aplicación. Se agregaron además el modelo del dominio y la especificación de los requisitos de software. Esta última sirvió de base para la identificación de los casos de uso del sistema, de los cuales se da su definición, diagramas y expansión.

CAPITULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

El análisis y diseño del sistema se consideran más cercanos al acceso de datos y a la arquitectura física de la aplicación. En el presente capítulo se abordan cuestiones fundamentales acerca del análisis y diseño de la aplicación con el objetivo de obtener una arquitectura sólida y facilitar en profundidad la comprensión de los requisitos.

Se incluyen el modelo de clases de análisis, los diagramas de colaboración y secuencia, diagramas de clase del diseño, el modelo de la base de datos. Finalmente se tratan los principios del diseño de la aplicación, así como las consideraciones a tener en cuenta en el tratamiento de errores, seguridad del sistema, interfaz y concepción de la ayuda.

3.1. Análisis.

El análisis forma parte del proceso de desarrollo de software, cuyo propósito primario es formular el modelo del dominio del problema. Durante el análisis, se analizan los requisitos que se describieron en la captura de requisitos, refinándolos y estructurándolos. El objetivo es desarrollar una serie de modelos que describan el software al trabajar para satisfacer un conjunto de requisitos definidos por el cliente. El análisis debe lograr tres objetivos primarios: describir lo que quiere el cliente, establecer una base para la creación de un diseño de software y definir un conjunto de requisitos que se puedan validar una vez que se construya el software.

La realización del análisis proporciona una visión general del sistema que puede ser más difícil de obtener mediante el estudio de los resultados del diseño y la implementación, por contener demasiados detalles.

3.1.1. Modelo de clases de análisis.

Una clase de análisis representa la abstracción de una o varias clases y/o subsistemas del diseño del sistema. Posee un conjunto de características tales como: se centra en el tratamiento de los requisitos funcionales y pospone los no funcionales, su comportamiento se define mediante responsabilidades a un nivel más alto y menos formal que las operaciones, define atributos y participa en relaciones, entre otras.

Cada clase posee uno de los estereotipos siguientes:

- **Clases de interfaz:** se utilizan para modelar la interacción entre el sistema y sus actores (usuarios y sistemas externos), la cual implica recibir y presentar información y peticiones, desde y hacia los usuarios y los sistemas externos. Las clases interfaz describen lo que se obtiene con la interacción, es decir, la información y las peticiones que se intercambian, no es necesario que

describan cómo se ejecuta físicamente la interacción, ya que esto se considerará en las actividades de diseño e implementación subsiguientes.

- **Clases de entidad:** se utilizan para modelar información que posee una vida larga o que es a menudo persistente, modelan la información y el comportamiento asociado de algún fenómeno o concepto, como una persona, un objeto, o un suceso del mundo real. Un objeto de entidad no ha de ser necesariamente pasivo y puede tener en ocasiones un comportamiento complejo relativo a la información que representa. Los objetos de entidad aíslan los cambios en la información que representan. Las clases de entidad suelen mostrar una estructura de datos lógica y contribuyen a comprender de qué información depende el sistema.
- **Clases de control:** Los aspectos dinámicos del sistema se modelan con clases de control, debido a que ellas manejan y coordinan las acciones y los flujos de control principales, y delegan trabajo a otros objetos (objetos de interfaz y de entidad). Representan coordinación, secuencia, transacciones, y control de otros objetos y se usan con frecuencia para encapsular el control de un caso de uso en concreto. Las clases de control también utilizan para representar derivaciones y cálculos complejos.

A continuación se muestran los diagramas de clases del análisis para los casos de uso más importantes.

Figura 3 Diagrama de clases de análisis. CU Autenticar usuario.

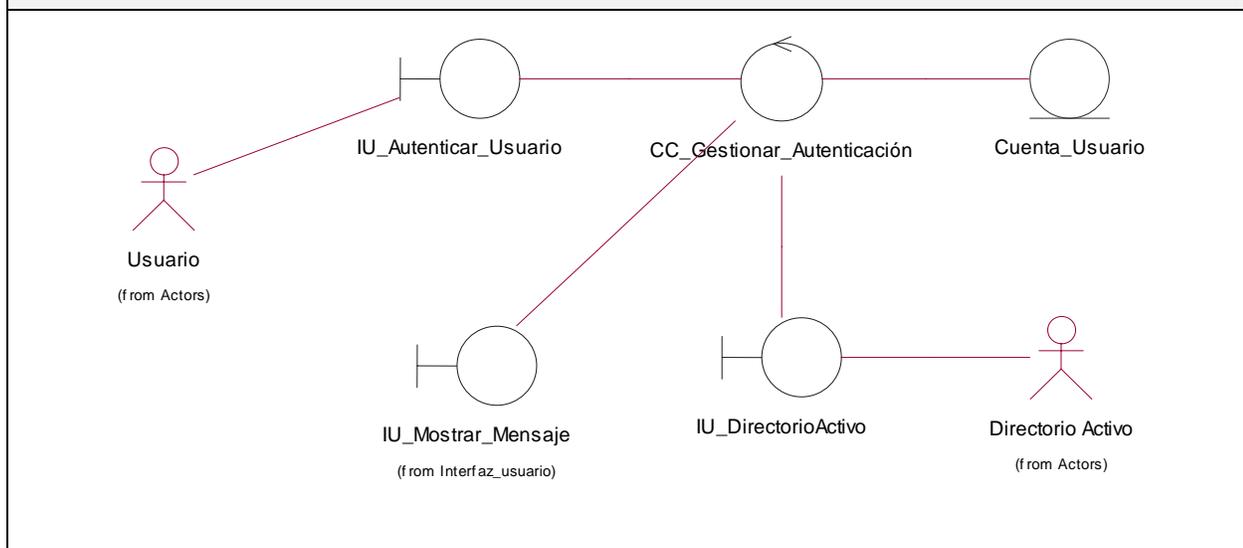


Figura 4 Diagrama de clases de análisis. CU Gestionar Producto.

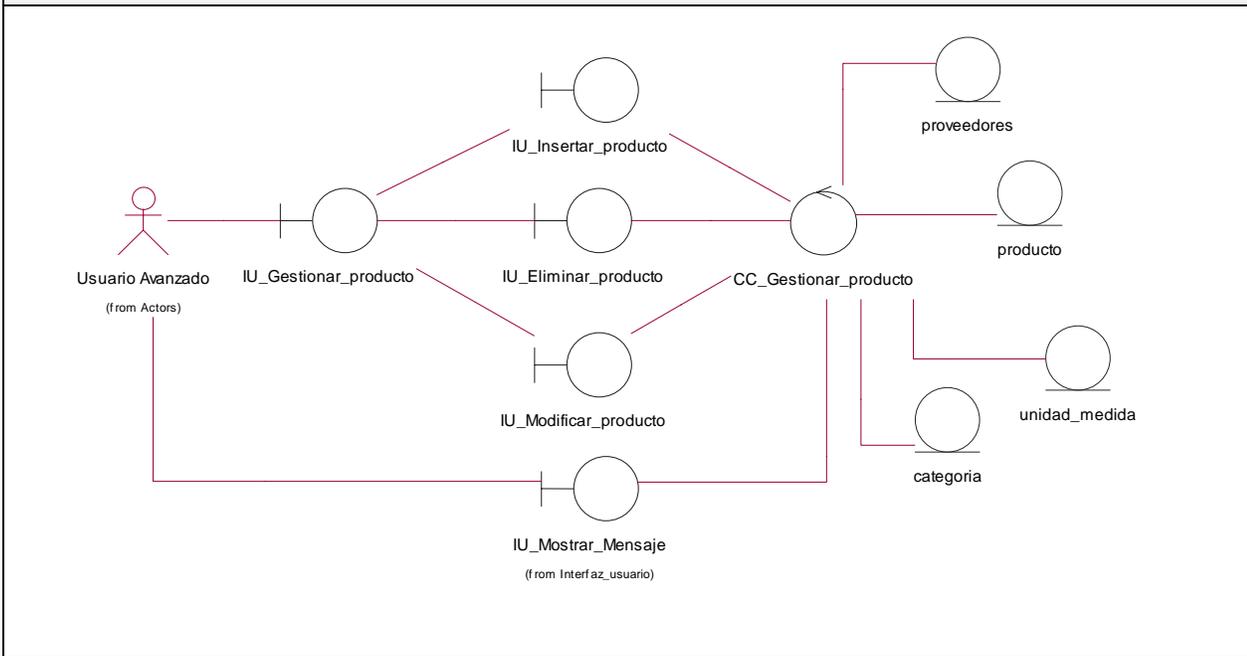


Figura 5 Diagrama de clases de análisis. CU Gestionar subred.

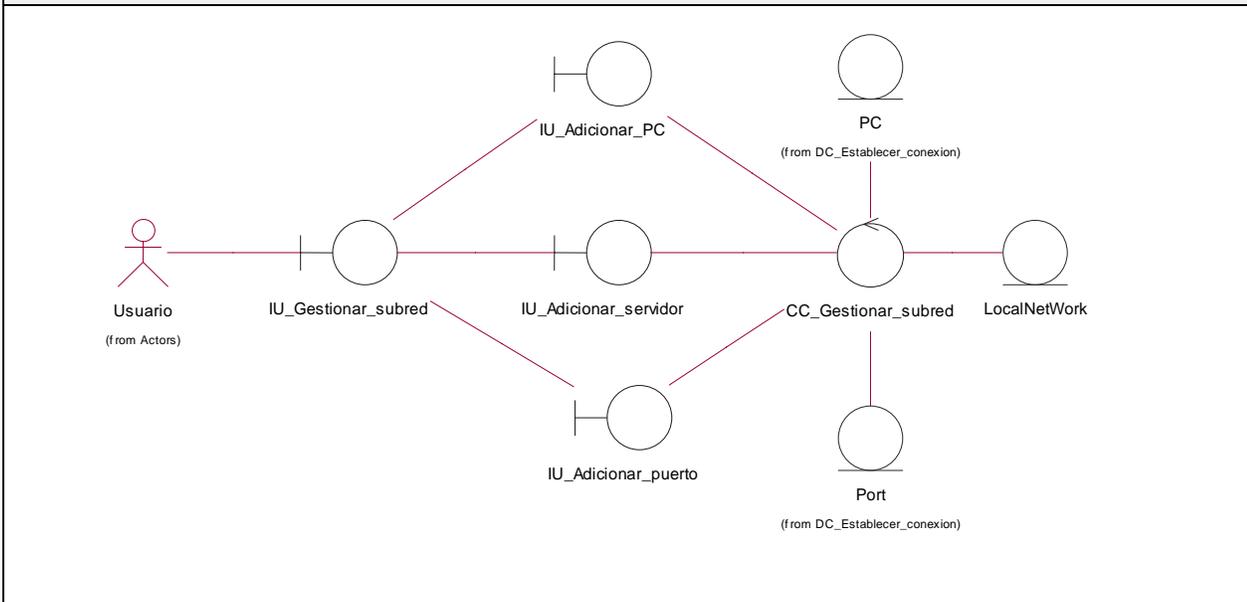


Figura 6 Diagrama de clases de análisis. CU Enviar paquete.

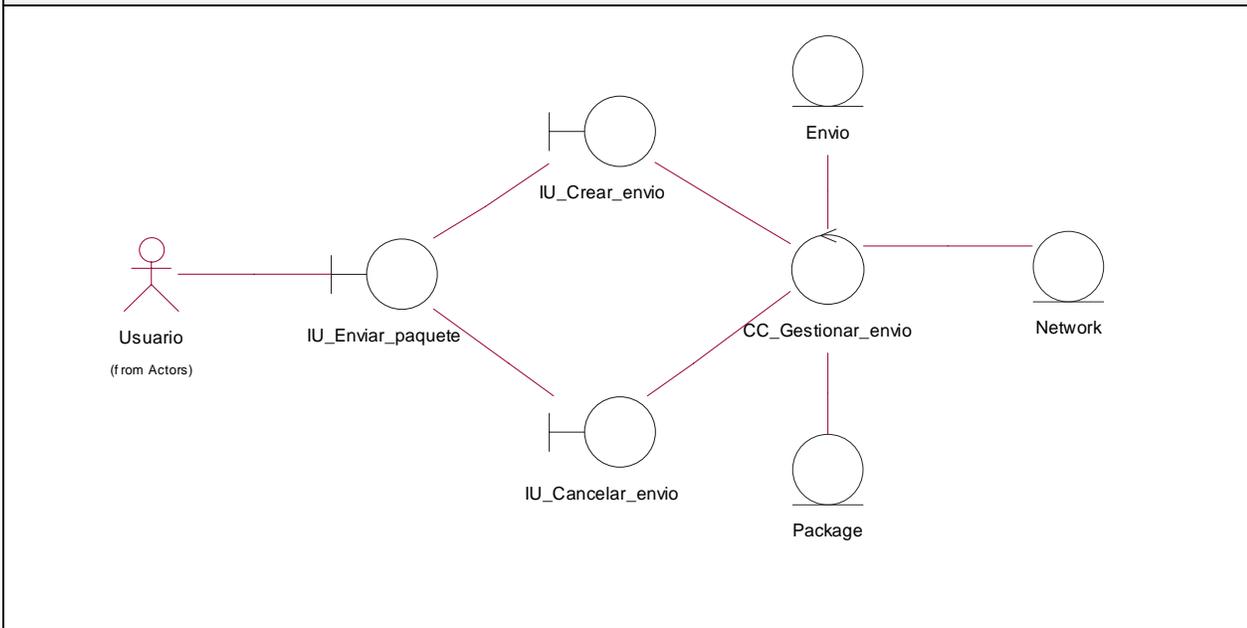


Figura 7 Diagrama de clases de análisis. CU Gestionar ejercicio.

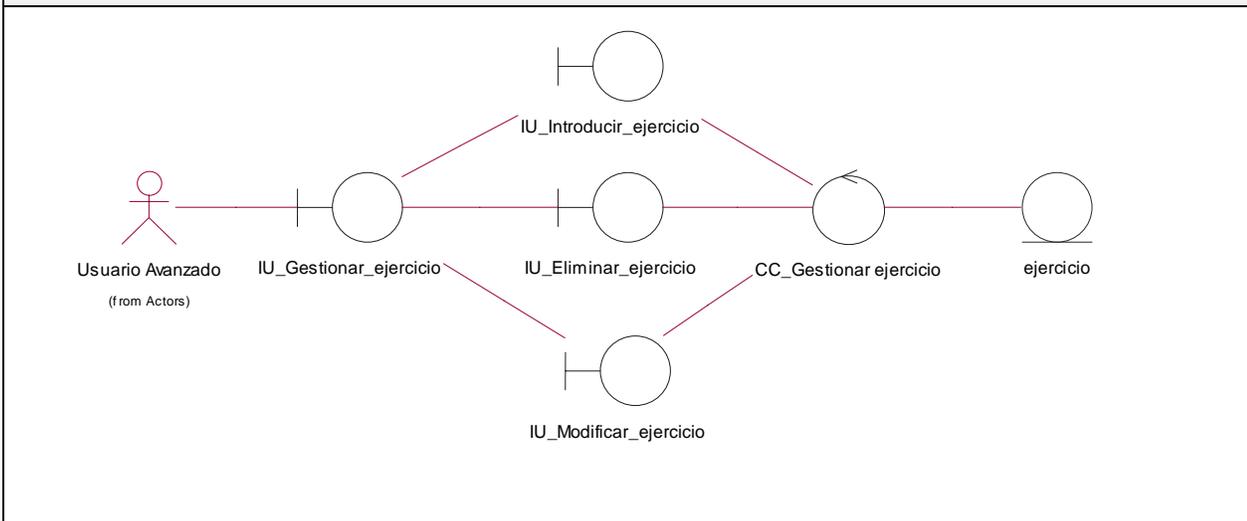
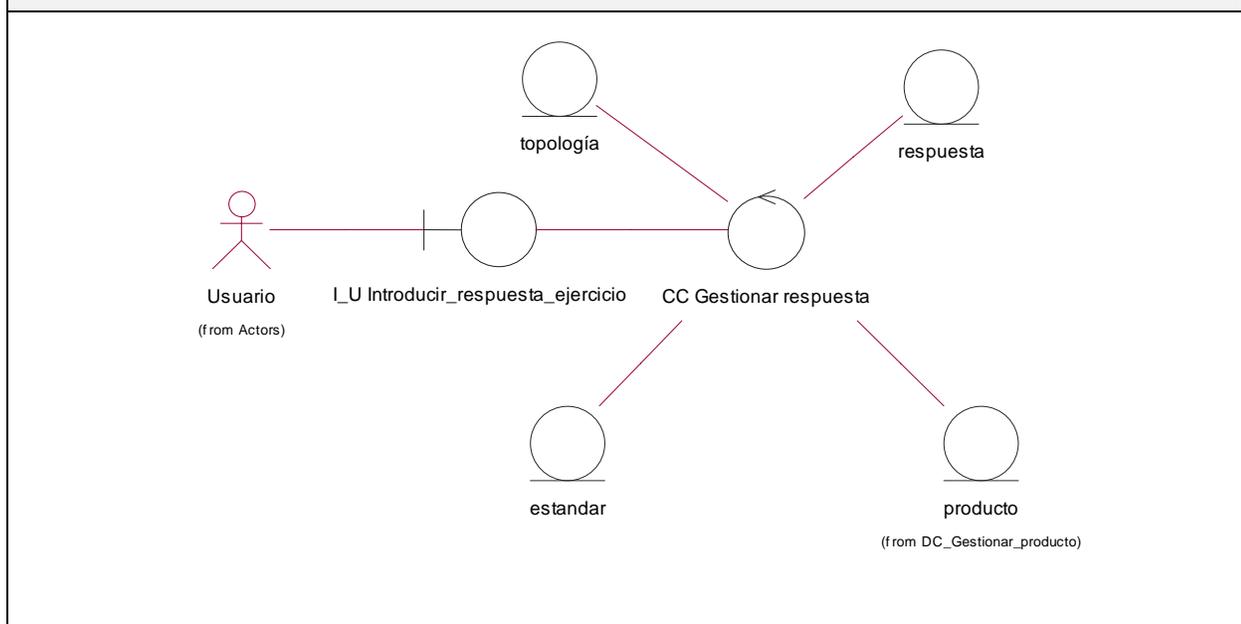


Figura 8 Diagrama de clases de análisis. CU Responder ejercicio.



En el **Anexo 2** se muestran los diagramas de clases restantes.

3.2. Diseño

El diseño del software se encuentra en el núcleo técnico de la ingeniería del software. Es la etapa del proceso de desarrollo donde se decide cómo se llevará a cabo el sistema. Una vez que se analizan y especifican los requisitos del software, el diseño es la primera de las tres actividades técnicas (diseño, generación del código y pruebas) que se requieren para construir y verificar el software.

El diseño se utiliza para modelar el sistema y encontrar su forma para que soporte todos los requisitos, incluyendo los requisitos no funcionales y demás restricciones. Se crea una entrada apropiada y un punto de partida para actividades de implementación subsiguientes capturando los requisitos o subsistemas individuales, interfaces y clases. El diseño debe ser una guía legible y comprensible para aquellos que generan código y para aquellos que comprueban y consecuentemente, dan soporte al software; deberá proporcionar una imagen completa del software, enfrentándose a los dominios de comportamiento, funcionales y de datos desde una perspectiva de implementación.

3.2.1. Diagramas de interacción.

La vista de interacción describe secuencias de intercambios de mensajes entre los roles que implementan el comportamiento de un sistema. Un rol es la descripción de un objeto, que desempeña un determinado papel dentro de una interacción, distinto de los otros objetos de la misma clase. Esta visión proporciona

una vista integral del comportamiento del sistema, es decir, muestra el flujo de control a través de muchos objetos. La vista de interacción se exhibe en dos diagramas centrados en distintos aspectos pero complementarios: centrados en los objetos individuales y centrados en objetos cooperantes.

Los diagramas de interacción muestran cómo se comunican los objetos en una interacción. En UML los diagramas de interacción pueden representarse a través de los Diagramas de Colaboración y/o de los Diagramas de Secuencia, ambos son representaciones alternas de interacciones. Los Diagramas de Secuencia muestran interacciones entre objetos basadas en el tiempo y los Diagramas de Colaboración muestran como los objetos se asocian unos con otros.

3.2.1.1. Diagramas de Secuencia

Los diagramas de secuencia son más adecuados para observar la perspectiva cronológica de las interacciones, muestran la secuencia explícita de mensajes y son mejores para especificaciones de tiempo real y para escenarios complejos. A diferencia de los diagramas de colaboración, los diagramas de secuencia incluyen secuencias temporales pero no incluyen las relaciones entre objetos. Pueden existir de forma de descriptor, describiendo todos los posibles escenarios, y en forma de instancia, describiendo un escenario real.

En el **Anexo 3** se muestran los diagramas de secuencia para cada caso de uso.

3.2.2. Diagrama de clases del diseño.

Una clase de diseño es una abstracción de una clase o construcción similar en la implementación del sistema. El lenguaje utilizado para especificar una clase del diseño es lo mismo que el lenguaje de programación. Consecuentemente, las operaciones, parámetros, atributos, tipos y demás son especificados utilizando la sintaxis del lenguaje de programación elegido. Al construir los diagramas de interacción se van utilizando clases procedentes del modelo conceptual, junto con otras creadas para encargarse de responsabilidades específicas. El conjunto de todas las clases usadas, junto con sus relaciones, forma el Diagrama de clases del diseño.

El Diagrama de Clases es el diagrama principal para el análisis y diseño. Un diagrama de clases presenta las clases del sistema con sus relaciones estructurales y de herencia. La definición de clase incluye definiciones para atributos y operaciones. El modelo de casos de uso aporta información para establecer las clases, objetos, atributos y operaciones.

Las formas de relación entre las clases son:

- Asociación y Agregación (vista como un caso particular de asociación): La asociación expresa una conexión bidireccional entre objetos. Una asociación es una abstracción de la relación existente en los enlaces entre los objetos. Puede determinarse por la especificación de multiplicidad (mínima...máxima). La agregación representa una relación *parte_de* entre objetos. Esta relación puede ser caracterizada con precisión determinando las relaciones de comportamiento y estructura que existen entre el objeto agregado y cada uno de sus objetos componentes.
 - Generalización/Especialización: Las relaciones de Agregación y Generalización forman jerarquías de clases. La Generalización consiste en factorizar las propiedades comunes de un conjunto de clases en una clase más general. Los nombres usados: clase padre - clase hija, otros nombres: superclase - subclase, clase base - clase derivada. Las subclases heredan propiedades de sus clases padre, es decir, atributos y operaciones de la clase padre están disponibles en sus clases hijas. La Generalización y Especialización son equivalentes en cuanto al resultado: la jerarquía y herencia establecidas.
- A continuación se muestran los diagramas de clases del diseño para los casos de uso más importantes.

Figura 9 Diagrama de clases del diseño. CU Autenticar usuario.

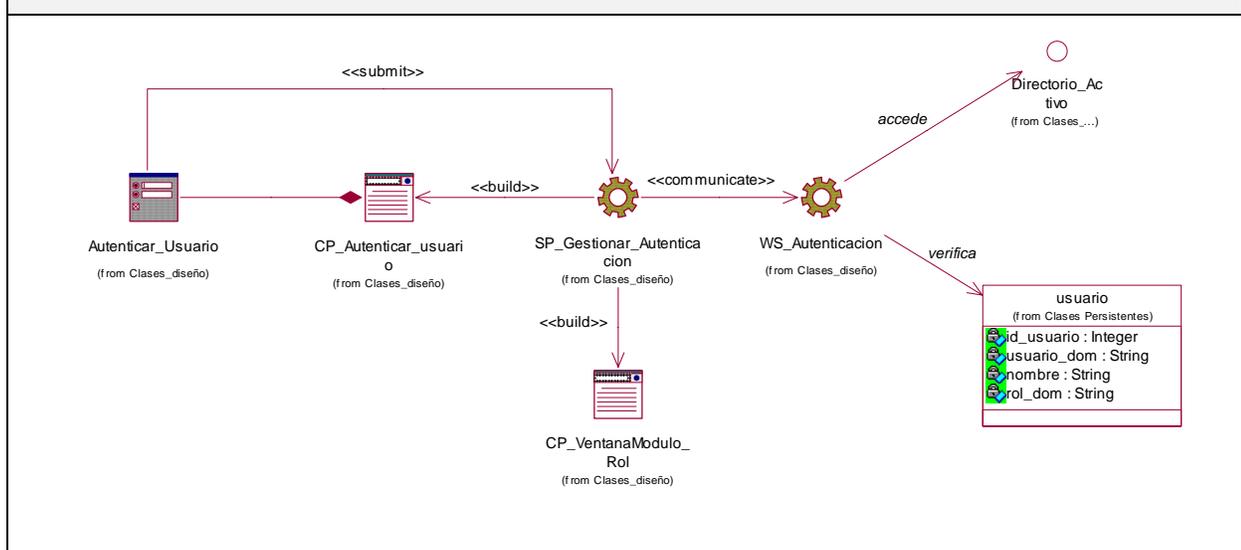


Figura 10 Diagrama de clases del diseño. CU Gestionar Producto.

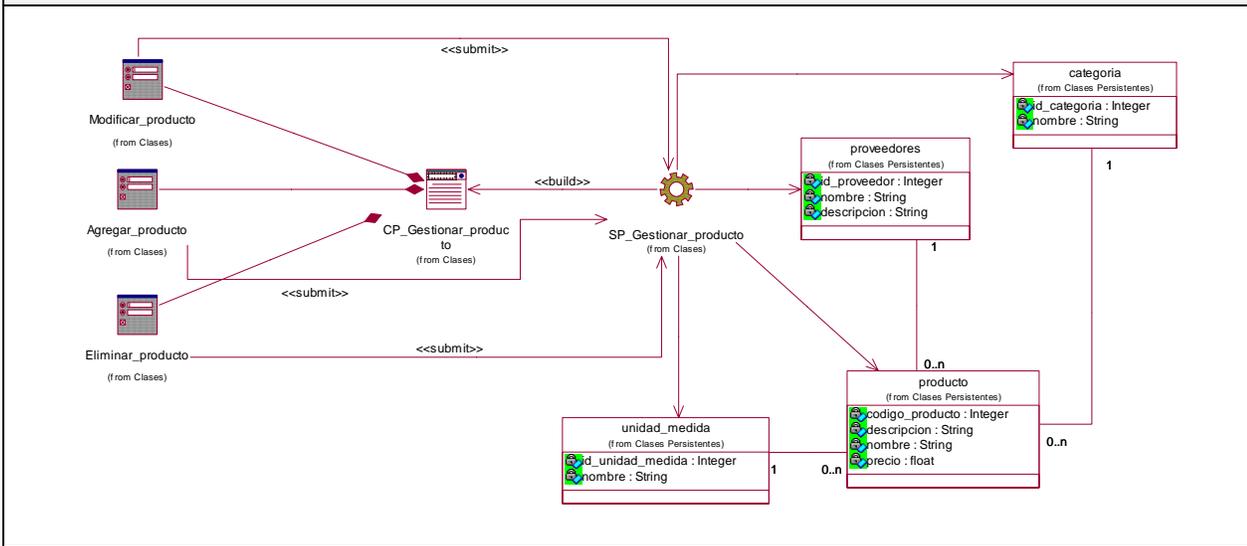


Figura 11 Diagrama de clases del diseño. CU Gestionar subred.

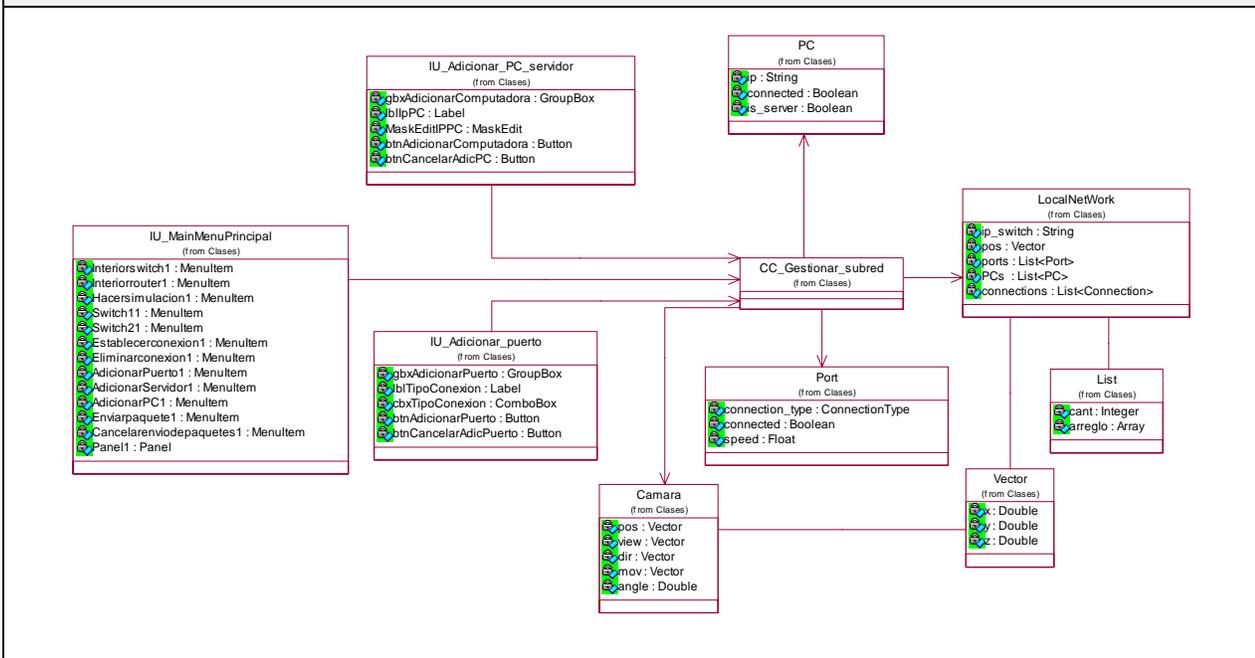


Figura 12 Diagrama de clases del diseño. CU Enviar paquete.

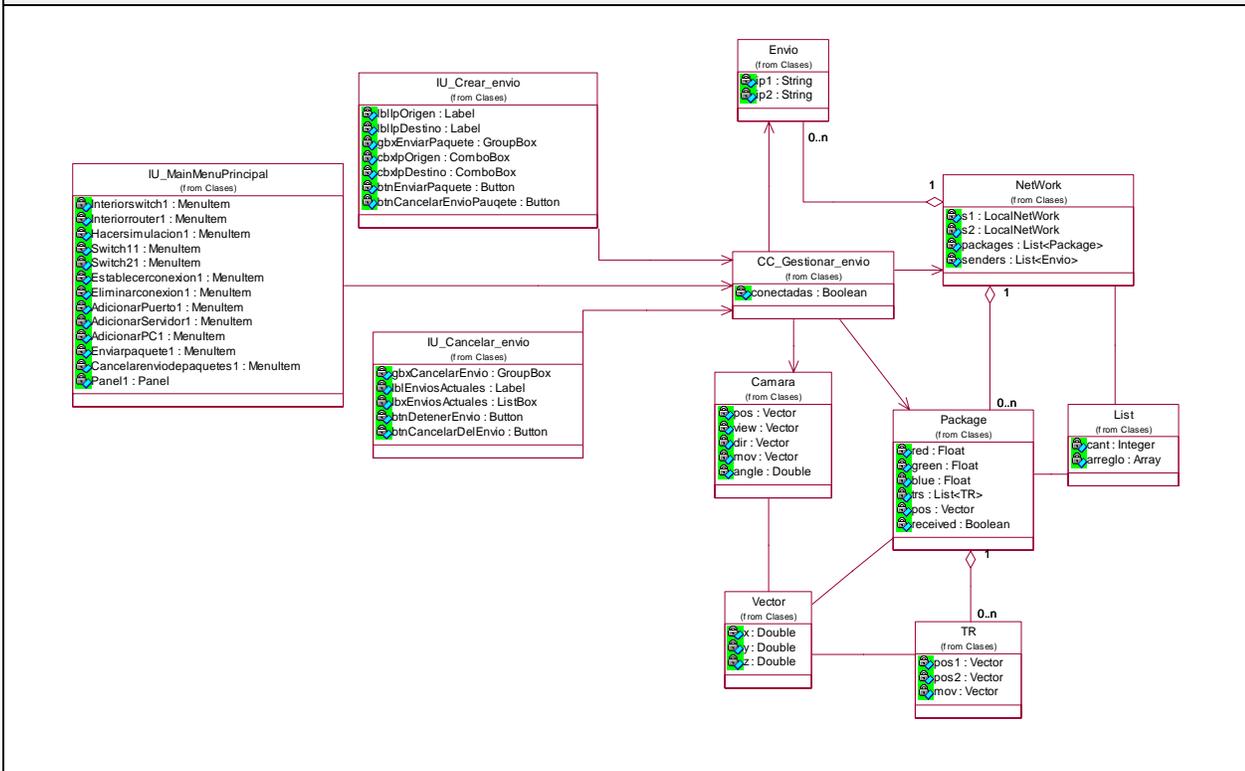


Figura 13 Diagrama de clases del diseño. CU Gestionar ejercicio.

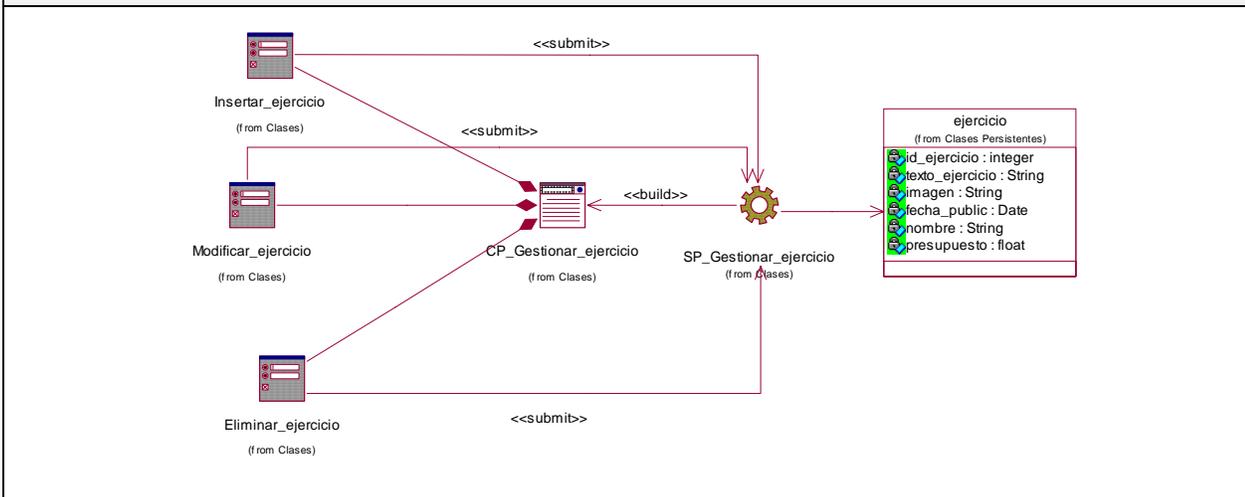
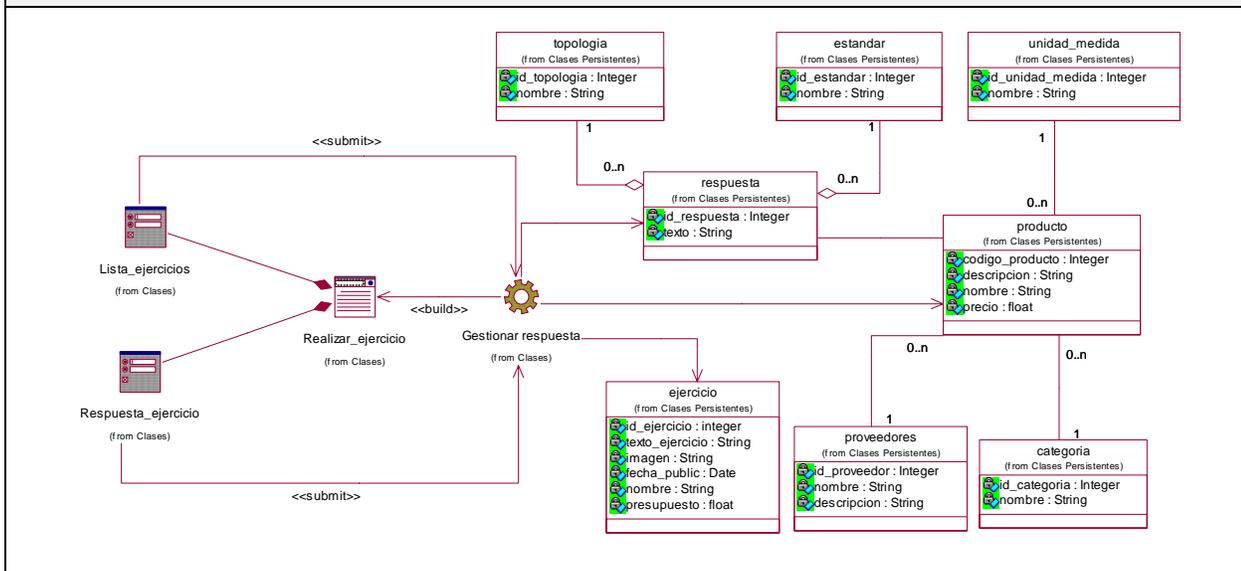


Figura 14 Diagrama de clases del diseño. CU Responder ejercicio.



En el **Anexo 4** se presentan los restantes diagramas de clases del diseño.

3.2.2.1. Descripción de las clases.

La descripción detallada de las clases, sus atributos y operaciones se relacionan en el **Anexo 5**.

3.3. Diseño de la base de datos (BD).

Una base de datos es un conjunto estructurado de datos interrelacionados y sin redundancias, registrados o almacenados sobre soportes accesibles por ordenador para satisfacer simultáneamente a varios usuarios en tiempo oportuno, y que pueden procesarse por uno o más sistemas de aplicación.

La sofisticación de la tecnología moderna de las bases de los datos es el resultado de la evolución que a lo largo de varias décadas ha tenido lugar en el procesamiento de los datos y en la gestión de la información. La tecnología de acceso a los datos se ha desarrollado desde los métodos primitivos de los años cincuenta hasta los potentes e integrados sistemas de hoy en día, arrastrados de un lado por las necesidades y las demandas de la administración y, de otro, restringida por las limitaciones de la tecnología.

Un modelo de datos está constituido por herramientas conceptuales para describir los datos, las relaciones que existen entre ellos, semántica asociada a los datos y las restricciones de consistencia.

Los modelos de datos se dividen en tres grupos:

1. Modelos lógicos basados en objetos: Se usan para describir datos en los niveles conceptual y de visión, es decir, con este modelo representamos los datos de tal forma como nosotros los captamos en el mundo real, tienen una capacidad de estructuración bastante flexible y permiten especificar restricciones de datos explícitamente. Existen diferentes modelos de este tipo, pero el más utilizado por su sencillez y eficiencia es el modelo Entidad-Relación.
2. Modelos lógicos basados en registros: Estos modelos se utilizan para describir datos en los niveles conceptual y físico. Utilizan registros e instancias para representar la realidad, así como las relaciones que existen entre estos registros o apuntadores. A diferencia de los modelos de datos basados en objetos, se usan para especificar la estructura lógica global de la base de datos y para proporcionar una descripción a nivel más alto de la implementación. Los tres modelos de datos más ampliamente aceptados son: el Modelo Relacional, el Modelo de Red y el Modelo Jerárquico.
3. Modelos físicos de datos: Se usan para describir a los datos en el nivel más bajo, aunque existen muy pocos modelos de este tipo, básicamente capturan aspectos de la implementación de los sistemas de base de datos.

El modelo utilizado para el diseño de la BD del Laboratorio Virtual de Redes es el relacional, donde se representan los datos y las relaciones entre estos, a través de una colección de tablas, en las cuales los renglones (tuplas) equivalen a cada uno de los registros que contendrá la base de datos y las columnas corresponden a las características (atributos) de cada registro localizado en la tupla. A la información almacenada se puede acceder mediante consultas SQL, que permiten la búsqueda de datos que han sido almacenados en más de una tabla y que se encuentran relacionados. La ventaja principal del modelo relacional es que los datos se almacenan, al menos conceptualmente, de un modo en que los usuarios los entienden con mayor facilidad. Este enfoque permite a los usuarios obtener información de la base de datos sin asistencia de sistemas profesionales de administración de información.

3.3.1.1. Modelo Entidad-Relación (ER).

Este modelo representa a la realidad a través de entidades, que son los objetos que existen y que se distinguen de otros por sus características particulares denominadas atributos; y las interrelaciones existentes entre ellas. La pareja objeto-relación es la piedra angular del modelo.

Este modelo tiene su base en una percepción del mundo real, y puede ser usado como una base para una vista unificada de los datos. En el contexto del análisis estructurado, el DER define todos los datos que se introducen, se almacenan, se transforman y se producen dentro de la aplicación.

A continuación se representa el diagrama ER para el sistema a implementar.

3.3.1.2. Descripción de las tablas.

En el **Anexo 6** se describen las tablas y cada uno de sus atributos.

3.4. Definiciones de diseño que se apliquen.

El diseño de interfaces de usuario es una tarea que ha adquirido relevancia en el desarrollo de sistemas. Las interfaces han de ser lo más uniforme posible, tratando de utilizar colores similares en todas las páginas, con textos concisos y claros evitando mezclar muchos tipos y tamaños de letra.

La navegación debe ser lo más rápida posible por lo que la presencia de imágenes innecesarias y todo lo que atente contra una navegación rápida y eficiente debe ser eliminado. En todas las páginas se respeta el esquema Cabecera-Navegador-Contenido. La cabecera contiene el logotipo de la aplicación en la esquina superior izquierda y una barra de navegación con enlaces de texto para cada uno de las funcionalidades del sistema. En el área del contenido se muestran los formularios de entrada, los reportes, etc.

Se utiliza una hoja de estilos para guardar la configuración del diseño de todas las páginas. Esta hoja de estilos establece el tipo y tamaño de fuente de los distintos elementos de cada página. Se utiliza en general la familia de fuentes Times New Roman, Times, serif, de tamaño 22 píxeles. Se establece el color de los vínculos, el color de fondo, el formato de los controles de formulario y las tablas.

3.5. Tratamiento de errores

El tratamiento de errores posibilita el buen funcionamiento de una aplicación dándole una mejor apariencia ante los clientes. En el sistema la validación para el tratamiento de errores serán referidos principalmente a errores de la interacción del usuario con el sistema, que son aquellos en los que se puede incurrir por la falta de conocimiento o inexperiencia en la explotación del sistema. Se tratará en todo momento de minimizar la posibilidad de ocurrencia de errores de este tipo aprovechando las posibilidades de la interfaz gráfica, para lo cual se le dará la opción de elegir o seleccionar la información, lo que facilitará la entrada de datos y la rapidez de la misma.

Utilizando mecanismos de validación en el lado del cliente y en el lado del servidor, se garantiza que los datos se almacenen íntegros y no existan inconsistencias. La validación en las páginas clientes será realizada con funciones Java Script, siempre se informará por medio de mensajes la ocurrencia de los errores, brindando la opción de recuperación de los mismos. Hay que tener en cuenta aquellas acciones que involucren modificación o eliminación de datos. Estas operaciones serán antecedidas por una opción de confirmación para realizarlas, pues son generalmente irreversibles.

3.6. Seguridad

La información es la primera línea de defensa en seguridad, detectando las amenazas efectivas, las actuaciones precursoras de ataques, hasta cualquier detección no sólo de ataques comprobados e identificados sino también de todo comportamiento “anormal” y, por lo tanto, sospechoso. En el Laboratorio Virtual de Redes se crearon niveles de acceso al sistema, definido por roles que desempeñarán los distintos usuarios, con el objetivo de garantizar la integridad y seguridad de los datos, así como el correcto funcionamiento de la aplicación e impedir la entrada de personal no autorizado a la misma. En el servidor de base de datos MySQL se crearon variables de sesión para restringir el acceso de los usuarios a determinadas páginas. Además de este control de acceso, para cada acción realizada por los usuarios se guardarán los datos que se manejaron, y los pasos que siguió en cada página visitada en un historial para poder identificar quien realizó cada acción, y cual fue el resultado.

3.7. Interfaz

Del diseño Web depende que la información sea útil, que los servicios se puedan usar. El diseño convierte a una aplicación Web en algo atractivo para el usuario, por su estética y su utilidad. Una aplicación con una interfaz bien diseñada debe tener una buena navegabilidad, usabilidad y distribución de los contenidos. Para el diseño de la interfaz de usuario del Laboratorio Virtual de Redes se tuvieron en cuenta los siguientes principios:

- Brindar la mayor cantidad de información sobre las opciones brindadas, de modo que el usuario sepa cuáles son las operaciones a las que puede acceder y en qué consiste exactamente cada una.
- Siempre que el usuario vaya a realizar una acción relevante sobre el sistema, se le debe mostrar un mensaje de confirmación.
- Se debe limitar el número de acciones que puede realizar el usuario sobre la aplicación evitando así al máximo los errores de estos.
- Requerir un mínimo proceso de aprendizaje y permitir su utilización desde el primer momento por cualquier persona que tenga un mínimo dominio de la informática.
- Requerir de los usuarios un mínimo esfuerzo para alcanzar sus objetivos.
- Mostrar al usuario solamente aquellas opciones a las que, dado su rol, puede ejecutar.

3.8. Concepción de la ayuda

Para los usuarios del sistema, en la página principal se muestran todas las opciones a las que tienen acceso, y una descripción de las mismas. Además dentro del flujo básico de trabajo de la aplicación se inserta una ayuda de modo que los usuarios entiendan los pasos que deben realizar.

Conclusiones

Las concepciones básicas para el análisis y diseño del sistema fueron tratadas en este capítulo. El principal resultado es el modelo de diseño, que conserva en gran medida la estructura del sistema impuesta por el modelo de análisis, y sirve como esquema para la implementación. Hasta aquí se logró confeccionar los diagramas de clases tanto del análisis como del diseño, se referenció el modelo ER utilizado para el diseño de la base de datos, así como la descripción de cada una de las tablas. Además se manifestaron las características del sistema en cuanto a las definiciones de diseño aplicadas, el tratamiento de errores, la seguridad y la concepción de la ayuda.

CAPITULO 4: IMPLEMENTACIÓN Y PRUEBA

En el presente capítulo se abordan las cuestiones fundamentales acerca de la implementación y prueba de la aplicación. Se incluyen el modelo de despliegue, el diagrama de componentes del modelo de implementación; y del modelo de prueba, la descripción de los casos de prueba de integración.

4.1. Implementación

En la implementación se comienza con el resultado del diseño y se implementa el sistema en términos de componentes. La mayor parte de la arquitectura del sistema es capturada durante el diseño, siendo el propósito principal de la implementación el desarrollar la arquitectura y el sistema como un todo. La implementación tiene como propósitos específicos: planificar las integraciones de sistema necesarias en cada iteración, distribuir el sistema asignando componentes ejecutables a nodos en el diagrama de despliegue, implementar las clases y subsistemas encontrados durante el diseño y probar los componentes individualmente, e integrarlos compilándolos.

La implementación es el centro durante las iteraciones de construcción, aunque también se lleva a cabo trabajo de implementación durante la fase de elaboración, para crear la línea base ejecutable de la arquitectura, y durante la fase de transición, para tratar defectos tardíos. 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 a aplicación.

4.1.1. Modelo de despliegue

El modelo de despliegue describe la distribución física del sistema, muestra cómo están distribuidos los componentes de software entre los distintos nodos de cómputo. Este modelo permite comprender la correspondencia entre la arquitectura software y la arquitectura hardware.

Un nodo es un elemento físico que existe en tiempo de ejecución y representa un recurso computacional, que generalmente tiene memoria y, a menudo, capacidad de procesamiento. Los nodos se utilizan para modelar la topología del hardware sobre el cual se ejecuta el sistema. Representa típicamente un procesador o un dispositivo sobre el que se pueden desplegar los componentes.

Los diagramas de despliegue pueden describir diferentes configuraciones de red, incluidas las configuraciones para pruebas y para simulación. La configuración física de la red tiene una gran influencia sobre la arquitectura del software, incluyendo las clases activas que se necesitan y la distribución de la

funcionalidad entre los nodos de la red. El Laboratorio Virtual de Redes utiliza un patrón de tres capas, donde el patrón cliente/servidor es un caso especial porque la lógica del negocio o de la aplicación se puede ubicar en la capa del cliente o en la de base de datos. Basándose en esta arquitectura, en el diagrama de despliegue se representan cuatro nodos. Uno de los nodos es el PC cliente, que representa las computadoras de los usuarios desde los cuales se accede, utilizando el protocolo HTTP, a la aplicación. La Aplicación Web se encuentra publicada en un Servidor, donde están todos los componentes de las capas de Presentación, Acceso a Datos y Negocio. A su vez, este se comunica con el Servidor de Base de Datos que contiene la base de datos del sistema (con MySQL como Sistema Gestor), a través del protocolo ADO para realizar consultas y actualizaciones de la información que se manipula. Las computadoras de los usuarios acceden mediante la aplicación, con el protocolo TCP/IP, a la aplicación desktop que se encuentra en el Servidor de la aplicación. El Servidor Web se comunica también con el Servidor del Directorio Activo para autenticar a los usuarios por el protocolo LDAP.

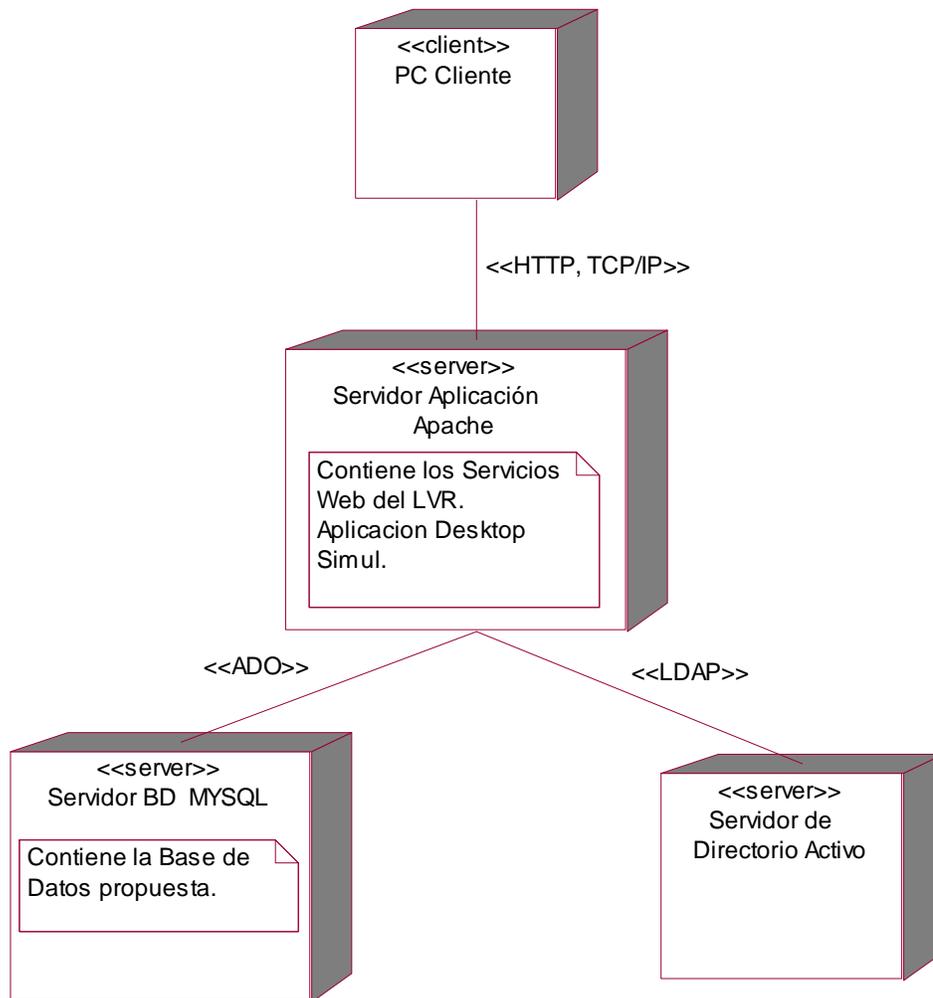


Figura 16 Diagrama de despliegue.

4.1.2. Modelo de implementación

El modelo de implementación denota la implementación actual del sistema en términos de componentes y subsistemas de implementación. Describe cómo se implementan los elementos del modelo de diseño. Los diagramas de componentes describen los elementos del sistema y sus relaciones, y muestran las opciones de realización, incluyendo código fuente, binario y ejecutable.

Los componentes representan todos los tipos de elementos de software que entran en la fabricación de las aplicaciones informáticas. Pueden ser archivos cuando representan un documento que contiene código fuente o datos, un documento, bibliotecas de objetos, estáticas o dinámicas, cargadas

dinámicamente, los ejecutables o tablas de una base de datos. En el **Anexo 7** se puede observar una vista general del diagrama de componentes del Laboratorio Virtual de Redes, así como una vista detallada de cada uno de los paquetes en que se han dividido los mismos, con vistas a lograr una mayor claridad y comprensión del modelo.

4.2. Modelo de prueba

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. El objetivo de la prueba de software es descubrir errores en la aplicación. Para conseguir este objetivo se planifica y se ejecuta una serie de pasos que van revisando los elementos del software más significativos. En todas las fases del desarrollo del proyecto hay que probar el software que se va construyendo, aunque en la fase de construcción se centran los mayores esfuerzos de este flujo.

El modelo de pruebas es una colección de casos de prueba, procedimientos de prueba y componentes de prueba, que permite probar los componentes ejecutables en el modelo de implementación. Cada prueba es especificada mediante un documento que establece las condiciones de ejecución, las entradas de la prueba, y los resultados esperados, denominado caso de prueba. Cada caso de prueba especifica una forma de probar el sistema, y son aplicados como pruebas de regresión en cada iteración.

Se pueden realizar muchos casos de prueba para determinar que un caso de uso es completamente satisfactorio. Entre los casos de prueba se puede distinguir dos tipos comúnmente utilizados: las llamadas pruebas de “caja negra” y las de “caja blanca”. Una prueba de “caja negra” es una prueba del comportamiento observable externamente del sistema, mientras que una prueba de “caja blanca” prueba la interacción interna entre los componentes del sistema.

En el Laboratorio Virtual de Redes se propone la realización de una prueba de “caja negra” para los casos de uso de flujos diferentes, para probar la interacción entre el usuario y el sistema, que se satisfagan las precondiciones y poscondiciones, y que se siga la secuencia de acciones intermedias especificadas por el caso de uso. Con el propósito de comprobar que todos los casos de uso de la aplicación son revisados, se diseñaron los casos de prueba más abarcadores para cada caso de uso, que permiten comprobar las funciones básicas del sistema.

En el **Anexo 8** se presentan los casos de prueba diseñados para comprobar las funcionalidades más importantes de la aplicación.

Conclusiones:

En este capítulo se define la línea base ejecutable de la arquitectura de la aplicación. Los diagramas de despliegue y componentes que se presentan, describen los componentes a construir y su organización y dependencia entre los nodos físicos. La etapa de prueba fue uno de los pasos más importantes realizados hasta el momento; puesto que ella refleja la calidad con que ha sido llevada a cabo la proyección del sistema. Por último, los casos de prueba diseñados en este capítulo permiten verificar el correcto funcionamiento de las principales funciones de la aplicación, para comprobar si fueron cumplidos los requerimientos del sistema.

CONCLUSIONES

En este trabajo se demostró la necesidad de diseñar e implementar un sistema que permita gestionar las actividades prácticas de la asignatura Teleinformática, que les permita a los estudiantes de la UCI responder ejercicios de diseño de redes de computadoras por entrada de parámetros orientados por los profesores, y simular el funcionamiento de algunos dispositivos.

Luego de un análisis de las tecnologías más usadas en la actualidad para la construcción de sistemas informáticos similares, se elaboró la propuesta desarrollar una aplicación sobre plataforma Web, basada en una arquitectura cliente servidor de 3 capas, utilizando como lenguaje de programación el PHP, y el Java Script en el navegador. Para el módulo de simulación del funcionamiento de algunos dispositivos de redes, se propone utilizar el lenguaje de programación C++. Se llegó a la conclusión también de que la metodología idónea para llevar a cabo el proceso de desarrollo es RUP (Proceso Unificado de Rational).

Se modeló el dominio; se definieron los requerimientos del sistema, tanto funcionales como no funcionales, y posteriormente se estructuró el modelo de casos de uso del sistema, describiéndose cada caso de uso para una mejor comprensión de la funcionalidad que brindan. También se diseñó el sistema, a través de diagramas de clases Web y el diagrama de clases persistentes. Se estructuró el modelo de datos, que es la representación física de la base de datos del sistema. Posteriormente se elaboró el modelo de despliegue y el de implementación.

Luego de todo este proceso de trabajo se puede concluir que el Laboratorio Virtual de Redes es un sistema que da solución a la situación problemática que lo originó y que su explotación significará una mejora considerable en la calidad y eficiencia de los procesos que automatiza.

Por todo lo anterior se concluye que los objetivos propuestos para el presente proyecto han sido cumplidos satisfactoriamente, incluyendo una serie de recomendaciones que deben tenerse en cuenta para el trabajo futuro.

RECOMENDACIONES

Se recomienda continuar perfeccionando la herramienta propuesta de simulación de dispositivos de redes Simul, incluyéndole detalles más específicos de varios dispositivos que les aporten conocimientos más profundos de la asignatura Teleinformática.

Se recomienda incluirle a la aplicación Web, Laboratorio Virtual de Redes otras funcionalidades importantes para la asignatura, y para el Departamento de Sistemas Digitales. Perfeccionar la manera de reportar los errores por ejercicio, es decir, que el profesor pueda dar varias respuestas correctas para que el sistema sea más flexible a la hora de calificar.

Se recomienda enriquecer la aplicación con más información de los contenidos por tema de la asignatura Teleinformática.

En vista a la política de migración hacia software libre llevada a cabo por nuestro país y a la que no se encuentra ajena nuestra universidad, se recomienda en etapas venideras hacer un profundo estudio sobre la migración del simulador **Simul** a software libre.

Implantar el sistema en la Intranet de la Universidad de las Ciencias Informáticas para prestar los servicios implementados.

BIBLIOGRAFÍA

1. *100% Cisco*. [En línea] <http://cisco.awardspace.com/>.
2. **Alfaro, Félix**. Arquitectura Cliente/Servidor. [En línea] 2006. <http://www.inei.gob.pe/biblioineipub/bancopub/inf/Lib5038/indice.HTM>.
3. **Boggs, Wendy y Boogs, Michael**. *UML with Rational Rose*. 2003. ISBN-13: 0025211440179.
4. **Comer, Douglas E**. *Redes Globales de Información con Internet y TCP/IP*. 1996. ISBN:968-880-541-6.
5. **Derfler, Frank**. *Descubre redes LAN & WAN*. Madrid : Prentice Hall, 1998. ISBN:84-8322-091-1.
6. **Gallego de Torres, Antonio**. *Enrutadores Cisco*. 2003. ISBN:84-415-1504-2.
7. **Grau, Xavier Ferré y Sánchez Segura, María Isabel**. *Desarrollo orientado a objetos con UML*. 2004.
8. HERRAMIENTAS SOFTWARE PARA LA SIMULACIÓN DE REDES DE COMUNICACIONES. [En línea] 2006. <http://nsl.csie.nctu.edu.tw/NCTUnsReferences/capitulo4.pdf>.
9. **Jacobson, Ivar, Bouch, Grady y Rumbaugh, James**. *El proceso unificado de desarrollo de software*. 2000.
10. **Larman, Craig**. *UML y Patrones, Una introducción al análisis y diseño orientado a objetos*. La Habana : Félix Varela, 2004.
11. **Miguel Jiménez, Juan Manuel**. *Simulador de redes telemáticas*. 2003.
12. Network Simulator. [En línea] 2006. <http://nsl.csie.nctu.edu.tw/NCTUnsReferences/simulatorbeschreibung.pdf>.
13. **Pressman, Roger S**. *Ingeniería de Software. Un enfoque práctico*. 2002.
14. Sistemas Gestores de Bases de Datos. *Wikipedia, la enciclopedia libre*. [En línea] <http://es.wikipedia.org/wiki/SGBD>.
15. **Vega, Jesús**. Desarrollo de Aplicaciones Web. [En línea] 2002. <http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node17.html>.
16. **Álvarez, Sofía, Hernández Anaisa**. *Metodología para el desarrollo de aplicaciones con tecnología Orientada a Objetos utilizando notación UML*. La Habana, 2000.
17. **Barrientos Enríquez, Aleida Mirian**. *El proceso Unificado de Modelado (RUP)*. [En línea] 2005 <http://www.monografias.com/trabajos16/lenguaje-modelado-unificado/lenguaje-modelado-unificado.shtml>.
18. **Date, C. J**. Introducción a los Sistemas de Bases de Datos. Editorial Félix Varela, La Habana, 2003.
19. *Rational Rose Enterprise Edition*. Rational Software Corporation. 2003.
20. **González, Carlos D**. Curso: Integral de diseño. *Programación de sitios dinámicos con MySQL y PHP*. [En línea] 2005 http://www.usabilidadweb.com.ar/x_int.php.

21. **González, Carlos D.** Curso: *Sitios Web dinámicos con Base de Datos PostgreSQL y PHP*. [En línea] 2005 <http://www.usabilidadweb.com.ar/postgre.php>
22. **Molpeceres, Alberto.** *Procesos de desarrollo: RUP, XP y FDD*. [En línea] 2002 <http://www.javahispano.org/articles.article.action?id=76>
23. Wikipedia, la enciclopedia libre. Sistemas Gestores de Bases de Datos.

ANEXOS

Anexo 1: Expansión de Casos de Uso restantes.

Tabla 28 CU Gestionar proveedor.

Caso de Uso	
CU-3	Gestionar proveedor
Propósito:	Permitir registrar y eliminar los proveedores de los productos.
Actor(es):	Usuario Avanzado(Inicia)
Resumen:	El caso de uso se inicia cuando el usuario avanzado selecciona la opción gestionar proveedor. El sistema permite al usuario avanzado, adicionar o eliminar un proveedor.
Referencias:	RF6, RF7, RF8
Acción del Actor	Respuesta del Sistema
1. El caso de uso se inicia cuando el usuario avanzado selecciona la opción que desea realizar.	2. Si el usuario avanzado selecciona: a) Registrar un proveedor, ir a la sección "Registrar proveedor" b) Eliminar un proveedor, ir a la sección "Eliminar proveedor".
Sección1: "Registrar proveedor"	
3. El usuario avanzado entra el nombre del proveedor para realizar su registro en la aplicación.	4. El sistema verifica que los campos obligatorios estén llenos.
	5. El sistema verifica que este proveedor no exista.
	6. El proveedor se almacena en el sistema.
	7. Se muestra un mensaje informándole al usuario avanzado que ha sido efectuado el registro del proveedor y finaliza el caso de uso.
Cursos Alternativos	
CA 4. Se emite un mensaje para que el usuario avanzado llene los campos obligatorios.	
CA 5. Si el proveedor existe se muestra un mensaje informativo y finaliza el caso de uso.	

Sección2: "Eliminar proveedor"	
3. El usuario avanzado selecciona el proveedor a eliminar.	4. El sistema pide confirmación.
	5. El sistema elimina el proveedor.
	6. Se muestra un mensaje de eliminación y finaliza caso de uso.
Puntos de extensión	

Tabla 29 CU Gestionar categoría.

Caso de Uso	
CU-4	Gestionar categoría
Propósito:	Permitir registrar y eliminar las categorías de los productos.
Actor(es):	Usuario Avanzado(Inicia)
Resumen:	El caso de uso se inicia cuando el usuario avanzado selecciona la opción gestionar categoría. El sistema permite al usuario avanzado, adicionar o eliminar una categoría.
Referencias:	RF9, RF10, RF11
Acción del Actor	Respuesta del Sistema
1. El caso de uso se inicia cuando el usuario avanzado selecciona la opción que desea realizar.	2. Si el usuario avanzado selecciona: a) Registrar una categoría, ir a la sección "Registrar categoría" b) Eliminar una categoría, ir a la sección "Eliminar categoría".
Sección1: "Registrar categoría"	
3. El usuario avanzado entra el nombre de la categoría para realizar su registro en la aplicación.	4. El sistema verifica que los campos obligatorios estén llenos.
	5. El sistema verifica que esta categoría no exista.
	6. La categoría se almacena en el sistema.
	7. Se muestra un mensaje informándole al usuario

	avanzado que ha sido efectuado el registro de la categoría y finaliza el caso de uso.
Cursos Alternativos	
CA 4. Se emite un mensaje para que el usuario avanzado llene los campos obligatorios.	
CA 5. Si la categoría existe se muestra un mensaje informativo y finaliza el caso de uso.	
Sección2: "Eliminar categoría"	
3. El usuario avanzado selecciona la categoría a eliminar.	4. El sistema pide confirmación.
	5. El sistema elimina la categoría.
	6. Se muestra un mensaje de eliminación y finaliza caso de uso.
Puntos de extensión	

Tabla 30 CU Gestionar unidad de medida.

Caso de Uso	
CU-5	Gestionar unidad de medida.
Propósito:	Permitir registrar y eliminar las unidades de medidas de los productos.
Actor(es):	Usuario Avanzado(Inicia)
Resumen:	El caso de uso se inicia cuando el usuario avanzado selecciona la opción gestionar unidad de medida. El sistema permite al usuario avanzado, adicionar o eliminar una unidad de medida.
Referencias:	RF12, RF13, RF14
Acción del Actor	Respuesta del Sistema
1. El caso de uso se inicia cuando el usuario avanzado selecciona la opción que desea realizar.	2. Si el usuario avanzado selecciona: a) Registrar una unidad de medida, ir a la sección "Registrar unidad de medida" b) Eliminar una unidad de medida, ir a la sección "Eliminar unidad de medida".
Sección1: "Registrar unidad de medida"	

3. El usuario avanzado entra el nombre de la unidad de medida para realizar su registro en la aplicación.	4. El sistema verifica que los campos obligatorios estén llenos.
	5. El sistema verifica que esta unidad de medida no exista.
	6. La unidad de medida se almacena en el sistema.
	7. Se muestra un mensaje informándole al usuario avanzado que ha sido efectuado el registro de la unidad de medida y finaliza el caso de uso.
Cursos Alternativos	
CA 4. Se emite un mensaje para que el usuario avanzado llene los campos obligatorios.	
CA 5. Si la unidad de medida existe se muestra un mensaje informativo y finaliza el caso de uso.	
Sección2: "Eliminar unidad de medida"	
3. El usuario avanzado selecciona la unidad de medida a eliminar.	4. El sistema pide confirmación.
	5. El sistema elimina la unidad de medida.
	6. Se muestra un mensaje de eliminación y finaliza caso de uso.
Puntos de extensión	

Tabla 31 CU Establecer conexión.

Caso de Uso	
CU-7	Establecer conexión.
Propósito:	Permitir registrar y eliminar conexiones en una subred.
Actor(es):	Usuario(Inicia)
Resumen:	El caso de uso se inicia cuando el usuario selecciona la opción hacer simulación. El sistema permite al usuario establecer una conexión entre una PC o servidor a un puerto de una subred.
Referencias:	RF18, RF19
Acción del Actor	Respuesta del Sistema

1. El caso de uso se inicia cuando el usuario selecciona la opción que desea realizar.	2. Si el usuario selecciona: a) Establecer una conexión, ir a la sección "Establecer una conexión" b) Eliminar una conexión, ir a la sección "Eliminar una conexión".
Sección1: "Establecer una conexión"	
3. El usuario selecciona la subred.	4. El sistema muestra las direcciones IP de las PC y los servidores que se encuentran sin conexión en la subred, y una lista de los puertos conectados y sin conectar.
5. El usuario selecciona el IP de la PC o del servidor y el puerto a conectar.	6. El sistema dibuja la conexión, dependiendo del tipo de conexión será el color de esta.
Sección2: " Eliminar una conexión"	
3. El usuario selecciona la conexión a eliminar.	4. El sistema pide confirmación.
	5. El sistema elimina la conexión.
Puntos de extensión	

Tabla 32 CU Simular interior dispositivos.

Caso de Uso	
CU-9	Simular interior dispositivos.
Propósito:	Permitir simular el interior de algunos dispositivos de red.
Actor(es):	Usuario(Inicia)
Resumen:	El caso de uso se inicia cuando el usuario avanzado selecciona la opción hacer simulación. El sistema permite al usuario observar el interior de algunos dispositivos de red.
Referencias:	RF22
Acción del Actor	Respuesta del Sistema
1. El caso de uso se inicia cuando el usuario selecciona el dispositivo de red a simular	2. El sistema dibuja el interior del dispositivo seleccionado.

internamente.	
Puntos de extensión	

Tabla 33 CU Gestionar topología.

Caso de Uso	
CU-12	Gestionar topología
Propósito:	Permitir registrar y eliminar las topologías de red.
Actor(es):	Usuario Avanzado(Inicia)
Resumen:	El caso de uso se inicia cuando el usuario avanzado selecciona la opción gestionar topología. El sistema permite al usuario avanzado, adicionar o eliminar una topología.
Referencias:	RF27, RF28, RF29
Acción del Actor	Respuesta del Sistema
1. El caso de uso se inicia cuando el usuario avanzado selecciona la opción que desea realizar.	2. Si el usuario avanzado selecciona: a) Registrar una topología, ir a la sección "Registrar topología" b) Eliminar una topología, ir a la sección "Eliminar topología".
Sección1: "Registrar topología"	
3. El usuario avanzado entra el nombre de la topología para realizar su registro en la aplicación.	4. El sistema verifica que los campos obligatorios estén llenos.
	5. El sistema verifica que esta topología no exista.
	6. La topología se almacena en el sistema.
	7. Se muestra un mensaje informándole al usuario avanzado que ha sido efectuado el registro de la topología y finaliza el caso de uso.
Cursos Alternativos	
CA 4. Se emite un mensaje para que el usuario avanzado llene los campos obligatorios.	
CA 5. Si la topología existe se muestra un mensaje informativo y finaliza el caso de uso.	

Sección2: "Eliminar topología"	
3. El usuario avanzado selecciona la topología a eliminar.	4. El sistema pide confirmación.
	5. El sistema elimina la topología.
	6. Se muestra un mensaje de eliminación y finaliza caso de uso.
Puntos de extensión	

Tabla 34 CU Gestionar estándar.

Caso de Uso	
CU-13	Gestionar estándar
Propósito:	Permitir registrar y eliminar algún estándar de red.
Actor(es):	Usuario Avanzado(Inicia)
Resumen:	El caso de uso se inicia cuando el usuario avanzado selecciona la opción gestionar estándar. El sistema permite al usuario avanzado, adicionar o eliminar un estándar.
Referencias:	RF30, RF31, RF32
Acción del Actor	Respuesta del Sistema
1. El caso de uso se inicia cuando el usuario avanzado selecciona la opción que desea realizar.	2. Si el usuario avanzado selecciona: a) Registrar un estándar, ir a la sección "Registrar estándar" b) Eliminar un estándar, ir a la sección "Eliminar estándar".
Sección1: "Registrar estándar"	
3. El usuario avanzado entra el nombre del estándar para realizar su registro en la aplicación.	4. El sistema verifica que los campos obligatorios estén llenos.
	5. El sistema verifica que este estándar no exista.
	6. EL estándar se almacena en el sistema.
	7. Se muestra un mensaje informándole al usuario avanzado que ha sido efectuado el registro del

	estándar y finaliza el caso de uso.
Cursos Alternativos	
CA 4. Se emite un mensaje para que el usuario avanzado llene los campos obligatorios.	
CA 5. Si el estándar existe se muestra un mensaje informativo y finaliza el caso de uso.	
Sección2: "Eliminar estándar"	
3. El usuario avanzado selecciona el estándar a eliminar.	4. El sistema pide confirmación.
	5. El sistema elimina el estándar.
	6. Se muestra un mensaje de eliminación y finaliza caso de uso.
Puntos de extensión	

Tabla 35 CU Gestionar reporte ejercicio.

Caso de uso	
CU- 14	Gestionar reporte ejercicio
Propósito	Mostrar al usuario la respuesta correcta del ejercicio, los errores que tuvo en el mismo, así como una calificación.
Actor(es)	Usuario(Inicia)
Resumen:	El usuario selecciona consultar reporte de un ejercicio realizado anteriormente, y el sistema le muestra el reporte del ejercicio con la respuesta correcta, los errores y la calificación alcanzada.
Referencias	RF33, RF34
Acción del actor	Respuesta del sistema
1. El caso de uso se inicia cuando un usuario selecciona la opción consultar reporte de un ejercicio.	2. El sistema muestra una ventana con la lista de todos los ejercicios resueltos que tiene el usuario.
3. El usuario selecciona el ejercicio del cual desea consultar reporte.	4. El sistema muestra una ventana con el reporte del ejercicio seleccionado. El mismo contiene la respuesta correcta del ejercicio, los errores, y la calificación propuesta.

Puntos de extensión

Tabla 36 CU Consultar estadísticas.

Caso de uso	
CU-15	Consultar estadísticas
Propósito	Mostrar estadísticas sobre los resultados obtenidos por estudiante o por ejercicio a los usuarios avanzados.
Actor(es)	Usuario Avanzado(Inicia)
Resumen:	El caso de uso se inicia cuando el usuario avanzado selecciona la opción mostrar estadísticas. El sistema muestra al usuario avanzado la estadística que él seleccione.
Referencias	RF35
Acción del actor	Respuesta del sistema
1. El caso de uso se inicia cuando el usuario avanzado selecciona la opción consultar estadísticas.	2. El sistema muestra una ventana para seleccionar el tipo de estadística a mostrar, por estudiante o por ejercicio.
3. El usuario avanzado selecciona la estadística a consultar.	4. El sistema muestra al usuario avanzado la estadística seleccionada.
Puntos de extensión	

Tabla 37 CU Operar historial usuario.

Caso de uso	
CU-16	Operar historial usuario.
Propósito	Mostrar historial de los estudiantes al usuario avanzado para evaluar desarrollo de estos.
Actor(es)	Usuario Avanzado(Inicia)
Resumen:	El caso de uso se inicia cuando el usuario avanzado selecciona la opción revisar historial de usuario. El sistema muestra al usuario avanzado la lista de usuarios y este selecciona el historial a revisar.
Referencias	RF36
Acción del actor	Respuesta del sistema

1. El caso de uso se inicia cuando el usuario avanzado selecciona la opción revisar historial de usuario.	2. El sistema muestra una ventana con la lista de usuarios.
3. El usuario avanzado selecciona el usuario para revisar historial.	4. El sistema muestra al usuario avanzado el historial del usuario.
Puntos de extensión	

Tabla 38 CU Publicar documentación.

Caso de Uso	
CU- 17	Publicar documentación
Propósito	Publicar documentación por parte del usuario avanzado.
Actor(es)	Usuario Avanzado(Inicia)
Resumen:	El caso de uso se inicia cuando el usuario avanzado selecciona la opción Publicar documentación. El sistema le permite guardar al usuario avanzado, documentación relacionada con la asignatura.
Referencias	RF37, RF38, RF39
Acción del Actor	Respuesta del Sistema
1. El caso de uso se inicia cuando el usuario avanzado selecciona la opción que desea realizar.	2. Si el usuario avanzado selecciona: a) Insertar una publicación, ir a la sección "Registrar publicación" b) Eliminar una publicación, ir a la sección "Eliminar publicación".
Sección1: "Registrar publicación"	
3. El usuario avanzado entra el nombre de la publicación para realizar su registro en la aplicación.	4. El sistema verifica que los campos obligatorios estén llenos.
	5. El sistema verifica que esta publicación no exista.
	6. La publicación se almacena en el sistema.
	7. Se muestra un mensaje informándole al usuario avanzado que ha sido efectuado el registro y finaliza el

	caso de uso.
Cursos Alternativos	
CA 4. Se emite un mensaje para que el usuario avanzado llene los campos obligatorios.	
CA 5. Si la publicación existe se muestra un mensaje informativo y finaliza el caso de uso.	
Sección2: "Eliminar publicación"	
3. El usuario avanzado selecciona la publicación a eliminar.	4. El sistema pide confirmación.
	5. El sistema elimina la publicación.
	6. Se muestra un mensaje de eliminación y finaliza caso de uso.
Puntos de extensión	

Tabla 39 CU Administrar usuarios.

Caso de uso	
CU-18	Administrar usuarios
Propósito	Permitir modificar el rol de los diferentes usuarios del sistema.
Actor(es)	Administrador(Inicia)
Resumen:	El caso de uso se inicia cuando el Administrador selecciona la opción de gestionar usuarios avanzados del sistema. El sistema permite al administrador seleccionar que usuarios del sistema tienen el rol de usuario avanzado.
Referencias	RF40, RF41, RF42
Acción del actor	Respuesta del sistema
1. El caso de uso se inicia cuando el Administrador selecciona la opción de administrar usuarios avanzados.	2. El sistema muestra una ventana con la opción de introducir datos de un usuario para realizar la búsqueda.
3. El Administrador introduce los datos de usuario.	4. El sistema verifica que este usuario exista.
	5. El sistema muestra al administrador los datos del usuario y la opción de modificarle el rol del sistema.
6. El administrador seleccionar el rol del	7. El sistema guarda los cambios efectuados.

sistema para asignarlo al usuario.	
	8. Se muestra un mensaje informándole al administrador que ha sido efectuada la modificación del usuario y finaliza el caso de uso.
Cursos Alternativos	
CA 4. Si el usuario no existe se muestra un mensaje informativo y finaliza el caso de uso.	
Puntos de extensión	

Anexo 2: Diagramas de clases del análisis restantes.

Figura 17 Diagrama de clases de análisis. CU Gestionar proveedor.

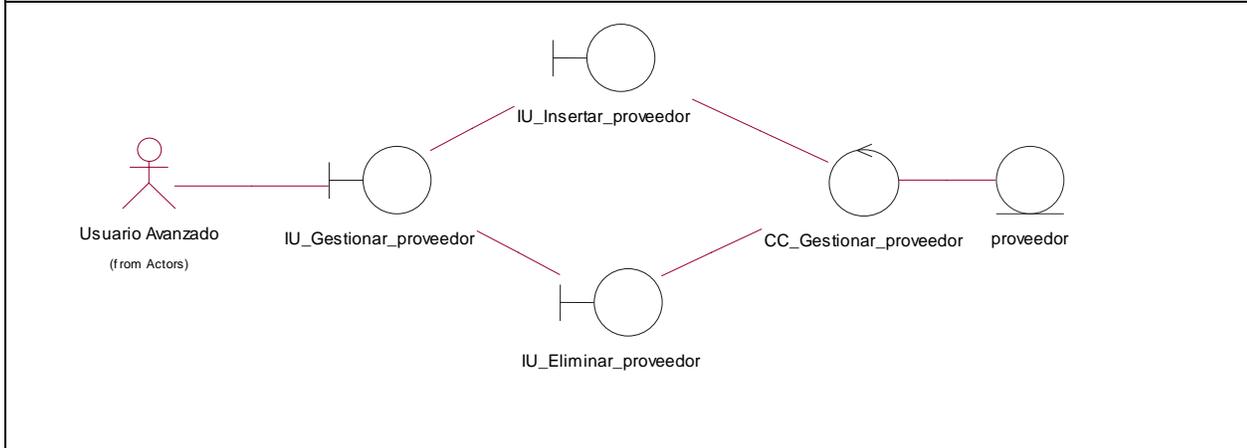


Figura 18 Diagrama de clases de análisis. CU Gestionar categoría.

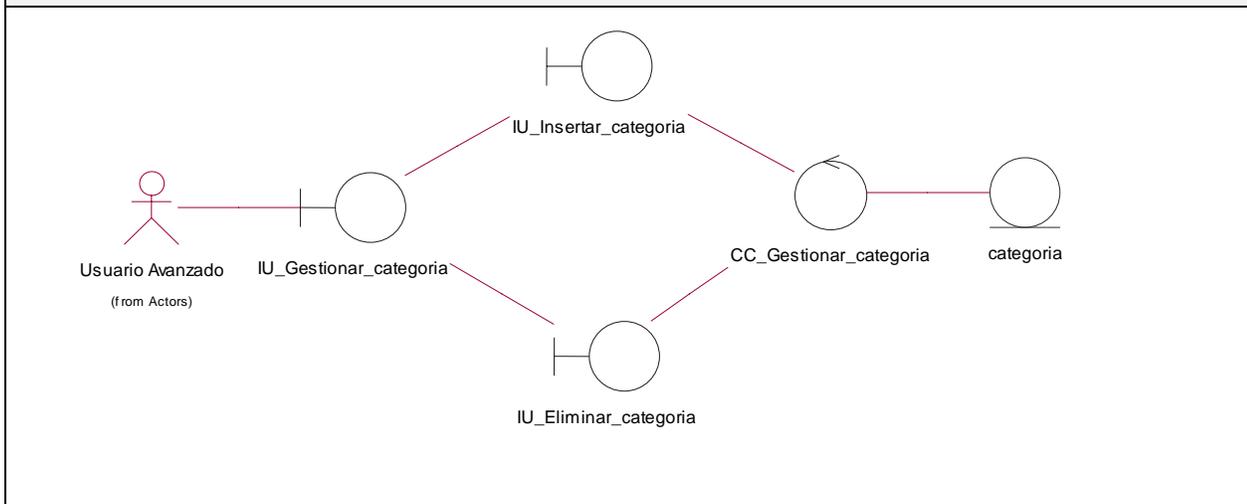


Figura 19 Diagrama de clases de análisis. CU Gestionar unidad de medida.

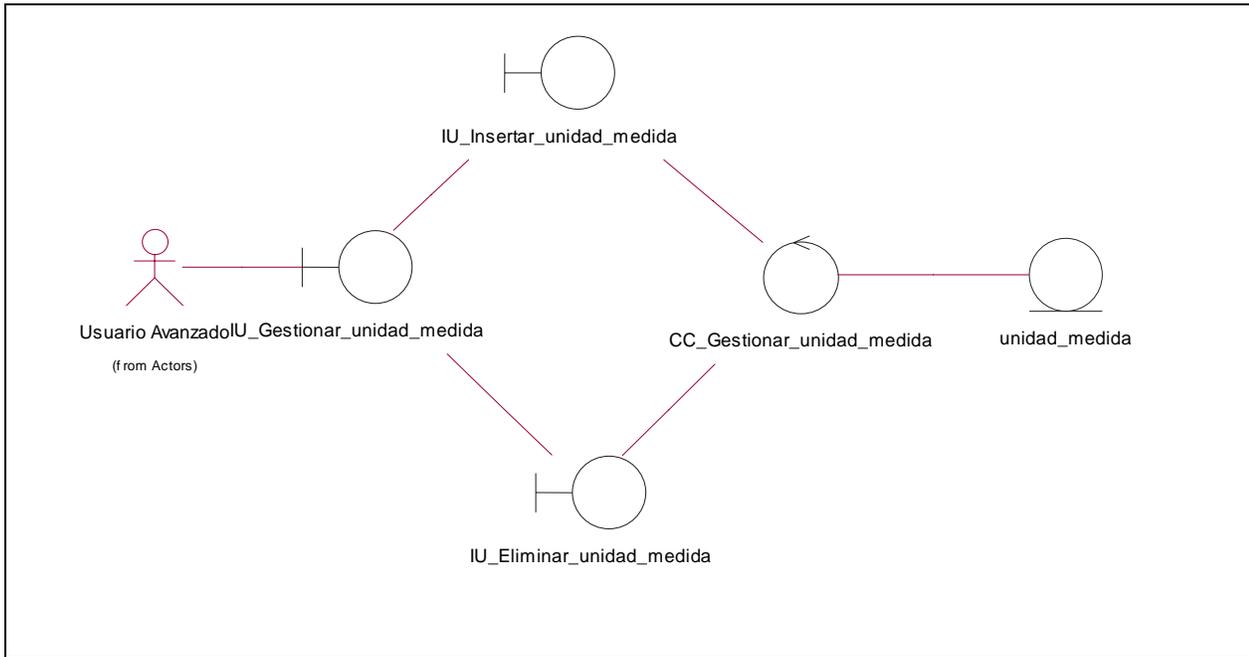


Figura 20 Diagrama de clases de análisis. CU Establecer conexión.

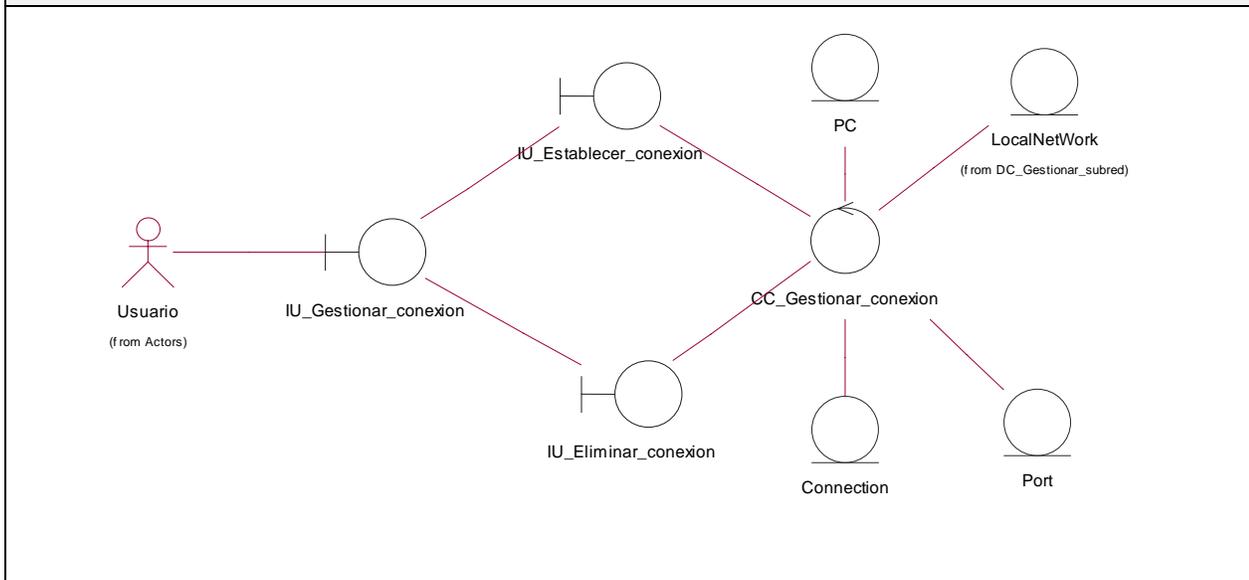


Figura 21 Diagrama de clases de análisis. CU Simular interior dispositivos.

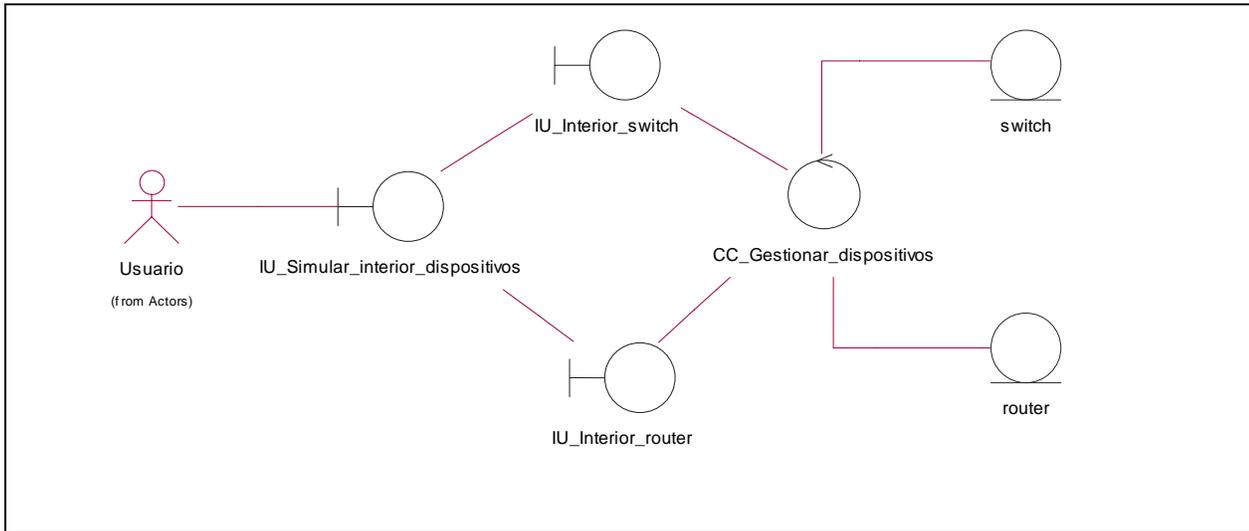


Figura 22 Diagrama de clases de análisis. CU Gestionar topología.

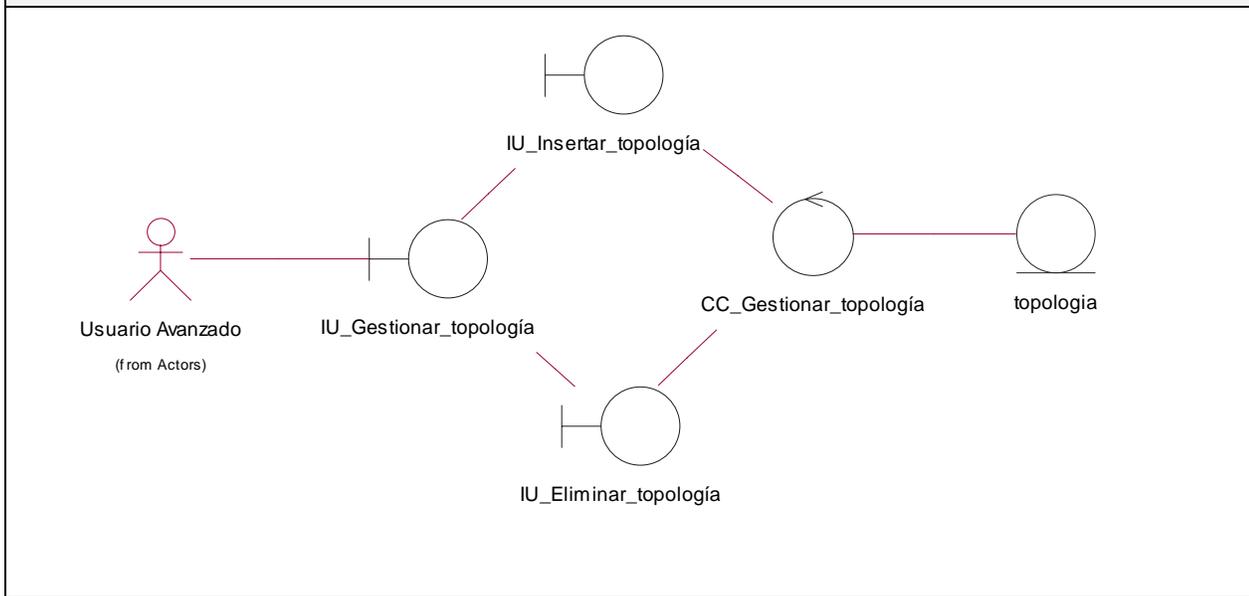


Figura 23 Diagrama de clases de análisis. CU Gestionar estándar.

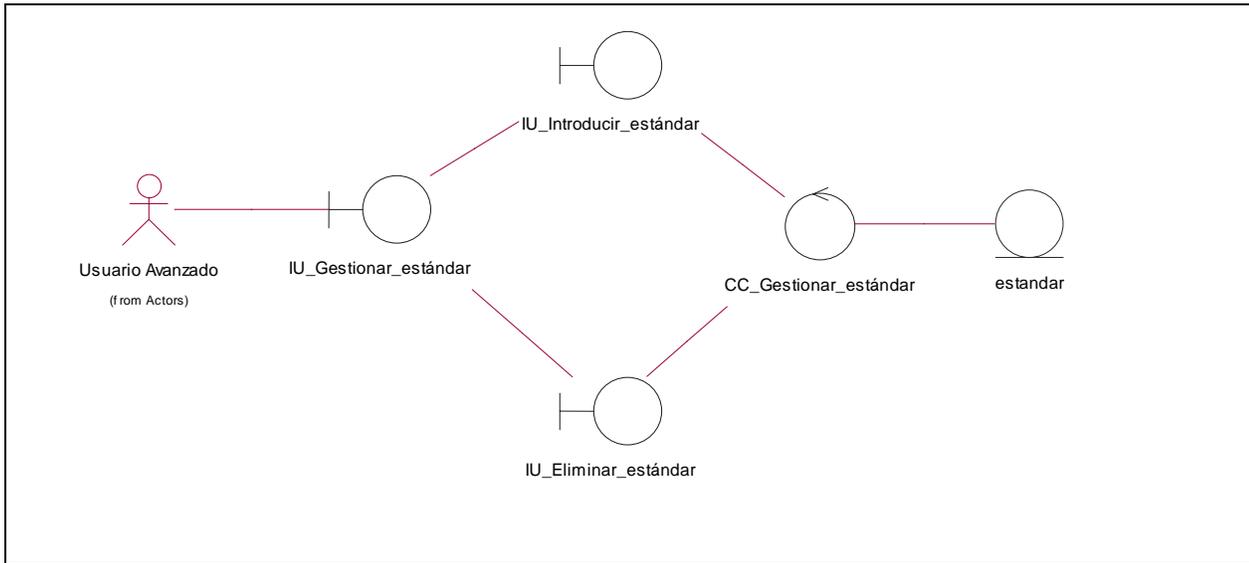


Figura 24 Diagrama de clases de análisis. CU Gestionar reporte ejercicio.

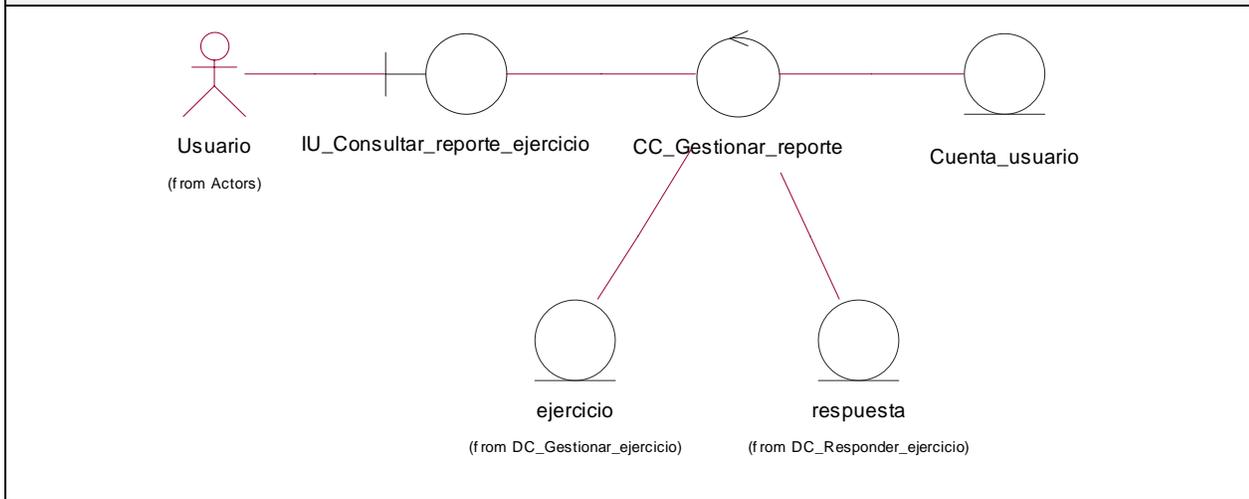


Figura 25 Diagrama de clases de análisis. CU Consultar estadísticas.

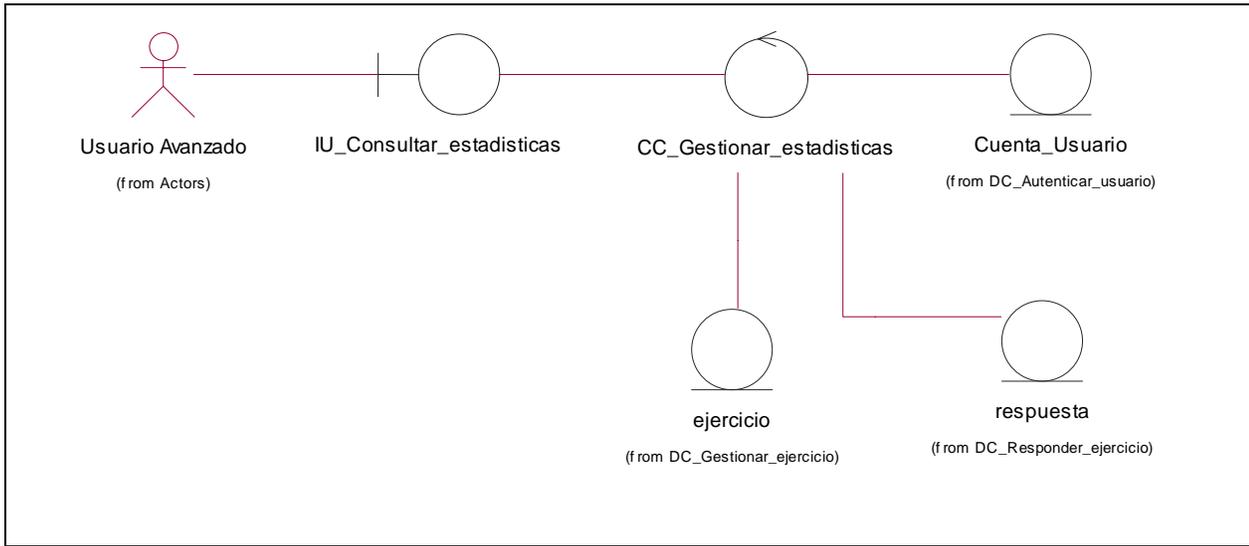


Figura 26 Diagrama de clases de análisis. CU Operar historial usuario.

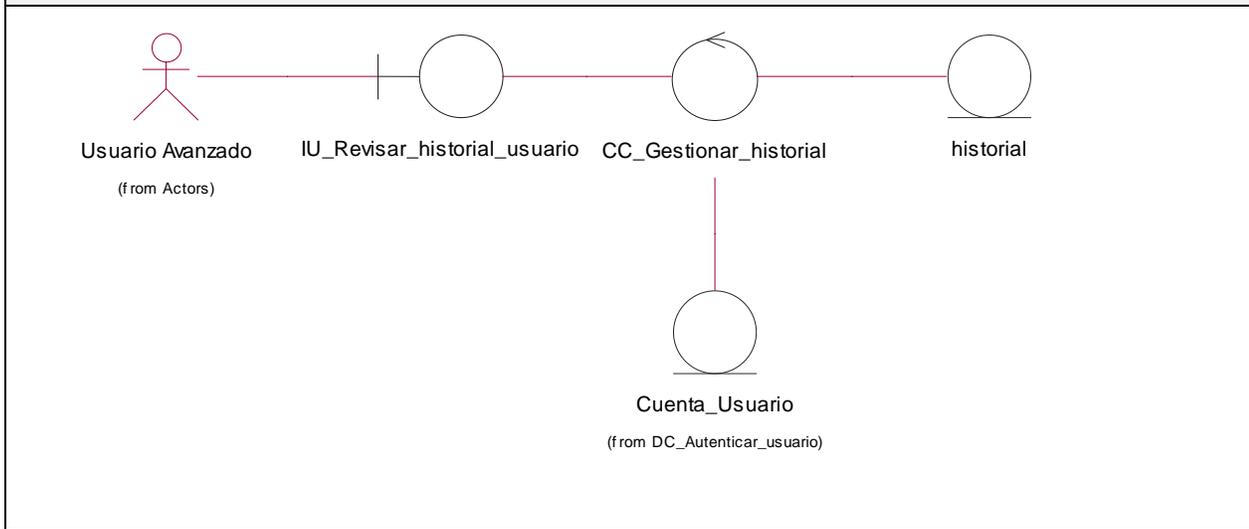


Figura 27 Diagrama de clases de análisis. CU Publicar documentación.

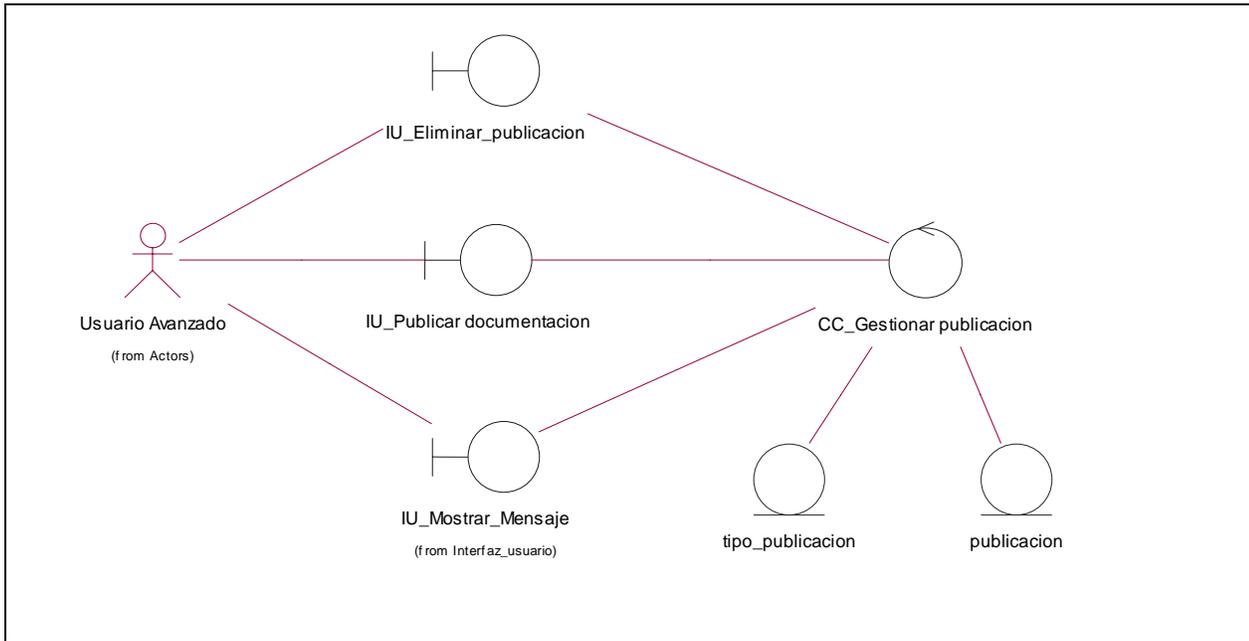
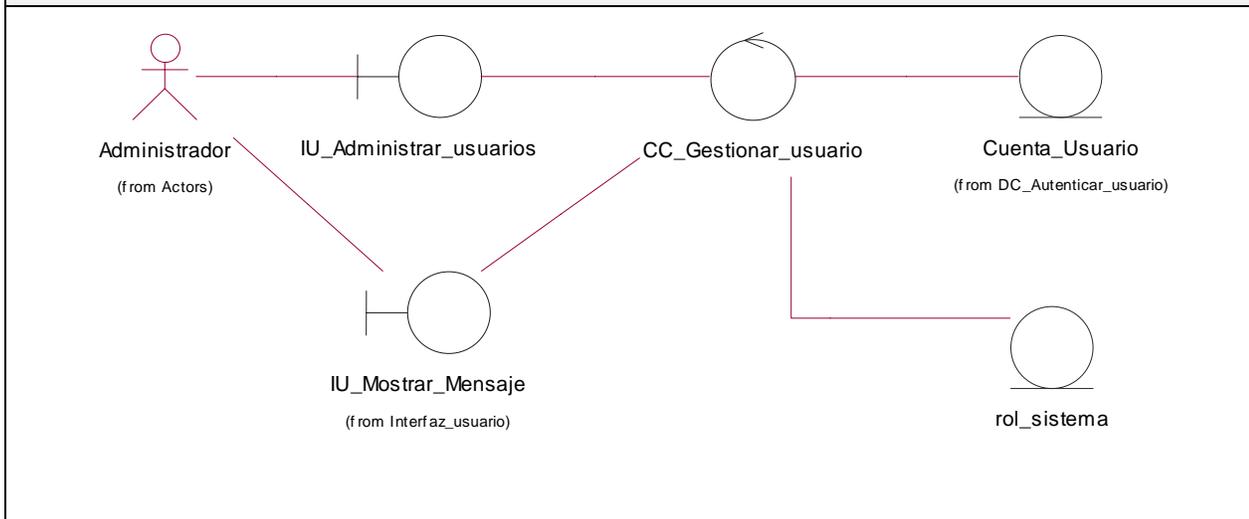


Figura 28 Diagrama de clases de análisis. CU Administrar usuarios.



Anexo 3: Diagramas de Secuencia para cada Caso de Uso.

Figura 29 Diagrama de secuencia CU Autenticar usuario.

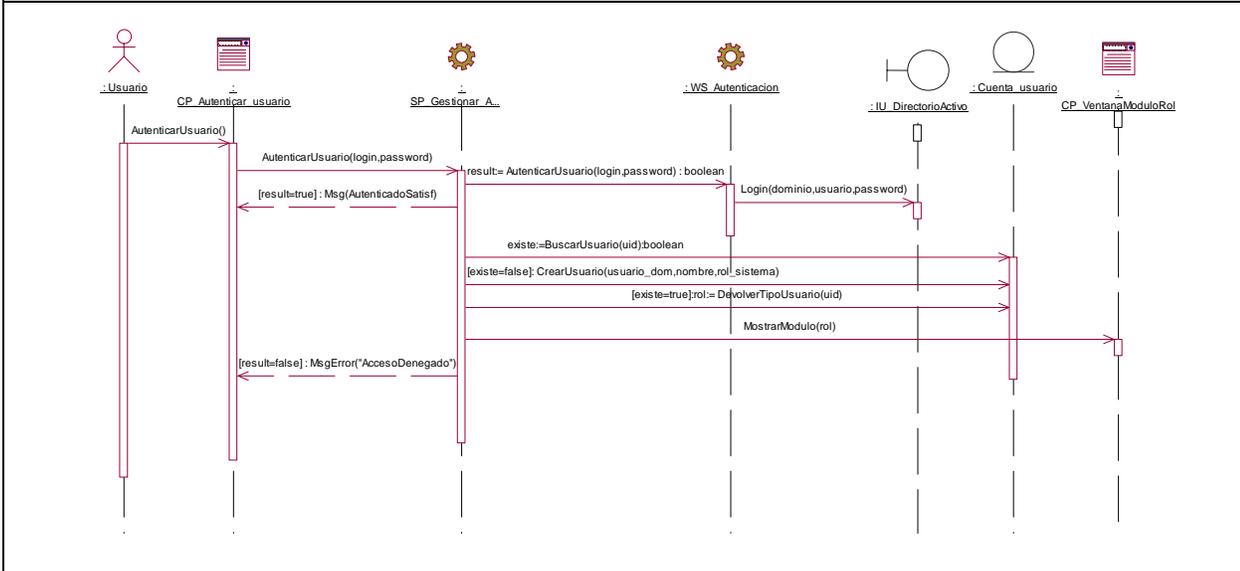


Figura 30 Diagrama de secuencia. CU Gestionar Producto. Agregar Producto.

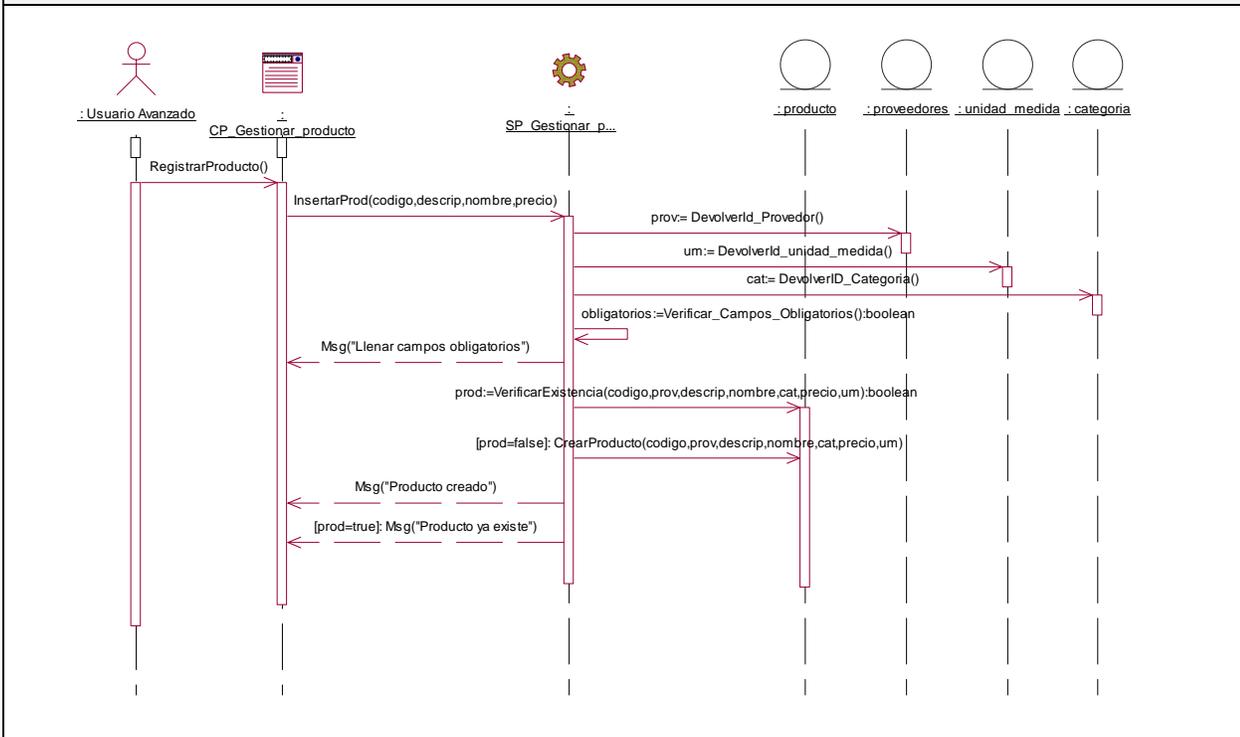


Figura 31 Diagrama de secuencia. CU Gestionar Producto. Modificar Producto.

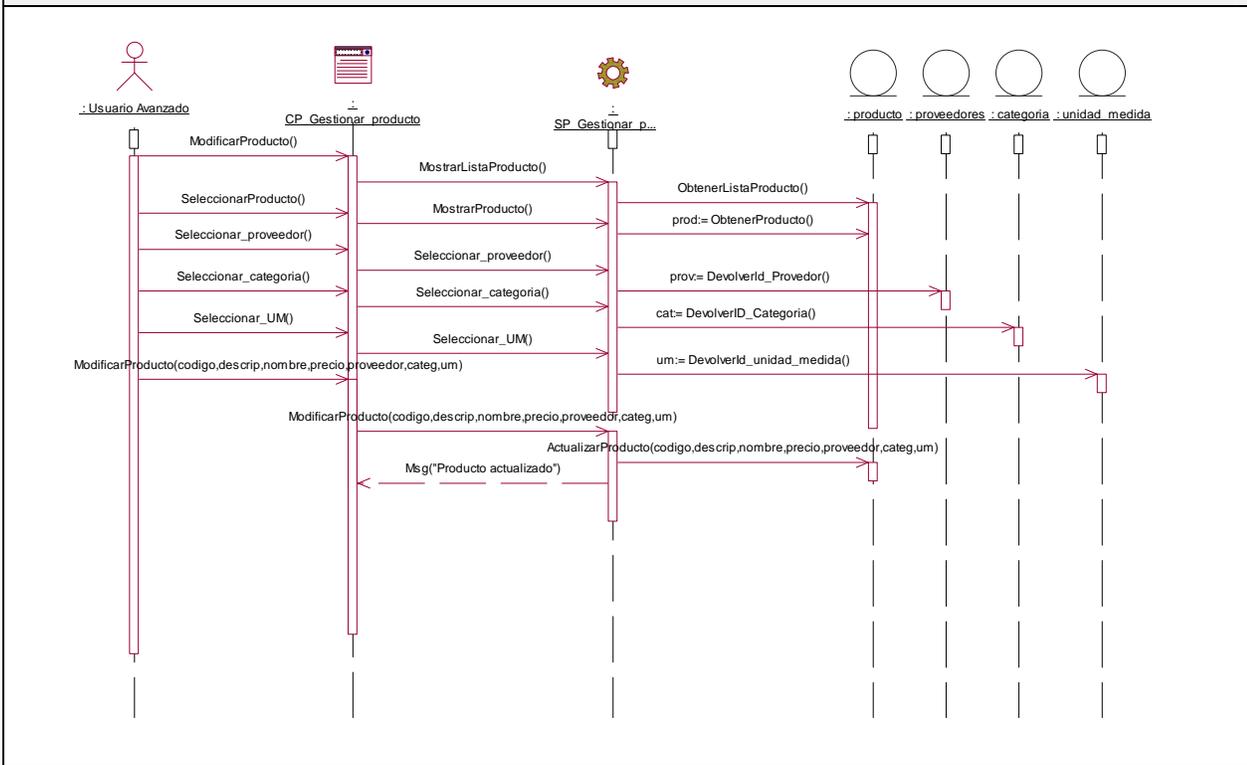


Figura 32 Diagrama de secuencia. CU Gestionar Producto. Eliminar Producto.

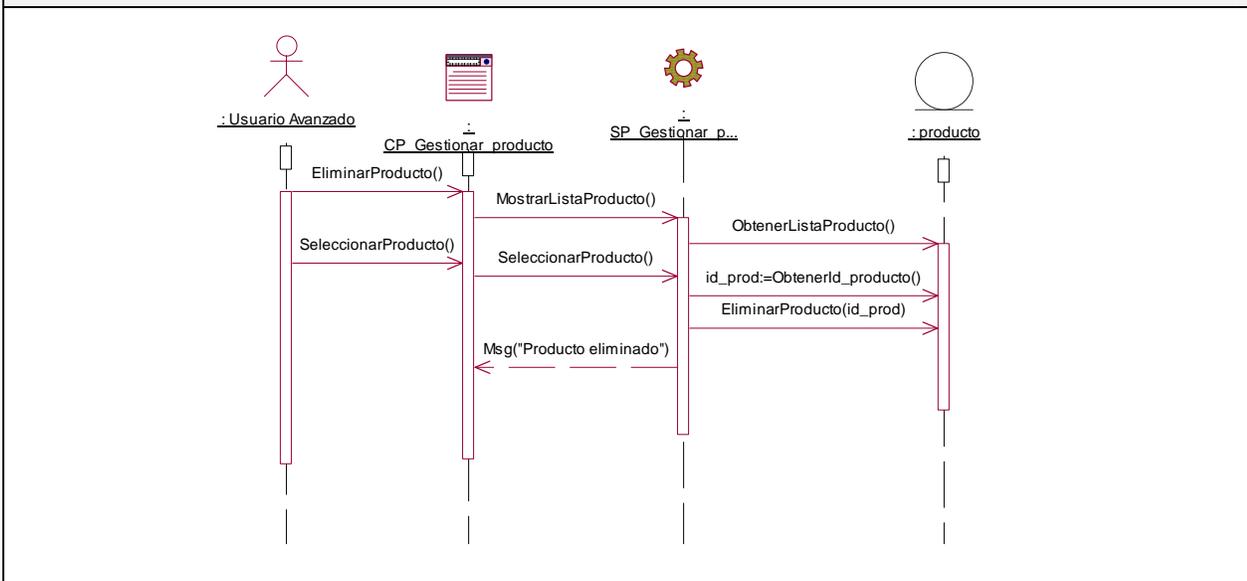


Figura 33 Diagrama de secuencia. CU Gestionar proveedor. Insertar proveedor.

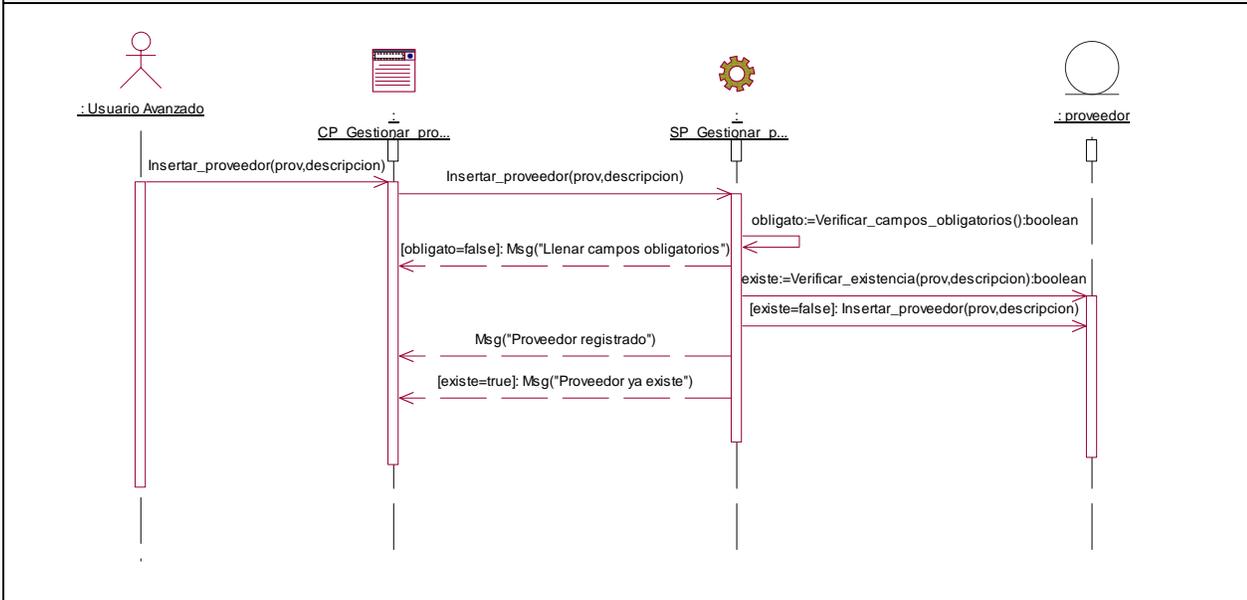


Figura 34 Diagrama de secuencia. CU Gestionar proveedor. Eliminar proveedor.

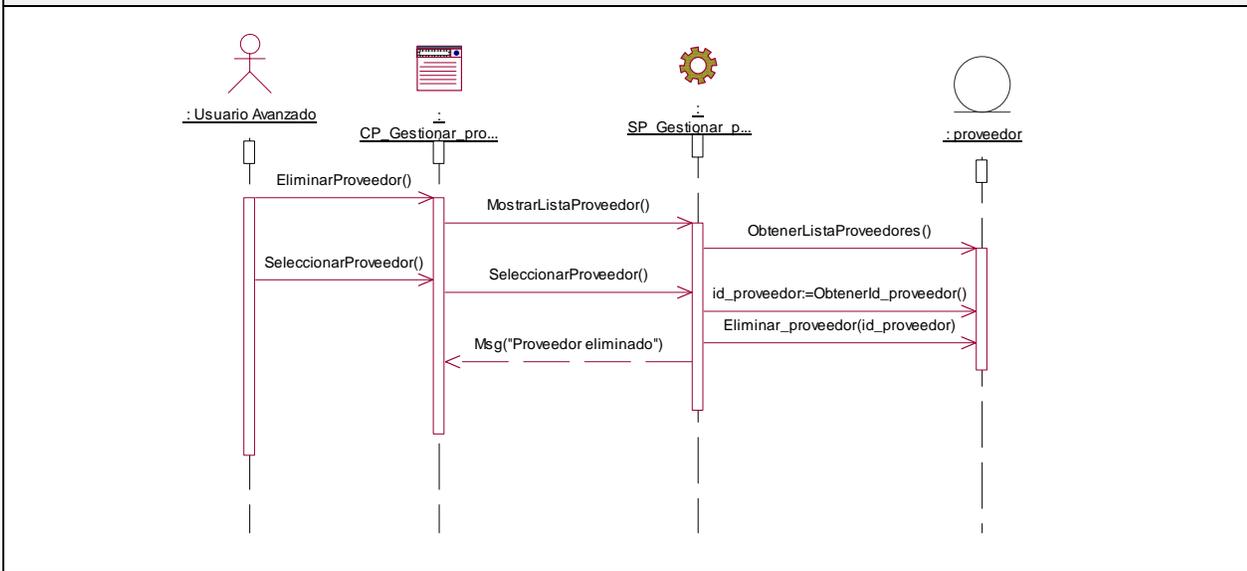


Figura 35 Diagrama de secuencia. CU Gestionar categoría. Insertar categoría.

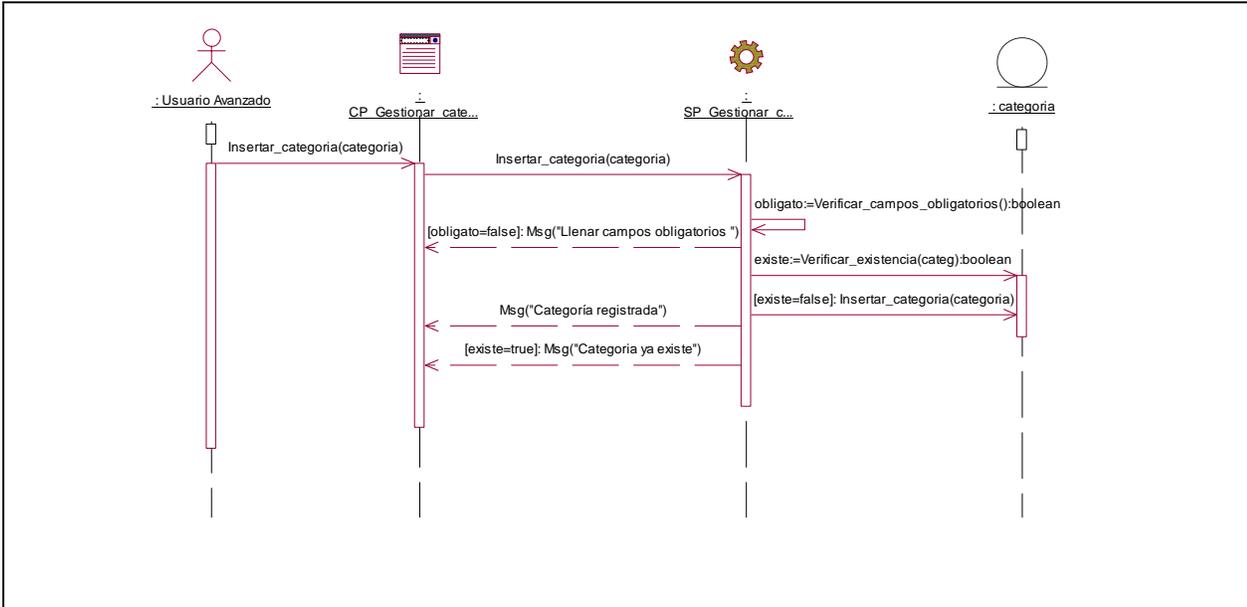


Figura 36 Diagrama de secuencia. CU Gestionar categoría. Eliminar categoría.

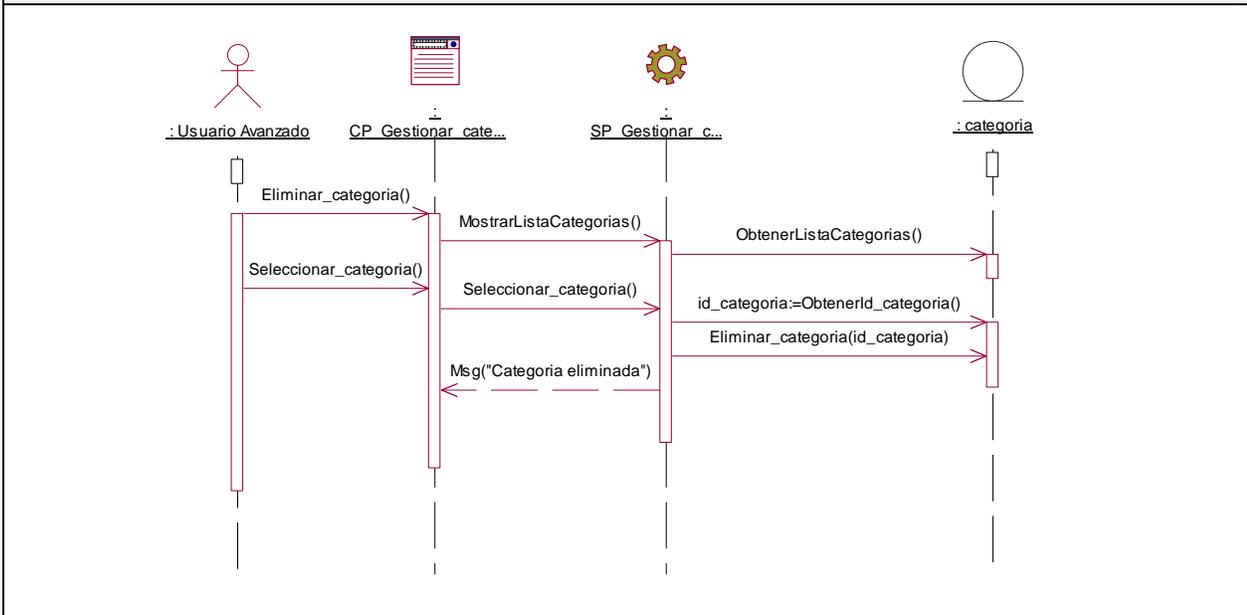


Figura 37 Diagrama de secuencia. CU Gestionar unidad de medida. Insertar UM.

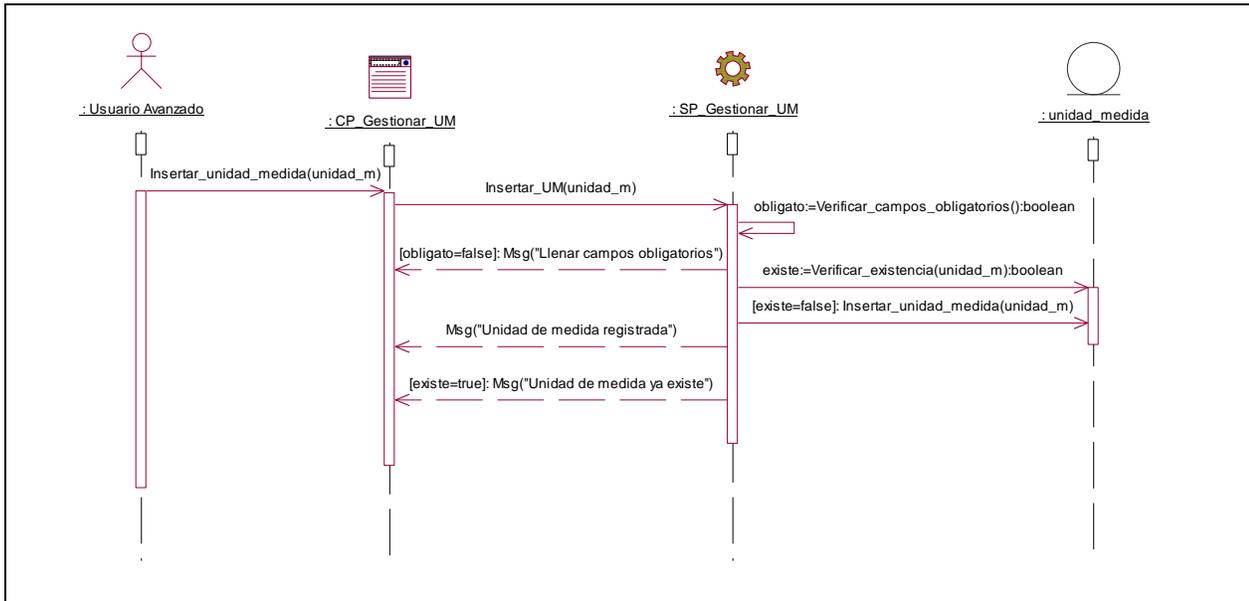


Figura 38 Diagrama de secuencia. CU Gestionar unidad de medida. Eliminar UM.

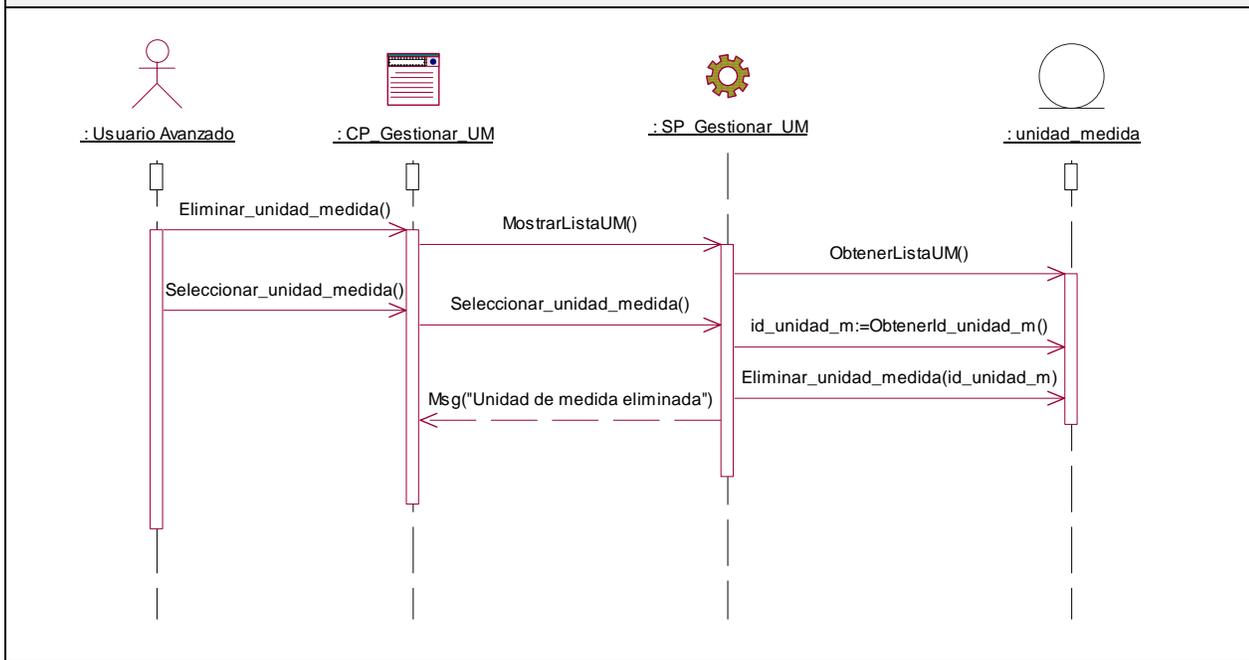


Figura 39 Diagrama de secuencia. CU Gestionar subred. Adicionar PC.

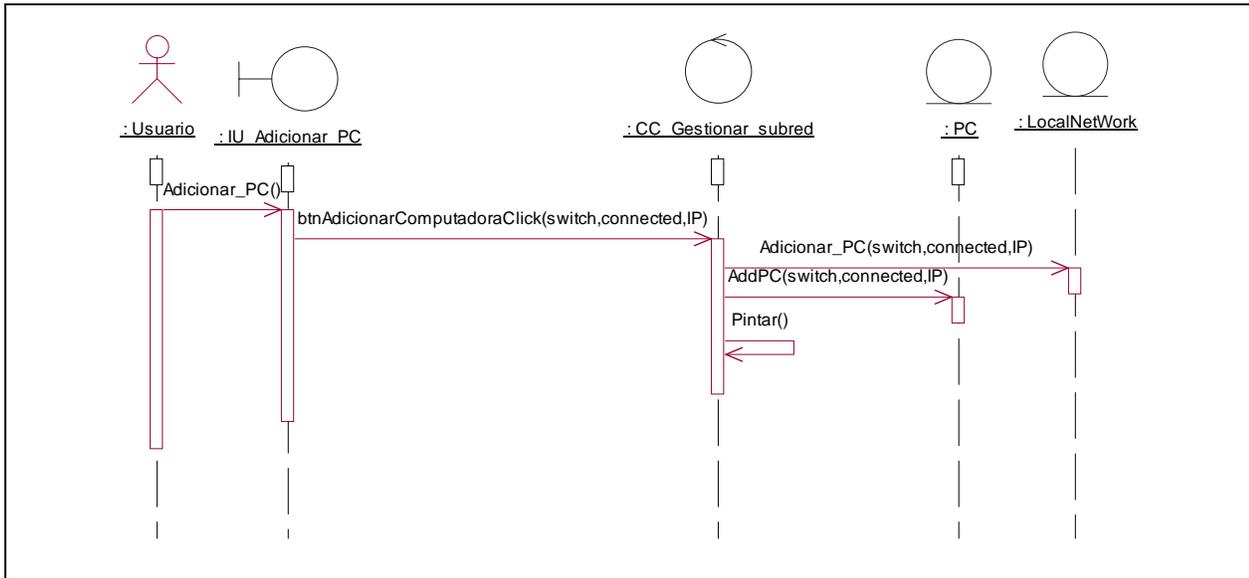


Figura 40 Diagrama de secuencia. CU Gestionar subred. Adicionar puerto.

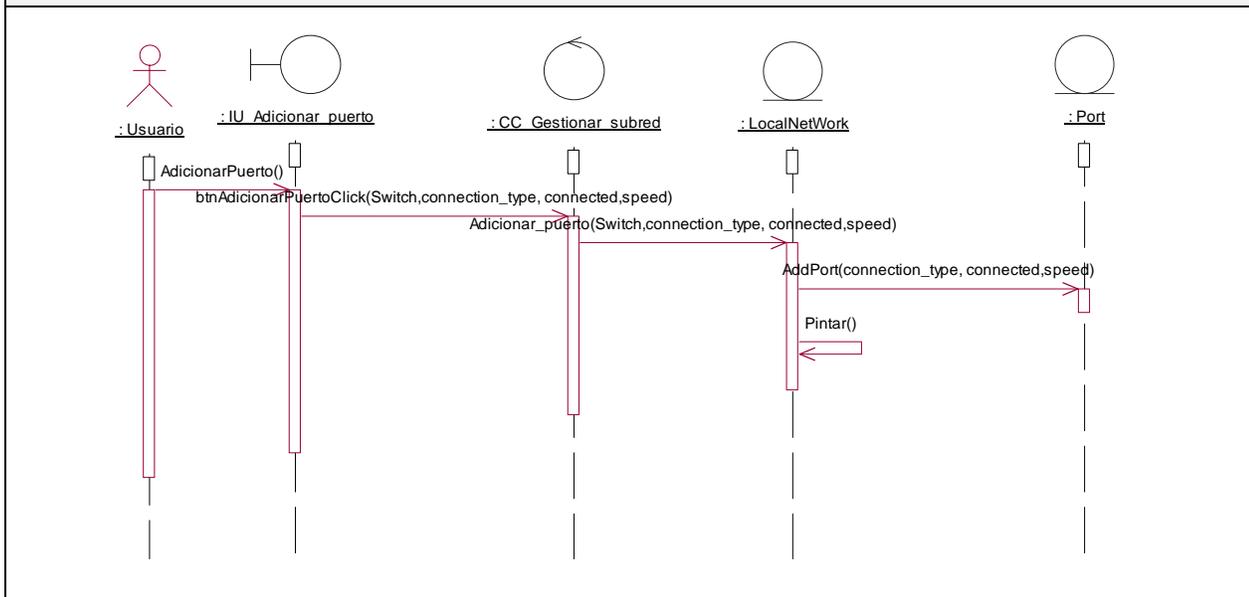


Figura 41 Diagrama de secuencia. CU Gestionar subred. Adicionar servidor.

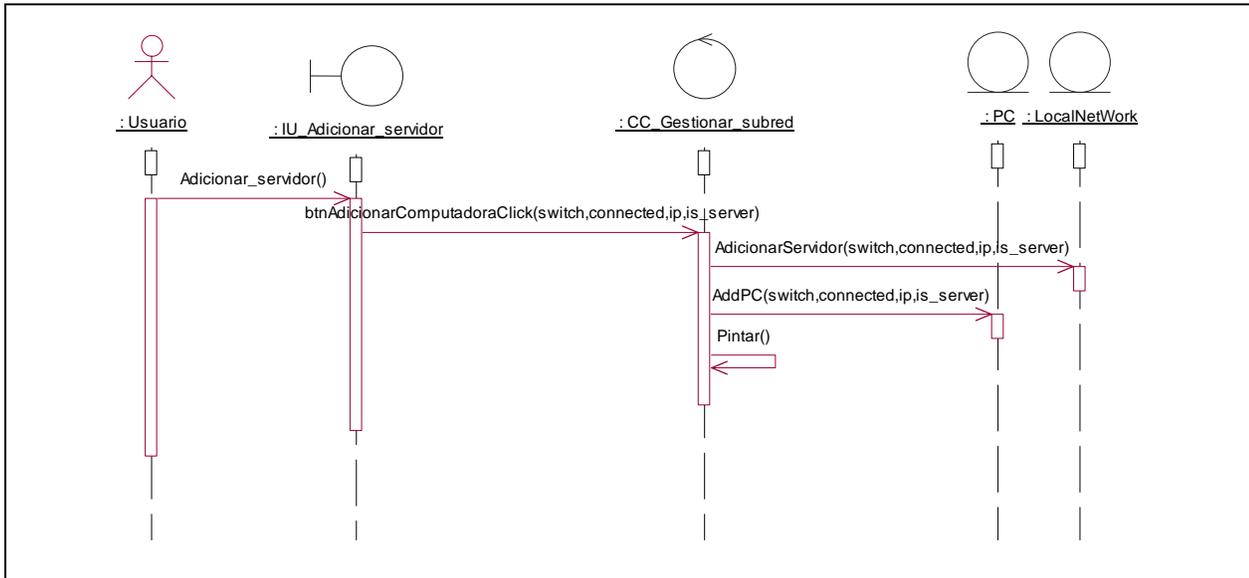


Figura 42 Diagrama de secuencia. CU Establecer conexión. Establecer conexión.

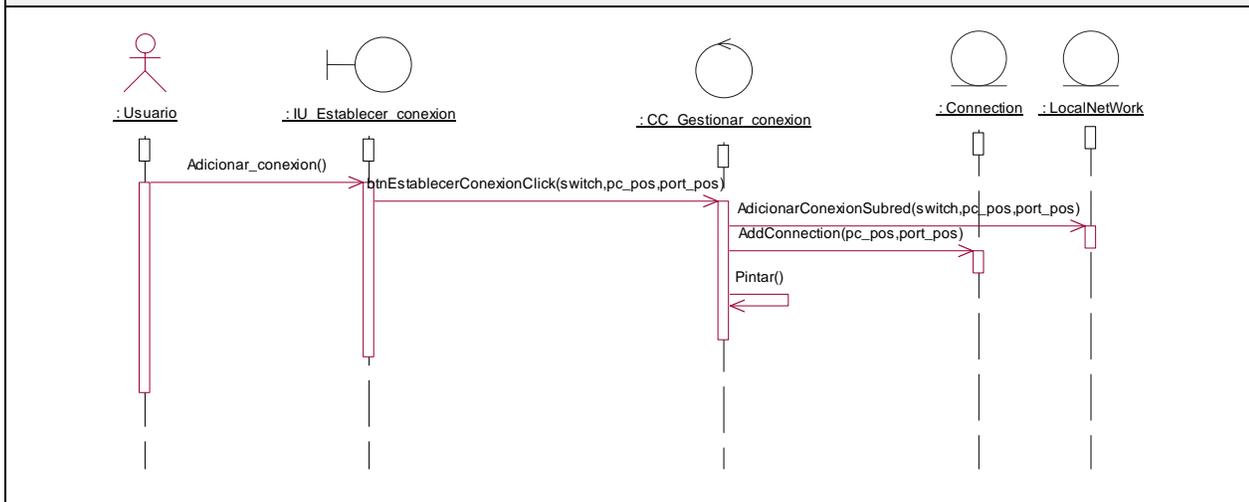


Figura 43 Diagrama de secuencia. CU Establecer conexión. Eliminar conexión.

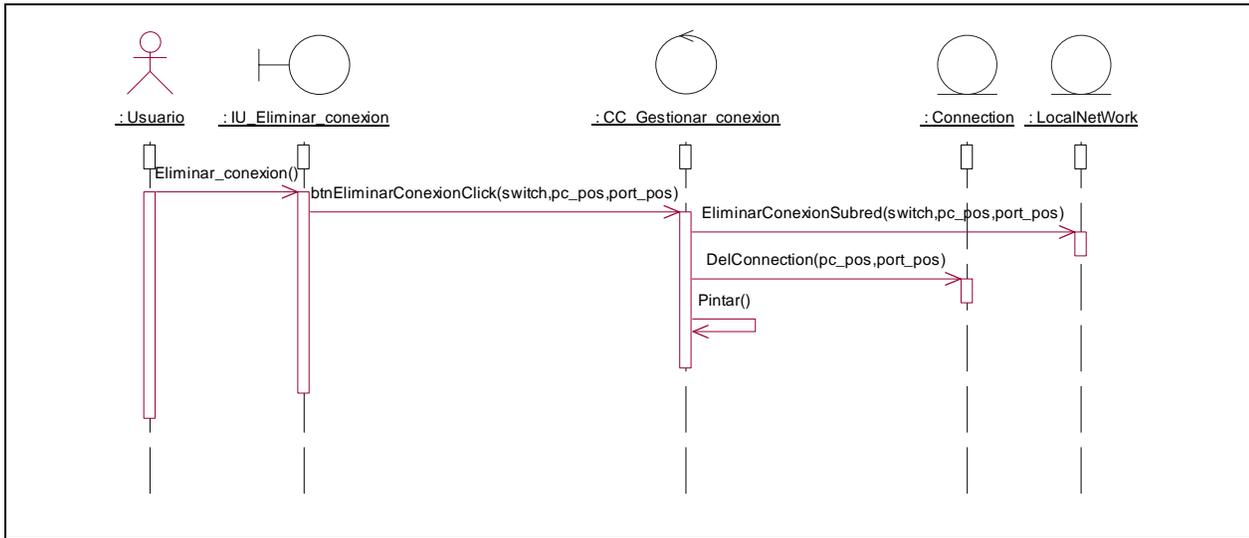


Figura 44 Diagrama de secuencia. CU Enviar paquete. Crear envío de paquete.

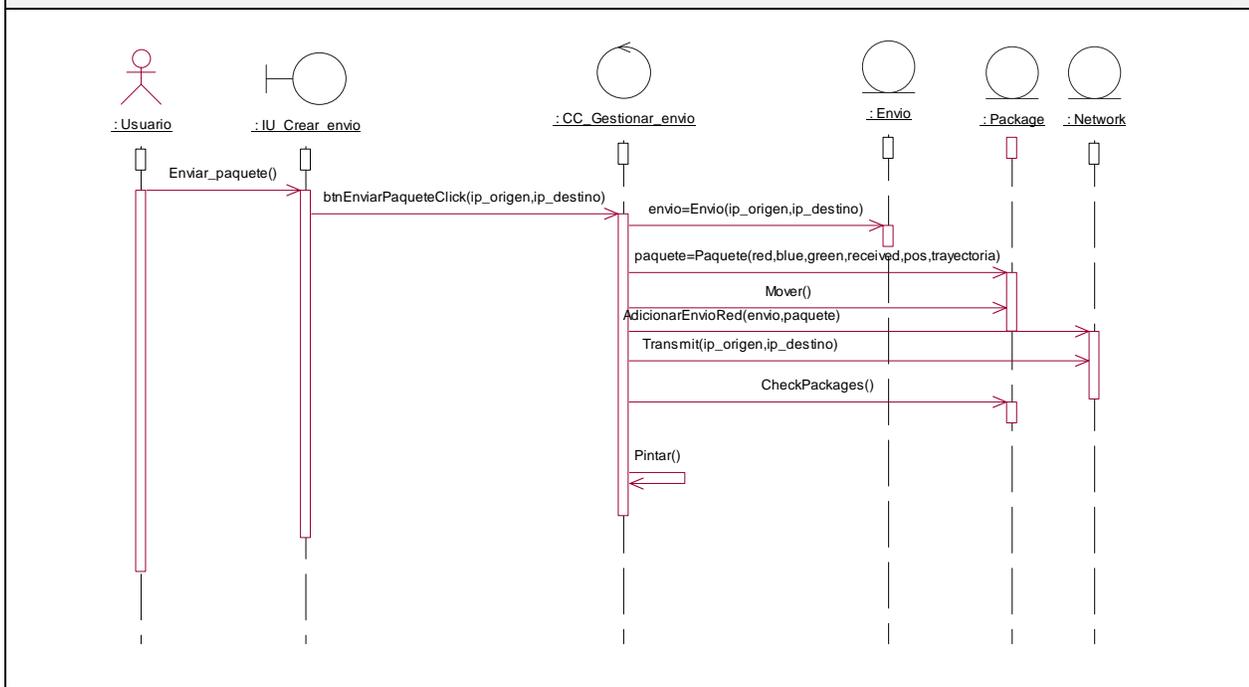


Figura 45 Diagrama de secuencia. CU Enviar paquete. Cancelar envío de paquete.

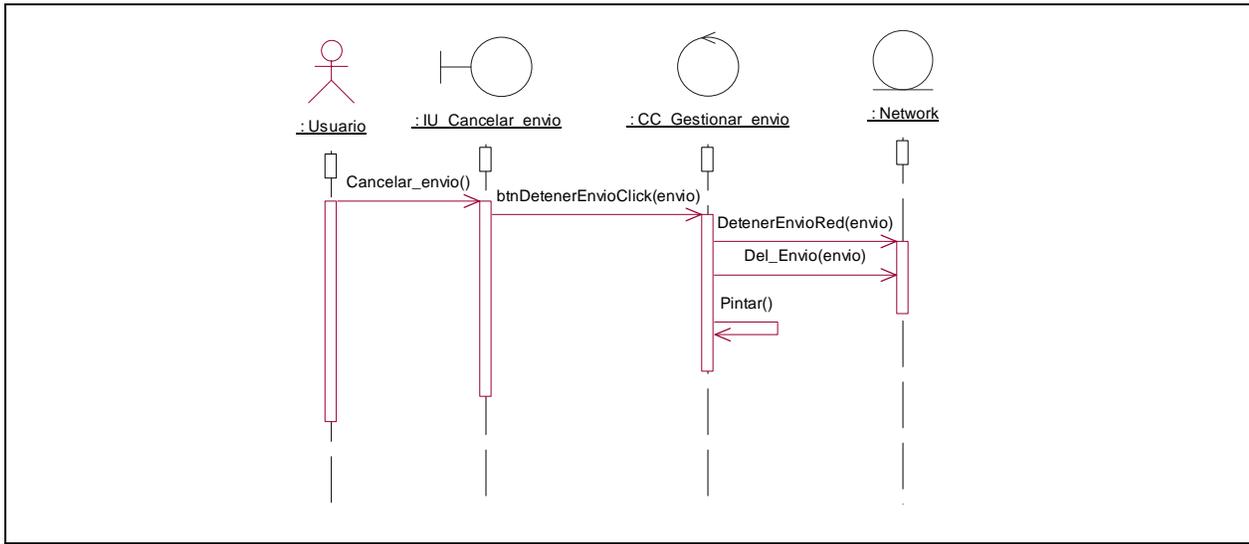


Figura 46 Diagrama de secuencia. CU Simular interior dispositivos. Interior router.

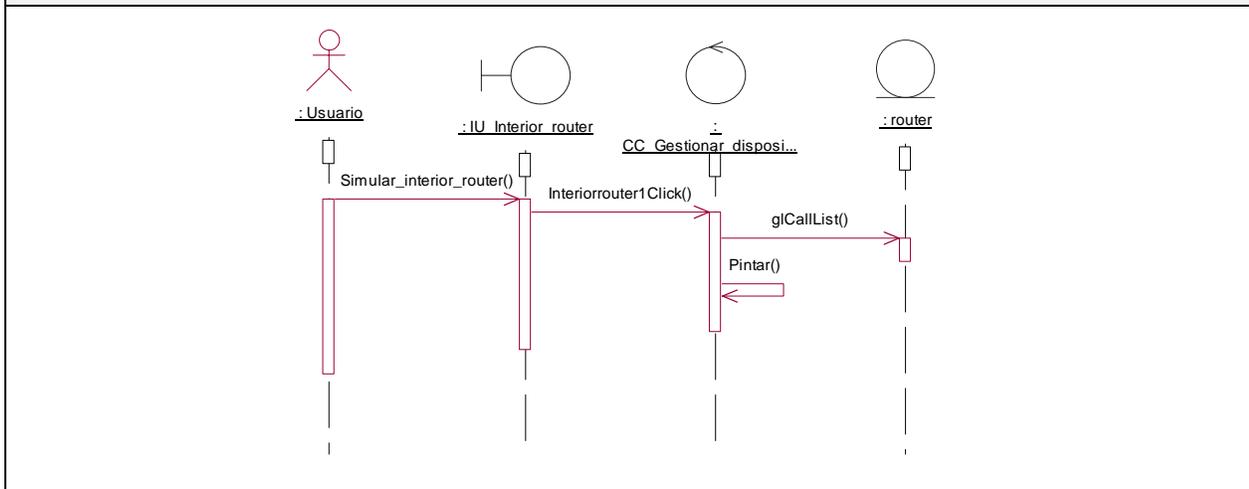


Figura 47 Diagrama de secuencia. CU Simular interior dispositivos. Interior switch.

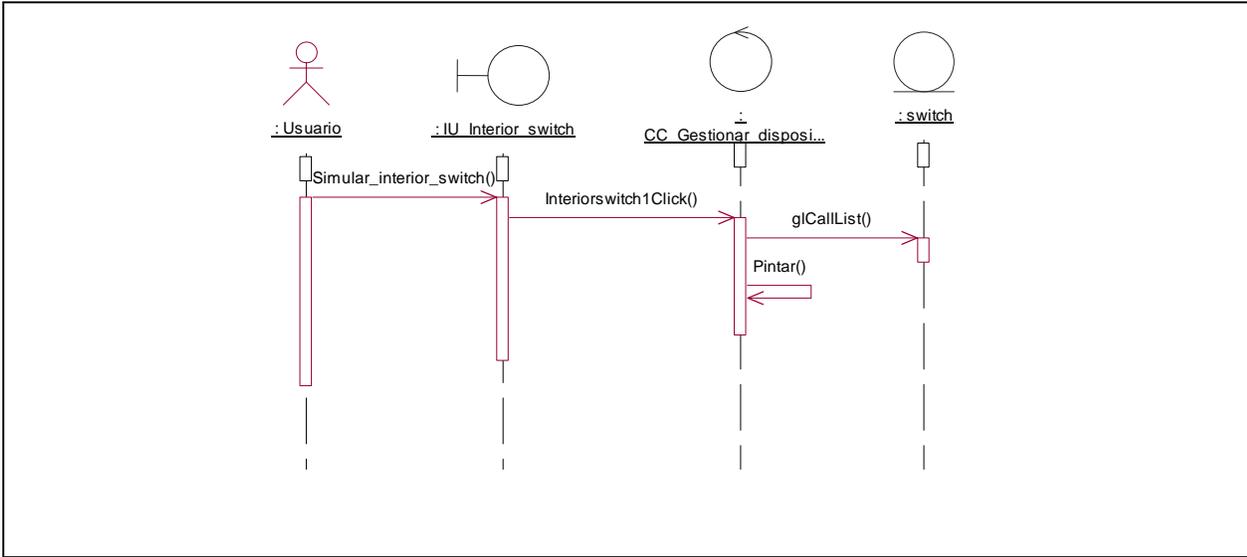


Figura 48 Diagrama de secuencia. CU Gestionar ejercicio. Insertar ejercicio.

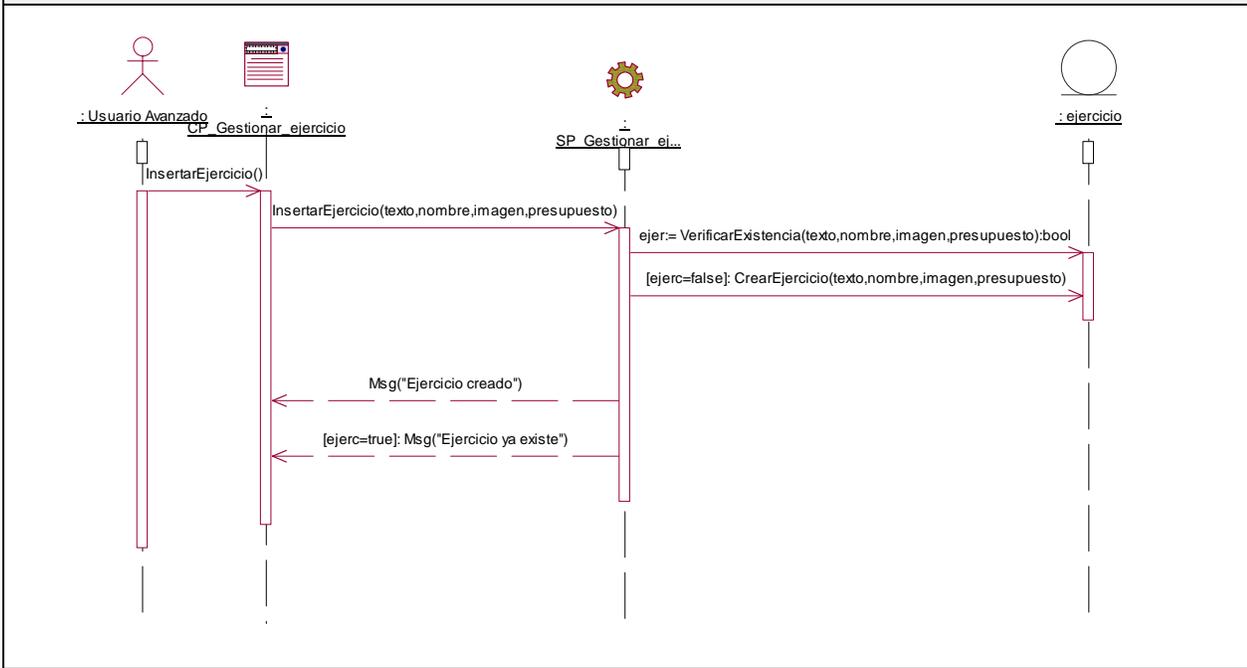


Figura 49 Diagrama de secuencia. CU Gestionar ejercicio. Modificar ejercicio.

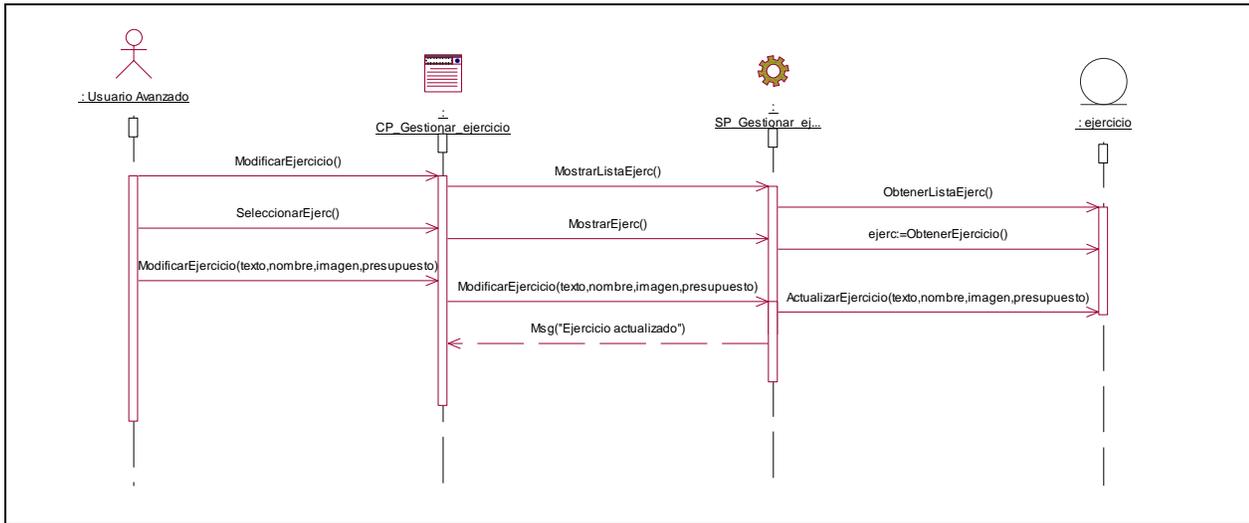


Figura 50 Diagrama de secuencia. CU Gestionar ejercicio. Eliminar ejercicio.

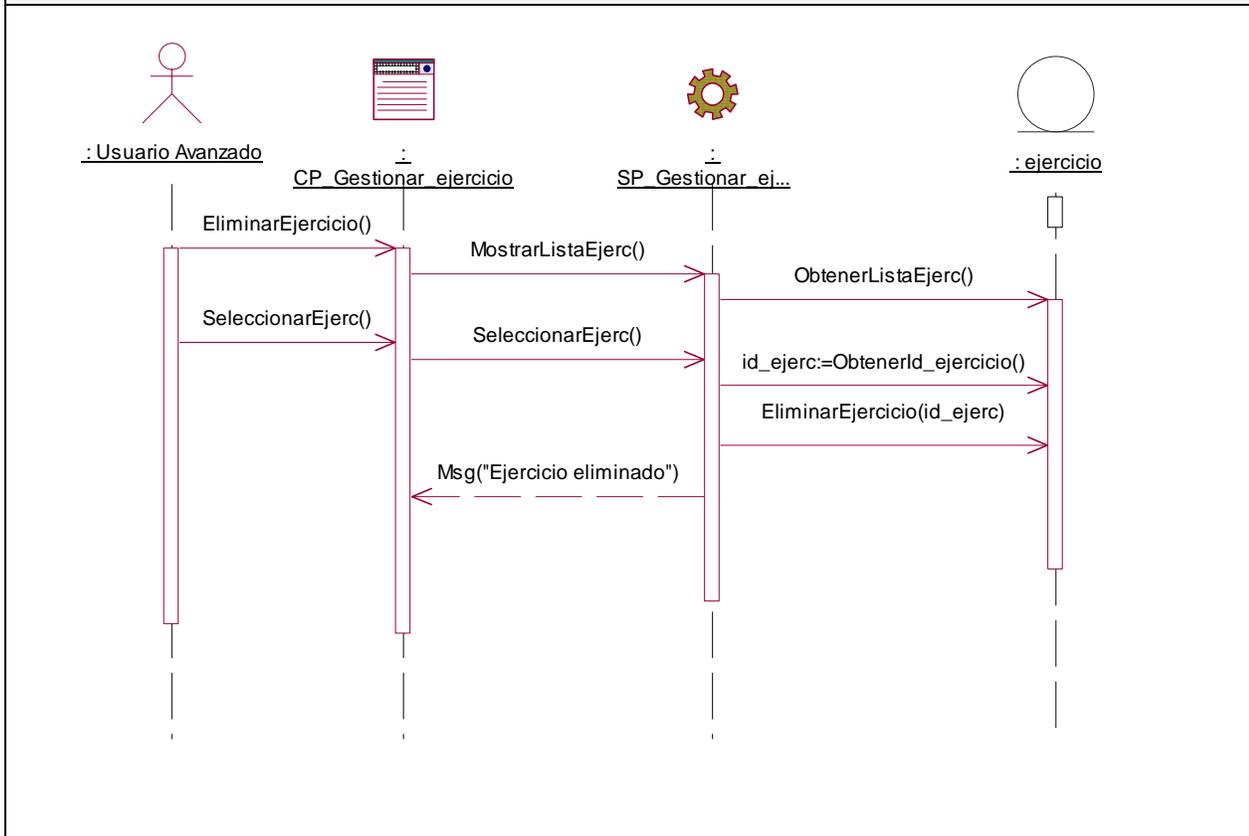


Figura 51 Diagrama de secuencia. CU Responder ejercicio.

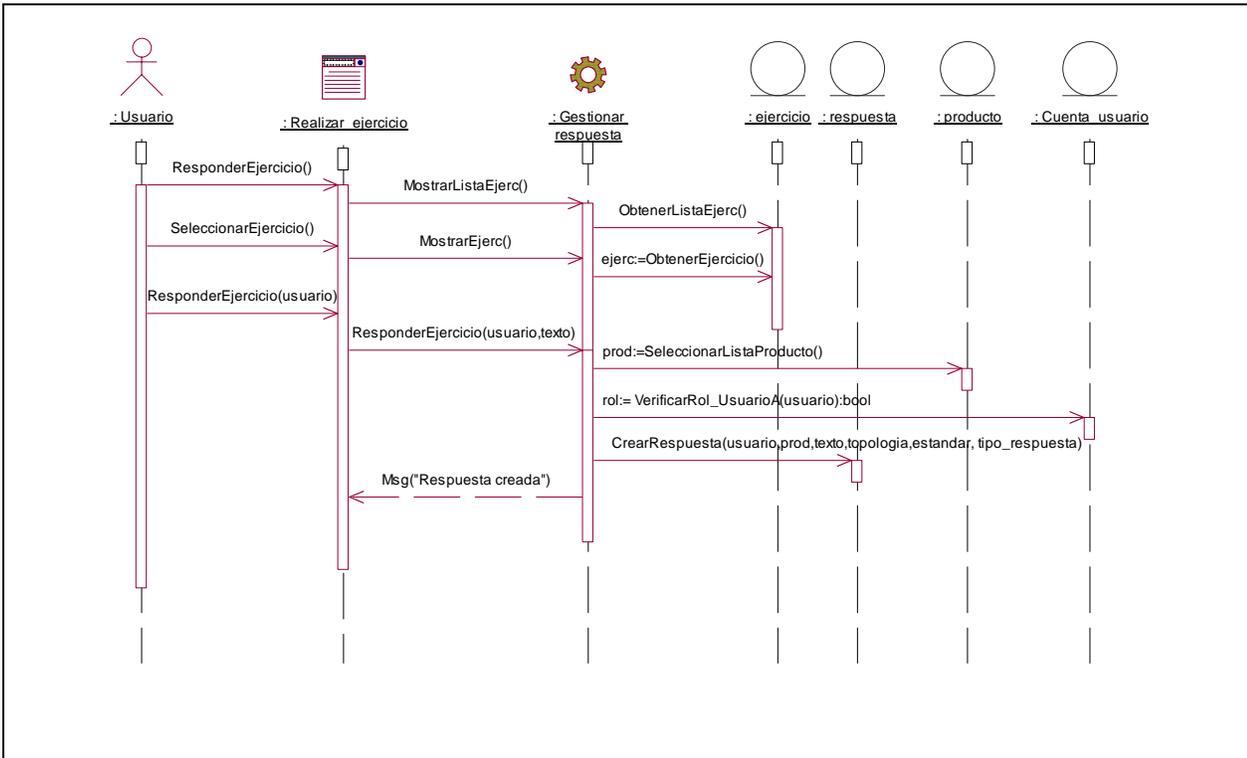


Figura 52 Diagrama de secuencia. CU Gestionar topología. Insertar topología.

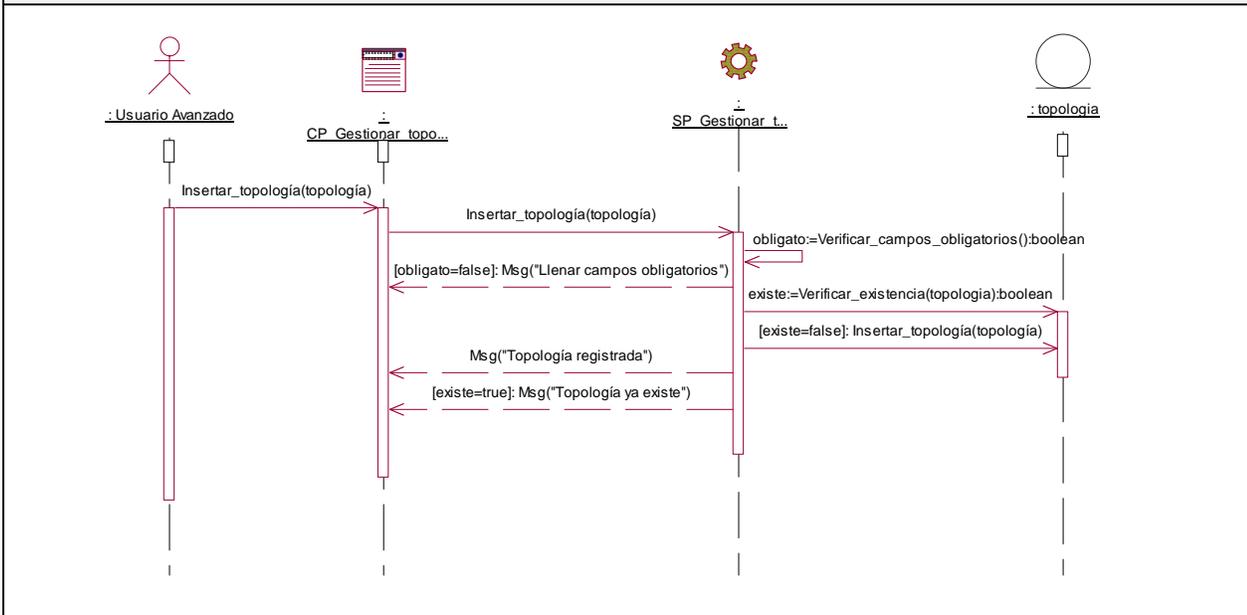


Figura 53 Diagrama de secuencia. CU Gestionar topología. Eliminar topología.

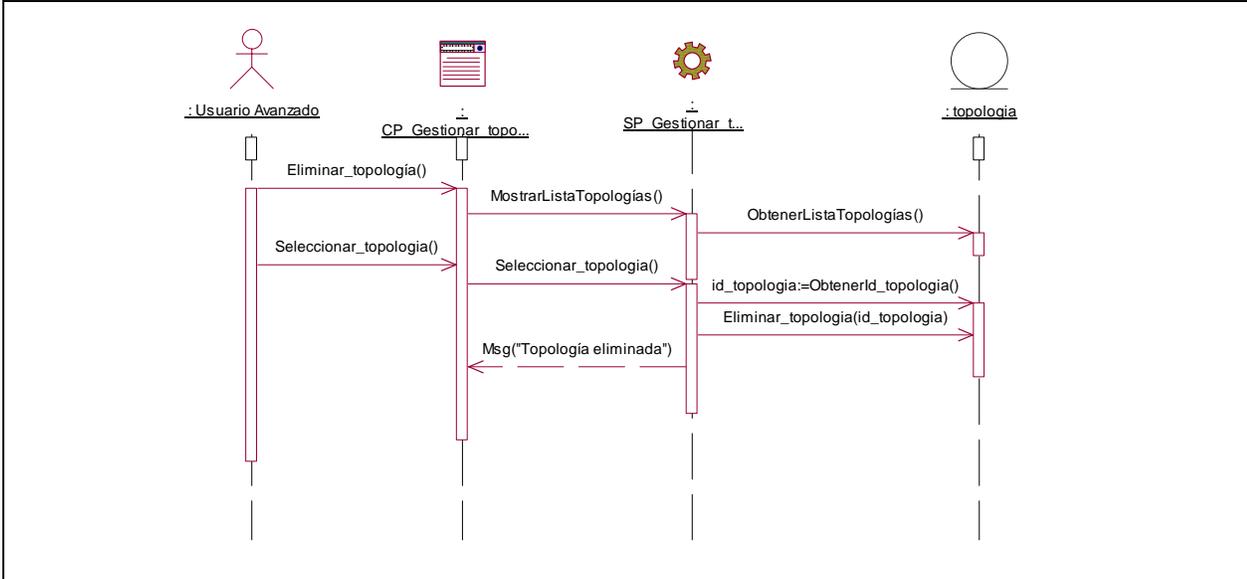


Figura 54 Diagrama de secuencia. CU Gestionar estándar. Insertar estándar.

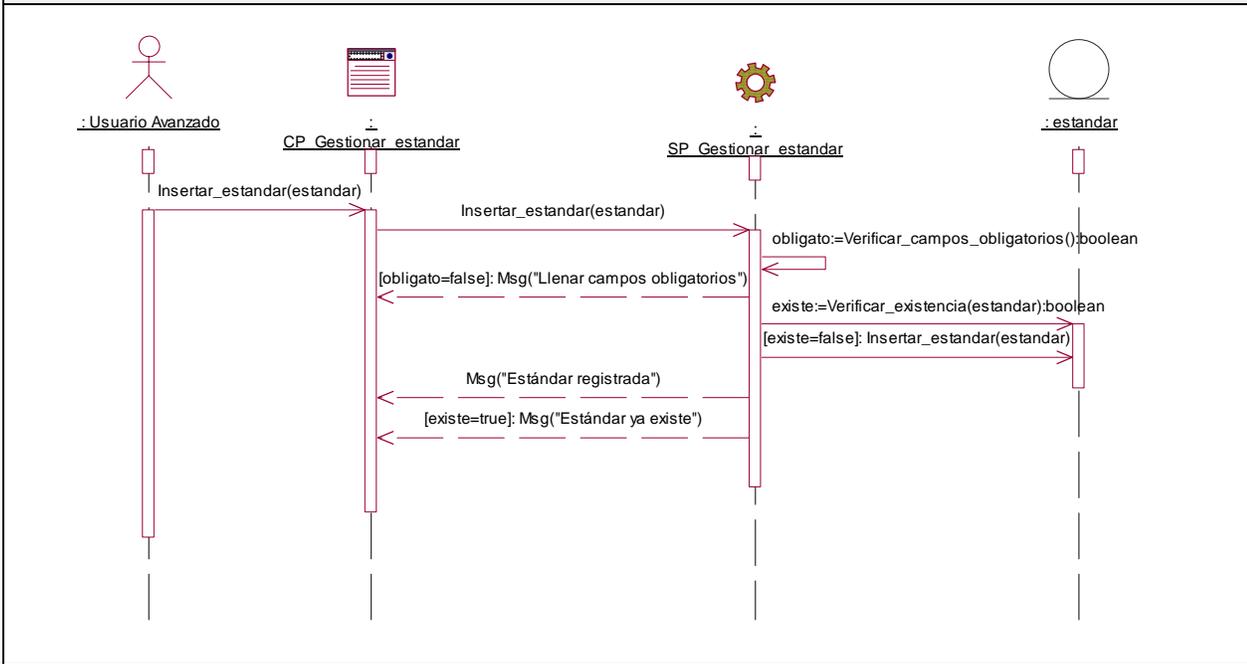


Figura 55 Diagrama de secuencia. CU Gestionar estándar. Eliminar estándar.

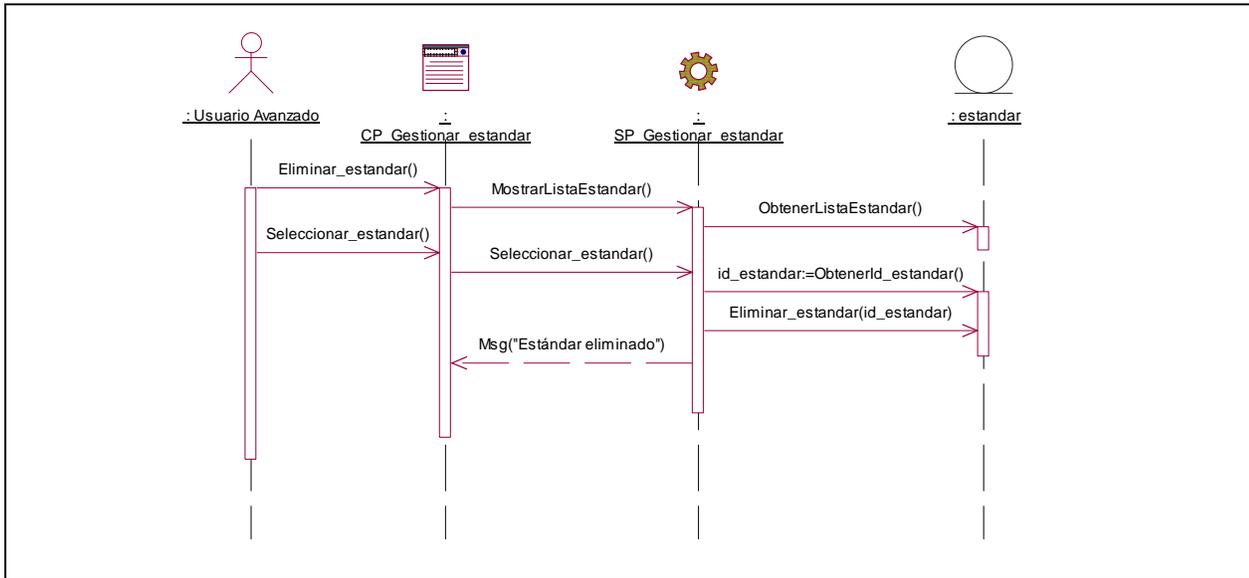


Figura 56 Diagrama de secuencia. CU Gestionar reporte ejercicio.

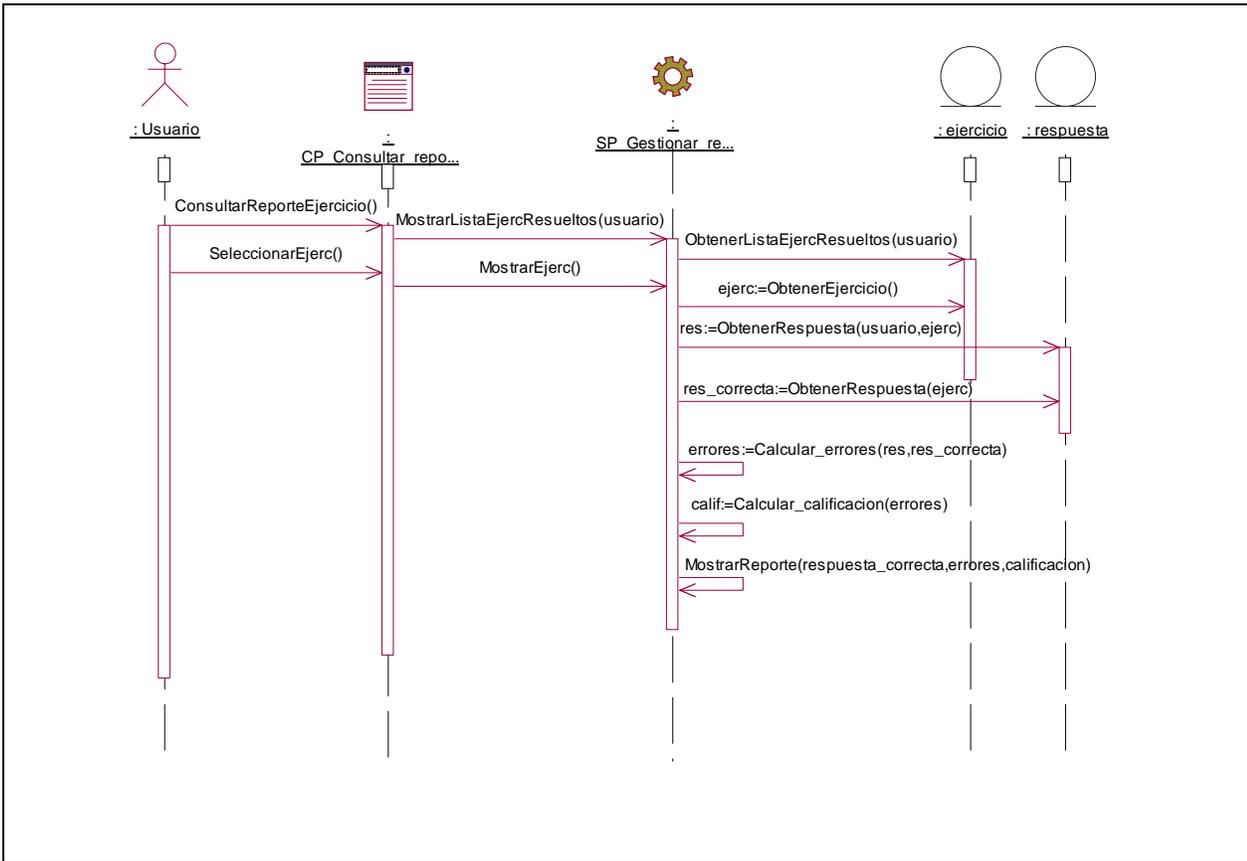


Figura 57 Diagrama de secuencia. CU Consultar estadísticas.

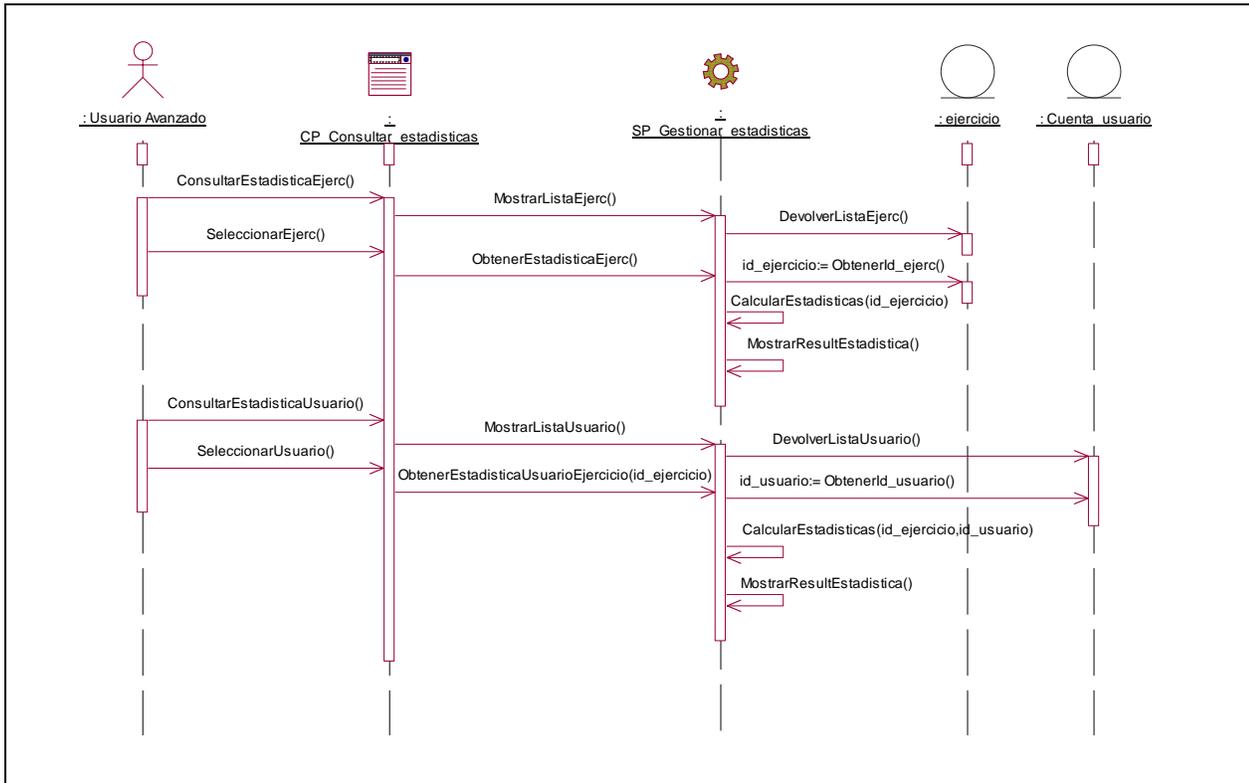


Figura 58 Diagrama de secuencia. CU Operar historial usuario.

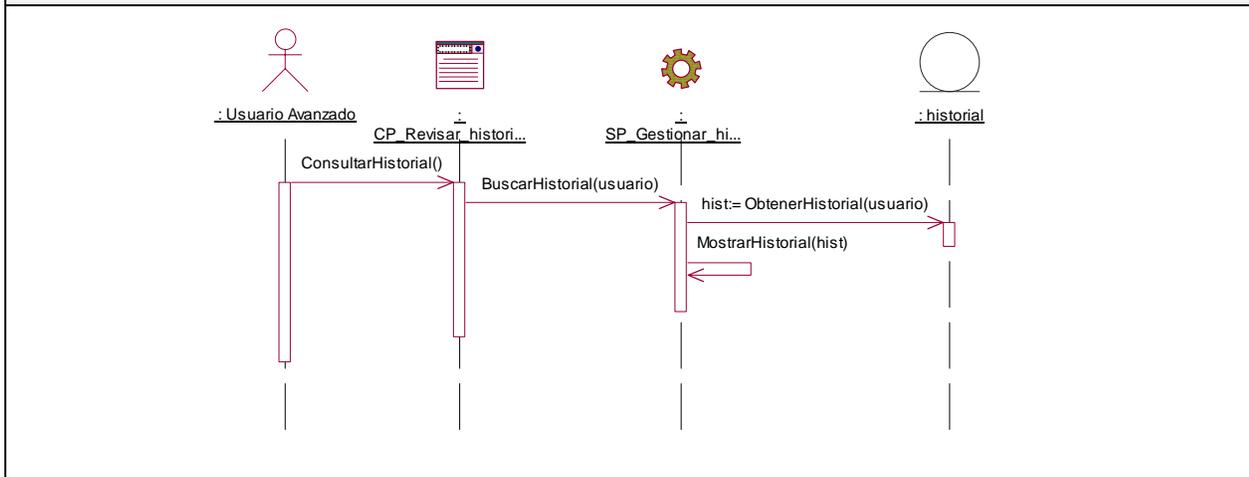


Figura 59 Diagrama de secuencia. CU Publicar documentación. Insertar publicación.

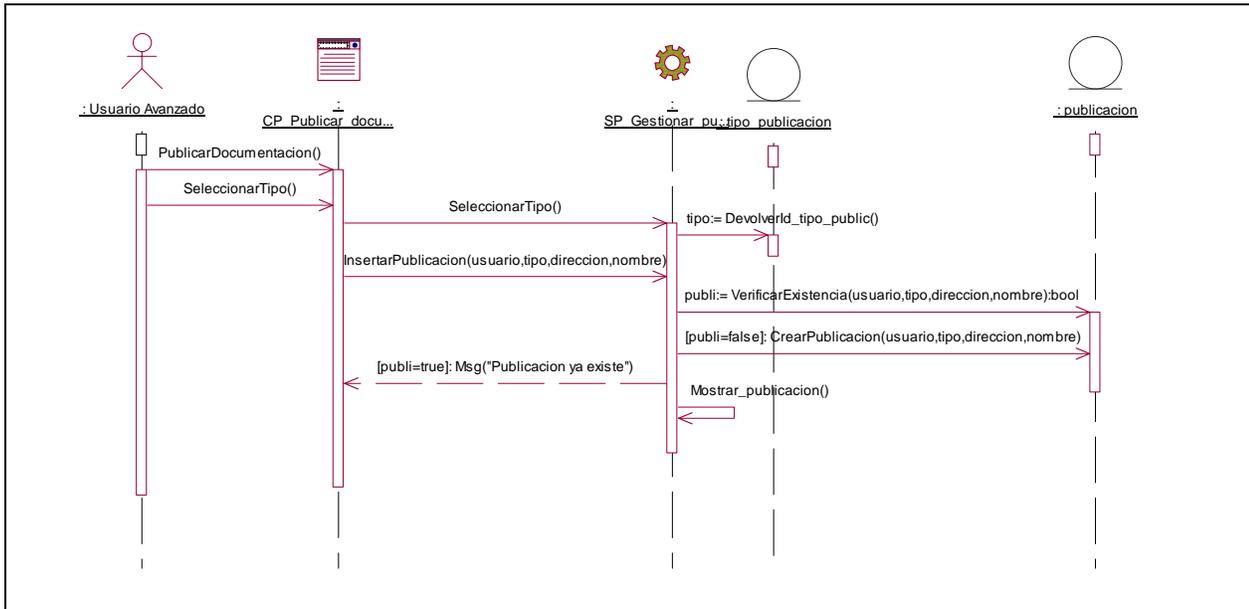


Figura 60 Diagrama de secuencia. CU Publicar documentación. Eliminar publicación.

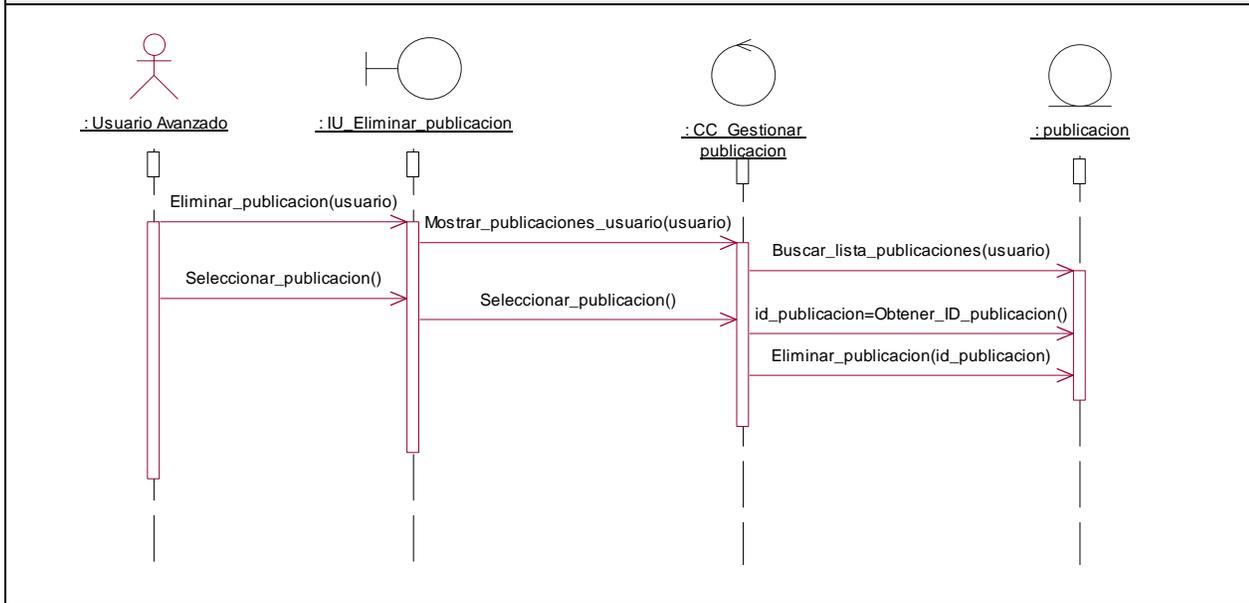
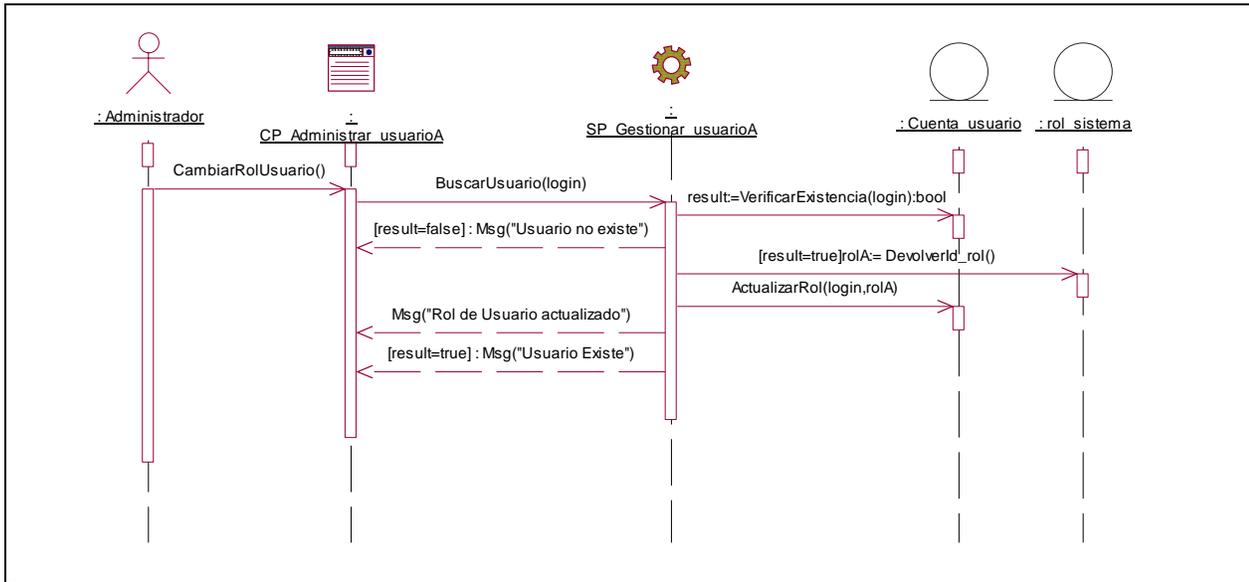


Figura 61 Diagrama de secuencia. CU Administrar usuarios.



Anexo 4: Diagramas de clases del diseño restantes.

Figura 62 Diagrama de clases del diseño. CU Gestionar proveedor.

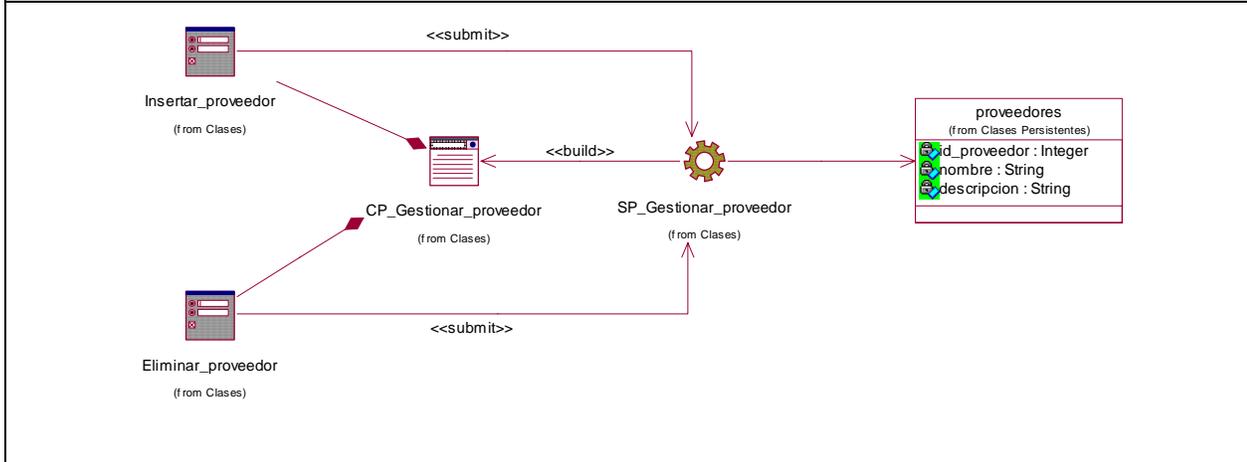


Figura 63 Diagrama de clases del diseño. CU Gestionar categoría.

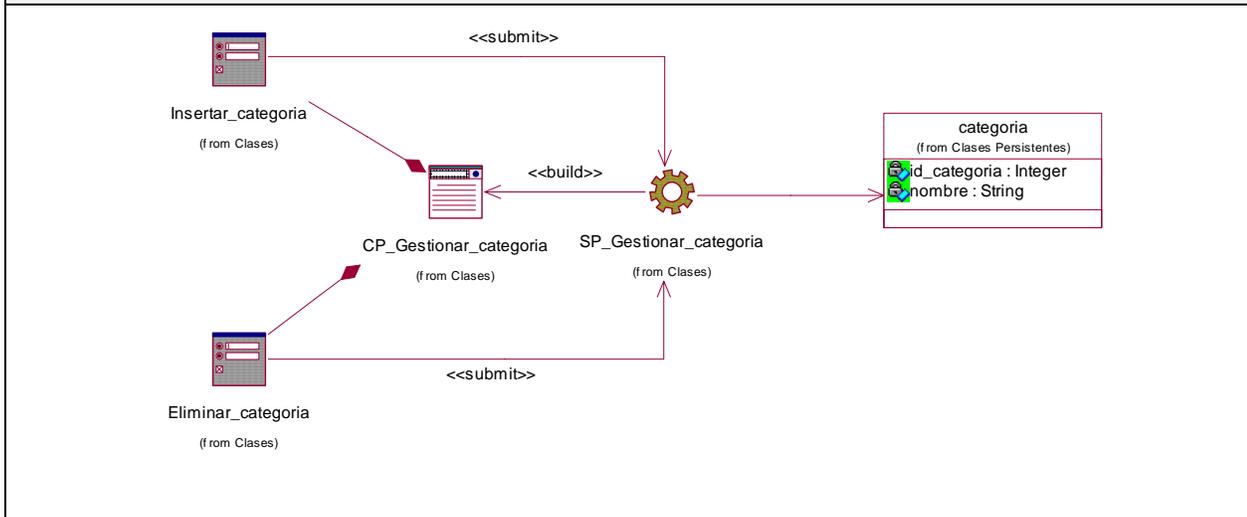


Figura 64 Diagrama de clases del diseño. CU Gestionar unidad de medida.

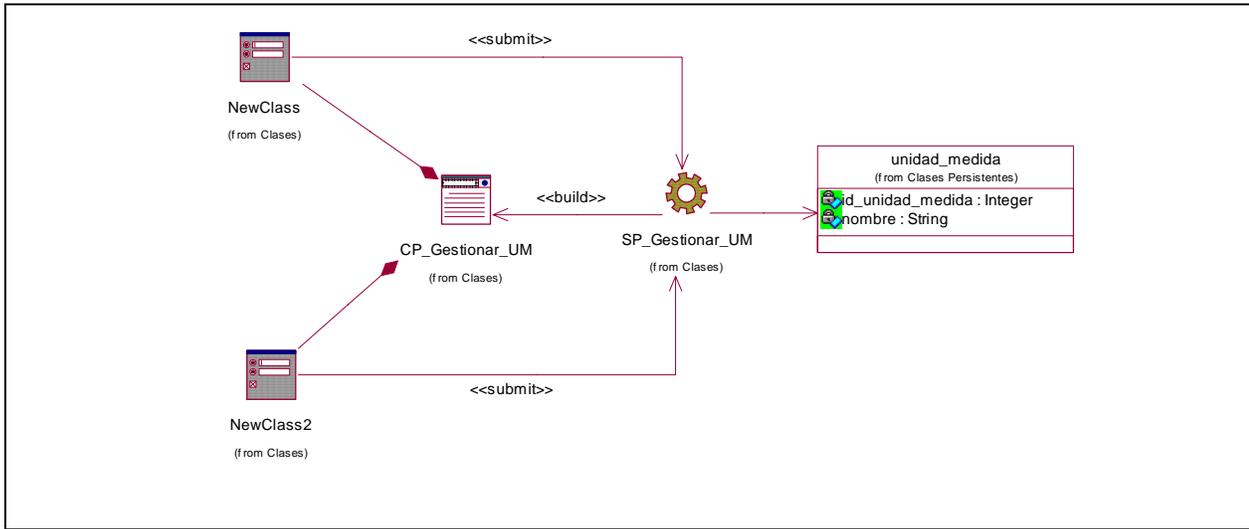


Figura 65 Diagrama de clases del diseño. CU Establecer conexión.

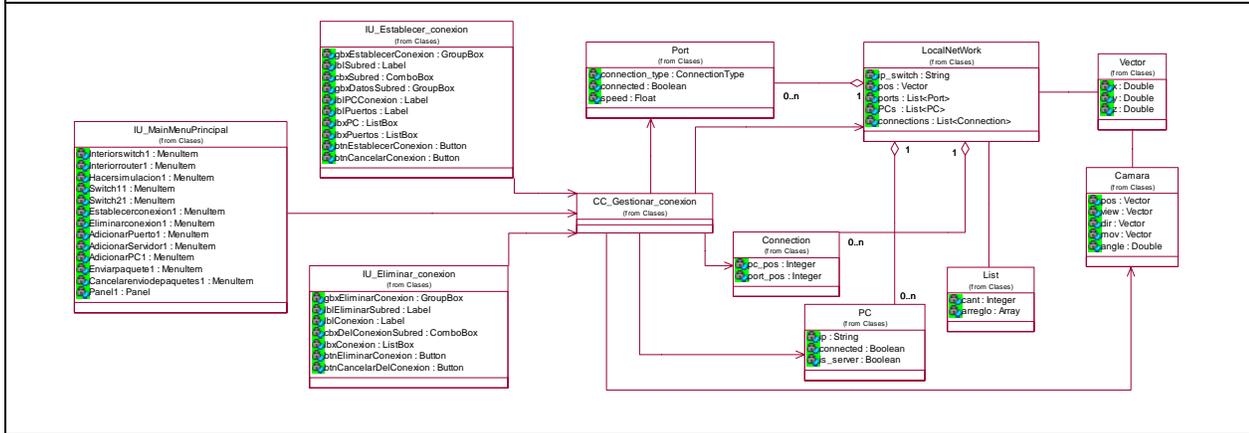


Figura 66 Diagrama de clases del diseño. CU Simular interior dispositivos.

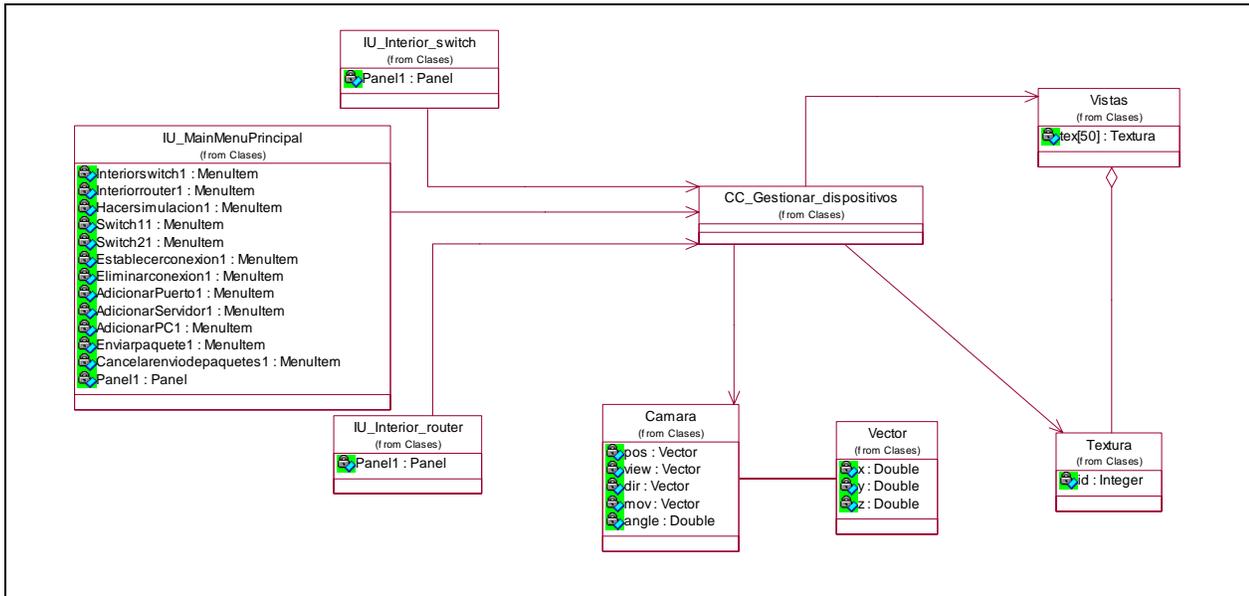


Figura 67 Diagrama de clases del diseño. CU Gestionar topología.

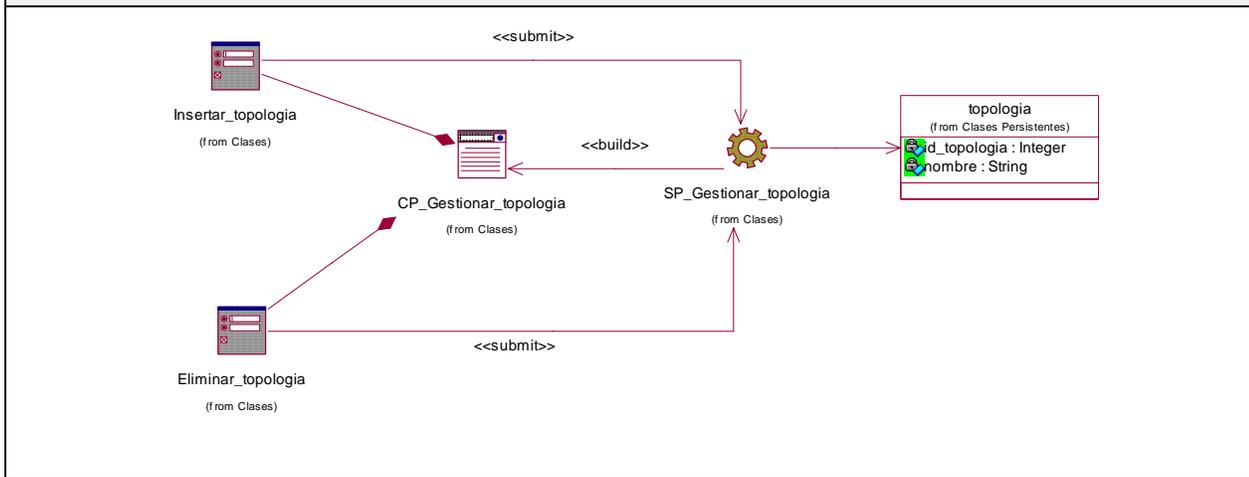


Figura 68 Diagrama de clases del diseño. CU Gestionar estándar.

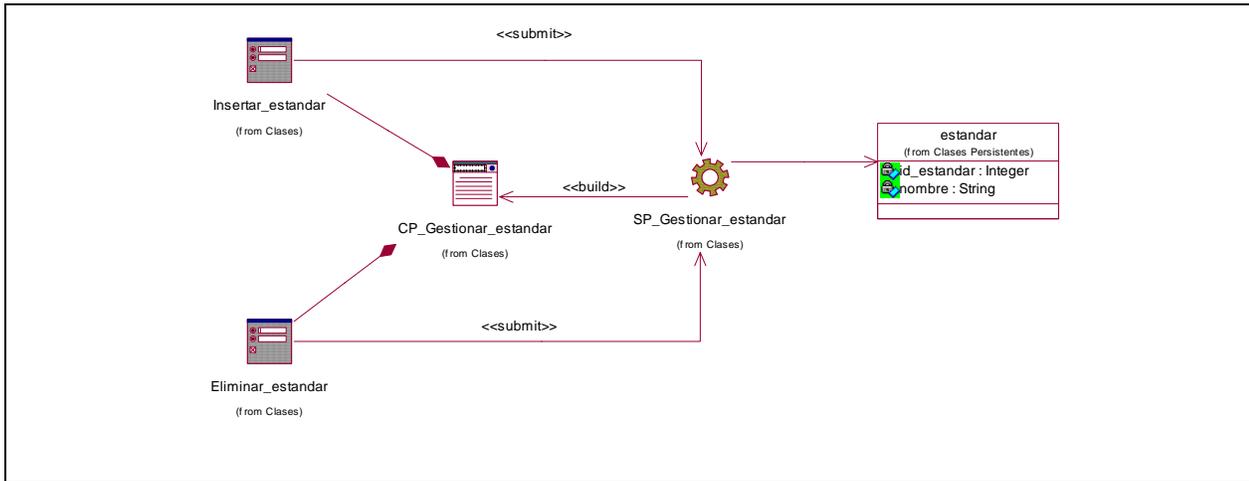


Figura 69 Diagrama de clases del diseño. CU Gestionar reporte ejercicio.

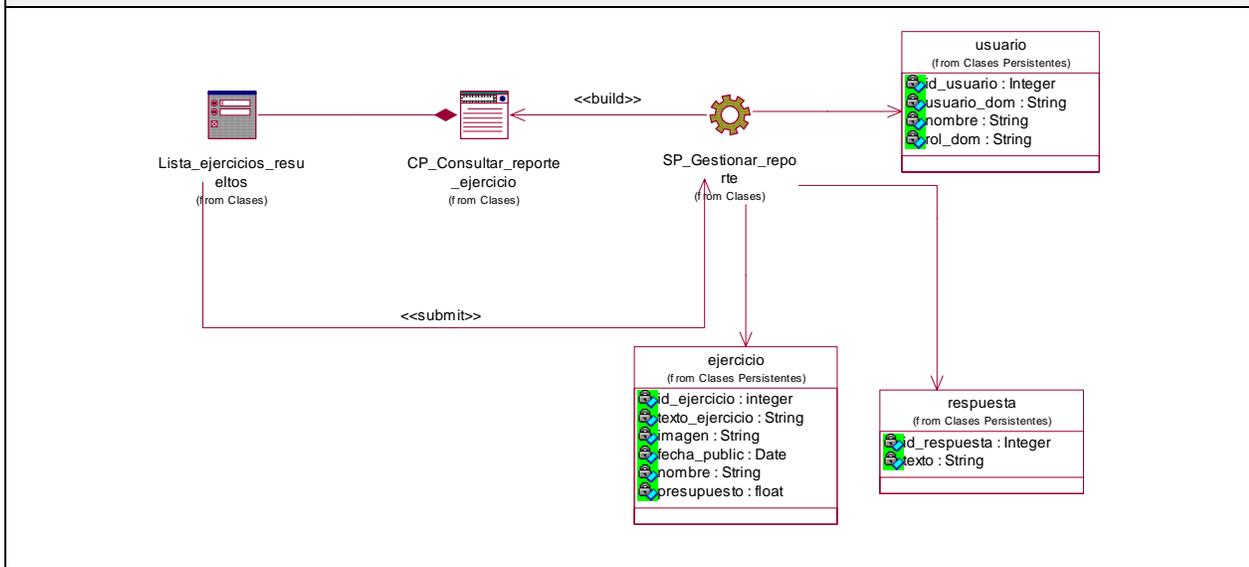


Figura 70 Diagrama de clases del diseño. CU Consultar estadísticas.

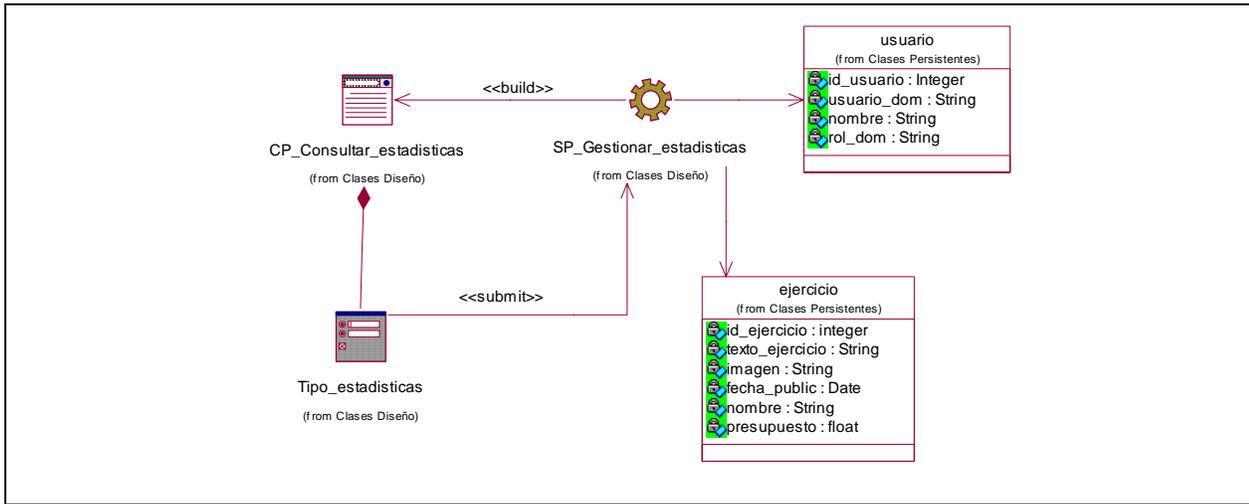


Figura 71 Diagrama de clases del diseño. CU Operar historial usuario.

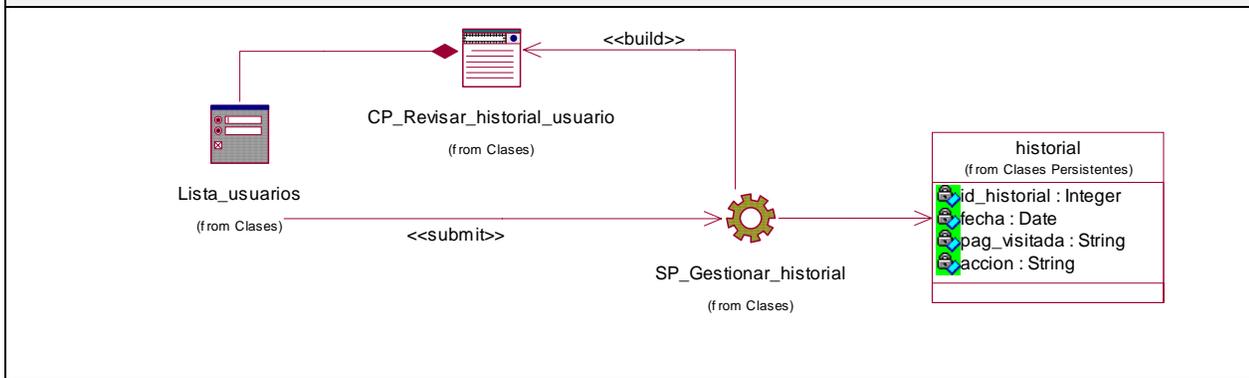


Figura 72 Diagrama de clases del diseño. CU Publicar documentación.

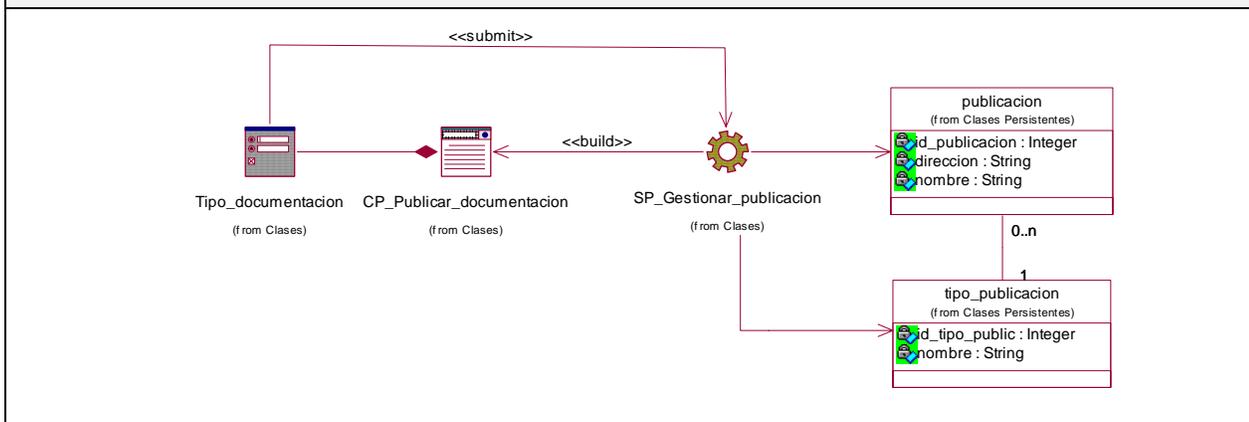
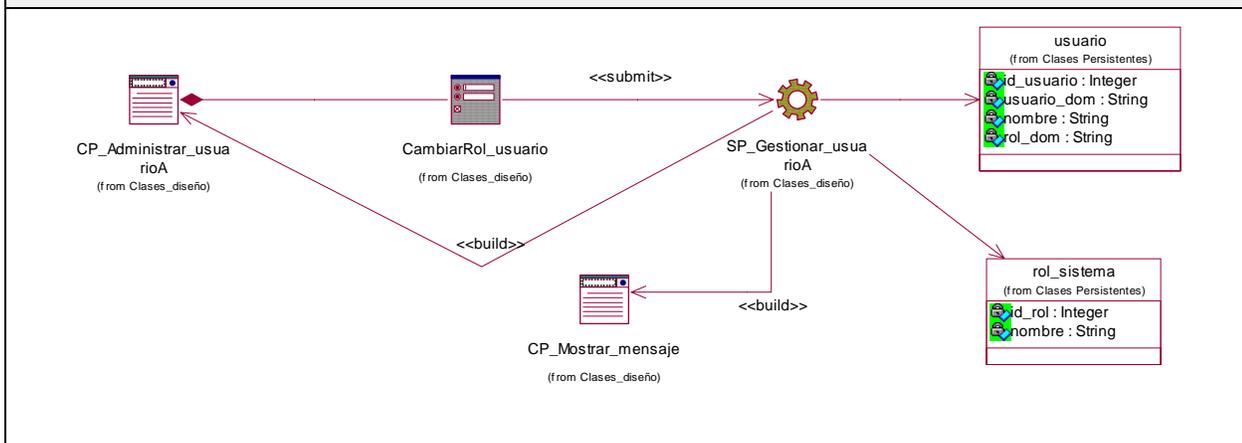


Figura 73 Diagrama de clases del diseño. CU Administrar usuarios.



Anexo 5: Descripción detallada de las clases, sus atributos y operaciones.

MÓDULO EJERCICIOS

Nombre: usuario	
Tipo de clase: Entidad	
Atributo	Tipo
id_usuario	Integer
usuario_dom	String
nombre	String
rol_dom	String
rol_sistema	Integer
Para cada responsabilidad:	
Nombre:	Insertar_Usuario_Nuevo(nombre,rol_dom,usuario,rol_sistema)
Descripción:	Insertar un usuario del sistema.
Nombre:	Buscar_rol_dom(rol)
Descripción:	Devolver el rol de dominio.
Nombre:	Buscar_idusuario(usuario)
Descripción:	Devolver el identificador de un usuario.
Nombre:	Eliminar_usuario(usuario)
Descripción:	Eliminar usuario del sistema.
Nombre:	Actualizar_Rol_Usuario(id_usuario,new_id_rol)
Descripción:	Cambiar el rol del sistema del usuario.
Nombre:	Devolver_rol_dom(usuario)
Descripción:	Devolver el rol de dominio de un usuario.
Nombre:	Obtener_Profesor(user,id_rol)
Descripción:	Devolver los datos de un usuario avanzado.
Nombre:	Buscar_usuario_id(\$id)
Descripción:	Devolver el nombre de un usuario.

Nombre: SP_Gestionar_usuarioA
Tipo de clase: Controladora

Para cada responsabilidad:	
Nombre:	Modificar_Rol_Usuario(id_usuario,new_id_rol)
Descripción:	Realiza las operaciones necesarias para cambiar el rol del sistema del usuario.
Nombre:	Buscar_Usuario(usuario)
Descripción:	Realiza las operaciones necesarias para devolver los datos de un usuario.
Nombre:	Buscar_Id_rol_sistema(rol)
Descripción:	Realiza las operaciones necesarias para devolver el identificador del rol de sistema.
Nombre:	Mostrar_Mensaje(tipo_mens)
Descripción:	Muestra un mensaje del resultado de las acciones realizadas.

Nombre: rol_sistema	
Tipo de clase: Entidad	
Atributo	Tipo
id_rol	Integer
nombre	String
Para cada responsabilidad:	
Nombre:	Devolver_id_rol(rol)
Descripción:	Devolver el identificador de un rol.
Nombre:	Insertar_rol_sistema(rol)
Descripción:	Insertar un nuevo rol de sistema.
Nombre:	Eliminar_rol_sistema(rol)
Descripción:	Eliminar un rol de sistema.

Nombre: SP_Gestionar_Autenticacion	
Tipo de clase: Controladora	
Atributo	Tipo
uci_domain	uci_ldap
Para cada responsabilidad:	
Nombre:	AutenticarUsuario(usuario).

Descripción:	Realiza las operaciones necesarias para autenticar el usuario en el sistema, utilizando el Web Services del Directorio Activo.
Nombre:	Mostrar_Mensaje(tipo_mens)
Descripción:	Muestra un mensaje del resultado de las acciones realizadas.

Nombre: SP_WS_Autenticacion	
Tipo de clase: Controladora	
Atributo	Tipo
name	String
mail	String
nickname	String
Para cada responsabilidad:	
Nombre:	Login(domain,user,password)
Descripción:	Realiza la autenticación del usuario.
Nombre:	search_user(domain,user,password,usrer_2)
Descripción:	Busca el usuario en el dominio UCI, devuelve si lo encontró o no.

Nombre: ejercicio	
Tipo de clase: Entidad	
Atributo	Tipo
id_ejercicio	Integer
texto_ejercicio	String
imagen	String
fecha_public	Date
nombre	String
presupuesto	Float
usuario	Integer
rol_sistema	Integer
Para cada responsabilidad:	
Nombre:	Insertar_Ejercicio(nomb,text,fecha,presu,imagen,user)

Descripción:	Insertar un nuevo ejercicio.
Nombre:	Eliminar_Ejercicio(id)
Descripción:	Eliminar un ejercicio.
Nombre:	Mostrar_Ejercicios()
Descripción:	Devolver una lista de los ejercicios.
Nombre:	Modificar_Ejercicio(nombre,text,fecha,presu,imagen,user)
Descripción:	Modificar los datos de un ejercicio.
Nombre:	Buscar_ejercicio(nomb)
Descripción:	Devolver los datos de un ejercicio.
Nombre:	Obtener_presupuesto(ejercicio)
Descripción:	Devolver el presupuesto de un ejercicio.

Nombre: SP_Gestionar_ejercicio	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Insertar_Ejerc(nomb,text,fecha,presu,imagen,user,rol)
Descripción:	Realiza las operaciones necesarias para insertar un nuevo ejercicio.
Nombre:	Eliminar_Ejerc(id)
Descripción:	Realiza las operaciones necesarias para eliminar un ejercicio.
Nombre:	Modificar_Ejerc(nombre,text,fecha,presu,imagen,user,rol)
Descripción:	Realiza las operaciones necesarias para modificar los datos de un ejercicio.
Nombre:	Mostrar_Ejercicio()
Descripción:	Realiza las operaciones necesarias para mostrar un ejercicio.
Nombre:	Mostrar_Mensaje(tipo_mens)
Descripción:	Muestra un mensaje del resultado de las acciones realizadas.

Nombre: SP_Gestionar_estadisticas	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Calcular_estadistica(id_ejercicio,id_usuario)

Descripción:	Realiza las operaciones necesarias para calcular las estadísticas por ejercicio y usuario.
Nombre:	Calcular_estad(id_ejercicio)
Descripción:	Realiza las operaciones necesarias para calcular las estadísticas por ejercicio.
Nombre:	Seleccionar_tipoE()
Descripción:	Realiza las operaciones necesarias para guardar el tipo de estadística.
Nombre:	Mostrar_Mensaje(tipo_mens)
Descripción:	Muestra un mensaje del resultado de las acciones realizadas.

Nombre: producto	
Tipo de clase: Entidad	
Atributo	Tipo
id_producto	Integer
codigo_producto	String
descripcion	String
nombre	String
precio	Float
proveedor	Integer
unidad_medida	Integer
categoria	Integer
especificacion	String
Para cada responsabilidad:	
Nombre:	Insertar_Producto(codigo,nomb,descrip,precio,prov,um,cat)
Descripción:	Insertar un nuevo producto.
Nombre:	Eliminar_Producto(cod)
Descripción:	Eliminar un producto.
Nombre:	Mostrar_Producto_categoria(catg)
Descripción:	Devuelve una lista de productos, dada la categoría.
Nombre:	Mostrar_Producto_prov(prov)
Descripción:	Devuelve una lista de productos según el proveedor.

Nombre:	Modificar_Producto(codigo,nomb,descrip,precio,prov,um,cat)
Descripción:	Actualizar los datos de productos.
Nombre:	Buscar_especificación()
Descripción:	Devolver la especificación de un producto.

Nombre: proveedores	
Tipo de clase: Entidad	
Atributo	Tipo
id_proveedor	Integer
nombre	String
descripcion	String
Para cada responsabilidad:	
Nombre:	Insertar_proveedor(nomb,des)
Descripción:	Insertar un nuevo proveedor.
Nombre:	Eliminar_proveedor(prov)
Descripción:	Eliminar un proveedor.
Nombre:	Buscar_proveedor(nomb)
Descripción:	Devuelve el identificador de un proveedor.
Nombre:	Mostrar_proveedor()
Descripción:	Devuelve una lista de los proveedores.

Nombre: SP_Gestionar_proveedor	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Insertar_proveedor(nomb,des)
Descripción:	Realiza las operaciones necesarias para insertar un nuevo proveedor.
Nombre:	Eliminar_proveedor(id_producto)
Descripción:	Realiza las operaciones necesarias para eliminar un proveedor.

Nombre: categoria

Tipo de clase: Entidad	
Atributo	Tipo
id_categoria	Integer
nombre	String
Para cada responsabilidad:	
Nombre:	Insertar_categoria(nom_catg)
Descripción:	Insertar una nueva categoría.
Nombre:	Eliminar_categoria(catg)
Descripción:	Eliminar una categoría.
Nombre:	Buscar_categoria(nomb)
Descripción:	Devuelve el identificador de una categoría.
Nombre:	Mostrar_categoria()
Descripción:	Devuelve una lista de categorías.

Nombre: SP_Gestionar_categoria	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Insertar_categoria(nomb,des)
Descripción:	Realiza las operaciones necesarias para insertar una nueva categoría.
Nombre:	Eliminar_categoria(id_producto)
Descripción:	Realiza las operaciones necesarias para eliminar una categoría.

Nombre: unidad_medida	
Tipo de clase: Entidad	
Atributo	Tipo
id_unidad_medida	Integer
nombre	String
Para cada responsabilidad:	
Nombre:	Insertar_unidad_medida(nom_unid)
Descripción:	Insertar una nueva unidad de medida.

Nombre:	Eliminar_unidad_medida(unid)
Descripción:	Eliminar una unidad de medida.
Nombre:	Buscar_unidad_medida(nomb)
Descripción:	Devuelve el identificador de una unidad de medida.

Nombre: SP_Gestionar_unidad_medida	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Insertar_UM(nomb,des)
Descripción:	Realiza las operaciones necesarias para insertar una nueva unidad de medida.
Nombre:	Eliminar_UM(id_producto)
Descripción:	Realiza las operaciones necesarias para eliminar una unidad de medida.

Nombre: SP_Gestionar_producto	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Agregar_producto(codigo,nomb,descrip,precio,prov,um,cat)
Descripción:	Realiza las operaciones necesarias para insertar un nuevo producto.
Nombre:	Modificar_producto(codigo,nomb,descrip,precio,prov,um,cat)
Descripción:	Realiza las operaciones necesarias para modificar un producto.
Nombre:	Eliminar_producto(id_producto)
Descripción:	Realiza las operaciones necesarias para eliminar un producto.
Nombre:	Mostrar_Mensaje(tipo_mens)
Descripción:	Muestra un mensaje del resultado de las acciones realizadas.

Nombre: SP_Gestionar_reporte	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Calcular_errores(id_ejercicio, usuario)
Descripción:	Realiza las operaciones necesarias para calcular la cantidad de errores.

Nombre:	Mostrar_descrip_errores()
Descripción:	Realiza las operaciones necesarias para mostrar una descripción de los errores cometidos.

Nombre: historial	
Tipo de clase: Entidad	
Atributo	Tipo
id_historial	Integer
fecha	Date
pag_visitada	String
accion	String
usuario	Integer
Para cada responsabilidad:	
Nombre:	Insertar_historial_usuario(fecha, pag, acción, id_usuario)
Descripción:	Insertar un nuevo historial para un usuario.
Nombre:	Buscar_historial_usuario_user(user)
Descripción:	Devolver los datos del historial de un usuario.

Nombre: SP_Gestionar_historial	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Mostrar_historial(usuario)
Descripción:	Realiza las operaciones necesarias para mostrar el historial de un usuario.
Nombre:	Guardar_historial(fecha, pag, acción, id_usuario)
Descripción:	Realiza las operaciones necesarias para guardar las acciones de un usuario en su historial.

Nombre: publicacion	
Tipo de clase: Entidad	
Atributo	Tipo

id_publicacion	Integer
direccion	String
nombre	String
tipo_public	Integer
Para cada responsabilidad:	
Nombre:	Insertar_publicacion(nombre,direccion,id_tipo_public)
Descripción:	Insertar una nueva publicación.
Nombre:	Mostrar_publicacion(id_publicacion)
Descripción:	Mostrar el contenido de la publicación.
Nombre:	Buscar_Id_publicacion(nombre)
Descripción:	Devolver el identificador de una publicación.

Nombre: tipo_publicacion	
Tipo de clase: Entidad	
Atributo	Tipo
id_tipo_public	Integer
nombre	String
Para cada responsabilidad:	
Nombre:	Insertar_tipo_public(nomb)
Descripción:	Insertar un nuevo tipo de publicación.
Nombre:	Eliminar_tipo_public(nomb)
Descripción:	Eliminar un tipo de publicación.
Nombre:	Buscar_tipo_public(nomb)
Descripción:	Devuelve el identificador de un tipo de publicación.
Nombre:	Mostrar_tipo_public()
Descripción:	Devuelve una lista de los tipos de publicación.

Nombre: SP_Gestionar_publicacion	
Tipo de clase: Controladora	
Para cada responsabilidad:	

Nombre:	Guardar_doc(nombre,direccion,id_tipo_public)
Descripción:	Realiza las operaciones necesarias para insertar una nueva publicación.

Nombre: respuesta	
Tipo de clase: Entidad	
Atributo	Tipo
id_respuesta	Integer
texto	String
usuario	Integer
ejercicio	Integer
estandar	Integer
topologia	Integer
lista_productos	producto
Para cada responsabilidad:	
Nombre:	Insertar_respuesta(id_ejercicio,id_usuario,id_topologia,id_estandar,texto,prod)
Descripción:	Insertar una nueva respuesta.
Nombre:	Eliminar_respuesta(id_respuesta)
Descripción:	Eliminar una respuesta.
Nombre:	Mostrar_lista_productos(id_respuesta)
Descripción:	Devuelve una lista de productos por respuesta.
Nombre:	Mostrar_respuesta_usuario(user,id_respuesta)
Descripción:	Mostrar los datos de una respuesta a un usuario.

Nombre: SP_Gestionar_respuesta	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Guardar_respuesta(id_ejercicio,id_usuario,id_topologia,id_estandar,texto,prod)
Descripción:	Realiza las operaciones necesarias para guardar una respuesta.
Nombre:	Mostrar_respuesta(id_respuesta)
Descripción:	Realiza las operaciones necesarias para mostrar una respuesta.

Nombre:	Buscar_tipo_respuesta(tipo)
Descripción:	Devuelve el identificador del tipo de respuesta.

Nombre: topología	
Tipo de clase: Entidad	
Atributo	Tipo
id_topologia	Integer
nombre	String
Para cada responsabilidad:	
Nombre:	Insertar_topologia(nom_topologia)
Descripción:	Insertar una nueva topología.
Nombre:	Eliminar_topologia(topologia)
Descripción:	Eliminar una topología.
Nombre:	Buscar_topologia(nomb)
Descripción:	Devuelve el identificador de una topología.
Nombre:	Mostrar_topologia()
Descripción:	Devuelve una lista de topologías.

Nombre: SP_Gestionar_topologia	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Insertar_topologia(nomb,des)
Descripción:	Realiza las operaciones necesarias para insertar una nueva topología.
Nombre:	Eliminar_topologia(id_topologia)
Descripción:	Realiza las operaciones necesarias para eliminar una topología.

Nombre: estándar	
Tipo de clase: Entidad	
Atributo	Tipo
id_estandar	Integer

nombre	String
Para cada responsabilidad:	
Nombre:	Insertar_estandar(nom_estandar)
Descripción:	Insertar un nuevo estándar.
Nombre:	Eliminar_estandar(estandar)
Descripción:	Eliminar un estándar.
Nombre:	Buscar_estandar(nomb)
Descripción:	Devuelve el identificador de un estándar.
Nombre:	Mostrar_estandar ()
Descripción:	Devuelve una lista de estándares.

Nombre: SP_Gestionar_estandar	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Insertar_estandar(nomb,des)
Descripción:	Realiza las operaciones necesarias para insertar un nuevo estándar.
Nombre:	Eliminar_estandar(id_estandar)
Descripción:	Realiza las operaciones necesarias para eliminar un estándar.

MÓDULO DE SIMULACIÓN

Nombre: IU_MainMenuPrincipal	
Tipo de clase: Interfaz	
Atributo	Tipo
Interiorswitch1	MenuItem
Interiorrouter1	MenuItem
Hacersimulacion1	MenuItem
Switch11	MenuItem
Establecerconexion1	MenuItem
Eliminarconexion1	MenuItem
AdicionarPuerto1	MenuItem

AdicionarServidor1	Menuitem
AdicionarPC1	Menuitem
Enviarpaquete1	Menuitem
Cancelarenviodepaquetes1	Menuitem
Panel1	Panel
Para cada responsabilidad:	
Nombre:	Interiorswitch1Click()
Descripción:	Mostrar la interfaz IU_Interior_switch para simular el interior de un conmutador.
Nombre:	Interiorrouter1Click()
Descripción:	Mostrar la interfaz IU_Interior_router para simular el interior de un enrutador .
Nombre:	Hacersimulacion1Click()
Descripción:	Habilita las opciones para simular el funcionamiento de una red.
Nombre:	AdicionarPuerto1Click()
Descripción:	Mostrar la interfaz IU_Adicionar_puerto para dibujar un nuevo puerto en una subred.
Nombre:	AdicionarServidor1Click()
Descripción:	Mostrar la interfaz IU_Adicionar_PC_servidor para dibujar un nuevo servidor en una subred.
Nombre:	AdicionarPC1Click()
Descripción:	Mostrar la interfaz IU_Adicionar_PC_servidor para dibujar una nueva PC en una subred.
Nombre:	Establecerconexion1Click()
Descripción:	Mostrar la interfaz IU_Establecer_conexion para dibujar una nueva conexión en una subred.
Nombre:	Eliminarconexion1Click()
Descripción:	Mostrar la interfaz IU_Eliminar_conexion para eliminar una conexión en una subred.
Nombre:	Enviarpaquete1Click()
Descripción:	Mostrar la interfaz IU_Crear_envio para simular el envío de paquete por la red.
Nombre:	Cancelarenviodepaquetes1Click()

Descripción:	Mostrar la interfaz IU_Cancelar_envio para cancelar un envío de paquetes por la red.
Nombre:	FormCreate()
Descripción:	Inicializar cámaras, dibujar PC, puertos y conexiones por defecto y organiza las posiciones de las PC.
Nombre:	ReSize(Width, Height)
Descripción:	Establecer el ancho y el largo de la pantalla.
Nombre:	InitOpendgl()
Descripción:	Inicializar una ventana de OpenGL en un Panel. Fijar el tamaño de la estructura. Se quiere la máscara estándar. Se quiere el tipo RGB y Alpha pixel.

Nombre: IU_Agregar_PC_servidor	
Tipo de clase: Interfaz	
Atributo	Tipo
gbxAgregarComputadora	GroupBox
lblIPPC	Label
MaskEditIPPC	MaskEdit
btnAgregarComputadora	Button
btnCancelarAdicPC	Button
Para cada responsabilidad:	
Nombre:	btnAgregarComputadoraClick()
Descripción:	Llamada a los métodos de la controladora para dibujar una PC o un servidor en una subred.
Nombre:	btnCancelarAdicPCClick()
Descripción:	Cancelar la acción de agregar una nueva PC.

Nombre: IU_Agregar_puerto	
Tipo de clase: Interfaz	
Atributo	Tipo
gbxAgregarPuerto	GroupBox

lblTipoConexion	Label
cbxTipoConexion	ComboBox
btnAdicionarPuerto	Button
btnCancelarAdicPuerto	Button
Para cada responsabilidad:	
Nombre:	btnAdicionarPuertoClick()
Descripción:	Llamada a los métodos de la controladora para dibujar un puerto en una subred.
Nombre:	btnCancelarAdicPuertoClick()
Descripción:	Cancelar la acción de adicionar un nuevo puerto.

Nombre: CC_Gestionar_subred	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	AdicionarPC()
Descripción:	Realiza las operaciones necesarias para dibujar una nueva PC.
Nombre:	AdicionarServidor()
Descripción:	Realiza las operaciones necesarias para dibujar un nuevo servidor.
Nombre:	AdicionarPuerto()
Descripción:	Realiza las operaciones necesarias para dibujar un nuevo puerto.
Nombre:	PosicionarCamara()
Descripción:	Realiza las operaciones necesarias para posicionar la cámara.
Nombre:	Pintar()
Descripción:	Borrar la pantalla. Establece la posición y la dirección del foco de luz. Pintar cada conmutador, puerto y conexión entre conmutadores.

Nombre: LocalNetWork	
Tipo de clase: Entidad	
Atributo	Tipo
ip_switch	String
pos	Vector

ports	List<Port>
PCs	List<PC>
connections	List<Connection>
Para cada responsabilidad:	
Nombre:	LocalNetWork(ipSwitch, pos)
Descripción:	Constructor de un objeto LocalNetWork.
Nombre:	SetIP(ip)
Descripción:	Cambiar el ip.
Nombre:	GetIP()
Descripción:	Devuelve el ip.
Nombre:	AddPC(newPC)
Descripción:	Adicionar una nueva PC a la lista de PCs.
Nombre:	AddPort(newPort)
Descripción:	Adicionar un nuevo puerto a la lista de Ports.
Nombre:	AddConnection(ip,portPos)
Descripción:	Adicionar una nueva conexión a la lista de Connections.
Nombre:	DelConnection(pos)
Descripción:	Eliminar una conexión de la lista de conexiones.
Nombre:	NumPortConnected(ip)
Descripción:	Devuelve la posición del puerto al que está conectada la PC con el IP ip.
Nombre:	NumPC(ip)
Descripción:	Devuelve la posición de la PC.
Nombre:	Reorganize(float y, float z, float esp = 2)
Descripción:	Reorganizar posición en el eje x de todas las PCs.

Nombre: PC	
Tipo de clase: Entidad	
Atributo	Tipo
ip	String
connected	Boolean

is_server	Boolean
Para cada responsabilidad:	
Nombre:	Pc(ipp,isServer)
Descripción:	Constructor de la clase PC.
Nombre:	Connect()
Descripción:	Devuelve si la Pc está conectada.
Nombre:	Disconnect()
Descripción:	Devuelve si la Pc está desconectada.
Nombre:	IP()
Descripción:	Devuelve el IP de la PC.
Nombre:	ConnectionStatus()
Descripción:	Devuelve el estado de conectividad.
Nombre:	IsServer()
Descripción:	Devuelve si es un servidor la PC.

Nombre: Camara	
Tipo de clase: Entidad	
Atributo	Tipo
pos	Vector
view	Vector
dir	Vector
mov	Vector
angle	Double
Para cada responsabilidad:	
Nombre:	Camara(pos,view,dir,angle)
Descripción:	Constructor de una Camara.
Nombre:	Show()
Descripción:	Enfoca la cámara.
Nombre:	Mover()
Descripción:	Cambia la posición de la cámara.

Nombre:	Rotar()
Descripción:	Rotar la cámara manteniendo la posición en el eje y.
Nombre:	AlejarAcercarCamara(int dir)
Descripción:	Desplazar la cámara en el eje x.
Nombre:	RotarZ(int dir)
Descripción:	Desplazar la cámara en el eje z.
Nombre:	RotarY(int dir)
Descripción:	Desplazar la cámara en el eje y.

Nombre: Port	
Tipo de clase: Entidad	
Atributo	Tipo
connection_type	ConnectionType
connected	Boolean
speed	Float
Para cada responsabilidad:	
Nombre:	Port(connectionType)
Descripción:	Constructor de un puerto.
Nombre:	Connection_Type()
Descripción:	Devuelve el tipo de conexión.
Nombre:	Connect()
Descripción:	Verifica si el puerto está conectado.
Nombre:	Disconnect()
Descripción:	Verifica si el puerto está desconectado.
Nombre:	ConnectionStatus()
Descripción:	Devuelve el estado de la conexión.
Nombre:	Speed()
Descripción:	Devuelve el valor de speed del puerto.

Nombre: Vector

Tipo de clase: Entidad	
Atributo	Tipo
x	Double
y	Double
z	Double
Para cada responsabilidad:	
Nombre:	Vector(_x,_y,_z)
Descripción:	Constructor de un vector.
Nombre:	+(Vector v)
Descripción:	Sobrecarga del operador + para operandos tipo Vector.
Nombre:	-(Vector v)
Descripción:	Sobrecarga del operador - para operandos tipo Vector.
Nombre:	+=(Vector v)
Descripción:	Sobrecarga del operador += para operandos tipo Vector.
Nombre:	-= (Vector v)
Descripción:	Sobrecarga del operador -= para operandos tipo Vector.
Nombre:	==(Vector v)
Descripción:	Sobrecarga del operador == para operandos tipo Vector.
Nombre:	!=(Vector v)
Descripción:	Sobrecarga del operador != para operandos tipo Vector.

Nombre: List	
Tipo de clase: Entidad	
Atributo	Tipo
cant	Integer
arreglo	Array
Para cada responsabilidad:	
Nombre:	List()
Descripción:	Constructor de una lista.
Nombre:	~List()

Descripción:	Destructor de una lista.
Nombre:	Len()
Descripción:	Devuelve la longitud de una lista.
Nombre:	Get(pos)
Descripción:	Devuelve el elemento de la lista, en la posición pos.
Nombre:	Insert(elem,pos)
Descripción:	Adicionar un nuevo elemento a la lista, en la posición pos.
Nombre:	Add(elem)
Descripción:	Adicionar un nuevo elemento a la lista.
Nombre:	Del(pos)
Descripción:	Eliminar un elemento de la lista.
Nombre:	Find(elem)
Descripción:	Buscar un elemento en la lista.
Nombre:	Clear()
Descripción:	Eliminar todos los elementos de una lista.

Nombre: IU_Establecer_conexion	
Tipo de clase: Interfaz	
Atributo	Tipo
gbxEstablecerConexion	GroupBox
lblSubred	Label
cbxSubred	ComboBox
gbxDatosSubred	GroupBox
lblPCConexion	Label
lblPuertos	Label
lbxPC	ListBox
lbxPuertos	ListBox
btnEstablecerConexion	Button
btnCancelarConexion	Button
Para cada responsabilidad:	

Nombre:	btnEstablecerConexionClick()
Descripción:	Llamada a los métodos de la controladora para dibujar una conexión en una subred.
Nombre:	btnCancelarConexionClick()
Descripción:	Cancelar la acción de establecer una nueva conexión.

Nombre: IU_Eliminar_conexion	
Tipo de clase: Interfaz	
Atributo	Tipo
gbxEliminarConexion	GroupBox
lblEliminarSubred	Label
lblConexion	Label
cbxDelConexionSubred	ComboBox
lbxConexion	ListBox
btnEliminarConexion	Button
btnCancelarDelConexion	Button
Para cada responsabilidad:	
Nombre:	btnEliminarConexionClick()
Descripción:	Llamada a los métodos de la controladora para eliminar una conexión en una subred.
Nombre:	btnCancelarDelConexionClick()
Descripción:	Cancelar la acción de eliminar una conexión.

Nombre: CC_Gestionar_conexion	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	AdicionarConexionSubred()
Descripción:	Realiza las operaciones necesarias para dibujar una conexión en una subred.
Nombre:	EliminarConexionSubred()
Descripción:	Realiza las operaciones necesarias para eliminar una conexión en una subred.

Nombre:	PosicionarCamara()
Descripción:	Realiza las operaciones necesarias para posicionar la cámara.
Nombre:	Pintar()
Descripción:	Borrar la pantalla. Establece la posición y la dirección del foco de luz. Pintar cada conmutador, puerto y conexión entre conmutadores.

Nombre: Connection	
Tipo de clase: Entidad	
Atributo	Tipo
pc_pos	Integer
port_pos	Integer
Para cada responsabilidad:	
Nombre:	Connection(pcPos,pcPort)
Descripción:	Constructor de una conexión.
Nombre:	PCPos()
Descripción:	Devuelve el número de la posición de una PC.
Nombre:	PortPos()
Descripción:	Devuelve el número de posición del puerto

Nombre: IU_Crear_envio	
Tipo de clase: Interfaz	
Atributo	Tipo
lblIpOrigen	Label
lblIpDestino	Label
gbxEnviarPaquete	GroupBox
cbxIpOrigen	ComboBox
cbxIpDestino	ComboBox
btnEnviarPaquete	Button
btnCancelarEnvioPauqete	Button
Para cada responsabilidad:	

Nombre:	btnEnviarPaqueteClick()
Descripción:	Llamada a los métodos de la controladora para simular el envío de paquetes.
Nombre:	btnCancelarEnvioPauqueteClick()
Descripción:	Cancelar la acción de crear un envío de paquetes.

Nombre: IU_Cancelar_envio	
Tipo de clase: Interfaz	
Atributo	Tipo
gbxCancelarEnvio	GroupBox
lblEnviosActuales	Label
lbxEnviosActuales	ListBox
btnDetenerEnvio	Button
btnCancelarDelEnvio	Button
Para cada responsabilidad:	
Nombre:	btnDetenerEnvioClick()
Descripción:	Llamada a los métodos de la controladora para detener un envío de paquetes.
Nombre:	btnCancelarDelEnvioClick()
Descripción:	Cancelar la acción de detener un envío de paquetes.

Nombre: CC_Gestionar_envio	
Tipo de clase: Controladora	
Atributo	Tipo
conectadas	Boolean
Para cada responsabilidad:	
Nombre:	AdicionarEnvioRed()
Descripción:	Realiza las operaciones necesarias para simular el envío de paquetes en la red.
Nombre:	DetenerEnvioRed()
Descripción:	Realiza las operaciones necesarias para detener el envío de paquetes.
Nombre:	PosicionarCamara()
Descripción:	Realiza las operaciones necesarias para posicionar la cámara.

Nombre: Envio	
Tipo de clase: Entidad	
Atributo	Tipo
ip1	String
ip2	String
Para cada responsabilidad:	
Nombre:	Envio(_ip1,_ip2)
Descripción:	Constructor de un envio.

Nombre: NetWork	
Tipo de clase: Entidad	
Atributo	Tipo
s1	LocalNetWork
s2	LocalNetWork
packages	cc
senders	List<Envio>
Para cada responsabilidad:	
Nombre:	NetWork()
Descripción:	Constructor de un objeto NetWork.
Nombre:	CheckPackages()
Descripción:	Chequear el recibo de los paquetes para eliminarlos.
Nombre:	Transmit(ip_origin,ip_destine)
Descripción:	En dependencia de la LocalNetWork a que pertenezcan los ip, crear las trayectorias de los paquetes (origen, intermedia y destino).

Nombre: Package	
Tipo de clase: Entidad	
Atributo	Tipo
red	Float

green	Float
blue	Float
trs	List<TR>
pos	Vector
received	Boolean
Para cada responsabilidad:	
Nombre:	Package()
Descripción:	Constructor de los paquetes.
Nombre:	Mover()
Descripción:	Cambiar el vector pos1 de la lista de trayectorias.

Nombre: TR	
Tipo de clase: Entidad	
Atributo	Tipo
pos1	Vector
pos2	Vector
mov	Vector
ctype	ConnectionType
Para cada responsabilidad:	
Nombre:	TR(p1,p2,ct)
Descripción:	Constructor de una trayectoria. Se establece una velocidad para cada tipo de conexión y se calcula el vector movimiento.
Nombre:	Mover()
Descripción:	Mientras la distancia de origen y destino sean diferentes cambiar pos1.
Nombre:	IsMoving()
Descripción:	Verificar que el Vector mov sea diferente de cero.

Nombre: IU_Interior_switch	
Tipo de clase: Interfaz	
Atributo	Tipo

Panel1	Panel
Para cada responsabilidad:	
Nombre:	Interiorswitch1Click()
Descripción:	Llamada a los métodos de la controladora para dibujar el interior de un switch.

Nombre: IU_Interior_router	
Tipo de clase: Interfaz	
Atributo	Tipo
Panel1	Panel
Para cada responsabilidad:	
Nombre:	Interiorrouter1Click()
Descripción:	Llamada a los métodos de la controladora para dibujar el interior de un router.

Nombre: CC_Gestionar_dispositivos	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	glCallList()
Descripción:	Método para dibujar los objetos de la lista, router, PC, servidor, switch.
Nombre:	Pintar()
Descripción:	Borrar la pantalla. Establece la posición y la dirección del foco de luz. Pintar cada conmutador, puerto y conexión entre conmutadores.
Nombre:	PosicionarCamara()
Descripción:	Realiza las operaciones necesarias para posicionar la cámara.

Nombre: Vistas	
Tipo de clase: Entidad	
Atributo	Tipo
tex[50]	Textura
Para cada responsabilidad:	
Nombre:	Vistas()

Descripción:	Constructor de una vista.
Nombre:	CrearListas()
Descripción:	Crear listas de objetos gl, con texturas, pintar pc, servidor, conmutador en forma de cubos. Utilizar librerías glut.h y glaux.h de OpenGL.
Nombre:	PintarPC(x, y, z)
Descripción:	Pintar el cubo que representa una PC.
Nombre:	PintarServer(x, y, z)
Descripción:	Pintar el cubo que representa un servidor.
Nombre:	PintarSwitch(x, y, z)
Descripción:	Pintar el cubo que representa un conmutador.
Nombre:	PintarPuertos(x, y, z, ports)
Descripción:	Pintar puertos en conmutador, dependiendo del tipo de puerto será su color.
Nombre:	PintarConexion(x1, y1, z1, x2, y2, z2, ctype)
Descripción:	Pintar líneas que unan las PCs o servidores con los puertos.

Nombre: Textura	
Tipo de clase: Entidad	
Atributo	Tipo
id	Integer
Para cada responsabilidad:	
Nombre:	Textura()
Descripción:	Constructor de una textura.
Nombre:	LoadBmp(const char *img)
Descripción:	Crear Textura por defecto si no se carga una imagen a utilizar como textura.
Nombre:	~Textura()
Descripción:	Destructor de una textura.

Anexo 6: Descripción de las tablas del modelo ER.

Nombre: usuario		
Descripción: Almacena los usuarios del sistema.		
Atributo	Tipo	Descripción
id_usuario	Integer	Identificador del usuario. Es el campo llave de la tabla.
id_rol	Integer	Identificador del rol del usuario en el sistema.
nombre	VARCHAR(30)	Nombre del usuario.
rol_dom	VARCHAR(20)	Rol del usuario del dominio.
usuario_dom	VARCHAR(10)	El nombre de usuario asignado por el dominio.

Nombre: rol_sistema		
Descripción: Almacena los roles de los usuarios del sistema.		
Atributo	Tipo	Descripción
id_rol	Integer	Identificador del rol. Es el campo llave de la tabla.
nombre	VARCHAR(30)	Nombre del rol.

Nombre: publicación		
Descripción: Almacena las publicaciones hechas por los usuarios avanzados en la aplicación.		
Atributo	Tipo	Descripción
id_publicacion	Integer	Identificador de la publicación. Es el campo llave de la tabla.
id_usuario	Integer	Identificador del usuario. Es campo llave de la tabla.
id_tipo_public	Integer	Identificador del tipo de publicación. Es campo llave de la tabla.
nombre	VARCHAR(80)	Nombre de la publicación.
direccion	VARCHAR(255)	Dirección URL donde se guardó la publicación.

Nombre: tipo_publicacion		
Descripción: Almacena los tipos de publicación para los documentos que se pueden publicar en la aplicación.		
Atributo	Tipo	Descripción
id_tipo_public	Integer	Identificador del tipo de publicación. Es el campo llave de la tabla.
nombre	VARCHAR(10)	Tipo de publicación.

Nombre: historial		
Descripción: Almacena el historial para cada usuario.		
Atributo	Tipo	Descripción
id_historial	Integer	Identificador del historial. Es el campo llave de la tabla.
fecha	DATETIME	Fecha en que el usuario trabaja en la aplicación.
pag_visitada	VARCHAR(255)	Página que el usuario visita.
accion	VARCHAR(255)	Acción que realiza en la página.
IP	VARCHAR(15)	Dirección IP de la computadora donde trabaja el usuario en la aplicación.

Nombre: ejercicio		
Descripción: Almacena los ejercicios propuestos por los usuarios avanzados.		
Atributo	Tipo	Descripción
id_ejercicio	Integer	Identificador del ejercicio. Es el campo llave de la tabla.
texto_ejercicio	LONGTEXT	Texto del ejercicio.
imagen	BLOB	Imagen de plano para ejercicio.
fecha_public	DATETIME	Fecha en que se publica el ejercicio.
nombre	VARCHAR(100)	Nombre del ejercicio.
presupuesto	Float	Presupuesto para el ejercicio.

Nombre: usuario_ejercicio		
Descripción: Almacena las relaciones entre los usuarios, los ejercicios y las respuestas a estos.		
Atributo	Tipo	Descripción
id_usuario_ejercicio	Integer	Identificador de la relación entre usuario y ejercicios. Es el campo llave de la tabla.
id_respuesta	Integer	Identificador de la respuesta. Es campo llave de la tabla.
id_usuario	Integer	Identificador del usuario. Es campo llave de la tabla.
id_ejercicio	Integer	Identificador del ejercicio. Es campo llave de la tabla.

Nombre: respuesta		
Descripción: Almacena las respuestas dadas por los usuarios a los ejercicios propuestos.		
Atributo	Tipo	Descripción
id_respuesta	Integer	Identificador de la respuesta. Es el campo llave de la tabla.
id_topologia	Integer	Identificador de la topología de red. Es campo llave de la tabla.
id_estandar	Integer	Identificador del estándar de red. Es campo llave de la tabla.
id_tiporespuesta	Integer	Identificador del tipo de respuesta. Es campo llave de la tabla.
texto	VARCHAR(2000)	Anotación importante sobre la respuesta.

Nombre: tipo_respuesta		
Descripción: Almacena los tipos de respuestas.		
Atributo	Tipo	Descripción
id_tiporespuesta	Integer	Identificador del tipo de respuesta. Es el

		campo llave de la tabla.
tipo	VARCHAR(50)	Tipo de respuesta.

Nombre: estándar		
Descripción: Almacena los estándares de red.		
Atributo	Tipo	Descripción
id_estandar	Integer	Identificador del estándar de red. Es el campo llave de la tabla.
nombre	VARCHAR(50)	Nombre del estándar de red.

Nombre: topología		
Descripción: Almacena las diferentes topologías de red.		
Atributo	Tipo	Descripción
id_topologia	Integer	Identificador de la topología de red. Es el campo llave de la tabla.
nombre	VARCHAR(50)	Nombre de la topología de red.

Nombre: respuesta_producto		
Descripción: Almacena los datos de la relación entre las respuestas y los productos.		
Atributo	Tipo	Descripción
id_respuesta_producto	Integer	Identificador de la relación entre las respuestas y los productos. Es el campo llave de la tabla.
id_respuesta	Integer	Identificador de la respuesta. Es campo llave de la tabla.
id_producto	Integer	Identificador del producto. Es campo llave de la tabla.
cantidad	Integer	Cantidad de un producto incluido en una respuesta.

Nombre: producto		
-------------------------	--	--

Descripción: Almacena los productos de red por proveedor.		
Atributo	Tipo	Descripción
id_producto	Integer	Identificador del producto. Es el campo llave de la tabla.
id_proveedor	Integer	Identificador del proveedor. Es campo llave de la tabla.
id_unidad_medida	Integer	Identificador de la unidad de medida. Es campo llave de la tabla.
id_categoria	Integer	Identificador de la categoría. Es campo llave de la tabla.
descripcion	LONGTEXT	Descripción del producto.
codigo_producto	VARCHAR(30)	Código del producto.
nombre	VARCHAR(255)	Nombre del producto.
precio	FLOAT	Precio del producto.
especificación	VARCHAR(50)	Especificación del tipo de producto.

Nombre: proveedores		
Descripción: Almacena los datos de los proveedores de los elementos de red.		
Atributo	Tipo	Descripción
id_proveedor	Integer	Identificador del proveedor. Es el campo llave de la tabla.
nombre	VARCHAR(50)	Nombre del proveedor.
descripción	LONGTEXT	Descripción del proveedor.

Nombre: unidad_medida		
Descripción: Almacena las unidades de medida para los elementos de red.		
Atributo	Tipo	Descripción
id_unidad_medida	Integer	Identificador de la unidad de medida. Es el campo llave de la tabla.
nombre	VARCHAR(50)	Nombre de la unidad de medida.

Nombre: categoría		
Descripción: Almacena las categorías de los elementos de red.		
Atributo	Tipo	Descripción
id_categoria	Integer	Identificador de la categoría de los elementos de red. Es el campo llave de la tabla.
nombre	VARCHAR(50)	Nombre de las categorías de los elementos de red.

Nombre: especificaciones		
Descripción: Almacena las especificaciones de los elementos de red.		
Atributo	Tipo	Descripción
id_especificacion	Integer	Identificador de la especificación de los elementos de red. Es el campo llave de la tabla.
nombre	VARCHAR(50)	Nombre de las especificaciones de los elementos de red.
tipo	VARCHAR(25)	Tipo de las especificaciones.

Anexo 7: Modelo de implementación

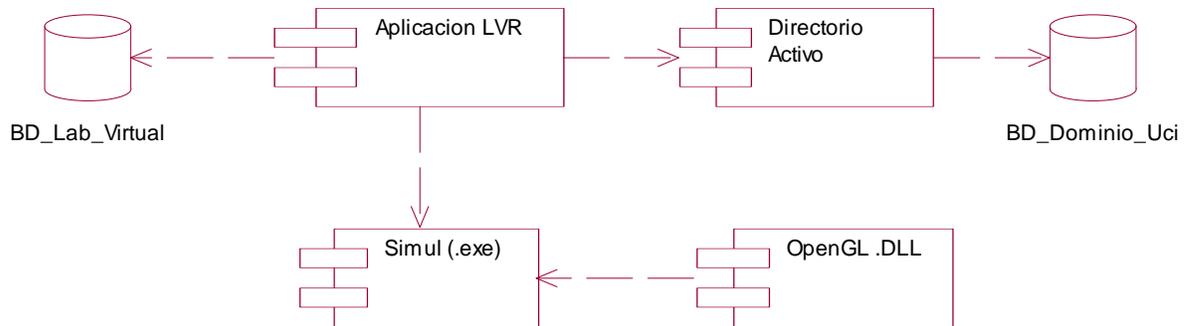


Figura 74 Diagrama de Componentes del Sistema.

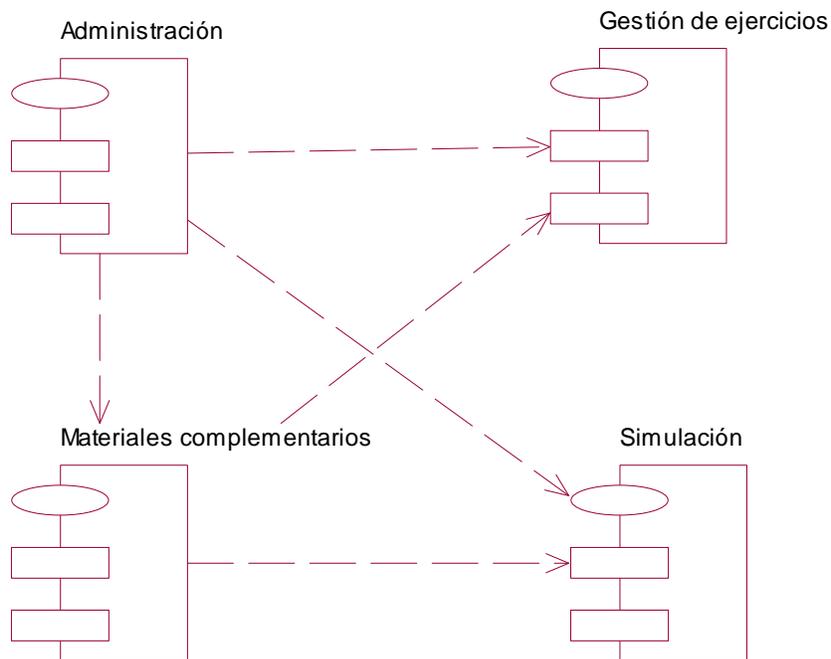


Figura 75 Diagrama de Componentes por Paquetes.

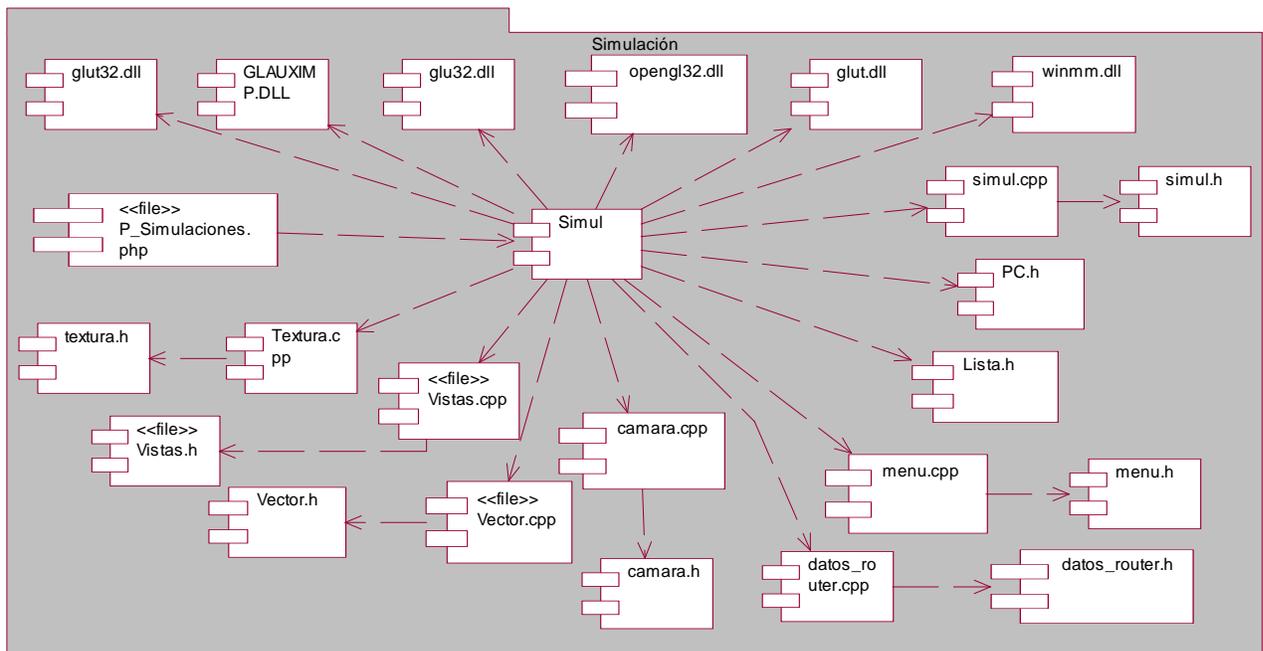


Figura 76 Diagrama de Componentes del Paquete Simulación.

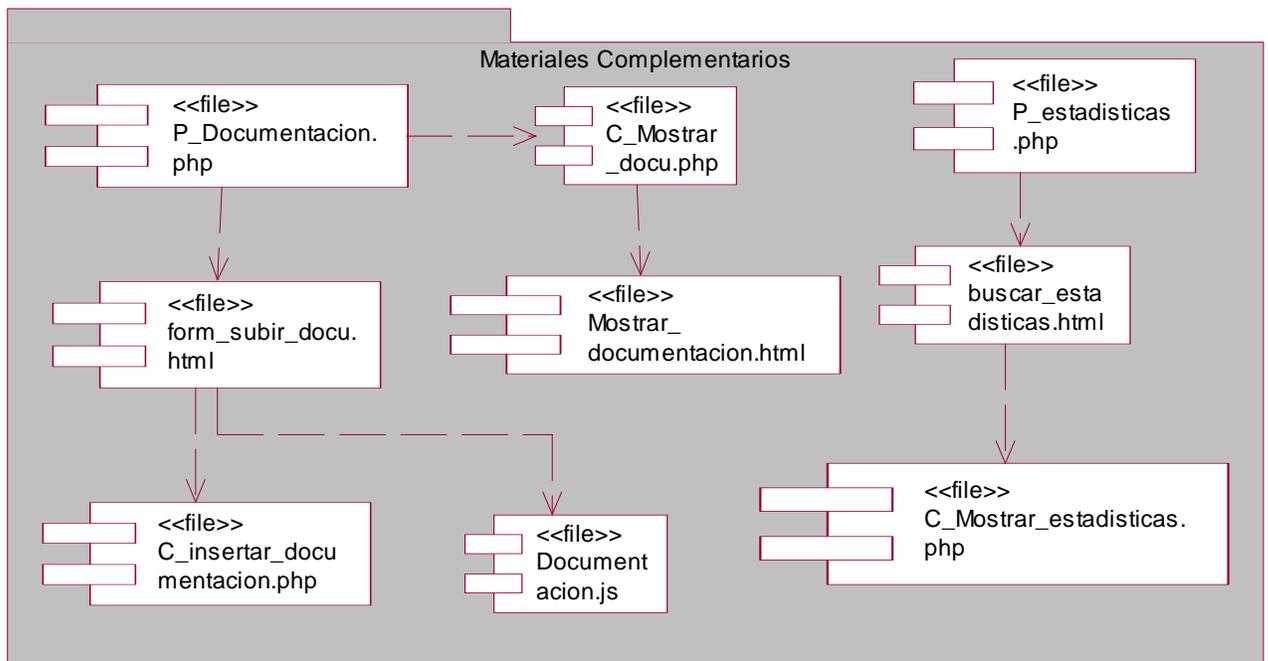


Figura 77 Diagrama de Componentes del Paquete Materiales Complementarios.

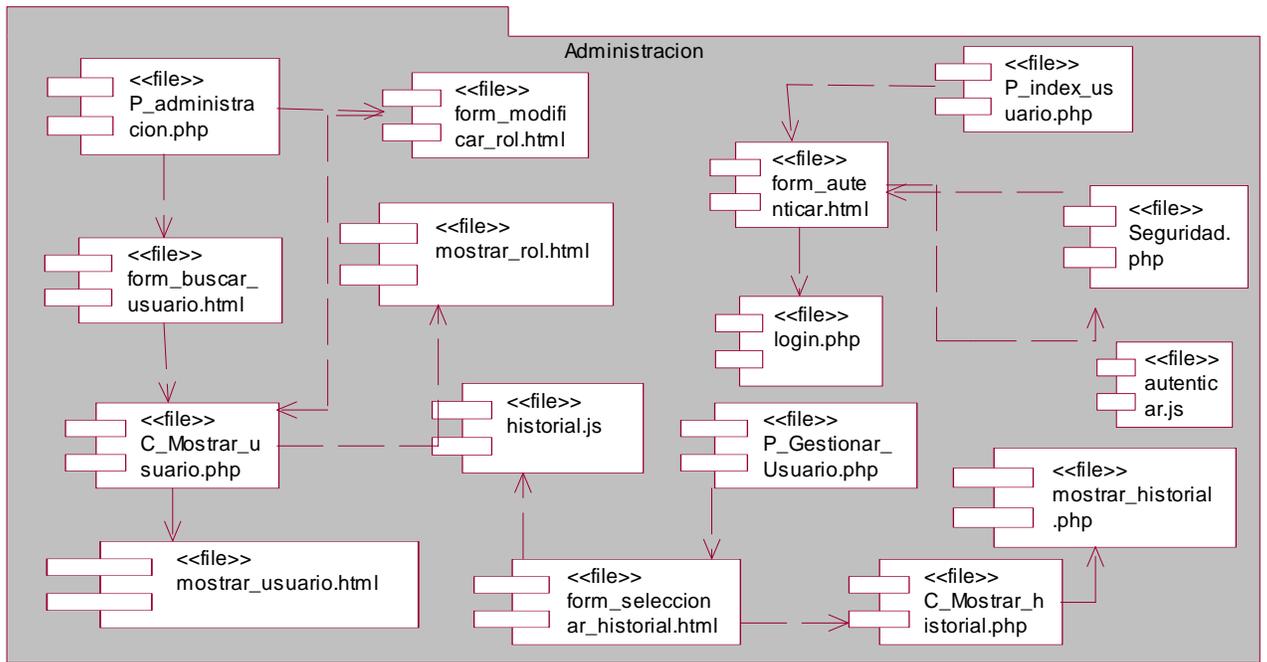


Figura 78 Diagrama de Componentes del Paquete Administración.

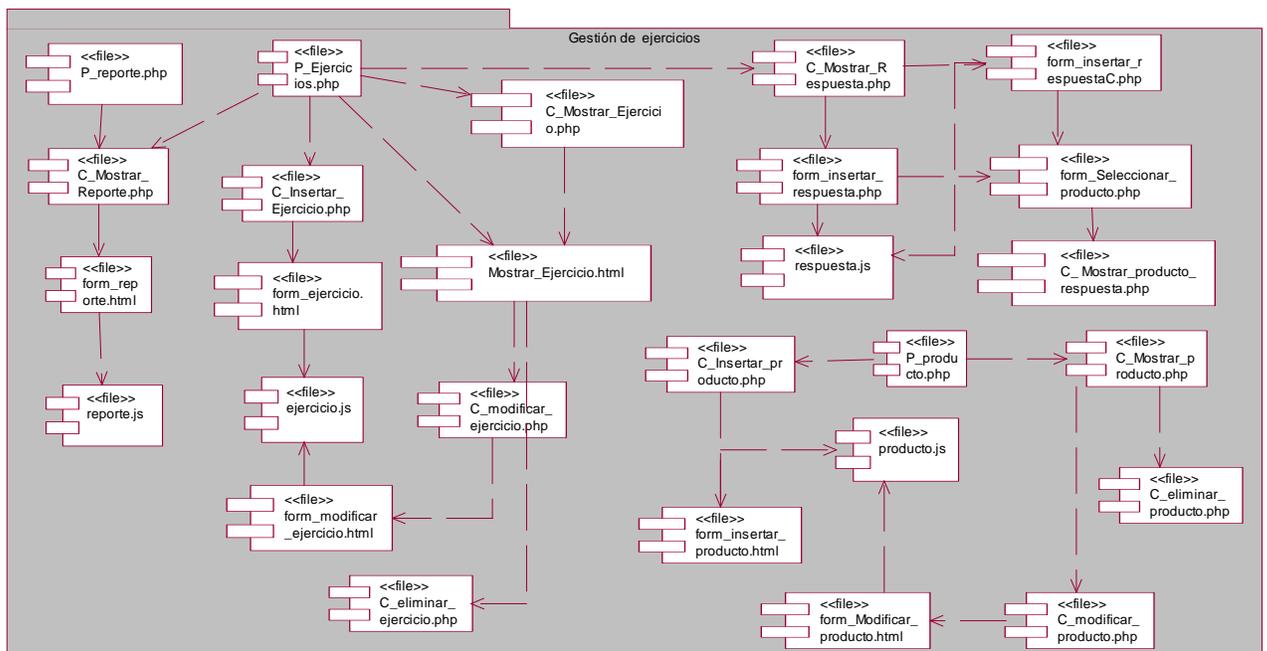


Figura 79 Diagrama de Componentes del Paquete Gestión de Ejercicios.

Anexo 8: Descripción de los Casos de Prueba.

Caso de Uso	Nombre:	Autenticar usuario		
	Descripción:	Para entrar en el Laboratorio Virtual de Redes se debe autenticar cada usuario, introduciendo el usuario y la contraseña del dominio.		
Casos de Prueba	Entrada	Resultados	Condiciones	
1.	El usuario introduce el usuario del dominio y la contraseña incorrectos, no existen en la base de datos. Usuario= "jpuertov" Contraseña="J173_pv"	El sistema muestra un mensaje informando al usuario que su usuario y contraseña no son correctos y se solicita que se introduzcan los datos nuevamente.	Como el usuario no existe en la base de datos del Directorio Activo de la UCI, el nombre entrado no coincide.	
2.	El usuario introduce el usuario del dominio válido y una contraseña que no existe en la base de datos. Usuario ="jpuerto" Contraseña="J173_pv"	El sistema muestra un mensaje informando al usuario que su contraseña no es la correcta y se solicita que se introduzcan los datos nuevamente.	Como el usuario que existe en la base de datos es jpuerto y la contraseña J173_pv entrada no existe en la base de datos del Directorio Activo de la UCI, la contraseña entrada no coincide.	
3.	El usuario introduce el usuario del dominio y la contraseña válidos. Usuario = "jpuerto" Contraseña="J17_pv*90"	El sistema guarda los datos del usuario que se obtuvieron del ldap como usuario del sistema y se muestran sólo aquellas funcionalidades a las que tiene acceso, con privilegio mínimo en la aplicación.	Como el usuario que existe en la base de datos es jpuerto y la contraseña entrada coincide, y como no existe en el sistema, los datos coinciden y el usuario se autentica satisfactoriamente.	

Caso de Uso	Nombre:	Gestionar producto		
	Descripción:	Permite registrar, modificar y eliminar los diferentes productos necesarios para el diseño de la red.		
Casos de Prueba	Entrada	Resultados	Condiciones	
4.	Se introducen datos de un producto que ya existe en el sistema. Nombre del Producto = "AT-567MSG 10/100TX" Código Producto = "0503452" Precio="521.20" Categoría="Dispositivos" Proveedor="PCMax" Unidad de Medida="Unidad" Descripción="Conmutador con 6 puertos GigaBit Ethernet" Especificación="Switch" Cantidad de Puertos = "24"	El sistema muestra un mensaje informando al usuario que el producto ya existe en el sistema, y se solicita que se introduzcan los datos nuevamente.	Como el producto existe ya en el sistema, no se puede realizar su registro.	
5.	Se introducen datos válidos de un producto. Nombre del Producto = "AT-125MSG 10/100TX" Código Producto = "0503459" Precio = "125.00" Categoría = "Dispositivos"	El producto se almacena en el sistema. Se muestra un mensaje informándole al usuario avanzado que ha sido efectuado el registro del producto.	Como todos los campos obligatorios sobre el producto están llenos, y el producto no existe en el sistema, se puede realizar su registro.	

Proveedor = "PCMax" Unidad de Medida = "Unidad" Descripción = "Conmutador Fast Ethernet" Especificación = "Switch" Cantidad de Puertos = "8"		
--	--	--

Caso de Uso	Nombre:	Gestionar proveedor	
	Descripción:	Permite registrar y eliminar los proveedores de los productos.	
Casos de Prueba	Entrada	Resultados	Condiciones
6.	Se introducen datos válidos de un proveedor para insertarlo en el sistema. Nuevo Proveedor = "TeCun" Descripción = "Firma comercializadora de equipamiento informático."	El proveedor se almacena en el sistema. Se muestra un mensaje informándole al usuario avanzado que ha sido efectuado el registro del proveedor.	Como los campos obligatorios están llenos, y este proveedor no existe, se puede realizar su registro.
7.	El usuario ejecuta el comando de adicionar un proveedor dejando campos obligatorios vacíos. Nuevo Proveedor = "" Descripción = ""	Se emite un mensaje para que el usuario avanzado llene los campos obligatorios para realizar el registro del proveedor.	Como los campos obligatorios no están llenos, no se puede realizar el registro del proveedor.
8.	Se selecciona un proveedor para eliminarlo del sistema. Nombre Proveedor =	El sistema elimina el proveedor, y se muestra un mensaje de eliminación.	Como el usuario confirma la eliminación del proveedor, se elimina el proveedor del

	"TeCun"		registro.
--	---------	--	-----------

Caso de Uso	Nombre:	Gestionar subred		
	Descripción:	Permite adicionar un nuevo puerto, una PC o un servidor a una subred.		
Casos de Prueba	Entrada	Resultados	Condiciones	
9.	Se selecciona adicionar una PC a la subred del switch1, y se introduce una dirección IP que ya existe en la subred. Ip de la computadora = "10.8.1.5"	El sistema muestra un mensaje informando que la dirección IP está en uso, y se muestra la opción de introducir otra dirección IP.	Como el IP introducido por el usuario ya está asignado a una computadora en la subred, no se puede dibujar una nueva PC.	
10.	Se selecciona adicionar una PC a la subred del switch2, y no se introduce una dirección IP. Ip de la computadora = "10.8.1. "	El sistema muestra un mensaje informando que la dirección IP no es correcta, y se solicita que se introduzca.	Como el IP introducido por el usuario no es correcto, no se puede dibujar una nueva PC.	
11.	Se selecciona adicionar un servidor a la subred del switch2, y se introduce una dirección IP válida. Ip de la computadora = "10.8.5.12"	El sistema dibuja el servidor en la subred del switch2.	Como el IP introducido por el usuario no está asignado a otra computadora en la red, se puede dibujar un nuevo servidor.	
12.	Se selecciona adicionar un puerto a la subred del switch1, y se introduce un tipo de conexión válido.	El sistema dibuja el puerto en el switch1, de color amarillo.	Como el tipo de conexión del puerto es GigaBitEthernet, se dibuja el puerto en el switch1 de	

	Tipo de conexión = "GigaBitEthernet"		color amarillo.
--	--------------------------------------	--	-----------------

Caso de Uso	Nombre:	Establecer conexión	
	Descripción:	Permite registrar y eliminar conexiones en una subred.	
Casos de Prueba	Entrada	Resultados	Condiciones
13.	Se selecciona la opción de establecer una conexión en el switch1, se introduce la subred, y se selecciona la PC y el puerto que se quieren conectar. Subred = "Switch L3 1" PC = "10.8.1.6" Puerto = "FastEthernet"	El sistema dibuja la conexión, de color azul.	Como la PC se encuentra sin conectar y el puerto libre, se dibuja la conexión, y como el puerto es FastEthernet, se dibuja la conexión de color azul.
14.	Se selecciona la opción de eliminar una conexión en el switch2, se introduce la subred, y se selecciona la conexión que se quieren eliminar. Subred = "Switch L3 2" Conexión = "10.8.5.121---Ethernet"	El sistema elimina la conexión entre la PC 10.8.5.121 y el puerto Ethernet, en el switch2.	Como la PC 10.8.5.121 se encuentra conectada al puerto Ethernet, se elimina esta conexión.

Caso de Uso	Nombre:	Enviar paquete	
	Descripción:	Permite simular el envío de paquetes de datos en una red.	

Casos de Prueba	Entrada	Resultados	Condiciones
15.	Se selecciona la opción de enviar paquetes en la red y se introducen el IP origen y el IP destino, válidos. IP origen = "10.8.1.5" IP destino = "10.8.5.121"	El sistema simula el envío de paquetes desde el IP origen hasta el IP destino, dependiendo la velocidad de los paquetes del tipo de conexión por la que pasen.	Como las PC 10.8.1.5 y 10.8.5.121 se encuentran conectadas en las subredes, se envían paquetes de la subred 1 a la subred 2.
16.	Se selecciona la opción de enviar paquetes en la red y se introducen el IP origen y el IP destino iguales. IP origen = "10.8.1.5" IP destino = "10.8.1.5"	El sistema muestra un mensaje informando que la dirección IP origen y destino no pueden coincidir.	Como las direcciones IP origen y destino son iguales, no se envían los paquetes.

Caso de Uso	Nombre:	Gestionar ejercicio
	Descripción:	Permite registrar, modificar y eliminar los diferentes ejercicios orientados de diseño de una red.

Casos de Prueba	Entrada	Resultados	Condiciones
17.	Se introducen datos válidos de un ejercicio para insertarlo en el sistema. Nombre ejercicio = "Diseño de red en docente2" Texto = "Diseñe una red para el docente 2 de la universidad." Presupuesto = "20058" Imagen = "E:\plano docente	El ejercicio se almacena en el sistema, y se muestra un mensaje informándole al usuario avanzado que ha sido efectuado el registro del ejercicio.	Como los campos obligatorios están llenos y este ejercicio no existe en el sistema, se realiza el registro.

	2.jpg”		
18.	Se dejan campos obligatorios de un ejercicio vacíos. Nombre ejercicio = “ejerc” Texto = “” Presupuesto = “” Imagen = “E:\plano auxiliar.jpg”	Se emite un mensaje para que el usuario avanzado llene los campos obligatorios.	Como los campos obligatorios no están llenos, no se realiza el registro del ejercicio.
19.	Se selecciona un ejercicio a eliminar. Nombre ejercicio = “ejerc1”	El sistema elimina el ejercicio, y se muestra un mensaje de eliminación.	Como el usuario confirma la eliminación del ejercicio, se elimina el ejercicio del registro del sistema.

Caso de	Nombre:	Responder ejercicio	
Uso	Descripción:	Permite guardar la respuesta dada por el usuario al ejercicio seleccionado.	
Casos de Prueba	Entrada	Resultados	Condiciones
20.	Se introducen datos correctos para responder un ejercicio. Usuario = “Janet Puerto Viera” Ejercicio = “Diseño de red en docente2” Topología = “Estrella” Estándar = “ISO” Producto = “Zyxel Omni 56K Lite USB Lite	El sistema guarda la respuesta del usuario.	Como los datos entrados por el usuario son correctos, se registra la respuesta al ejercicio Diseño de red en docente2.

	Modem” Cantidad = 2		
--	------------------------	--	--

Caso de Uso	Nombre:	Operar historial usuario		
	Descripción:	Permite mostrar el historial de los estudiantes al usuario avanzado.		
Casos de Prueba	Entrada	Resultados	Condiciones	
21.	Se introduce el usuario para consultar el historial. Usuario = “Janet Bárbara Puerto Viera”	Se muestra el historial del usuario.	Como el usuario está registrado en el sistema se muestran las acciones, las páginas a las que ha tenido acceso, la dirección IP de donde accedió a la aplicación y la fecha.	

Caso de Uso	Nombre:	Publicar documentación		
	Descripción:	Permite publicar documentación por el usuario avanzado.		
Casos de Prueba	Entrada	Resultados	Condiciones	
22.	Se introducen los datos de la publicación. Nombre para Mostrar = “How LAN Switches Work” Tipo archivo = “doc” Archivo a publicar = “E:\Yordan\simulacion\How LAN Switches Work.doc”	La publicación se almacena en el sistema, y se muestra un mensaje informándole al usuario avanzado que ha sido efectuado el registro.	Como los campos obligatorios están llenos, y esta publicación no existe en el sistema, se realiza el registro	

			de la publicación.
--	--	--	--------------------

Caso de Uso	Nombre:	Administrar usuarios		
	Descripción:	Permite modificar el rol de los diferentes usuarios del sistema.		
Casos de Prueba	Entrada	Resultados	Condiciones	
23.	Se selecciona el usuario al que se le quiere cambiar el rol del sistema, y se selecciona el nuevo rol. Usuario = "lperez" Nuevo rol = "Usuario Avanzado"	El sistema guarda el nuevo rol del sistema del usuario, y se muestra un mensaje informándole al administrador que ha sido efectuada la modificación del usuario.	Como el usuario existe en el registro de usuarios del sistema, se actualiza el rol del sistema.	

GLOSARIO

1. **Web:** Es una página, sitio o conjunto de sitios que proveen información por los navegadores mostrándolos en la pantalla del usuario, también puede referirse a la enorme e interconectada red disponible prácticamente en todos los sitios de Internet.
2. **HTTP:** El protocolo de transferencia de hipertexto (HyperText Transfer Protocol) usado en cada transacción de la Web.
3. **OSI:** El modelo de referencia de Interconexión de Sistemas Abiertos (OSI, Open System Interconnection), este modelo de red descriptivo ayuda a los fabricantes a crear redes que sean compatibles con otras redes.
4. **Ethernet:** Define las características de cableado y señalización de nivel físico y los formatos de trama del nivel de enlace de datos del modelo OSI.
5. **ARP:** Protocolo de resolución de direcciones (Address Resolution Protocol). Es un protocolo de nivel de red responsable de encontrar la dirección hardware que corresponde a una determinada dirección IP.
6. **ICMP:** El Protocolo de Control de Mensajes de Internet o ICMP (Internet Control Message Protocol) es el subprotocolo de diagnóstico y notificación de errores del Protocolo de Internet (IP).
7. **IP:** El Protocolo de Internet (Internet Protocol) es un protocolo no orientado a conexión usado tanto por el origen como por el destino para la comunicación de datos a través de una red de paquetes conmutados.
8. **TCP/IP:** Protocolo de comunicaciones estándar en Internet (Transmission Control Protocol/Internet Protocol).
9. **IIS:** Es el servidor Web de Microsoft que corre sobre plataformas Windows. (Internet Information Server).
10. **FTP:** Es un protocolo de transferencia de ficheros entre sistemas conectados a una red TCP (File Transfer Protocol), desde un equipo cliente nos podemos conectar a un servidor para descargar ficheros desde él o para enviarle nuestros propios archivos independientemente del sistema operativo utilizado en cada equipo.
11. **UDP:** Es un protocolo del nivel de transporte (User Datagram Protocol) basado en el intercambio de datagramas, permite el envío de datagramas a través de la red sin que se haya establecido previamente una conexión.
12. **RTP:** Es un Protocolo de Transporte de Tiempo real (Real-time Transport Protocol), es un protocolo de nivel de aplicación (no de nivel de transporte, como su nombre podría hacer pensar) utilizado para la transmisión de información en tiempo real.

13. **Tcl/TK, Otcl, TclCL20:** son lenguajes interpretados de programación visual de fácil aprendizaje y potente, que genera código portable. Se usan principalmente en programas rápidos, aplicaciones "script", entornos gráficos y pruebas.

14. **Linux:** Es un sistema operativo tipo Unix. Es uno de los paradigmas más prominentes del software libre y del desarrollo del código abierto, cuyo código fuente está disponible públicamente y cualquier persona puede libremente usarlo, estudiarlo, redistribuirlo y, con los conocimientos informáticos adecuados, modificarlo.

15. **Windows:** Es un sistema operativo con interfaz gráfica para computadoras personales cuyo propietario es la empresa Microsoft.

16. **Sun Solaris:** Es un sistema operativo desarrollado por Sun Microsystems. Aunque Solaris en sí mismo aún es software propietario, la parte principal del sistema operativo se ha liberado como un proyecto de software libre denominado Opensolaris.

17. **HTML:** Es el Lenguaje de Marcado de Hipertexto (Hypertext Markup Language) en el que se escriben las páginas a las que se accede a través de navegadores WWW. Admite componentes hipertextuales y multimedia.

SOAP: Es un protocolo estándar (Simple Object Access Protocol) que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML, es uno de los protocolos utilizados en los servicios Web. En SOAP los mensajes se forman con un formato determinado por medio del XML para encapsular los parámetros de la petición. Cada mensaje está compuesto por tres partes: sobre, encabezado y cuerpo. El sobre envuelve al mensaje y contiene al encabezado y al cuerpo; el encabezado es opcional y proporciona información para el enrutamiento del mensaje; por su parte el cuerpo contiene los datos en formato XML.

18. **XML:** Es un lenguaje extensible de etiquetas (Extensible Markup Language) que permite definir lenguajes de marcado adecuado a usos determinados. Se propone como lenguaje de bajo nivel (a nivel de aplicación, no de programación) para intercambio de información estructurada entre diferentes plataformas.

19. **DHTML:** El HTML Dinámico o DHTML (Dynamic HTML) designa el conjunto de técnicas que permiten crear sitios web interactivos utilizando una combinación de lenguaje HTML estático, un lenguaje interpretado en el lado del cliente (como Java Script), el lenguaje de hojas de estilo en cascada (CSS) y la jerarquía de objetos de un DOM.

20. **WWW:** Se le llama World Wide Web o la Web, es un sistema de navegador web para extraer elementos de información llamados documentos o páginas web.
21. **SGML:** Es el Lenguaje de Marcación Generalizado (Standard Generalized Markup Language) es un sistema para la organización y etiquetado de documentos.
22. **Java Script:** Lenguaje desarrollado por Netscape. Aunque es parecido a Java se diferencia de él en que los programas están incorporados en el fichero HTML.
23. **Perl:** Lenguaje Práctico para la Extracción e Informe (Practical Extraction and Report Language).
24. **ASP:** (Active Server Pages) es una tecnología del lado servidor de Microsoft para páginas web generadas dinámicamente, que ha sido comercializada como un anexo a IIS.
25. **JSP:** (Java Server Pages) es una tecnología Java que permite a los programadores generar contenido dinámico para la web, en forma de documentos HTML, XML o de otro tipo.
26. **Ruby:** Es un lenguaje de programación interpretado en una sola pasada y su implementación oficial es distribuida bajo una licencia de software libre. Combina una sintaxis inspirada en Python, Perl con características de programación orientada a objetos. Comparte también funcionalidad con otros lenguajes de programación.
27. **Python:** Es un lenguaje de programación que permite dividir el programa en módulos reutilizables desde otros programas Python. Viene con una gran colección de módulos estándar que se pueden utilizar como base de los programas (o como ejemplos para empezar a aprender Python).
28. **Teleformación:** Es un sistema de formación apoyado en el uso intensivo de las Tecnologías de la Información y las Comunicaciones, que combina elementos pedagógicos y tecnológicos para dar soporte parcialmente al sistema de pregrado y postgrado de la UCI.
29. **Routers:** Un router (enrutador o encaminador) es un dispositivo hardware o software de interconexión de redes de computadoras que opera en la capa 3 (Nivel de Red) del modelo OSI. Trabaja con tablas de encaminamiento o enrutado con la información que generan los protocolos, deciden si hay que enviar un paquete o no, deciden cual es la mejor ruta para enviar un paquete, y cual es la mejor ruta para enviar la información de un equipo a otro. Este dispositivo interconecta segmentos de red o redes enteras. Hace pasar paquetes de datos entre redes tomando como base la información de la capa de red.
30. **Switch:** Un switch (conmutador) es un dispositivo electrónico de interconexión de redes de computadoras que opera en la capa 2 (Nivel de enlace de datos) del modelo OSI. Un switch interconecta dos o más segmentos de red, funcionando de manera similar a los puentes (bridges), pasando datos de un segmento a otro, de acuerdo con la dirección MAC de destino de los datagramas en la red. Los

switches se utilizan cuando se desea conectar múltiples redes, fusionándolas en una sola. Al igual que los bridges, mejoran el rendimiento y la seguridad de las LAN.

31. **MAC:** En redes de computadoras la dirección MAC (Media Access Control) es un identificador hexadecimal de 48 bits que se corresponde de forma única con una tarjeta o interfaz de red.

32. **RouterSim Network Visualizer:** Es un simulador de redes que permite diseñar, construir y configurar una red con tan solo arrastrar y soltar elementos. El programa ofrece 470 comandos y más de 220 e-labs.

33. **LAN:** Redes de área local, ambientes de computo (Ethernet, Fast Ethernet, Token-ring, Arcnet, FDDI/TP-PMD)

34. **VLAN: (Virtual LAN, 'red de área local virtual')** es una red de computadoras lógicamente independiente. Consiste en una red de ordenadores que se comportan como si estuviesen conectados al mismo cable, aunque pueden estar en realidad conectados físicamente a diferentes segmentos de una red de área local. Una de las mayores ventajas de las VLANs surge cuando se traslada físicamente una computadora a otra ubicación: puede permanecer en la misma VLAN sin necesidad de ninguna reconfiguración hardware.

35. **UML:** El Lenguaje Unificado de Modelado prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan. Mientras que ha habido muchas notaciones y métodos usados para el diseño orientado a objetos, ahora los modeladores sólo tienen que aprender una única notación.

36. **CGI:** Common Gateway Interface (Interfaz Común de Pasarela) es una importante tecnología de la World Wide Web que permite a un cliente (explorador web) solicitar datos de un programa ejecutado en un servidor web. Las aplicaciones CGI fueron una de las primeras maneras prácticas de crear contenido dinámico para las páginas web. En una aplicación CGI, el servidor web pasa las solicitudes del cliente a un programa externo. La salida de dicho programa es enviada al cliente en lugar del archivo estático tradicional.

37. **C++:** El C++ es un lenguaje de programación, diseñado a mediados de los años 1980, por Bjarne Stroustrup, como extensión del lenguaje de programación C. Es un lenguaje que abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos.

38. **Conectores:** Un conector de ordenadores es cualquier conector dentro de un ordenador, o para conectar ordenadores a redes, impresoras u otros dispositivos. Generalmente, estos conectores tienen

nombres específicos que permiten una identificación más precisa, y el uso de estos nombres se recomienda. Algunos conectores de ordenadores comunes son USB, FireWire, BNC, RJ-11 y varios conectores de pin y de "socket". La mayor parte de los conectores de ordenador son eléctricos u ópticos.

39. **Jack hembra:** El conector Jack es un conector de audio utilizado en numerosos dispositivos para la transmisión de sonido en formato analógico.

40. **Jumper dúplex:** Es un elemento para interconectar dos terminales de manera temporal sin tener que efectuar una operación que requiera herramienta adicional, dicha unión de terminales cierran el circuito eléctrico del que forma parte.

41. **Modems:** Un modem es un dispositivo que permite la transmisión y recepción de información binaria (es decir, datos de ordenador) a través de un medio analógico (o sea, la línea telefónica); para poder realizar este proceso, es necesario convertir la señal digital en analógica, y viceversa, y esa es la función del modem. El modulador emite un fuerte sonido y una señal analógica constante denominada portadora. A medida que se desea transmitir datos digitales, se modifica alguna característica de la señal portadora, de manera que se indica si se está transmitiendo un "cero" o un "uno". El demodulador interpreta los cambios en la señal portadora para reconstruir el flujo de datos digitales.

42. **UPS:** Sistema de alimentación ininterrumpida (Uninterruptible Power Supply, en español abreviado como SAI).

43. **Canaletas:** Las canaletas son tubos metálicos o plásticos que conectados de forma correcta proporcionan al cable una segunda pantalla o protección.

44. **Cables de fibra óptica:** Tienen circuitos de fibra óptica con filamentos de vidrio (compuestos de cristales naturales) o plástico (cristales artificiales), del espesor de un pelo (entre 10 y 300 micrones). Llevan mensajes en forma de haces de luz que realmente pasan a través de ellos de un extremo a otro, donde quiera que el filamento vaya (incluyendo curvas y esquinas) sin interrupción. Las fibras ópticas pueden ahora usarse como los alambres de cobre convencionales, tanto en pequeños ambientes autónomos (tales como sistemas de procesamiento de datos de aviones), como en grandes redes geográficas (como los sistemas de largas líneas urbanas mantenidos por compañías telefónicas).

45. **Cables coaxiales:** El cable coaxial consiste de un núcleo sólido de cobre rodeado por un aislante, una combinación de blindaje y alambre de tierra y alguna otra cubierta protectora. En el pasado del cable coaxial tenía rasgos de transmisión superiores que el cable par trenzado, pero ahora las técnicas de transmisión para el par trenzado igualan o superan los rasgos de transmisión del cable coaxial.

46. **Firewall:** Un cortafuegos es un elemento de hardware o software utilizado en una red de computadoras para controlar las comunicaciones, permitiéndolas o prohibiéndolas según las políticas de red que haya definido la organización responsable de la red. La ubicación habitual de un cortafuegos es el punto de conexión de la red interna de la organización con la red exterior, que normalmente es Internet; de este modo se protege la red interna de intentos de acceso no autorizados desde Internet, que puedan aprovechar vulnerabilidades de los sistemas de la red interna. También es frecuente conectarlo a una tercera red, llamada zona desmilitarizada o DMZ, en la que se ubican los servidores de la organización que deben permanecer accesibles desde la red exterior.

47. **Detector de intrusos:** Un sistema de detección de intrusos (o IDS de sus siglas en inglés Intrusion Detection System) es un programa usado para detectar accesos desautorizados a un computador o a una red. Estos accesos pueden ser ataques de habilidosos hackers, o de Script Kiddies que usan herramientas automáticas. El IDS suele tener sensores virtuales (por ejemplo, un sniffer de red) con los que el núcleo del IDS puede obtener datos externos (generalmente sobre el tráfico de red). El IDS detecta, gracias a dichos sensores, anomalías que pueden ser indicio de la presencia de ataques o falsas alarmas.

48. **Print Server:** Un Servidor de Impresión es un concentrador, o más bien un servidor, que conecta una impresora a una red, para que cualquier PC pueda acceder a ella e imprimir trabajos, sin depender de otro PC para poder utilizarla, como es el caso de las impresoras compartidas. Actualmente existen servidores de impresora tanto para interfaz paralela, como USB.

49. **Proxy:** El uso más común es el de servidor proxy, que es un ordenador que intercepta las conexiones de red que un cliente hace a un servidor de destino. De ellos, el más famoso es el servidor proxy de web (comúnmente conocido solamente como "proxy"). Intercepta la navegación de los clientes por páginas web, por varios motivos posibles: seguridad, rendimiento, anonimato, etc. También existen proxies para otros protocolos, como el proxy de FTP. El proxy ARP puede hacer de enrutador en una red, ya que hace de intermediario entre ordenadores.

50. **Delphi:** Es un entorno de desarrollo de software diseñado para la programación de propósito general con énfasis en la programación visual. En Delphi se utiliza como lenguaje de programación una versión moderna de Pascal llamada Object Pascal. En sus diferentes variantes, permite producir archivos ejecutables para Windows, Linux y la plataforma .NET.

51. **IDE (Entorno de Desarrollo Integrado):** 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. Los IDEs pueden ser aplicaciones por si solas o pueden ser parte de aplicaciones existentes.

52. API(Interfaz de Programación de Aplicaciones): Representa un interfaz de comunicación entre componentes software. Se trata del conjunto de llamadas a ciertas librerías que ofrecen acceso a ciertos servicios desde los procesos y representa un método para conseguir abstracción en la programación, generalmente (aunque no necesariamente) entre los niveles o capas inferiores y los superiores del software. Uno de los principales propósitos de una API consiste en proporcionar un conjunto de funciones de uso general,

53. Bitmaps: Una imagen rasterizada, también llamada bitmap, es una estructura o fichero de datos que representan generalmente una rejilla rectangular de pixeles o puntos de color, denominada raster, en un monitor, papel u otro dispositivo de representación.

54. Framework: Es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Un framework puede incluir soporte de programas, bibliotecas y un lenguaje de scripting entre otros softwares para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio.