Universidad de las Ciencias Informáticas
Facultad 5

HADIS: Herramienta para la administración del proceso de desarrollo de software para proyectos docentes de la asignatura de Ingeniería de Software.

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

Autor(es):
Yadira Ramírez Rodríguez
Juan De Jesús Grave de Peralta Rodríguez

Tutor
Ing. Amado Espinosa Hidalgo

Junio 2007

La mayor parte de las ideas fundamentales de la ciencia son esencialmente sencillas y, por regla general, pueden ser expresadas en un lenguaje comprensible para todos.

Albert Einstein.

DECLARACIÓN DE AUTORÍA

Por este medio declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas (UCI) para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmamos el presente trabajo de diploma a los ___ días del mes de _____ del año _____.

Yadira Ramírez Rodríguez
Firma del autor

Juan de Jesús Grave de Peralta Rodríguez
Firma del autor

Amado Espinosa Hidalgo
Firma del tutor

A mi mamá Magalys y a mi abuela Aleyda que desde lejos me dieron ánimo y apoyo para lograr uno de mis mayores sueños y de ellas en esta vida, verme graduada.

A mi tía Mayda, a mi tío Alexis y a mi primo Alexito que estuvieron dándome su apoyo a lo largo de toda mi carrera aquí en La Habana.

A Mary, a Esther y a Fefa que se han comportado conmigo como mi propia familia, apoyándome y dándome su apoyo en todo momento.

A mi novio Alexey que estuvo estos cincos años de mi vida dándome aliento para poder llegar a la recta final de esta carrera.

Yadira

A mis padres Enma y Peralta, por haberme dado toda la confianza y amor en mi vida, que han sabido guiarme por el buen camino y sembrando en mi los valores que me caracterizan, ya que son ejemplos de firmeza, audacia e inteligencia, dándome su apoyo para graduarme.

A mi hermana Yuneidi le dedico con cariño este trabajo para que le sirva de ejemplo y guía para su carrera.

A mi novia Aniolis por haberme comprendido, ayudado, dándome su apoyo y esperanza, por haber esperado tanto para ver llegar este momento y compañera de mi vida.

Peralta

A la profesora de Seminario de Tesis Yenieris que siempre estuvo dispuesta ayudarnos antes los dilemas que nos hemos encontrado en el transcurso de del desarrollo de la investigación realizada.

A Abdelaziz que nos ayudó desinteresadamente en el diseño y logo de nuestra aplicación.

A Daylin que prácticamente sin conocerla puso su granito de arena en la introducción de nuestra aplicación, brindándonos ideas y sugerencias para la misma.

A Juan Carlos que estuvo dispuesto ayudarnos antes dificultades que se nos presentaron en el desarrollo de la aplicación.

A Yurdenia que siempre está dispuesta ayudarnos en cualquier tema relacionada con diseño.

A todos nuestros compañeros y compañeras de aula (y de año) por estar con nosotros todos estos años.

A mi mamá. A mi abuela. A mi novio.

Yadira

A mi mamá. A mi papá. A mi hermana. A mi novia.

Peralta

A la Revolución y en especial al Comandante en Jefe Fidel Castro Ruz por habernos dado todas las condiciones y posibilidades de llevar a cabo esta labor.

Gracias

RESUMEN

La Universidad de las Ciencias Informáticas (UCI) posee una infraestructura tecnológica, que permite la creación de una ciudad digital, donde su principal objetivo es la automatización de los procesos involucrados en el quehacer cotidiano. El correcto funcionamiento del proceso docente-educativo es de vital importancia para un desempeño triunfante del centro; es por tal motivo que surge la necesidad de desarrollar una herramienta para la administración del proceso de desarrollo de software para proyectos docentes de la asignatura de Ingeniería de Software con el objetivo de automatizar todo el proceso docente de la asignatura de ingeniería de software. El presente trabajo tiene como objetivo modelar un sistema informático para la automatización de los procesos de administración, planificación y evaluación en los proyectos docentes de la asignatura de Ingeniería de Software, proporcionándole un control por parte de los profesores de las actividades que desarrollan los estudiantes para llevar a cabo sus proyectos docentes en dicha asignatura.

La puesta en vigor de este sistema permitirá dar solución a problemas actuales existentes en el proceso docente de la asignatura de ingeniería de software, lo que se traduce en un control estricto por parte del profesor en las tareas de asignación de roles, creación de artefactos según el rol otorgado, la evaluación de artefactos, entre otras actividades relacionadas con el proceso de desarrollo de software realizadas por los estudiantes.

Para su desarrollo se siguieron los pasos que proponen el Proceso Unificado del Software. Está implementado en el lenguaje de programación PHP, servidor Web Apache y como gestor de base de datos se utilizó MySQL.

INDICE

INTRODUCCION

La introducción de las Tecnologías de la Información y las Comunicaciones (TIC) en el sistema educativo marca una nueva etapa de desarrollo que responda a las necesidades del proceso de formación. La integración de las TIC al proceso docente-educativo es una realidad y una necesidad social impuesta por el desarrollo tecnológico en todo el mundo.

Cuba enfrenta el reto de informatizar su sociedad con el propósito de integrarse plenamente a la infraestructura de la información y hacer uso óptimo de las nuevas tecnologías, lo que permitirá que ninguna esfera económica o social pueda pensar en el desarrollo sino no es con la presencia de las grandes tecnologías. La Universidad de las Ciencias Informáticas no se encuentra ajena a este desarrollo.

La Universidad de las Ciencias Informáticas, primera universidad creada al calor de La Batalla de Ideas, con nuevas concepciones de lo que debe ser una universidad y con el principal objetivo de convertirse en una universidad de excelencia como dijera nuestro comandante llega a sus 5 primeros años de existencia, por lo que una de las tareas inmediatas lo constituye consolidar y mejorar cada día más el proceso docente educativo que necesita una universidad de este tipo. Como cualquier sistema que se inicia presenta algunas deficiencias que pueden ser mejoradas a través de sistemas informáticos, nuevas ideas y el constante control y evaluación del mismo.

Actualmente en la Universidad de las Ciencias Informáticas el proceso docente de la asignatura de Ingeniería de Software se realiza a través de proyectos de curso; lo cual trae consigo que no exista un ambiente que facilite el intercambio de artefactos y de evaluación entre el profesor y el estudiante, éstos se realizan mediante el correo. Además se presenta dificultad a la hora de dividir el proyecto o las tareas del proyecto debido a la naturaleza de la metodología que es bidimensional. Al mismo tiempo no existe un mecanismo que permita las tareas de adicionar integrantes a un proyecto, la asignación de roles, la creación de artefactos según el rol otorgado, el control de tareas asignadas, entre otras actividades vinculados al proceso de desarrollo de software, éstas se realizan manualmente y no se guían por pasos para hacer más eficiente el desarrollo de las mismas, lo que trae consigo que no haya un control por parte del profesor del trabajo que realizan los estudiantes.

Por tanto el **problema** a resolver queda formulado a modo de interrogante de la siguiente forma: ¿Cómo erradicar las deficiencias en la organización, planificación y evaluación de los

proyectos docentes de la asignatura de Ingeniería de Software en la Universidad de las Ciencias Informáticas?

El **objeto de estudio** lo constituye el proceso docente educativo de la asignatura de Ingeniería de Software.

Y el **campo de acción** de acción que abarca este trabajo, es el desarrollo de proyectos docentes en la asignatura de Ingeniería de Software.

La **idea a defender** consiste en: con la creación de una herramienta informática que apoye el proceso de desarrollo de software de los proyectos docentes, se mejorará el proceso docente educativo de la asignatura de Ingeniería de Software.

El **objetivo general** de esta investigación es: Modelar un sistema informático para la automatización de los procesos de administración, planificación y evaluación en los proyectos docentes de la asignatura de Ingeniería de Software.

Con vista al cumplimiento del objetivo propuesto se plantea la realización de las siguientes **tareas**:

- Estudiar el proceso docente de la asignatura de ingeniería de software.
- Analizar los sistemas actuales que permiten la administración y la planificación de proyectos docentes.
- Estudiar en profundidad la metodología de desarrollo de software RUP, haciendo uso de la ayuda extendida del Rational Rose.
- Seleccionar la Metodología de Análisis y Diseño de Sistemas Informáticos, que facilite la creación y garantice la calidad del sistema.
- Seleccionar las herramientas para llevar a cabo el proyecto y la elección de la plataforma en la que se desarrollará la aplicación, fundamentando dicha elección.
- Implementar una herramienta que permita dar solución al objetivo propuesto.

El presente documento se estructura en resumen, introducción, tres capítulos de contenidos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía consultada, glosario de términos y anexos donde se incluye todo lo relacionado con el trabajo investigativo realizado.

En el Capítulo 1: Fundamentación teórica se describe algunos de los conceptos que están vinculados con la solución al problema que se va a resolver. Además se hace referencia a las tecnologías existentes en la actualidad que se deben considerar para hacer la selección de

aquellas que se van a utilizar en la realización de la aplicación propuesta. Finalmente se plantea dicha selección a modo de propuesta, con cada uno de sus aspectos debidamente fundamentado.

En el Capítulo 2: Características del sistema, se aborda lo referente al funcionamiento del negocio: la descripción del mismo y las mejoras que se proponen, así como sistemas automatizados que existen vinculados a este trabajo. Se describen los actores y trabajadores del negocio y se exponen el diagrama de casos de uso del negocio, el diagrama de actividad y el modelo de objetos, así como los requerimientos funcionales y no funcionales que debe cumplir la aplicación. También se exponen los actores y el diagrama de casos de uso del sistema con una descripción de cada caso de uso.

En el Capítulo 3: Construcción de la solución propuesta, se modelan los diagramas de clases del diseño con estereotipos Web y los diagramas de secuencia para cada flujo de evento, ambos describiendo la forma en que se implementará la aplicación; además del diagrama de clases persistentes y el modelo de datos. Se hará referencia a los principios de diseño y al modelo de implementación mediante el diagrama de despliegue y los diagramas de componentes.

1

CAPÍTULO

FUNDAMENTACION TEORICA

Introducción

En el presente capítulo se describen los conceptos fundamentales que sirven de punto de partida en este trabajo entre los que se incluyen proceso docente-educativo, ingeniería de software, proyecto de desarrollo de software, administración de proyectos, planificación de proyectos y proceso de desarrollo de software. Además se hace un análisis de las tecnologías y tendencias que existen en la actualidad a nivel mundial y que pudieran ser útiles en el desarrollo de la propuesta de solución. Se tienen en cuenta los servidores Web, los lenguajes de programación para la Web, los Sistemas Gestores de Bases de Datos mayormente utilizados a escala internacional, las distintas metodologías de desarrollo de software, así como JavaScript y XML asíncronos (Ajax) y las hojas de estilo en cascada (CSS). Finalmente, se seleccionan las más apropiadas teniendo en cuenta que las que se utilicen deben garantizar el cumplimiento de los intereses de los usuarios finales y de la universidad en general.

1.1 Proceso docente-educativo.

“El proceso docente-educativo, se desarrolla con el fin de alcanzar los objetivos propuestos, de ahí que la lógica que se sigue no responde ni a la lógica de la ciencia ni a la del programa, sino a la asimilación de los contenidos por los estudiantes conjuntamente con el desarrollo de sus capacidades cognoscitivas e independencia”(DJOHNSON 2006). Los eslabones fundamentales de este se sintetizan en:

1.6.7Planificación y organización del proceso.

1.6.7Motivación y comprensión del contenido.

1.6.7Sistematización de los contenidos.

1.6.7Evaluación del aprendizaje.

“En el proceso docente-educativo se expresa la unión entre lo instructivo y lo educativo, desde una relación dialéctica entre, la instrucción: que es el resultado de de la asimilación por el

estudiante del contenido de la enseñanza y la educación: que es la formación en el estudiante de los rasgos más estables de su personalidad"(GONZÁLEZ 2007).

1.2 Ingeniería de software.

"La Ingeniería de Software es una tecnología multicapa en la que, según Pressman, se pueden identificar: los métodos (indican cómo construir técnicamente el software), el proceso (es el fundamento de la Ingeniería de Software, es la unión que mantiene juntas las capas de la tecnología) y las herramientas (soporte automático o semiautomático para el proceso y los métodos)"((DIS) 2005).

Esta rama está vinculada con diversas áreas de de la Informática y de las Ciencias de la Computación, ejemplo de ellas lo constituyen la construcción de compiladores, sistemas operativos o desarrollos de Intranet/Internet, abordando todas las fases del ciclo de vida del desarrollo de cualquier tipo de sistemas de información y aplicables a una infinidad de áreas tales como: negocios, investigación científica, medicina, producción, logística, banca, control de trafico, meteorología, el mundo del derecho, la red de redes Internet, redes Intranet y Extranet, entre otras.

1.3 ¿Qué es un proyecto de desarrollo de software?

"Un proyecto es un esfuerzo temporal emprendido para crear un producto o un servicio único. Así, el resultado final buscado puede diferir con la misión de la organización que la emprende, ya que el proyecto tiene determinado específicamente un plazo y el esfuerzo es temporal"(WIKIMEDIA FOUNDATION 2006d).

Partiendo de lo anterior se puede decir que un proyecto de desarrollo de software es aquel que tiene como fin realizar un producto o mejorar uno existente, con una fecha de inicio y finalización determinados, así como, alcance, presupuesto y recurso asignado para el logro de los objetivos propuestos.

Todos los proyectos tienen ciertas características en común.

- Todos ellos tienen un inicio y un fin.
- Todos los proyectos son únicos. Pueden ser similares a proyectos anteriores, pero son únicos en términos de su programación, recursos, ambiente de negocios, etc.
- El resultado de ejecutar proyectos es la creación de uno o más entregables.
- Los proyectos también tienen recursos asignados – ya sea de tiempo completo o tiempos parciales o ambos.

- Existe un objetivo claro.
- Se puede identificar un conjunto de tareas.
- Necesaria la intervención de especialistas.
- Existen limitaciones en los recursos.
- Se requiere un nivel de calidad y una planificación.

1.4 Administración de proyectos.

La administración de proyectos es el proceso de combinar sistemas, técnicas y personas para completar un proyecto dentro de las metas establecidas de tiempo, presupuesto y calidad, es decir, es guiar los procesos dentro de la organización para obtener los resultados deseados.

La administración de proyectos enseña que para alcanzar el objetivo deseado el proyecto debe seguir un proceso específico, el cual se conoce como el ciclo de vida del proyecto.

Las etapas del ciclo de vida son:

- Factibilidad: En esta etapa se conocen los recursos financieros con los que se cuenta para el proyecto, se establecen presupuestos totales y se hace una organización preliminar. Se aplican estudios de factibilidad para saber si se puede resolver el problema o no y al término de esta etapa hay una decisión formal de continuar o no con el proyecto.
- Diseño: Es muy parecida a la etapa de factibilidad en la que se refiere a la organización y a la administración, pero en esta se detalla mejor el presupuesto, la calendarización y el financiamiento que le otorgan al proyecto.
- Producción: Se realiza en todas las actividades concernientes a la creación del proyecto. Esta etapa se caracteriza por ser totalmente diferente a las anteriores, ya que la fase de factibilidad y la de diseño son de carácter evolutivo, mientras que la de producción es de alto grado mecanicista.
- Culminación y puesta en marcha: En esta etapa se hacen pruebas finales al proyecto. También se da mantenimiento periódicamente verificando que no tenga fallas lógicas.

1.5 Planificación de proyectos.

La planificación de proyectos distribuye el proyecto en tareas y estima el tiempo y los recursos requeridos para completar cada tarea, organiza las tareas de forma concurrente para hacer mejor uso de la fuerza laboral, minimiza dependencias entre tareas para evitar retrasos debidos

a que una tarea espere a la terminación de otra y depende de la intuición y experiencia de los administradores.

“La planificación de proyectos de software tiene como objetivo proporcionar un marco de trabajo que permita al gestor hacer estimaciones razonables de recursos, costos y planificación temporal. Estas estimaciones se hacen dentro de un marco de tiempo limitado al comienzo de un proyecto software, y debería actualizarse regularmente a medida que progresa el proyecto. Además, las estimaciones deberían definir los escenarios del ¨mejor caso¨ y ¨peor caso¨ de forma que los resultados del proyecto puedan limitarse”(CONCEPCIÓN 2005).

El objetivo de la planificación de proyectos es establecer y mantener planes que define las actividades del proyecto.

Las tareas que conlleva la planificación de proyectos son:

- Desarrollar un plan inicial del proyecto.
- Establecer una relación adecuada con todas las personas involucradas en el proyecto.
- Obtener compromiso con el plan.
- Mantener el plan durante el desarrollo del proyecto.

El plan incluye estimación de los elementos de trabajo y tareas, recursos necesarios, negociación de compromisos, establecimiento de un calendario, e identificación y análisis de los posibles riesgos que pueda tener el proyecto.

El plan de proyectos es un herramienta de trabajo viva que se debe de actualizar con mucha frecuencia ya que los requisitos cambiarán, habrá que reestimar, habrá riesgos que desaparezcan y otros que surjan nuevos, habrá que tomar acciones correctivas.

1.6 Proceso de desarrollo de software.

El proceso de desarrollo de software "es aquel en que las necesidades del usuario son traducidas en requerimientos de software, estos requerimientos transformados en diseño y el diseño implementado en código, el código es probado, documentado y certificado para su uso operativo"(JACOBSON 1998). Concretamente "define quién está haciendo qué, cuándo hacerlo y cómo alcanzar un cierto objetivo"(JACOBSON 1998).

El proceso de desarrollo de software requiere por un lado un conjunto de conceptos, una metodología y un lenguaje propio. A este proceso también se le llama el ciclo de vida del software que comprende cuatro grandes fases: concepción, elaboración, construcción y

transición. La concepción define el alcance del proyecto y desarrolla un caso de negocio. La elaboración define un plan del proyecto, especifica las características y fundamenta la arquitectura. La construcción crea el producto y la transición transfiere el producto a los usuarios.

1.7 La política de migración hacia software libre.

En la actualidad se ha venido observando como el mundo casi entero está haciendo uso del software libre y la Universidad de las Ciencias Informáticas no se queda atrás. Cada día se promociona más la migración de las instituciones, empresas, etc., que presentan licencia comercial hacia el software libre.

La UCI como una institución avanzada en el campo de la informática está prácticamente obligada a llevar a cabo, y cuánto antes mejor, esta migración. El presente trabajo a realizar parte de esa premisa y se propone la construcción de un sistema que satisfaga las necesidades que lo originaron, haciendo uso de herramientas y tecnologías libres.

1.8 Tecnologías actuales a considerar.

El desarrollo de una herramienta informática lleva consigo un estudio preliminar de las diferentes metodologías y herramientas que existen a nivel internacional; luego de haber finalizado su estudio, llevar a cabo su elección de acuerdo a sus características y condiciones y comenzar a desarrollar un producto con la calidad requerida.

1.8.1 Aplicaciones Web.

“Las aplicaciones Web son una especialización y concreción de de las aplicaciones cliente/servidor”((DTP) 2003). Estas aplicaciones generan dinámicamente una serie de páginas en un formato estándar, que está soportado por navegadores Web comunes como HTML o XHTML. Se utilizan lenguajes interpretados del lado del cliente, tales como JavaScript, para añadir elementos dinámicos a la interfaz de usuario, unidos a los lenguajes script que corren al lado del servidor tales como ASP, Java, PHP, Perls, etc., para desarrollar la lógica del negocio dentro del servidor y permitir el acceso a los sistemas gestores de base de datos.

Los componentes de la arquitectura Web son: el servidor Web, la red física que permite la comunicación y un navegador o cliente.

La aplicaciones Web se estructuran como una aplicación de tres-capas comúnmente, la primera capa la compone un navegar Web, la segundo un motor usando alguna tecnología Web dinámica (ejemplo: CGI, PHP, Java Servlets o ASP) y una base de datos como última, siendo ésta su forma más común.

En la actualidad el desarrollo de una aplicación Web permite publicar: un catálogo electrónico de productos, manejo de inventarios, órdenes de compra, publicación de información con acceso restringido a ciertos usuarios, actualización y mantenimiento del sitio Web y en general, permite publicar cualquier tipo de información que se pueda incorporar a una base de datos; además los clientes sólo necesitan un navegador y para hacer uso de las aplicaciones Web no se tiene que instalar ningún componente de software adicional Por tanto se propone que el sistema a desarrollar consista en una aplicación Web.

1.8.2 La seguridad de las trasmisiones en la Web.

La seguridad indica que un sistema está libre de todo peligro, daño o riesgo. Actualmente la seguridad en las aplicaciones Web es un elemento de gran importancia para el mantenimiento de la confidencialidad, integridad y disponibilidad de la información manejada, ya que si es mal implementada puede dar pie a que piratas informáticos penetren en la red y alteren o deterioren la información que se utiliza y transmita.

“La seguridad es una propiedad dinámica: el que hoy seamos seguros no implica que mañana lo sigamos siendo”(HUERTA 2005).

A raíz de todo esto surgieron tecnologías que persiguen mejorar la seguridad de todas estas comunicaciones.

Protocolo SSL (Secure Sockets Layer)

El protocolo SSL, uno de los más seguros para la Web “fue desarrollado por Netscape para permitir confidencialidad y autenticación en Internet. SSL es una capa por debajo de HTTP y tal como lo indica su nombre esta a nivel de socket por lo que permite ser usado no tan solo para proteger documentos de hipertexto sino también servicios como FTP, SMTP, TELNET entre otros”(SEPULVEDA 2006).

La idea que persigue SSL es cifrar los datos intercambiados entre el servidor y el cliente con un algoritmo de cifrado simétrico, típicamente el RC4 o IDEA, y cifrar la clave de sesión de RC4 o IDEA mediante un algoritmo de cifrado de clave pública, típicamente el RSA. La clave de

sesión es la que se utiliza para cifrar los datos que vienen del y van al servidor seguro. Con funciones hash utiliza el algoritmo criptográfico MD5.

Este sistema de seguridad ideado para acceder a un servidor garantizando la confidencialidad de los datos mediante técnicas de encriptación modernas proporcionando cifrado de datos, autenticación de servidores, integridad de mensajes y, opcionalmente, autenticación de cliente para conexiones TCP/IP.

1.8.3 Servidor Web.

"Un servidor Web es un programa que implementa el protocolo HTTP (hypertext transfer protocol). Este protocolo está diseñado para transferir lo que llamamos hipertextos, páginas Web o páginas HTML (hypertext markup language): textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de música"(WIKIMEDIA FOUNDATION 2006f).

En la actualidad existen una serie de servidores Web, entre los más importante tenemos a: Apache, IIS (Internet Information Services) y Cherokee.

IIS (Internet Information Services).

IIS, Internet Information Services (o Server), es una serie de servicios para los ordenadores que funcionan con Windows y simplifica la publicación de información en Internet o en la Intranet. Inicialmente era parte de Option Pack para Windows NT, más tarde fue integrado en otros sistemas operativos de Microsoft, como Windows 2000 o Windows Server 2003; Windows XP Profesional posee una versión ilimitada del mismo. Los servicios que ofrece son: FTP, SMTP, NNTP y HTTP/HTTPS.

IIS incluye una amplia gama de funciones administrativas para controlar sitios Web y el servidor Web. Con funciones de programación como páginas Active Server (ASP), puede crear e implementar aplicaciones Web flexibles y escalables.

Su gran desventaja: es propiedad de Microsoft Corporation y no es un software libre.

Apache (Servidor HTTP Apache).

"El Servidor HTTP Apache es un servidor HTTP de código abierto para plataformas Unix, Windows y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual"(WIKIMEDIA

FOUNDATION 2006e). Se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation.

Entre sus características resaltan:

- Mensajes de error altamente configurables
- Bases de datos de autenticación
- Negociado de contenido
- Encriptación de 128 bits.
- Código fuente completo.
- Sencilla administración basada en la configuración de un único archivo.

En la actualidad (2006), Apache es el servidor HTTP más usado, siendo el servidor HTTP del 68% de los sitios Web en el mundo y creciendo aún su cuota de mercado.

Cherokee.

Cherokee es un servidor Web libre ideado por Álvaro López Ortega, un madrileño de 26 años que trabaja en Dublín, con código abierto, creado con el objetivo de que sea accesible por el mayor número de personas y exista una comunidad de usuarios y desarrolladores que lo soporten y mejoren.

Velocidad, facilidad de uso y modularidad son las principales características del nuevo servidor. Su diseño es muy modular y eficiente; hasta puede usarse para servir en baratos sistemas empotrados por su bajo consumo de recursos.

En la actualidad Cherokee funciona con Linux, BSD, Solaris, MacOS y, muy pronto también con Windows.

Selección del servidor Web a utilizar.

Se seleccionó el Servidor HTTP Apache para utilizarlo en la propuesta de solución, por todas las características expuestas anteriormente es considerado uno de los servidores Web líder en el mercado ofreciendo una perfecta combinación entre estabilidad y sencillez. Además posee un coste gratuito, funciona en cualquier plataforma, presenta gran fiabilidad y extensibilidad que lo convierten en una herramienta potente y muy configurable.

1.8.4 Lenguajes de Programación para la Web.

"Los lenguajes de programación son una técnica estándar de comunicación que permite expresar las instrucciones que han de ser ejecutadas en una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen un lenguaje informático"(WIKIMEDIA FOUNDATION 2006b). Entre los distintos lenguajes de programación para la Web que existen, se destacan dos grupos, los cuales se diferencian entre sí, por el lugar que ocupan en la arquitectura Cliente/Servidor características de los sistemas Web. El primer grupo está compuesto por los lenguajes de programación al lado del cliente, entre los que se destacan: Javascript, JScript, Vbscript, que son las funciones que se corren en el navegador y no necesitan hacer peticiones al servidor. El segundo grupo lo componen los lenguajes de programación al lado del servidor, como ejemplos tenemos: ASP, JSP, PERL, PHP, son las funciones que corren y necesitan hacer pedido al servidor de de alguna información.

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 Internet Information Server (IIS).

ASP ha pasado por cuatro iteraciones mayores, ASP 1.0 (distribuido con IIS 3.0), ASP 2.0 (distribuido con IIS 4.0), ASP 3.0 (distribuido con IIS 5.0) y ASP.NET (parte de la plataforma .NET de Microsoft). Las versiones pre-.NET se denominan actualmente (desde 2002) como ASP clásico.

En el último ASP clásico, ASP 3.0, hay seis objetos integrados disponibles para el programador, Application, ASPError, Request, Response, Server y Session. Cada objeto corresponde a un grupo de funcionalidades frecuentemente usadas y útiles para crear páginas Web dinámicas.

Las páginas pueden ser generadas mezclando código de scripts del lado del servidor (incluyendo acceso a base de datos) con HTML y código del lado del servidor.

Desde 2002, el ASP clásico está siendo reemplazado por ASP.NET, que, entre otras cosas, reemplaza los lenguajes interpretados como VBScript o JScript por lenguajes compilados a código intermedio (llamado MSIL o Microsoft Intermediate Language) como Visual Basic, C#, o cualquier otro lenguaje que soporte la plataforma .NET.

JSP (Java Pages Server).

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 permite 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 (.class) 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.

Sin embargo JSP no se puede considerar un script al 100% ya que antes de ejecutarse el servidor Web compila el script y genera un servlet, por lo tanto, se puede decir que aunque este proceso sea transparente para el programador no deja de ser una aplicación compilada. La ventaja de esto es algo más de rapidez y disponer del API de Java en su totalidad.

Perl (Practical Extracting and Reporting Language).

Lenguaje de programación de propósito general originalmente desarrollado para la manipulación de texto y que ahora es utilizado para un amplio rango de tareas incluyendo administración de sistemas, desarrollo Web, programación en red, desarrollo de GUI y más.

Sus principales características son que es fácil de usar, soporta tanto la programación estructurada como la programación orientada a objetos y la programación funcional, tiene incorporado un poderoso sistema de procesamiento de texto, una enorme colección de módulos y cientos de bibliotecas disponibles.

Lenguaje de programación que permite resolver los problemas fáciles fácilmente, y resolver también problemas difíciles. Es rápido hacer una pequeña aplicación Web.

Las ventajas principales son: se trata de un lenguaje muy maduro, que lleva mucho tiempo funcionando, y que tiene cientos de bibliotecas operativas y listas para ser usadas. Por otra parte, si uno tiene paciencia puede encontrar el 90% del trabajo que tenía que hacer ya hecho por otra persona, y el código que tiene que escribir es sólo el 10% más trivial.

PHP (Hypertext Pre-processor).

Lenguaje interpretado usado para la creación de aplicaciones para servidores, o creación de contenido dinámico para sitios Web.

Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, Postgres, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite; lo cual permite la creación de aplicaciones Web muy robustas.

PHP también tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos tales como UNIX (y de ese tipo, como Linux), Windows y Mac OS X, y puede interactuar con los servidores de Web más populares ya que existe en versión CGI, módulo para Apache, e ISAPI.

Sus principales usos son: programación de páginas Web dinámicas, habitualmente en combinación con el motor de base datos MySQL, aunque cuenta con soporte nativo para otros motores, incluyendo el estándar ODBC (Open DataBase Connectivity), lo que amplía en gran medida sus posibilidades de conexión, programación en consola, al estilo de Perl o Shell scripting y creación de aplicaciones gráficas independientes del navegador, por medio de la combinación de PHP y GTK (GIMP Tool Kit), lo que permite desarrollar aplicaciones de escritorio en los sistemas operativos en los que está soportado.

Entre sus ventajas encontramos que el lenguaje PHP cuenta con capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, lee y manipular datos desde diversas fuentes, incluyendo datos que pueden ingresar los usuarios desde formularios HTML, tiene la capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones), posee una amplia documentación en su página oficial, permite las técnicas de Programación Orientada a Objetos y nos permite crear los formularios para la Web.

Actualmente está en su última versión la 5.2.0.

Selección del lenguaje a utilizar.

PHP, debido a todas las ventajas que presenta; multiplataforma y sobre todo un lenguaje de programación libre, por lo que se presenta como una alternativa de fácil acceso para todos e incluye el estándar ODBC lo que sus posibilidades de conexión en gran medida. Por ello se propone su uso como lenguaje del lado del servidor.

Como lenguaje al lado del cliente se propone Java Script, que es un lenguaje de programación interpretado y dirigido por eventos, no requiere de compilación. Permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones, estructuras de datos complejas, etc. Además, Javascript pone a disposición del programador todos los elementos que forman la página Web, para que éste pueda acceder a ellos y modificarlos dinámicamente.

Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado.

1.8.5 Hojas de Estilo en Cascada (Cascade Style Sheets, CSS).

"Las hojas de estilo en cascada (Cascading Style Sheets, CSS) son un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML)"(WIKIMEDIA FOUNDATION 2006a).

Entre sus ventajas resaltan, una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o incluso a elección del usuario, los navegadores le permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio Web, con el CSS no es necesario usar imágenes invisibles para hacer sangría(la propiedad text-indent se encarga de eso) o usar una tabla para ubicar un elemento en un determinado lugar de la pantalla(las CSS permite posicionar con precisión cualquier elemento), ésta y otras son algunas de las ventajas que caracterizan a la hojas de estilo, que aunque potente son sencillas y fáciles de usar.

Las hojas de estilo representan un gran paso adelante para la Web. En resumen, las hojas de estilo permiten separar el formato visual de las páginas de contenido. Por la ventaja que representan, se propone su utilización en la propuesta de solución.

1.8.6 Ajax (JavaScript y XML asíncronos).

Ajax (JavaScript y XML asíncronos) "es una técnica de desarrollo web para crear aplicaciones interactivas. Éstas se ejecutan en el cliente, es decir, en el navegador del usuario, y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la interactividad, velocidad y usabilidad en la misma"(WIKIMEDIA FOUNDATION 2007).

No es una tecnología, sino la unión de varias tecnologías que juntas pueden lograr cosas realmente impresionantes. Ajax incorpora:

- Presentación basada en estándares usando XHTML y CSS para el diseño que acompaña a la información.
- Exhibición e interacción dinámicas usando el Document Object Model accedido con un lenguaje de scripting por parte del usuario.
- Intercambio y manipulación de datos usando XML and XSLT ;
- Recuperación de datos asincrónica usando XMLHttpRequest ;
- Y JavaScript poniendo todo junto.

En un desarrollo basado en AJAX es posible enviar peticiones al servidor enviando solo la información necesaria. El servidor recibirá la petición, la procesará y regresarán sólo los datos necesarios a la página original. Con todo ello, lógicamente, ganamos bastante en ancho de banda y por tanto, en tiempo de respuesta del servidor. Al final, en el performance o funcionamiento de los desarrollos también se verá mejorado.

1.8.7 Sistemas Gestores de Bases de Datos (SGBD).

“Los Sistema de gestión de base de datos son un tipo de software muy específico, dedicado a servir de interfaz entre la Base de datos y el usuario, 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. En los textos que tratan este tema, o temas relacionados, se mencionan los términos SGBD y DBMS, siendo ambos equivalentes, y acrónimos, respectivamente, de Sistema Gestor de Bases de Datos y DataBase Management System, su expresión inglesa”(WIKIMEDIA FOUNDATION 2006g).

En la actualidad existen múltiples SGBD divididos en SGBD libres y SGBD comerciales. Dentro de los SGBD comerciales encontramos a: Microsoft SQL Server, Microsoft Access, Oracle, entre otros, siendo **Oracle** considerado como una de los Sistemas Gestores de Base de Datos más completos. Entre sus características se encuentran: el soporte de transacciones, su gran estabilidad y escalabilidad, así como que es un sistema multiplataforma, entre otras ventajas.

La tecnología Oracle se encuentra prácticamente en todas las industrias alrededor del mundo. Aunque hasta hace poco su dominio en el mercado de servidores empresariales era casi total, recientemente sufre la competencia de recientemente sufre la competencia del Microsoft SQL Server de Microsoft y de la oferta de otros SGBD libres.

Microsoft SQL Server es un SGBD relacionales basada en el lenguaje SQL, capaz de poner a disposición de múltiples usuarios grandes cantidades de información de forma simultánea. Sus características más destacadas son la gran seguridad y estabilidad que presenta. Para el desarrollo de aplicaciones más complejas (tres o más capas), Microsoft SQL Server incluye interfaces de acceso para la mayoría de las plataformas de desarrollo, incluyendo .NET. A pesar de todas sus ventajas no es multiplataforma y al igual que Oracle son Sistemas Gestores de Base de Datos no son libres.

Entre los SGBD libres encontramos a: PostgreSQL, MySQL, Firebird, entre otros, que pueden ser usados y modificados por cualquier persona y su código fuente puede ser bajado sin tener que pagar.

MySQL "es un sistema de gestión de base de datos, multihilo y multiusuario con más de seis millones de instalaciones"(SCHUMACHER and LENTZ 2006).

Entre sus ventajas se destaca, su gran portabilidad entre sistemas, posibilita manipular datos del orden de seis mil tablas y alrededor de cincuenta millones de registros, y hasta 32 índices por tabla, dispone de APIs para múltiples plataformas, permite manejar multitud de tipos para columnas y registros de longitud fija o variable.

"MySQL es muy utilizado en aplicaciones web como MediaWiki o Drupal, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación Web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. MySQL es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones Web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones"(WIKIMEDIA FOUNDATION 2006c).

Este gestor de bases de datos es, probablemente, el gestor más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración.

Por otra parte se encuentra **PostgreSQL,** el cual es considerado ampliamente el motor de base de datos de código abierto más avanzado del mundo, es servidor de base de datos

relacional libre, liberado bajo la licencia BSD y además una alternativa a otros sistemas de bases de datos de código abierto), así como sistemas propietarios como Oracle o DB2.

Entre sus características se encuentran: tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL, aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas, ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multi-versión, soporte multi-usuario, transactions, optimización de consultas, herencia, y arrays, ésta son algunas de las ventajas de este potente SGBD.

Actualmente PostgreSQL corre en la mayoría de las plataformas Unix, en particular en GNU/Linux, a partir de la versión 8(beta) corre en sistemas basados en Windows NT, como WinXp, Win2000, Win2003.

Selección del SGBD a utilizar.

Según lo expuesto anteriormente se considera utilizar MySQL debido a ser un sistema gestor de base de datos rápido, ya que es usado en las pequeñas compañías y ser un sistema de código libre que se integra a la perfección con PHP (lenguaje de programación a utilizar) y por ser un sistema de código abierto.

1.8.8 Metodologías de Desarrollo de Software.

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos software, es decir, van indicando paso a paso todas las actividades a realizar para lograr el producto informático deseado, indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben de tener. Además detallan la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla.

Estas tecnologías se dividen en dos grandes grupos; las metodologías ligeras, en la que encontramos a XP (eXtreme Programming), Crystal Light Methods y metodologías pesadas como son, SW-CMM (SoftWare Capability Maturity Model), RUP (Rational Unified Process). Las primeras se basan en la idea de conseguir el objetivo común por medio de orden y documentación, mientras que las segundas tratan de lograrlo por medio de la comunicación directa e inmediata entre aquello que intervienen en el proceso. Analizaremos a continuación tres de las más conocidas con sus características, ventajas y desventajas.

Proceso Unificado de Rational (RUP).

Proceso Unificado de Rational, en inglés Rational Unified Process es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado (UML), constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos en la actualidad, pues está pensada para adaptarse a una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipo de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos. Es el resultado de la experiencia de más de 30 años de trabajo y los autores [James Rumbaugh, Grady Booch e Ivar Jacobson] confirman que es la solución al problema del software.

RUP se divide en cuatro fases (Inicio, Elaboración, Construcción, Transición) y presenta nueve flujos de trabajos, de ellos seis son de ingeniería (Modelamiento del negocio, Requerimientos, Análisis y Diseño, Implementación, Prueba, Instalación o Distribución) y tres de apoyo (Configuración y administración del cambio, Administración de proyectos, Ambiente).

Una particularidad de esta metodología es que, en cada ciclo de iteración, se hace exigente el uso de artefactos, siendo por este motivo, una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo del software y tiene la ventaja de venir acompañada de una potente herramienta que soporta todos los procesos básicos de RUP: Rational Rose Enterprise Edition 2003.

Programación Extrema (XP).

La metodología Programación Extrema, en inglés Extreme Programing consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. Ha sido diseñada para solucionar el eterno problema del desarrollo de software por encargo: entregar el resultado que el cliente necesita a tiempo.

La metodología se basa en:

- Pruebas Unitarias: se basa en las pruebas realizadas a los principales procesos, de tal manera que adelantándonos en algo hacia el futuro, podamos hacer pruebas de las fallas que pudieran ocurrir.

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

- Programación en pares: una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo.

Entre sus características podemos resaltar: los diseñadores y programadores se comunican efectivamente con el cliente y entre ellos mismos, los diseños del software se mantienen sencillos y libres de complejidad o pretensiones excesivas, el software es liberado en entregas frecuentes tan pronto como sea posible, los cambios se implementan rápidamente tal y como sea posible, es usada en proyectos de corto plazo, entre otras.

Con estas características no es sorprendente que XP sea la metodología más apropiada para un entorno caracterizado por requerimientos cambiantes originados por un mercado fluctuante y los propios avances de la tecnología y los negocios.

Desarrollo Guiado por la Funcionalidad (FDD)

La metodología Desarrollo Guiado por la Funcionalidad, en inglés Feature Driven Development, es un proceso ligero que se considera a medio camino entre RUP y XP. Está pensado para proyectos con tiempo de desarrollo relativamente cortos (menos de un año). Se basa en un proceso iterativo con iteraciones cortas que producen un software funcional que el cliente y la dirección de la empresa pueden ver y monitorizar.

Las iteraciones se deciden en base a features o funcionalidades (de ahí el nombre del proceso), que son pequeñas partes del software con significado para el cliente. Se divide en 5 partes para el desarrollo de un proyecto: desarrollo de un modelo general, construcción de la lista de funcionalidades, plan de releases en base a las funcionalidades a implementar, diseñar en base a las funcionalidades, implementar en base a las funcionalidades.

En el proceso de implementar la funcionalidad se contemplan la preparación y ejecución de pruebas, así como revisiones del código e integración de las partes que componen el software. Define métricas para seguir el proceso de desarrollo de la aplicación, útiles para el cliente y la dirección de la empresa, y que pueden ayudar, además para conocer el estado actual del desarrollo, a realizar mejores estimaciones en proyectos futuros.

Selección de la metodología a utilizar.

En el caso particular de este proyecto, se decidió la utilización de la metodología RUP, por todas las ventajas de organización que brinda, por ser orientada a objetos, por dividir su trabajo

en roles, por estar preparada para desarrollar grandes y complejos proyectos a diferencia de XP y FDD que se utilizan para proyectos de corto plazo y por considerarse una metodología muy organizada.

1.8.9 Lenguaje Unificado de Modelado UML)

UML oficialmente se presenta cuando Rumbaugh, Booch y Jacobson unifican sus estudios con una semántica y notación, para lograr compatibilidad en el análisis y diseño orientado a objetos, permitiendo que los proyectos se asentaran en un lenguaje de modelado maduro, enfocando a los constructores de herramientas en producir características más útiles.

"El Lenguaje de Modelamiento Unificado (UML - Unified Modeling Language) es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software. UML entrega una forma de modelar cosas conceptuales como lo son procesos de negocio y funciones de sistema, además de cosas concretas como lo son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables"(CARO and KAHLER 1996).

El desarrollo de sistemas con UML siguiendo el proceso unificado incluye actividades específicas, cada una de ellas a su vez contienen otras subactividades las cuales sirven como una guía de cómo deben ser las actividades desarrolladas y secuenciadas con el fin de obtener sistemas exitosos; consecuentemente el desarrollo de los sistemas puede variar de desarrollador en desarrollador, de proyecto en proyecto, de empresa en empresa adoptando siempre un Proceso de Desarrollo.

1.8.10 Propuesta

Luego del análisis llevado a cabo, se puede plantear una propuesta que consiste en desarrollar una aplicación sobre plataforma Web, utilizando como lenguaje de programación del lado del servidor el PHP dada su portabilidad y eficiencia, y el JavaScript para lograr una interactividad con el usuario en el navegador. Se propone también la utilización de MySQL como SGBD, y de Apache como servidor Web. Como metodología de desarrollo se utilizará RUP junto con el Lenguaje Unificado de Modelado (UML) y la potente herramienta que soporta todos sus procesos básicos: Rational Rose Enterprise Edition. Para referenciar toda la bibliografía consultada se hizo uso del EndNote.

Conclusiones

En la realización de este capítulo se abordaron los conceptos como, proceso docente-educativo, proyecto de software, administración de proyectos, planificación de proyectos, ingeniería de software y proceso de desarrollo de software, los cuales servirán para que el lector tengo una idea general de lo que se quiere con la investigación realizada. Además se realizó un estudio detallado de los diferentes lenguajes de programación, sistemas gestores de base de datos, servidores Web, metodologías de desarrollo de software, de mayor impacto a nivel internacional, entre otras tecnologías como hojas de estilo, Ajax y la seguridad en las aplicaciones Web, los cuales sirvieron de punto de partida en la elección de las herramientas a utilizar para llevar a cabo el desarrollo de la herramienta informática que dará solución a la problémica existente. Proponiéndose como lenguaje de programación PHP, sistemas gestor de base de datos MySQL, servidor Web Apache y como metodología de desarrollo de software RUP.

2

CAPÍTULO

CARACTERISTICAS DEL SISTEMA

Introducción

En el presente capítulo se describe la propuesta de solución. Primeramente se hace referencia al estado actual que presenta el negocio a modelar, luego se identifican los actores, trabajadores y los casos de uso correspondientes, realizándose el modelo de casos de uso del negocio, el de objeto y la descripción y diagramas de actividades correspondientes a los casos de uso identificados. Se hace regencia a sistemas automatizados existentes y se elabora la propuesta de solución. Se plantean los requisitos funcionales y no funcionales de la aplicación a desarrollar. Se modela el sistema, identificando los actores y casos de uso del sistema, desarrollando el diagrama de casos de uso del sistema y describiendo dichos casos de uso.

2.1 Estado actual del negocio

En la Universidad de la Ciencias Informáticas el proceso docente de la asignatura ingeniería de software se realiza a través de proyectos finales, los cuales les son otorgados a los estudiantes al comienzo de la asignatura.

Para realizar este proceso de entrega del proyecto, el profesor divide el grupo en una cantidad x de estudiantes para formar los equipos de trabajo y así asignarle un proyecto final que será con el que trabajaran hasta concluida la asignatura.

Luego de haber asignado al equipo de trabajo el proyecto final, el profesor le otorga a cada integrante del equipo un rol a desempeñar dentro del proceso de desarrollo de software en el proyecto asignado. Seguido a esto el profesor planifica las iteraciones que van a tener los proyectos finales y define todos los flujos de trabajo a evaluar, así como las actividades y artefactos a realizar por cada estudiante según el rol que desempeña. Estas actividades

ejecutadas por el profesor, las realiza manualmente y no se guía por pasos para hacer más eficiente el desarrollo de las mismas.

Por otra parte el profesor no lleva el control de las tareas y artefactos asignados al estudiante según su rol, ya que una vez que el profesor otorga las actividades y artefactos a desarrollar, el estudiante se las envía por correo; luego el profesor la descarga, corrigiéndole los errores y mandándole las plantillas de los artefactos corregidos a través del correo unido con la nota que ha obtenido en ese cierre de iteraciones del proyecto final, lo que implica que no existe un ambiente que facilite el intercambio de artefactos y evaluación.

A raíz del lo expuesto anteriormente el problema a resolver queda formulado a modo de interrogante de la siguiente forma: ¿Cómo erradicar las deficiencias en la organización, planificación y evaluación de los proyectos docentes de la asignatura de Ingeniería de Software?

2.2 Modelo del negocio

El modelo del negocio es un modelo que describe los procesos de un negocio (casos de uso del negocio) y su interacción con elementos externos (actores), tales como socios y clientes, es decir, describe las funciones que el negocio pretende realizar y su objetivo básico es describir cómo el negocio es utilizado por sus clientes y socios.

2.2.1 Actores del negocio.

Los actores del negocio son aquellas personas o sistemas que obtienen un resultado de valor de uno o varios procesos del negocio. Los actores del negocio estudiado se definen en la siguiente tabla.

Tabla 2- 1 Actores del negocio.

Actores del Negocio	Justificación
Estudiante	El estudiante es el que inicia todas las acciones que dan comienzo a los procesos de negocio analizados, y al mismo tiempo es el principal beneficiado con el resultado de dichos procesos de negocio.

2.2.2 Trabajadores del negocio.

Los trabajadores del negocio son aquellas personas o sistemas que están involucrados en uno o más procesos del negocio, que participan en ellos, pero no obtienen ningún resultado de valor. Los trabajadores del negocio estudiado se definen en la siguiente tabla.

Tabla 2- 2 Trabajadores del negocio.

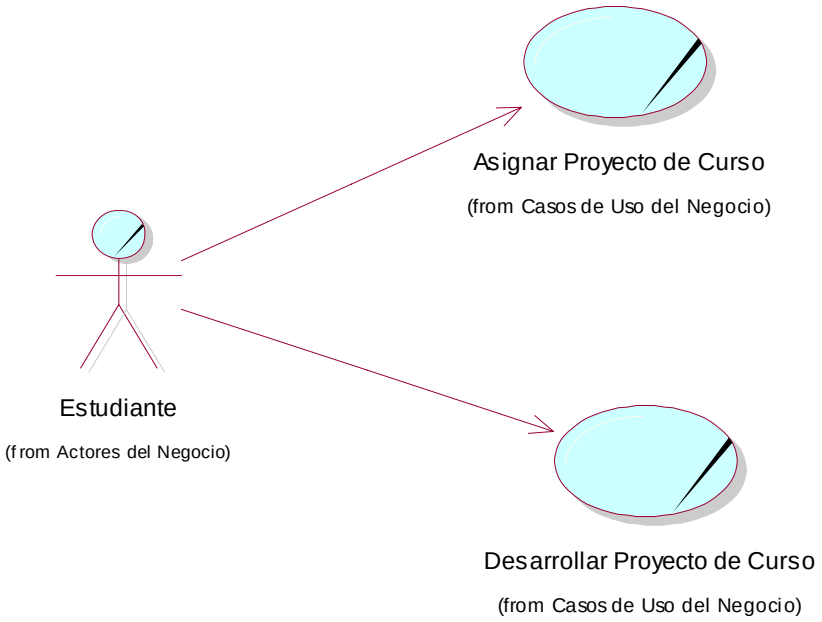
Trabajadores del Negocio	Justificación
Profesor	Es el encargado de orientar y revisar el proyecto de curso al estudiante. No se beneficia en ningún momento de las acciones ejecutadas en los procesos de negocio, sino que se limita a ejecutarlas.

2.2.3 Diagrama de casos de uso del negocio

El diagrama de casos de uso del negocio representa gráficamente a los procesos del negocio y su interacción con los actores del negocio. El modelo de casos de uso del negocio se define en la siguiente tabla.

Tabla 2- 3 Diagrama de casos de uso del negocio.

Diagrama de casos de uso del negocio



2.2.4 Descripción de los casos de uso del negocio.

En este epígrafe se describen los casos de uso del negocio.

Nombre del Caso de Uso	**01 Asignar Proyecto de Curso**
Actores	Estudiante
Propósito	Permitir al profesor crear las condiciones necesarias para que los estudiantes comiencen a trabajar en el proyecto de curso asignado.
Resumen	El caso de uso se inicia cuando el estudiante se presenta en el aula y el profesor confecciona los equipos de trabajo, se le realiza un test de roles a cada integrante del equipo, para saber el rol que va a desempeñar dentro del equipo de trabajo, luego el profesor entrega el tema a desarrollar y los artefactos que debe de realizar cada rol asignado. Finalizando así el caso de uso.
Curso Normal de los eventos	
Acciones del Actor	**Respuesta del proceso de negocio**
1. El estudiante se presenta en el aula.	1.1 El profesor analiza la matrícula del grupo para confeccionar los equipos de

	trabajo según los proyectos de curso.
	2.1 El profesor confecciona los equipos de trabajo.
	3.1 El profesor confecciona el test de roles.
	4.1 Reúne a los estudiantes por los equipos de trabajo confeccionados para realizar el test de roles.
	5.1 El profesor examina el test de roles al estudiante.
6. El estudiante realiza el test de roles, para saber que rol desempeñar dentro del equipo de trabajo que pertenece.	6.1 El profesor evalúa el test de roles realizado al estudiante. 6.2 Le comunica al estudiante el rol que va desempeñar dentro de su equipo de trabajo.
7. Recibe la noticia del rol a desempeñar en el equipo de trabajo asignado.	7.1 Luego el profesor otorga a cada equipo de trabajo el tema que van a desarrollar para su evaluación de final.
8. El estudiante toma el tema del trabajo final que va a desarrollar.	8.1 Le entrega a cada rol definido los artefactos a evaluar en el proyecto de curso.
9. El estudiante toma los artefactos que debe de entregar y le serán evaluados.	
10. El estudiante se retira del aula.	
Curso Alternativo de los eventos	

Prioridad	Crítico
Mejoras	
Otros	

Diagrama de actividad. CUN 01 Asignar Proyecto de Curso. (Anexos I, Figura. 2-1).

Nombre del Caso de Uso	**02 Desarrollar Proyecto de Curso**
Actores	Estudiante
Propósito	Permitir al profesor realizar el papel como revisor técnico, evaluándole al estudiante el cierre de las iteraciones de su proyecto de curso, y asignándole la nota final de la asignatura de Ingeniería de Software.
Resumen	

Curso Normal de los eventos	
Acciones del Actor	**Respuesta del proceso de negocio**
1. El estudiante se presenta en el aula para recibir las tareas en el curso de Ingeniería de Software.	1.1 El profesor adapta la metodología de desarrollo de software a utilizar en los proyectos de cursos de acuerdo al plan de estudio.
	2.1 Define los flujos de trabajo a evaluar en el proyecto final de curso en los cuales se va a trabajar.
	3.1 Define los artefactos a evaluar en el proyecto final de curso.
	4.1 Define las actividades a evaluar en el proyecto final de curso.
	5.1 Luego planifica las iteraciones que les va a evaluar a los estudiantes del proyecto final de curso.
	6.1 El profesor asigna las actividades y artefactos según el rol otorgado durante la creación del proyecto. 6.2 Comunica al estudiante el día de la evaluación del cierre de las iteraciones del proyecto de curso.
7. El estudiante se presenta para la evaluación del cierre de las iteraciones de su proyecto de curso.	7.1 El profesor le comunica a los estudiantes el orden de exposición que van a seguir para la evaluación del cierre de las iteraciones del proyecto final de curso.

8. El estudiante según el rol que desempeña le expone al profesor las actividades y artefactos correspondientes al flujo de trabajo que se evalúa en el cierre de las iteraciones.	8.1 El profesor revisa los artefactos y actividades por el estudiante. 8.2 Corrige los errores encontrados en las actividades y artefactos expuestos. 8.3 Le da al estudiante la evaluación del cierre de iteración expuesto.
9. El estudiante recibe la evaluación de los cierres de iteración expuestos.	9.1 El profesor comunica al estudiante la fecha y el lugar de la exposición final del proyecto de curso. 9.2 Orienta al estudiante que se preparen en el rol que desempeñan y corrijan los errores encontrados.
10. El estudiante se presenta en el lugar escogido por el profesor para la exposición final.	
11. El estudiante expone todas las actividades y artefactos del cierre todas las iteraciones de su proyecto de curso.	11.1 El profesor evalúa la exposición final del proyecto de curso. 11.2 Corrige errores encontrados en la exposición final del proyecto de curso. 11.3 Le da al estudiante la evaluación final del proyecto de curso.
12. El estudiante recibe la evaluación final del proyecto de curso.	
	13.1 El profesor revisa las evaluaciones de los cierres de iteraciones y la trayectoria del estudiante en la asignatura ingeniería de software. 13.2 Realiza un listado con la evaluación final del estudiante en la asignatura. 13.3 Da la nota final de la asignatura al estudiante.
14. El estudiante recibe la evaluación final de la asignatura de Ingeniería de Software.	
15. El estudiante se retira.	

Curso Alternativo de los eventos	
9. El estudiante pide una nueva revisión del cierre de iteración expuesto.	9.1 El profesor realiza una nueva revisión de los artefactos y actividades expuestas por el estudiante. 9.2 Le de al estudiante la nueva evaluación del cierre de la iteración.
12. El estudiante pide una nueva revisión del cierre final de las iteraciones del proyecto de curso.	12.1 El profesor analiza el trabajo de curso completo. 12.2 Le da al estudiante la nueva evaluación final del proyecto de curso.
14. El estudiante pide que le analicen la nota final en la asignatura de Ingeniería de software.	14.1 El profesor realiza una nueva revisión de las evaluaciones anteriores y de su trayectoria. 14.2 Le da al estudiante una nueva evaluación final en la asignatura Ingeniería de Software.
Prioridad	Crítico
Mejoras	
Otros	

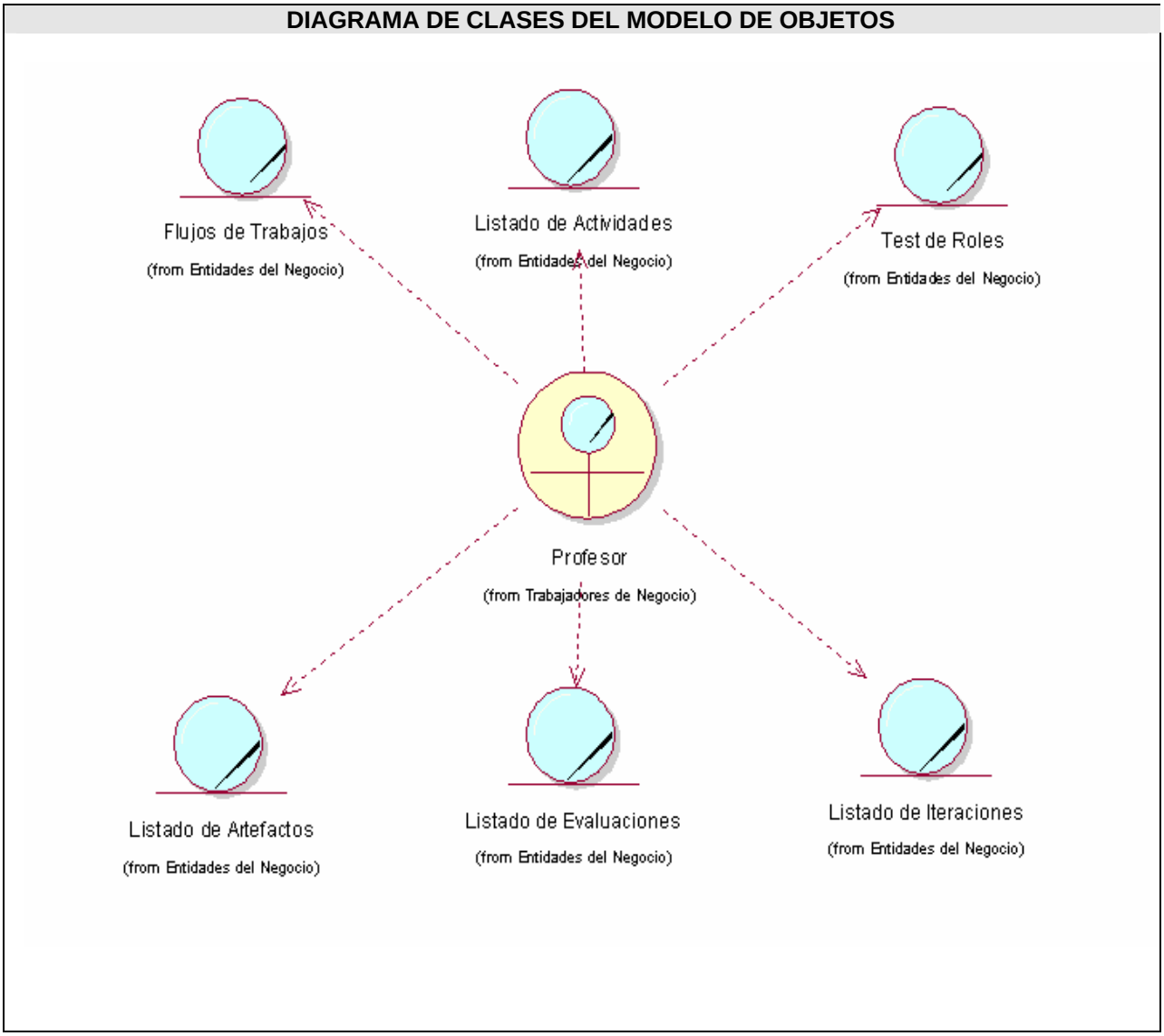
Diagrama de actividad. CUN 02 Desarrollar Proyecto de Curso. (Anexos I, Figura 2-2).

2.2.5 Modelo de objetos

El modelo de objetos es la participación de los trabajadores y entidades del negocio y la relación entre ellos. El modelo de objetos se define en la siguiente tabla.

Tabla 2- 4 Modelo de objetos.

DIAGRAMA DE CLASES DEL MODELO DE OBJETOS



2.3 Sistemas automatizados existentes

En la Universidad de las Ciencias Informáticas se está explotando en la actualidad la plataforma Entorno Virtual de Aprendizaje la cual se encarga de todo el proceso docente que se lleva a cabo en la universidad; la misma está destinada a estudiantes y profesores de pregrado, a estudiantes y profesores de postgrado y a trabajadores (de la UCI y externos) que reciban cursos de capacitación. Esta plataforma cuenta con una gama de curso y otras funciones entre los que incluyen, la formación de pregrado que reciben los estudiantes en cada semestre por año, cursos optativos, cursos del perfil, los cursos de nivelación, las pruebas de nivelación, las comunidades de desarrollo, la formación de postgrado que son los cursos que

reciben los profesores para subir de categoría, los cursos a distancia, los tutoriales de moodle y la organización del proceso docente.

Este es un entorno sencillo, intuitivo y amigable que brinda la posibilidad de que el estudiante adquiera conocimientos a través de distintos productos (videos, power point, multimedia, libro digitales, tutoriales, etc.) elaborados o seleccionados por el profesor, además le permite al estudiante conocer el programa de cada asignatura y tienen a su disposición materiales didácticos, guías de estudio y el asesoramiento del profesor, se les programa al estudiante una series de actividades que guían su proceso de estudio y la realización de diversos ejercicios de autoevaluación que le permiten conocer y controlar el aprendizaje del estudiante por parte del profesor, entre otras funcionalidades que brinda esta plataforma siempre con el objetivo de combinar todos los medios tecnológicos y pedagógicos para garantizar el soporte total o parcial a todo el sistema de pregrado y postgrado de la universidad.

Este entorno de aprendizaje está basado en el proceso docente clásico, en él cual se presentan en cada una de las asignaturas o cursos que incluye la plataforma, conferencias, clases prácticas, seminarios y otros materiales de apoyo que sirven para profundizar un poco más en el tema que se aborda. Sin embargo, a pesar de todas estas ventajas que presenta el entorno en el proceso docente que se lleva a cabo a cabo en la universidad, carece de una ambiente que facilite en profundidad los procesos internos de cada asignatura.

2.4 Solución propuesta: HADIS

Con la confección de la herramienta se pretende automatizar el proceso de desarrollo de proyectos de curso de la asignatura de Ingeniería de Software, mezclando para ello elementos de administración, planificación, formación a distancia y evaluación de los mismos, adaptados a la metodología de desarrollo de software.

A partir de la necesidad expuesta se propone como solución la elaboración de una herramienta informática que permita la automatización de los procesos de administración, planificación y evaluación en los proyectos docentes de la asignatura de Ingeniería de Software orientando en la versión inicial a la metodología de desarrollo de software RUP. A esta aplicación se le ha denominado “HADIS”, la cual incluirá fundamentalmente las siguientes funcionalidades:

- Permitir crear un proyecto de curso, registrándoles las siguiente características: el nombre del proyecto, tipo de proyecto, la complejidad del proyecto y los roles a utilizar en el mismo; así como modificar sus datos o eliminarlo en dependencia de quien realice esa actividad.

- Asignar los recursos humanos necesarios para la elaboración del proyecto, tales como: el nombre de la persona, primer apellido, segundo apellido, el teléfono, el email y el rol que va desempeñar dentro del proyecto.
- Adaptar el proyecto a la metodología RUP, visualizando las cuatro fases del ciclo de vida del proyecto y sus nueves flujos de trabajo, así como planificar las iteraciones que va a tener el proyecto y a la actividades y artefactos que se van a desarrollar dentro de ellas según el rol que se le haya otorgado al recurso humano.
- Permitir subir en cada artefacto la plantilla que le corresponde para que luego ser descargada por el estudiante.
- Permitir el intercambio entre estudiantes y profesores de todos los artefactos con sus respectivas plantillas.
- Asignar al estudiante la evaluación según los artefactos desarrollados, así como modificarla o eliminarla.

Además debe incluir una serie de funcionalidades de administración del sistema, que permitan gestionar toda la información con la que éste trabaja y que brinden la mayor flexibilidad posible, con vistas a su adaptación a cualquier tipo de situación que pueda presentarse.

2.5 Modelo del sistema.

A partir de este punto se comienza a modelar el sistema que se va a construir. Para ello se identifican sus requisitos, tanto funcionales como no funcionales, y se modelan los funcionales en términos de casos de uso del sistema.

2.5.1 Requisitos funcionales

R1 Autenticar usuario.

1. El sistema debe permitir al usuario autenticarse usando su nombre de usuario y contraseña.
2. El sistema debe permitir mostrar al usuario las opciones a las que tiene acceso según el rol o permiso que tiene asignado.
3. El sistema debe integrarse a la autenticación (NTLM) que provee una red con un dominio Windows.

R2 Gestionar proyecto.

1. El sistema debe permitir crear un nuevo proyecto y registrar sus datos como: nombre del proyecto, tipo de proyecto, roles a utilizar y la complejidad del proyecto.

2. El sistema debe permitir modificar los datos del proyecto (tales como: nombre del proyecto, tipo de proyecto, roles a utilizar y la complejidad del proyecto).
3. El sistema debe permitir eliminar el proyecto y todos los datos que estén relacionados.

R3 Gestionar recursos humanos.

1. El sistema debe permitir registrar un nuevo integrante al proyecto, especificando nombre, apellidos, rol que desempeña en dentro del proyecto, teléfono y email.
2. El sistema debe permitir modificar los datos de un determinado integrante del proyecto.
3. El sistema debe permitir eliminar a un integrante de un determinado proyecto.

R4 Adaptar al proyecto la metodología de RUP.

1. El sistema debe de permitir mostrar los flujos de trabajo y las fases que define RUP.
2. El sistema debe permitir definir los flujos de trabajo a realizar acorde a los definidos por el proceso RUP.
3. El sistema debe permitir especificar el número de iteraciones a realizar para cada fase acorde a las definidas en RUP.
4. El sistema debe permitir especificar para cada iteración las actividades a realizar en cada flujo de trabajo.
5. El sistema debe permitir especificar para cada iteración los artefactos a entregar según las tareas seleccionadas.
6. El sistema debe permitir mostrar toda la planificación realizada al proyecto desde las fases con sus respectivas iteraciones con sus flujos de trabajo y dentro las actividades y artefactos desarrollados.

R5 Gestionar artefactos.

1. El sistema debe de permitir descargar la plantilla de los artefactos por los estudiantes.
2. El sistema debe de permitir subir el artefacto desarrollado por cada integrante según el rol que juegue dentro del proyecto.
3. El sistema debe permitir definir el lugar donde se almacenaran todos los artefactos del proyecto.

R6 Controlar intercambios entre profesor y estudiantes.

1. El sistema debe permitir al profesor modificar o actualizar los artefactos en su revisión.
2. El sistema debe permitir notificar a los estudiantes cuando un artefacto ha sido revisado por el profesor y la nota.

3. El sistema debe permitir al profesor definir los hitos en los que se realizaran los cortes de proyecto, especificando la fecha de entrega.
4. El sistema debe permitir al profesor bloquear los artefactos una vez cumplida la fecha de culminación de una iteración o corte de proyecto.
5. El sistema debe permitir al profesor emitir una evaluación en cada revisión.

R7 Gestionar Usuario.

1. El sistema debe permitir registrar datos de un usuario (tales como: nombre, apellidos, especialidad, teléfono y email).
2. El sistema debe permitir modificar datos de un usuario (tales como: nombre, apellidos, rol que desempeña en dentro del proyecto, teléfono y email).
3. El sistema debe permitir eliminar datos de un usuario (tales como: nombre, apellidos, rol que desempeña en dentro del proyecto, teléfono y email).

R8 Gestionar plantillas

1. El sistema debe de permitir colocar las plantillas en caso de que la tenga de cada uno de los artefactos a desarrollar acorde a la configuración del proyecto.
2. El sistema debe de permitir cambiar las plantillas de los artefactos en caso de ocurrir algún cambio en la plantilla.

R9 El sistema debe permitir cambiar la contraseña de los usuarios.

R10 Gestionar evaluación.

1. El sistema debe de permitir emitir la evaluación de un estudiante de acuerdo a su proyecto.
2. El sistema debe de permitir modificar la evaluación de un estudiante de acuerdo a su proyecto.
3. El sistema debe de permitir eliminar la evaluación de un estudiante de acuerdo a su proyecto.

2.5.2 Requisitos no funcionales

Usabilidad:

3.5 El sistema podrá ser usado por cualquier estudiante y profesor que estén vinculados con la asignatura de Ingeniería de Software en la Universidad de las Ciencias Informáticas.

Soporte:

- Garantía de instalación y pruebas del sistema.

Seguridad:

- Transmisión de datos por la red a través de un protocolo seguro.
- Verificación sobre acciones irreversibles (eliminaciones).
- Garantizar que las funcionalidades del sistema se muestren de acuerdo al nivel de acceso del usuario activo.
- Protección contra acciones no autorizadas o que puedan afectar la integridad de los datos.
- Protección contra las inyecciones de código.

Legales:

- El empleo del sistema debe estar regido por un manual de normas y procedimientos que debe ser aprobado por la Dirección de la Universidad y estar basado en las disposiciones legales vigentes.

Confiabilidad:

- Validación de las entradas del usuario.

Apariencia o interfaz externa:

- El sistema debe poseer una interfaz amigable y de fácil navegabilidad para los usuarios de la misma.
- Diseño encuadrado para resoluciones de 800x600.

Software:

- La aplicación correrá en sistema operativo Windows desde versión ´98 o superior con interfaz gráfica y soporte para red.
- Navegador compatible o superior con Internet Explorer 4, o Mozilla Firefox 1.8 o superior.

Diseño e Implementación:

- El sistema se implementará utilizando el lenguaje de programación PHP y el NuSphere 4.6 y el Zend Studio 5.2 como editores de texto del mismo.
- El sistema utilizará una Base de Datos implementada en MySQL Server 5.0.
- Utilizar Apache Web Server 2.2.3 como servidor Web.

2.5.3 Actores del sistema.

Los actores de un sistema son agentes externos: aquellas personas o sistemas que interactúan con él. Los actores del sistema estudiado se definen en la siguiente tabla.

Tabla 2- 5 Actores del negocio.

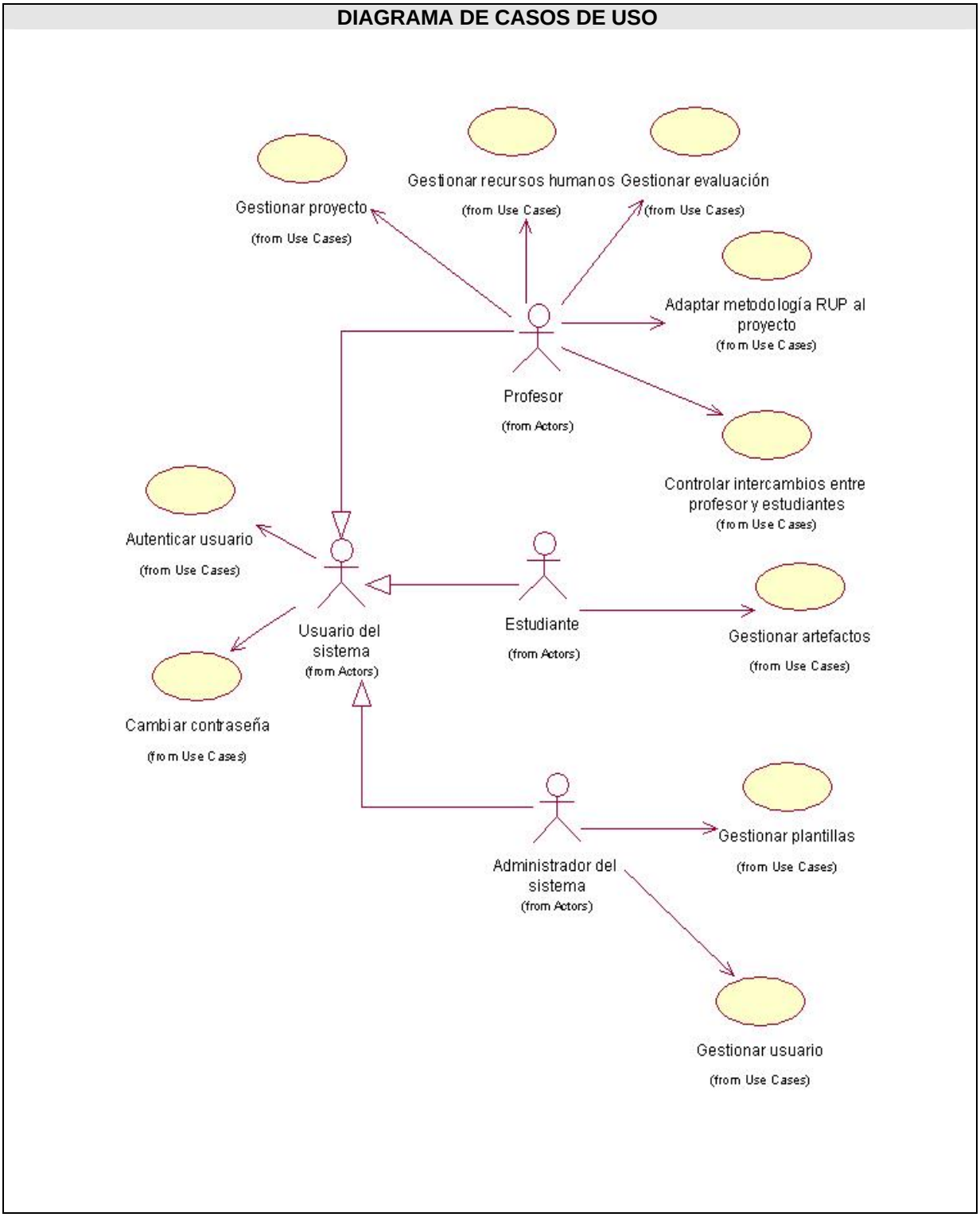
Actores del Sistema	Justificación
Usuario del sistema	Es un usuario que generaliza el rol de autenticación al sistema y de cambiar la contraseña de acceso al sistema.
Administrador del sistema	Es el encargado de gestionar usuarios y de gestionar las plantillas de cada uno de los artefactos que propone RUP.
Profesor	Es el encargado de gestionar los datos de los proyectos, de los RRHH, la evaluación de los estudiantes de adaptar la metodología RUP al proyecto y de controlar los intercambios entre el estudiante y él.
Estudiante	Es el encargado de gestionar los artefactos que propone RUP.

2.5.4 Diagrama de casos de uso del sistema

Los casos de uso son fragmentos de funcionalidad del sistema. En ellos se describe la secuencia determinada de eventos que realiza un actor en interacción con la aplicación. El diagrama de casos de uso del sistema realizado se define en la siguiente tabla.

Tabla 2- 6 Diagrama de casos de uso del sistema.

DIAGRAMA DE CASOS DE USO



2.5.5 Descripción de los casos de uso.

En este epígrafe se describen los casos de uso del sistema.

Nombre del Caso de Uso	**01 Autenticar usuario.**
Actores	Usuario del sistema(Inicia)
Propósito	Permitir al usuario del sistema poder acceder al sistema.
Resumen	El caso de uso inicia cuando el usuario del sistema entra los datos que se le especifican para acceder al sistema, estos se verifican y finaliza dándole los permisos y habilitándole la entrada.
Referencias	R1
Precondiciones	Se habilitan las funcionalidades según lo privilegios.
Poscondiciones	Se ha autenticado un usuario en el sistema.
Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1. El usuario entra usuario y contraseña del dominio UCI.	1.1 El sistema encripta la contraseña. 1.2 Verifica que le usuario y la contraseña sean correctos. 1.3 En caso de ser correcto se le asignan los permisos.
Curso Alternativo de los eventos	
	1.2 En caso de no ser correctos el usuario y la contraseña, le muestra un mensaje de error ¨Acceso denegado¨.
	1.3 En caso de no existir se envía un mensaje de aviso.
Prioridad	Crítico

Nombre del Caso de Uso	**02 Gestionar proyecto.**
Actores	Profesor
Propósito	Permitir registrar, modificar y eliminar datos acerca de un proyecto.
Resumen	El caso de uso se inicia cuando el profesor decide registrar, modificar

	o eliminar datos acerca de un proyecto.
Referencias	R2
Precondiciones	Profesor ya autenticado.
Poscondiciones	Se registra un proyecto, se modifican los datos o se elimina el mismo.

Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1. El administrador del sistema necesita registrar, modificar o eliminar los datos de un proyecto.	1.1 El sistema ejecuta algunas de las siguientes acciones: a) Si decide registrar un proyecto, ir a la sección "Registrar proyecto" b) Si decide modificar los datos de un proyecto, ir a la sección "Modificar proyecto". c) Si decide eliminar un proyecto, ir a la sección "Eliminar proyecto".
Sección "Registrar proyecto"	
Acciones del Actor	**Respuesta del Sistema**
2. El profesor entra los datos necesarios para registrar un proyecto al sistema (nombre del proyecto, roles a utilizar, la descripción del proyecto y selecciona el tipo de proyecto y la complejidad.).	2.1 El sistema verifica que todos los campos estén llenos. 2.2 El sistema verifica que este proyecto no exista. 2.3 El proyecto se almacena en el sistema. 2.4 Se muestra un mensaje informándosele al profesor que ya ha sido efectuado el registro del proyecto satisfactoriamente y finaliza así el caso de uso.
Curso Alternativo de los eventos	
	2.1 Se emite un mensaje para que llene todos los campos.

	2.2 Si el proyecto existe se emite un mensaje informando la existencia del mismo y se finaliza así el caso de uso.
Sección "Modificar proyecto"	
Acciones del Actor	**Respuesta del Sistema**
2. El profesor selecciona la opción de modificar el proyecto por el cual esté navegando.	2.1 El sistema le muestra los datos del proyecto para que pueda modificar los deseados.
3. El profesor realiza las actualizaciones deseadas.	3.1 Se verifica que todos los campos estén llenos. 3.2 Se actualiza la información incorporada al proyecto y se le muestra un mensaje informándole que sus cambios se realizaron satisfactoriamente. Finalizando así el caso de uso.
Curso Alternativo de los eventos	
	3.1 Se emite un mensaje informando que todos los campos deben de estar llenos.
Sección "Eliminar proyecto"	
Acciones del Actor	**Respuesta del Sistema**
2. El profesor selecciona la opción de eliminar el proyecto por el cual esté navegando.	2.1 El sistema pide la confirmación de la eliminación del proyecto. 2.2 El sistema elimina el proyecto seleccionado y finaliza así el caso de uso.
Curso Alternativo de los eventos	
Prioridad	Crítico

Nombre del Caso de Uso	**03 Gestionar recursos humanos.**
Actores	Profesor.

Propósito	Permitir registrar, modificar y eliminar datos acerca de un estudiante.
Resumen	El caso de uso se inicia cuando el profesor decide registrar, modificar o eliminar datos acerca de un estudiante.
Referencias	R2
Precondiciones	El proyecto para el cual se quieren gestionar los recursos humanos debe de estar registrado.
Poscondiciones	Se registra un estudiante, se modifican los datos o se elimina el mismo.

Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1. El profesor necesita registrar, modificar o eliminar los datos de un estudiante.	1.1 El sistema ejecuta algunas de las siguientes acciones: a) Si decide registrar un estudiante, ir a la sección "Registrar recursos humanos" b) Si decide modificar los datos de un estudiante, ir a la sección "Modificar recursos humanos". c) Si decide eliminar un estudiante, ir a la sección "Eliminar recursos humanos".
Sección "Registrar recursos humanos"	
Acciones del Actor	**Respuesta del Sistema**
2. El profesor entra los datos necesarios para registrar un estudiante al sistema (nombre, apellidos, rol que desempeña dentro del proyecto, usuario, contraseña y confirmar contraseña, teléfono y email).	2.1 El sistema verifica que todos los campos estén llenos. 2.2 El sistema verifica que esta persona no exista. 2.3 El estudiante se almacena en el sistema. 2.4 Se muestra un mensaje informándosele al profesor que ya ha sido efectuado el registro del estudiante y finaliza así el caso de

	uso.
Curso Alternativo de los eventos	
	2.1 Se emite un mensaje para que llene todos los campos.
	2.2 Si el estudiante existe se emite un mensaje informando la existencia del mismo y se finaliza así el caso de uso.
Sección "Modificar recursos humanos"	
Acciones del Actor	**Respuesta del Sistema**
2. El profesor selecciona el usuario del estudiante a modificar.	2.1 El sistema brinda la posibilidad de modificar los datos.
3. El profesor realiza las actualizaciones deseadas.	3.1 Se verifica que todos los campos estén llenos. 3.2 Se actualiza la información incorporada al estudiante y se le muestra un mensaje informándole la modificación satisfactoria de los datos del estudiante. Finalizando así caso de uso.
Curso Alternativo de los eventos	
	3.1 Se emite un mensaje informando que todos los campos deben de estar llenos.
Sección "Eliminar recursos humanos"	
Acciones del Actor	**Respuesta del Sistema**
2. El profesor selecciona el usuario del estudiante a eliminar.	2.1 El sistema pide la confirmación de la eliminación del estudiante. 2.2 El sistema elimina al estudiante seleccionado y finaliza así el caso de uso.
Curso Alternativo de los eventos	
Prioridad	Crítico

Nombre del Caso de Uso	**04 Adaptar metodología RUP al proyecto.**
Actores	Profesor
Propósito	Permitir al profesor adaptar la metodología a un proyecto determinado.
Resumen	El caso de uso se inicia cuando el profesor decide adaptar la metodología a un proyecto, el actor selecciona la cantidad de iteraciones, artefactos y actividades a realizar.
Referencias	R4
Precondiciones	Debe existir un proyecto y sus roles ya deben estar definidos.
Poscondiciones	Se definieron las iteraciones, actividades, artefactos para el proyecto seleccionado.

Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1. El profesor necesita planificar las iteraciones en cada una de las fases, definir las actividades y artefactos según el rol que desempeñe el estudiante o mostrar la planificación del proyecto hasta el momento (comenzando por las fases con sus iteraciones y dentro de éstas las actividades y artefactos planificados para realizar en el proyecto).	1.1 El sistema ejecuta algunas de las siguientes acciones: a) Si desea planificar las iteraciones de cada fase ir a la sección “Definir iteración”. b) Si desea definir las actividades y artefactos según el rol que desempeñe el estudiante en las iteraciones definidas en las cuatro fases ir a la sección “Definir actividades y artefactos”. c) Si decide mostrar la planificación del proyecto hasta el momento ir a la sesión “Mostrar planificación”
Sección “Definir iteración”	
Acciones del Actor	**Respuesta del Sistema**
2. El profesor entra el número de la iteración a definir en cada una de las fases y las envía.	2.1 El sistema toma los datos entrados.

	2.2 Muestra cada fase con la iteración entrada por el profesor y dentro de cada una de éstas los nueves flujos de trabajo. Finalizando así el caso de uso.
Sección "Definir actividades y artefactos"	
Acciones del Actor	**Respuesta del Sistema**
2. El profesor selecciona la opción de definir actividades y artefactos según rol.	2.1 El sistema le muestra las cuatro fases y dentro de cada una las iteraciones que le fueron especificadas y dentro de cada iteración los nueve flujos de trabajos.
3. El profesor selecciona la fase con la cual desea trabajar.	3.1 El sistema le muestra las iteraciones de la fase seleccionada.
4. El profesor selecciona una de las iteraciones dentro de la fase seleccionada.	4.1 El sistema le muestra dentro de la iteración los nueves flujos de trabajo.
5. El profesor selecciona el flujo de trabajo con el cual desea trabajar dentro de la iteración seleccionada.	5.1 El sistema le muestra los roles ya definidos anteriormente, además de todas las actividades y artefactos que el flujo de trabajo seleccionada.
6. El profesor selecciona un rol.	6.1 El sistema le muestra las actividades y artefactos correspondientes al rol seleccionado.
7. El profesor marca las actividades y artefactos que desea que se ejecuten por el rol seleccionado y luego los envía. .	7.1 El sistema almacena los roles con sus respectivas actividades y artefactos seleccionados. 7.2 El sistema muestra un mensaje diciéndole que se efectúo satisfactoriamente la inserción en el flujo seleccionado. Finalizando así el caso de uso.
Curso Alternativo de los eventos	

Sección "Mostrar planificación"	
Acciones del Actor	**Respuesta del Sistema**
2. El profesor selecciona la opción de mostrar la planificación realizada que hasta el momento lleva el proyecto.	2.1 El sistema le muestra toda la planificación realizada del proyecto hasta el momento (empezando por las fases con sus iteraciones, dentro los flujos de trabajo con sus actividades y artefactos planificados por el profesor). Finalizando así el caso de uso.
Prioridad	Crítico

Nombre del Caso de Uso	**05 Gestionar usuario.**
Actores	Administrador del sistema.
Propósito	Permitir registrar, modificar y eliminar datos acerca de un usuario.
Resumen	El caso de uso se inicia cuando el administrador del sistema decide registrar, modificar o eliminar datos acerca de un usuario.
Referencias	R2
Precondiciones	Administrador del sistema ya autenticado.
Poscondiciones	Se registra un usuario, se modifican los datos o se elimina el mismo.

Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1. El administrador del sistema necesita registrar, modificar o eliminar los datos de un usuario.	1.1 El sistema ejecuta algunas de las siguientes acciones: d) Si decide registrar un usuario, ir a la sección "Registrar usuario" e) Si decide modificar los datos de un usuario, ir a la sección "Modificar usuario". f) Si decide eliminar un usuario, ir a la sección "Eliminar

	usuario".
Sección "Registrar usuario"	
Acciones del Actor	**Respuesta del Sistema**
2. El administrador del sistema entra los datos necesarios para registrar un usuario al sistema (nombre, especialidad, apellidos, teléfono, usuario, contraseña, confirmar contraseña, categoría y email).	2.1 El sistema verifica que todos los campos estén llenos. 2.2 El sistema verifica que esta persona no exista. 2.3 El usuario se almacena en el sistema. 2.4 Se muestra un mensaje informándosele al administrador que ya ha sido efectuado el registro del usuario y finaliza así el caso de uso.
Curso Alternativo de los eventos	
	2.1 Se emite un mensaje para que llene todos los campos.
	2.2 Si el usuario existe se emite un mensaje informando la existencia del mismo y se finaliza así el caso de uso.
Sección "Modificar usuario"	
Acciones del Actor	**Respuesta del Sistema**
2. El administrador del sistema selecciona el usuario a modificar.	2.1 El sistema brinda la posibilidad de modificar los datos.
3. El administrador del sistema realiza las actualizaciones deseadas.	3.1 Se verifica que todos los campos estén llenos. 3.2 Se actualiza la información incorporada al usuario y se emite un mensaje informándole que la modificación se realizó satisfactoriamente. Finalizando así el caso de uso.
Curso Alternativo de los eventos	
	3.1 Se emite un mensaje informando

	que todos los campos deben de estar llenos.
Sección "Eliminar usuario"	
Acciones del Actor	**Respuesta del Sistema**
2. El administrador del sistema selecciona la categoría (profesor o administrador del sistema) del usuario a eliminar.	2.1 El sistema le muestra una tabla con los usuarios (nombre, primer apellido, segundo apellido) pertenecientes a la categoría seleccionada.
3. El administrador del sistema selecciona el usuario a eliminar.	2.1 El sistema pide la confirmación de la eliminación del usuario. 2.2 El sistema elimina al usuario seleccionado y emite un mensaje informando que la eliminación se realizó satisfactoriamente. Finalizando así el caso de uso.
Curso Alternativo de los eventos	
Prioridad	Crítico

Nombre del Caso de Uso	**06 Gestionar artefactos.**
Actores	Estudiante
Propósito	Permitir realizar las operaciones relacionadas con las plantillas correspondientes a los artefactos generados en el desarrollo del sistema.
Resumen	El caso de uso inicia cuando el estudiante decide realizar alguna operación con los artefactos definidos en el proyecto, que pueden ser descargar la plantilla y subir una actualización, ver las respuestas de los artefactos revisados por el profesor o ver el listado de los artefactos existentes en el proyecto en general.
Referencias	R5
Precondiciones	Estudiante con rol asignado ya autenticado, el artefacto seleccionado para realizar alguna operación debe haber sido incorporado dentro de la configuración del proyecto.

Poscondiciones	Se descargan las plantillas del artefacto y una nueva actualización del artefacto queda disponible para posterior evaluación, se exponen los artefactos ya revisados por el profesor o muestran el listado con las plantillas ya realizadas por el estudiante.

Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1. El estudiante necesita descargar la plantilla del artefacto a realizar según el rol que desempeña dentro del proyecto, descargar y subir la plantilla desarrollada, observar las plantillas de los artefactos ya revisados por el profesor o ver el listado de todos los artefactos con sus respectivas plantillas realizadas de su proyecto.	1.1 El sistema ejecuta algunas de las siguientes acciones: a) Si decide descargar la plantilla del artefacto a desarrollar ir a la sección "Descargar y subir plantilla de artefacto". b) Si decide subir la respuesta de la plantilla desarrollada ir a la sección "Ver respuesta del profesor". c) Si decide ver el listado con todos los artefactos y sus respectivas plantillas del proyecto completo ir a la sección "Listado de artefactos desarrollados".
Sección "Descargar y subir plantilla de artefacto"	
Acciones del Actor	**Respuesta del Sistema**
2. El estudiante decide descargar la plantilla del artefacto a elaborar.	2.1 El sistema le muestra los artefactos con sus respectivas plantillas al lado.
3. El estudiante da clic sobre la plantilla a descargar.	3.1 El sistema le muestra un cartel con las opciones de abrir, guardar o cancelar la planilla.
4. El estudiante descarga la plantilla del artefacto.	

5. El estudiante después de desarrollada la plantilla del artefacto correspondiente, decide ir subir la plantilla elaborada.	5.1 El sistema le muestra los nombres de los artefactos y al lado un botón para buscar la plantilla elaborada.
6. El estudiante da clic sobre el botón y busca la dirección donde tiene guardada la plantilla del artefacto.	6.1 El sistema le muestra un botón para subir la plantilla elaborada.
7. El estudiante sube la plantilla del artefacto realizado.	7.1 El sistema le muestra que la plantilla ha sido subida satisfactoriamente y finaliza así el caso de uso.
Curso Alternativo de los eventos	
Sección "Ver respuesta del profesor"	
Acciones del Actor	**Respuesta del Sistema**
2. El estudiante decide ver las plantillas ya revisadas por el profesor.	2.1 El sistema le muestra por orden desde fase con sus iteraciones y dentro de éstas sus flujos de trabajo y dentro los artefactos que le corresponden a él ya revisados por el profesor.
3. El estudiante decide descargar la plantilla del artefacto.	3.1 El sistema le muestra la opción de descargar la plantilla.
4. El estudiante descarga la plantilla del artefacto.	4.1 El sistema le brinda la plantilla y finaliza así el caso de uso.
Curso Alternativo de los eventos	
Sección "Listado de artefactos desarrollados"	
Acciones del Actor	**Respuesta del Sistema**
2. El estudiante decide ver el estado del proyecto completo con todos los artefactos y sus respectivas plantillas.	2.1 El sistema le muestra el listado de todos los artefactos y sus respectivas plantillas al lado.
3. El estudiante accede a la que desee.	3.1 El sistema le brinda la posibilidad de descargarla; finalizando así el caso de uso.
Curso Alternativo de los eventos	

Prioridad	Crítico

Nombre del Caso de Uso	**07 Controlar intercambios entre profesor y estudiantes.**
Actores	Profesor
Propósito	Definir un espacio para el intercambio entre los profesores y los estudiantes.
Resumen	El profesor entra al área de intercambios, define los hitos del proyecto, bloquea los artefactos y revisa los artefactos elaborados por el estudiante.
Referencias	R6
Precondiciones	Proyecto ya registrado. Artefactos ya desarrollada por los distintos roles de RUP.
Poscondiciones	Se modifican los artefactos y se emite la evaluación por cada corte del proyecto y del proyecto en general.

Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1. El profesor decide modificar la plantilla de un artefacto subido al sistema por el estudiante, notificar que un artefacto ha sido revisado y al lado poner la nota, definir los hitos en los que se realizaran los cortes de proyecto, bloquear los artefactos una vez cumplida la fecha de culminación de una iteración o corte del proyecto o emitir una evaluación por cada corte de proyecto y del proyecto completo.	1.1 El sistema ejecuta algunas de las siguientes acciones: a) Si decide modificar la plantilla de los artefactos realizados por el estudiante y darle una nota ir a la sección “Revisar artefacto” b) Si decide definir los hitos en los que se realizaran los cortes de proyecto ir a la sección “Definir hitos”. c) Si decide bloquear los artefactos una vez cumplida la fecha de culminación de una iteración o corte de un proyecto ir a la sección

	"Bloquear artefactos".
Sección "Revisar artefacto"	
Acciones del Actor	**Respuesta del Sistema**
2. El profesor selecciona el rol y el estudiante al cual le va a revisar los artefactos realizados.	2.1 El sistema le muestra las cuatro fases de RUP con sus iteraciones y dentro de cada una de éstas los flujos de trabajo que se acordaron desarrollar en las iteraciones y en cada flujo de trabajo los artefactos con respectivas plantillas al lado que el estudiante ha subido.
3. El profesor da clic sobre el artefacto que desea revisar para descargarlo.	3.1 El sistema le muestra la opción descargarlo o abrirlo.
4. El profesor descarga la plantilla del artefacto.	
5. El profesor revisa la plantilla del artefacto y le hace las modificaciones necesarias.	
6. El profesor da clic sobre el botón "Buscar" para subir la plantilla revisada.	6.1 El sistema le pide la dirección de donde tiene la plantilla revisada.
7. El profesor sube la plantilla revisada.	7.1 El sistema le muestra la plantilla subida satisfactoriamente al lado del artefacto.
Curso Alternativo de los eventos	
Sección "Definir hitos"	
Acciones del Actor	**Respuesta del Sistema**
2. El profesor entra las fechas en los que se efectuaran los hitos de los cortes de proyecto.	2.1 El sistema toma los datos entrados. 2.2 El sistema muestra las fechas en las que se efectuarán los hitos de los cortes de proyecto. Finalizando así el caso de uso.
Curso Alternativo de los eventos	
Sección "Bloquear artefactos"	
Acciones del Actor	**Respuesta del Sistema**

2. El profesor entra la fecha en las que los artefactos se inhabilitarán.	2.1 El sistema tomas los datos entrados. 2.2 El sistema le muestra un mensaje al profesor diciéndole que en la fecha entrada se le inhabilitarán los artefactos a los estudiantes. Finalizando así el caso de uso.
Curso Alternativo de los eventos	
Prioridad	Crítico

Nombre del Caso de Uso	**08 Gestionar plantilla.**	
Actores	Administrador del sistema.	
Propósito	Permitir colocar las plantillas en los artefactos que la contengan y cambiar las plantillas de los artefactos en de existir una nueva.	
Resumen	El caso de uso si inicia cuando el administrador del sistema decide colocar las plantillas en los artefactos que la contengan o cambiar las plantillas de los artefactos en caso de ocurrir algún cambio.	
Referencias	R8	
Precondiciones	Administrador del sistema ya autenticado, debe existir el artefacto al cual se le desea agregar alguna plantilla o modificar la existente.	
Poscondiciones	Se colocan plantillas en los artefactos que la contengan o se cambia la plantilla de un artefacto en caso de cambio.	
Curso Normal de los eventos		
Acciones del Actor		**Respuesta del sistema**
1. El administrador del sistema decide colocar las plantillas en los artefactos que la contengan y cambiar las plantillas de los artefactos en caso de que ocurra algún cambio en las mismas.		1.1 El sistema ejecuta algunas de las siguientes acciones: a) Si decide colocar las plantillas en los artefactos que la contengan ir a la sección “Colocar plantilla”. b) Si decide cambiar las plantillas de los artefactos en

	caso de ocurrir algún cambio ir a la sección “Cambiar plantilla”.
Sección “Colocar plantilla”	
2. El administrador del sistema da clic sobre el flujo de trabajo en el cual desea subir la plantilla a los artefactos correspondientes	2.1 El sistema le muestra los artefactos correspondientes a ese flujo con su botón “Buscar” al lado.
3. El administrador del sistema da clic sobre el botón “Buscar”.	3.1 El sistema le pide la dirección donde tiene la plantilla del artefacto deseado.
4. El administrador del sistema busca la dirección correcta de la plantilla a subir y le da al botón subir plantilla.	4.1 El sistema verifica si lo que el administrador del sistema ha subido es un documento.
	5.1 El sistema le muestra que el documento ha sido subido satisfactoriamente. Finalizando así el caso de uso.
Curso Alternativo de los eventos	
	4.1 Se emite un mensaje de informándole al administrador del sistema que sólo se pueden subir documentos.
Sección “Cambiar plantilla”	
Acciones del Actor	**Respuesta del sistema**
2. El administrador del sistema descargara la plantilla del artefacto que desea modificar.	2.1 El sistema le pide la dirección donde quiere que se guardar la plantilla modificar.
3. El administrador del sistema descarga la plantilla en la dirección que le especificó al sistema.	
4. El administrador del sistema realiza los cambios en la plantilla descargada.	
5. El administrador del sistema le da al botón ¨Buscar¨ para subir la plantilla con los cambios	5.1 El sistema le pide la dirección donde tiene la plantilla modificada

realizados.	del artefacto deseado.
6. El administrador del sistema busca la dirección correcta de la plantilla modificada a subir y le da al botón subir plantilla.	6.1 El sistema verifica si lo que el administrador del sistema ha subido es un documento.
	7.1 El sistema le muestra que la plantilla modificada ha sido subida satisfactoriamente.
Curso Alternativo de los eventos	
	6.1 Se emite un mensaje de informándole al administrador del sistema que sólo se pueden subir documentos.
Prioridad	Crítico

Nombre del Caso de Uso	**09 Cambiar contraseña.**	
Actores	Usuario del sistema(Inicia)	
Propósito	Permitir al usuario del sistema cambiar su contraseña.	
Resumen	El caso de uso inicia cuando el usuario del sistema decide cambiar su contrañesa, para lo cual entra los datos necesarios para lleva a cabo dicha acción.	
Referencias	R9	
Precondiciones	Usuario ya autenticado.	
Poscondiciones	Se cambia la contraseña de un usuario del sistema.	
Curso Normal de los eventos		
Acciones del Actor		**Respuesta del sistema**
1. El usuario del sistema entra contraseña anterior, la nueva contraseña, la confirmación de la misma y el usuario del dominio.		1.1 El sistema verifica los datos introducidos. 1.2 El sistema muestra un mensaje informándole al usuario del sistema que su contraseña ha sido cambiada satisfactoriamente. Finalizando así el caso de uso.
Curso Alternativo de los eventos		

	1.2 El sistema muestra un mensaje informándole al usuario del sistema que los datos están incorrectos y le permite llenarlos nuevamente Finalizando así el caso de uso.
Prioridad	Secundario

Nombre del Caso de Uso	**10 Gestionar evaluación.**
Actores	Profesor
Propósito	Permitir registrar, modificar y eliminar la evaluación de un estudiante.
Resumen	El caso de uso se inicia cuando el profesor decide registrar, modificar o eliminar la evaluación de un estudiante.
Referencias	R10
Precondiciones	Profesor ya autenticado. Artefactos subidos por el estudiante ya revisados por el profesor.
Poscondiciones	Se registra una evaluación, se modifica o se elimina la misma.

Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1. El administrador del sistema necesita registrar, modificar o eliminar la evaluación de un estudiante	1.1 El sistema ejecuta algunas de las siguientes acciones: d) Si decide registrar una evaluación, ir a la sección "Registrar evaluación" e) Si decide modificar a evaluación, ir a la sección "Modificar evaluación". f) Si decide eliminar una evaluación, ir a la sección "Eliminar evaluación".
Sección "Registrar evaluación"	
Acciones del Actor	**Respuesta del Sistema**
2. El profesor decide emitir la evaluación de un determinado estudiante.	2.1 El sistema le muestra una tabla con el usuario, nombre, primer

	apellido y segundo apellido de los estudiantes del proyecto por el cual esté navegando el profesor.
3. El profesor selecciona el usuario del estudiante al cual quiere introducirle una determinada nota.	3.1 El sistema le muestra las notas (2, 3, 4, 5) para que el profesor escoja la que el estudiante se ganó.
4. El profesor selecciona la nota y la envía.	4.1 El sistema almacena la nota que el profesor seleccionó. 4.2 El sistema le muestra un cartel al profesor informándole que nota fue subida satisfactoriamente. 4.3 El sistema le muestra por orden todas las notas asignadas al estudiante. Finalizando así el caso de uso.
Curso Alternativo de los eventos	
Sección "Modificar evaluación"	
Acciones del Actor	**Respuesta del Sistema**
2. El profesor selecciona el usuario del estudiante al cual le va a modificar la nota.	2.1 El sistema le muestra las notas que hasta el momento tiene el estudiante por orden.
3. El profesor entra la nota que va a modificar y la envía.	3.1 El sistema almacena los datos entrados por el profesor. 3.2 El sistema le muestra un mensaje informándole que la nota fue modificada satisfactoriamente. 3.3 El sistema muestra el listado de notas del estudiante con los cambios realizados. Finalizando así el caso de uso.
Curso Alternativo de los eventos	
Sección "Eliminar evaluación"	
Acciones del Actor	**Respuesta del Sistema**

2. El profesor selecciona el usuario del estudiante a eliminar.	2.1 El sistema muestra el nombre, primer apellido, segundo apellido y el listado de las notas que tiene hasta el momento.
3. El profesor selecciona la nota a eliminar.	3.1 El sistema pide la confirmación de eliminación.
4. El profesor elimina la nota deseada.	4.1 El sistema elimina la nota especificada por el profesor. 4.2 El sistema muestra un mensaje de informándole al profesor que la eliminación se realizó satisfactoriamente. Finalizando así el caso de uso.
Curso Alternativo de los eventos	
Prioridad	Secundario

Conclusiones

En este capítulo luego de haber estudiado todo el proceso que se lleva en el proceso docente de la asignatura de ingeniería se modeló el negocio actual a través de: la identificación de los actores del negocio, trabajadores de negocio, entidades del negocio y los casos de uso correspondientes, y a raíz de esto se realizaron los diagramas de casos de uso, de actividades y el modelo de objeto. Se elaboró la propuesta de solución y se comenzó con la captura de requisitos tanto funcionales como no funcionales con los que debe cumplir el sistema a desarrollar y de ahí se modeló el mismo en términos de casos de uso del sistema. A partir de este punto se puede comenzar a construir el sistema que constituye la propuesta de solución.

3

CAPÍTULO

CONSTRUCCION DE LA SOLUCION PROPUESTA

Introducción

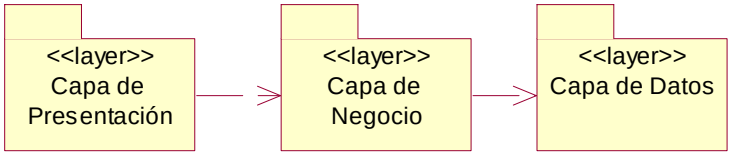
En el presente capítulo se diseña la propuesta de solución, estableciendo el patrón de arquitectura y los de diseño a utilizar, se modela el modelo de diseño, creando los diagramas de clases Web para cada caso de uso con sus respectivos diagramas de secuencia por flujo de eventos, el diagrama de clases persistentes que da entrada al modelo de la base de datos, los principios del diseño y el tratamiento de excepciones. Finalmente, se entra en el flujo de implementación con los diagramas de componentes y el modelo de despliegue para una mejor descripción de la solución.

3.1 Patrones de arquitectura

“Los patrones de arquitectura expresan el esquema fundamental de organización para sistemas de software. Proveen un conjunto de subsistemas predefinidos; especifican sus responsabilidades e incluyen reglas y guías para organizar las relaciones entre ellos”(WELICKI 2005).

Después de hacer un estudio de los patrones existentes se estableció que el mejor enfoque para la construcción de la aplicación Web fuese patrón de arquitectura de 3 capas, donde en la capa de presentación encontramos a todas las clases interfaces, en la capa de negocio, encontramos toda la lógica del negocio, es decir las clases entidades y controladoras y ya por último en la capa de datos tenemos las clases que permiten el acceso a la base de datos. La siguiente figura muestra este modelo.

Figura 3-1. Modelo de tres capas.



3.2 Modelo de diseño

El modelo del diseño es un modelo de objetos que describe la realización de los casos de uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tiene impacto en el sistema a considerar, y constituye una entrada fundamental de las actividades de implementación.

3.2.1 Patrones de diseño

"Un patrón de diseño es básicamente una solución (un diseño) que surge de la experiencia práctica con varios proyectos, y los equipos de desarrollo han encontrado que se puede aplicar en diversos contextos. Cada patrón de diseño describe a un conjunto de objetos y clases comunicadas. El conjunto se ajusta para resolver un problema de diseño en un contexto específico".

En el diseño de la aplicación se aplicaron los patrones de GRASP, patrones para asignación de responsabilidades, fundamentalmente el patrón Experto, que plantea que se debe de asignar una responsabilidad al experto en información, o sea, la clase con toda la información necesaria para llevarla a cabo.

3.2.2 Diagramas de clases Web

Los diagramas de clases Web se realizaron empleando estereotipos Web (formularios, páginas clientes, páginas servidoras), para un mayor entendimiento y lógica de los mismos. Se realizó un diagrama de clases por cada caso de uso y por cada flujo de evento de los diagramas se realizó un diagrama de secuencia.

Los diagramas de clases Web se definen en las siguientes tablas.

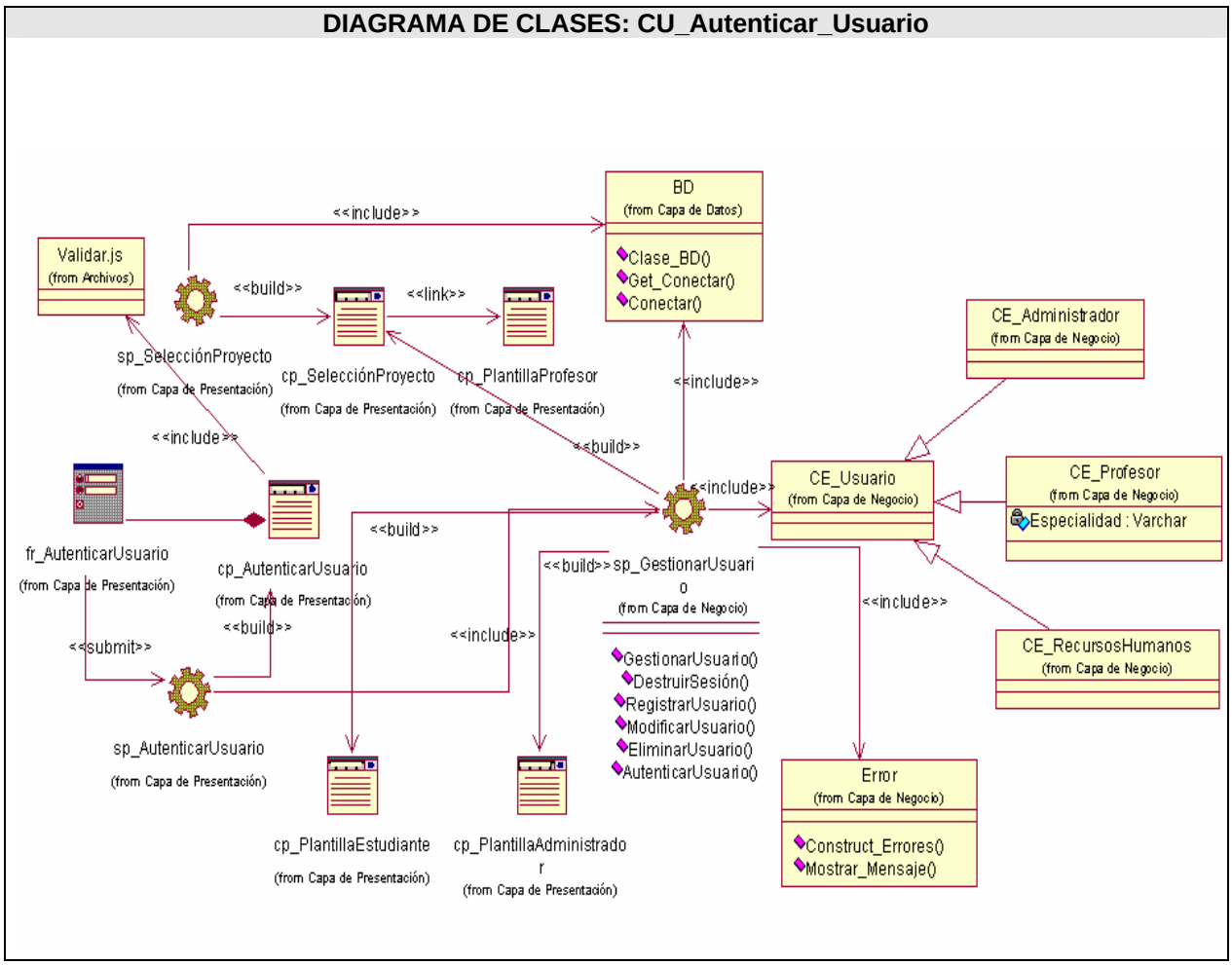


Tabla 3-1. Diagrama de clases. Caso de uso Autenticar usuario.

Diagrama de secuencia. CU Autenticar usuario. Escenario "Autenticar Usuario" (Anexos II, Figura 3-2).

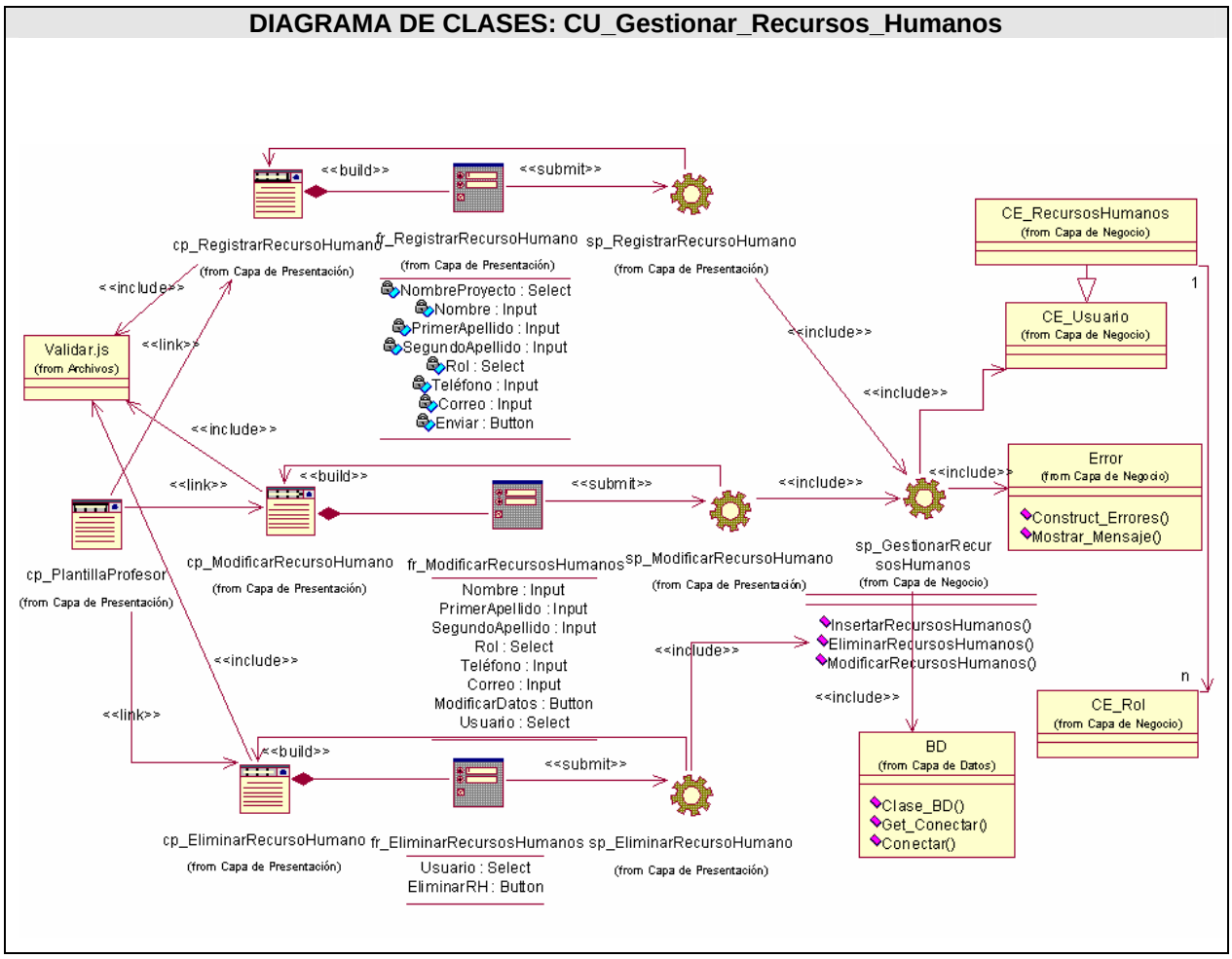


Tabla 3-2. Diagrama de clases. Caso de uso Gestionar recurso humano.

Diagrama de secuencia. CU Gestionar recurso humano. Escenario "Registrar recurso humano" (Anexos II, Figura 3-3).

Diagrama de secuencia. CU Gestionar recurso humano. Escenario "Modificar recurso humano" (Anexos II, Figura 3-4).

Diagrama de secuencia. CU Gestionar recurso humano. Escenario "Eliminar recurso humano" (Anexos II, Figura 3-5).

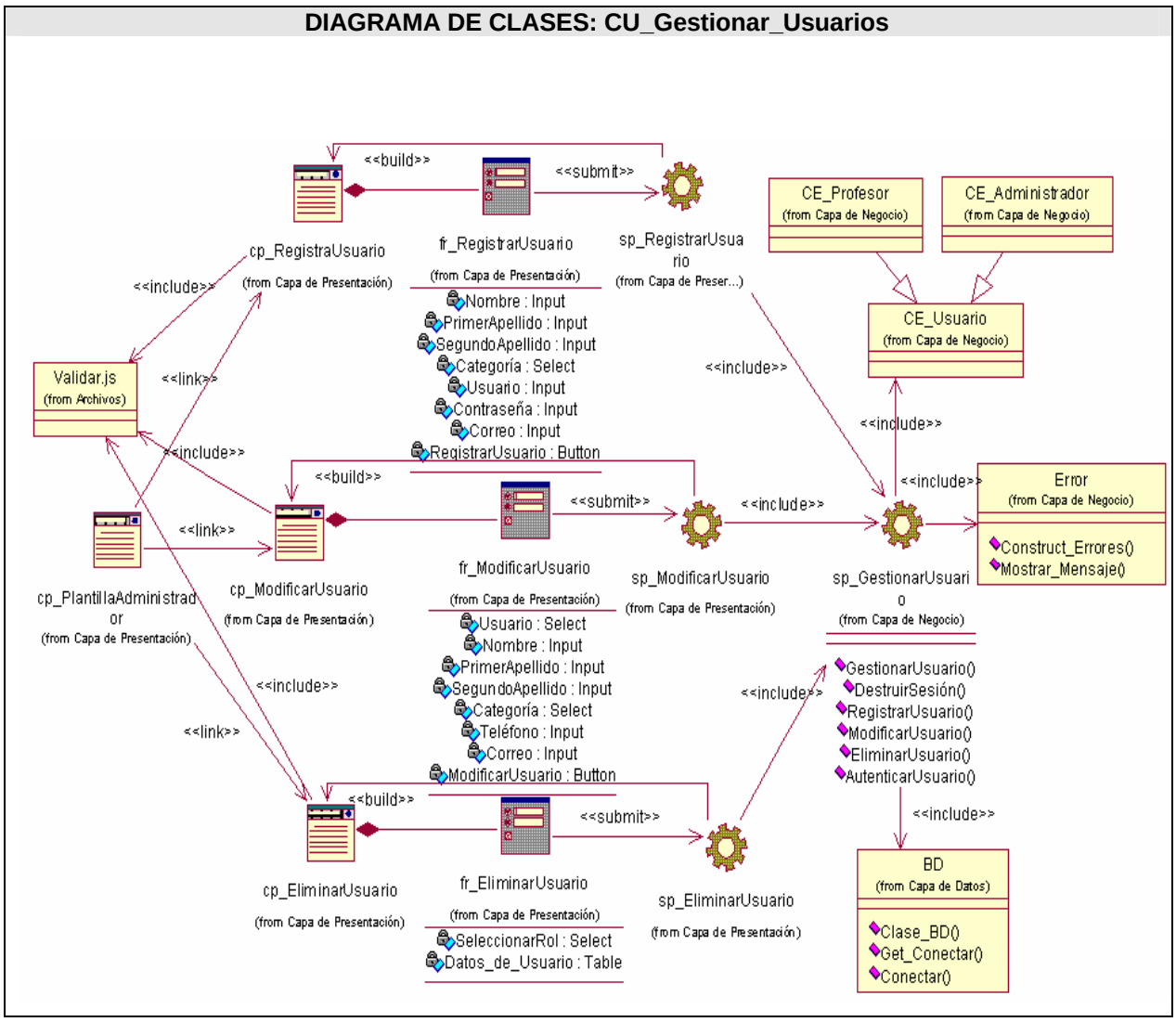


Tabla 3-3. Diagrama de clases. Caso de uso Gestionar usuario.

Diagrama de secuencia. CU Gestionar usuario. Escenario “Registrar usuario” (Anexos II, Figura 3-6).

Diagrama de secuencia. CU Gestionar usuario. Escenario “Modificar usuario” (Anexos II, Figura 3-7).

Diagrama de secuencia. CU Gestionar usuario. Escenario “Eliminar usuario” (Anexos II, Figura 3-8).

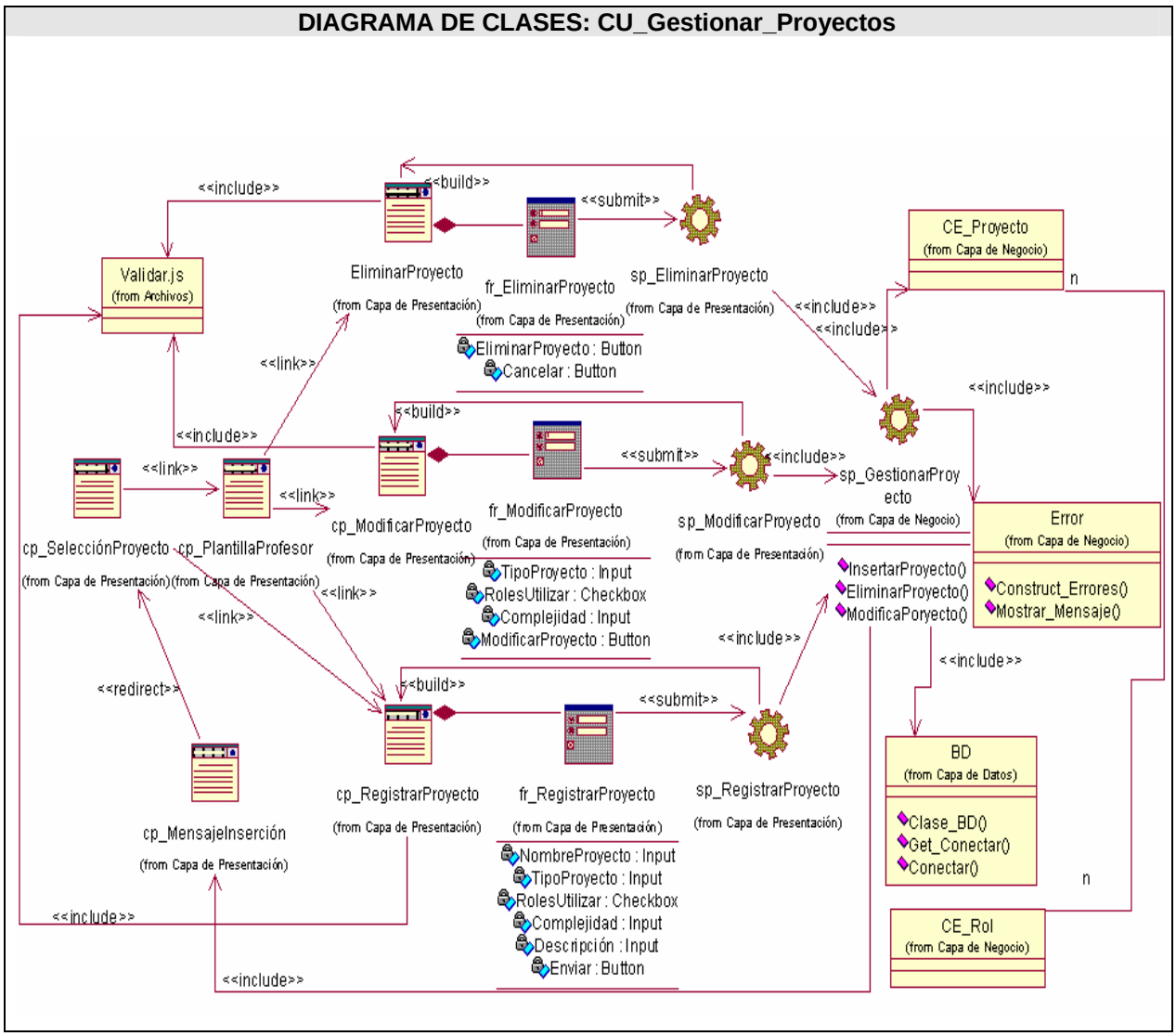


Tabla 3-4. Diagrama de clases. Caso de uso Gestionar proyecto.

Diagrama de secuencia. CU Gestionar proyecto. Escenario "Registrar proyecto" (Anexos II, Figura 3-9).

Diagrama de secuencia. CU Gestionar proyecto. Escenario "Modificar proyecto" (Anexos II, Figura 3-10).

Diagrama de secuencia. CU Gestionar proyecto. Escenario "Eliminar proyecto" (Anexos II, Figura 3-11).

DIAGRAMA DE CLASES: CU_Adaptar_Proyecto_Metodología_RUP

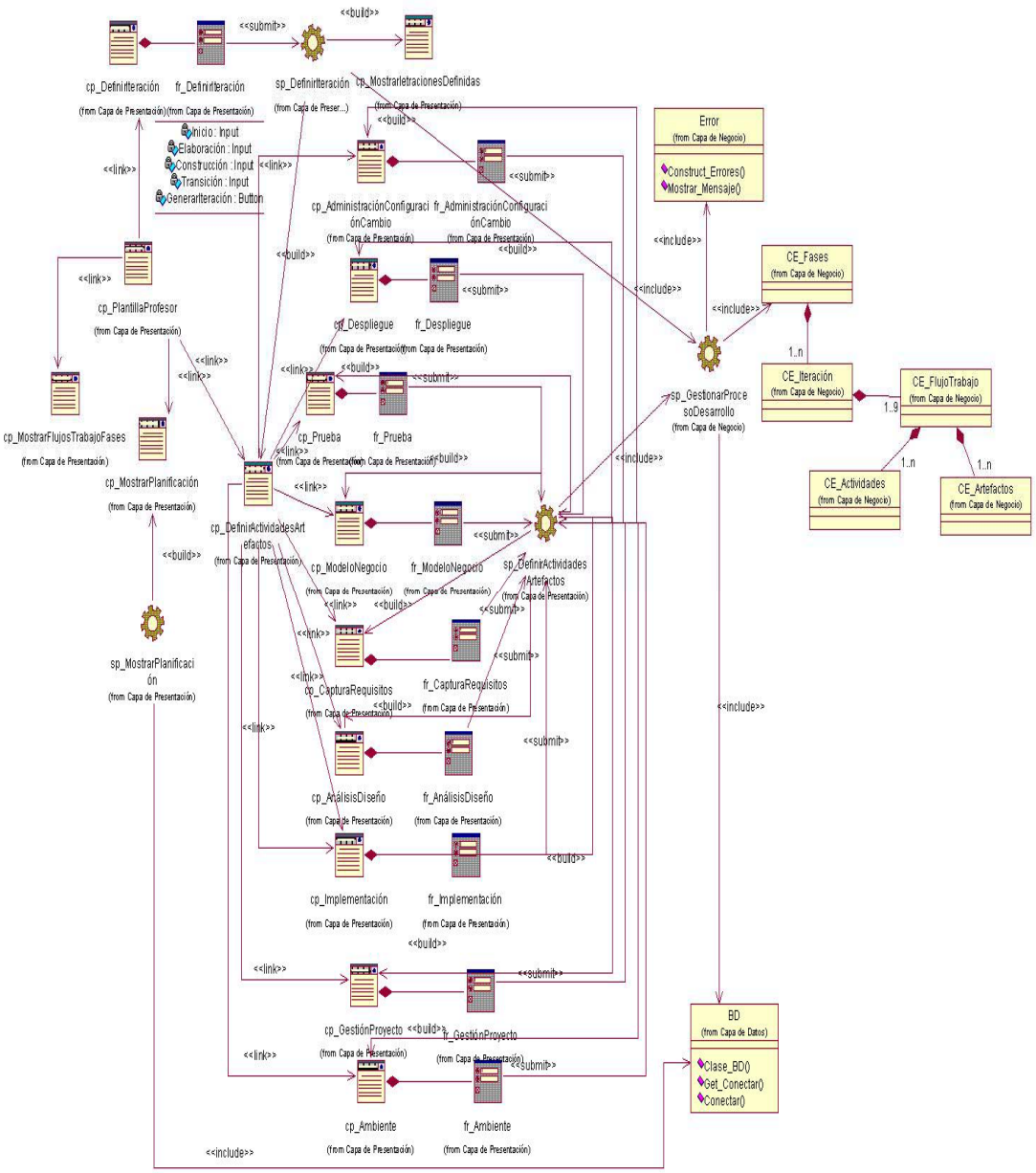


Tabla 3-5. Diagrama de clases. Caso de uso Adaptar al proyecto la metodología RUP.

Diagrama de secuencia. CU Adaptar al proyecto la metodología RUP. Escenario "Definir actividades y artefactos" (Anexos II, Figura 3-12).

DIAGRAMA DE CLASES: CU_GestionarPlantillas(FE:CambiarPlantilla Parte 1)

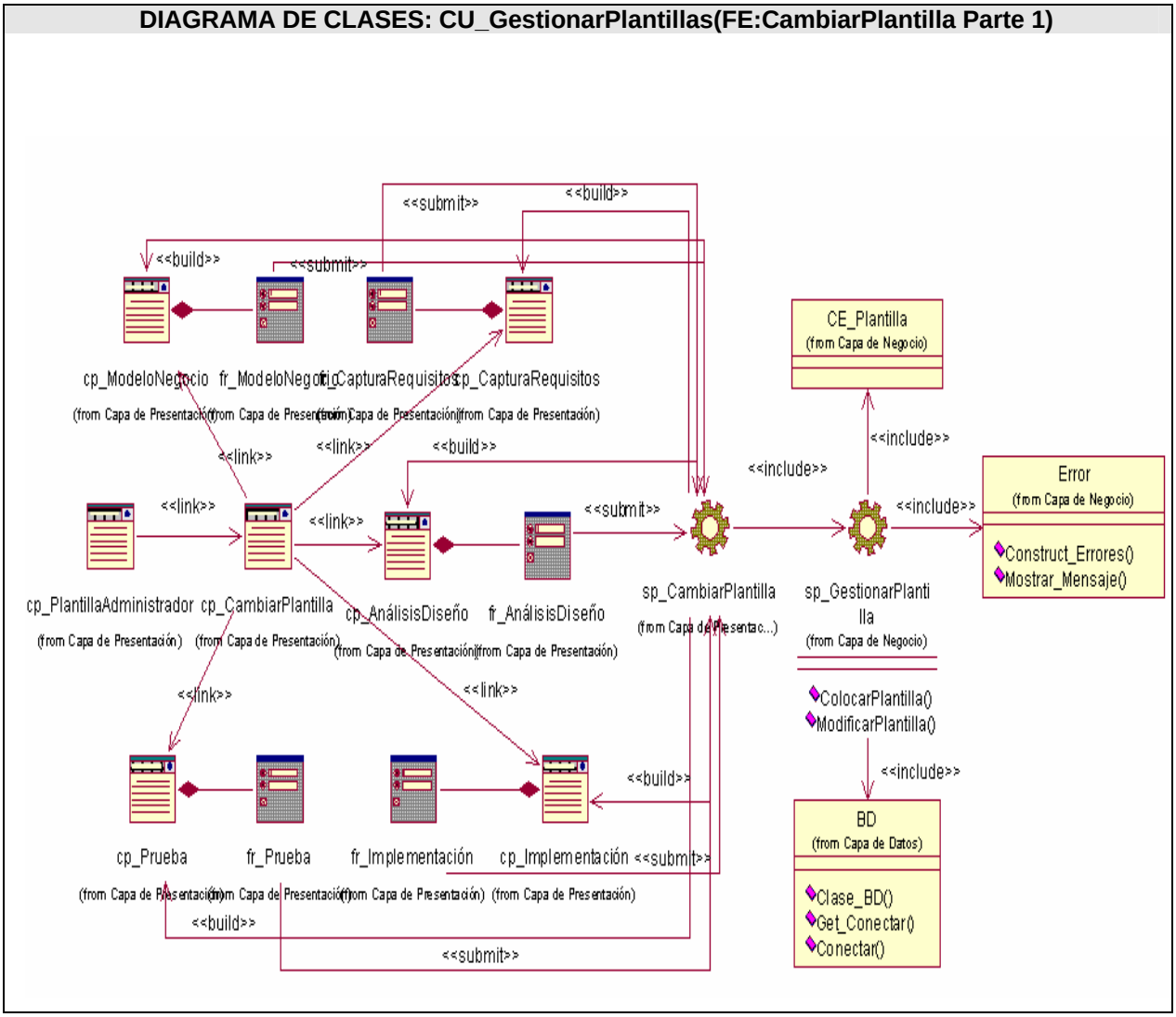


Tabla 3-6.Diagrama de clase. Caso de uso Gestionar plantillas. Parte 1.

DIAGRAMA DE CLASES: CU_GestionarPlantillas(FE:CambiarPlantilla2)

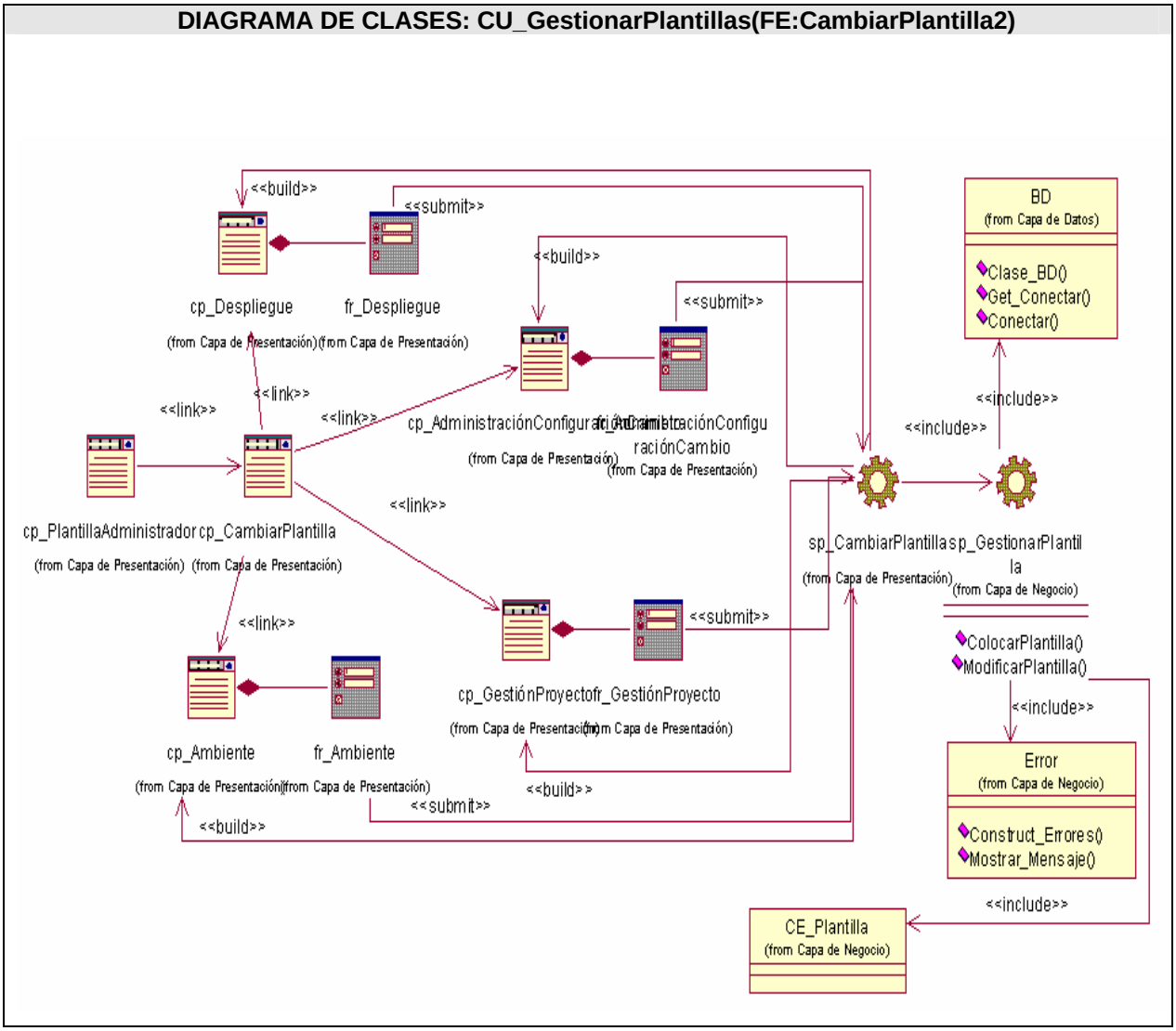


Tabla 3-7. Diagrama de clases. Caso de uso Gestionar plantillas. Parte 2.

DIAGRAMA DE CLASES: CU_GestionarPlantillas(FE:ColocarPlantilla1)

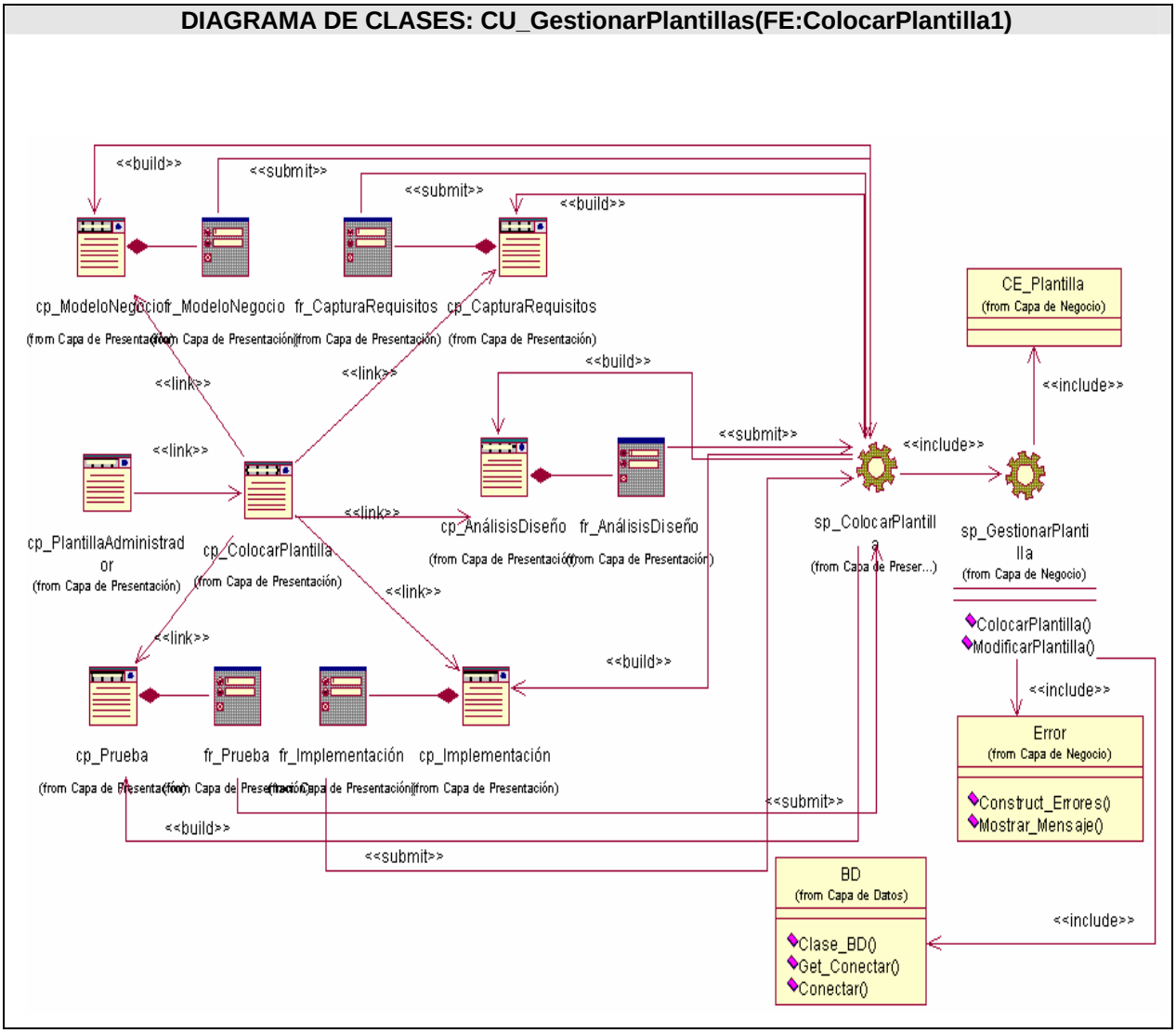


Tabla 3-8. Diagrama de clases. Caso de uso Gestionar plantillas. Parte 3

DIAGRAMA DE CLASES: CU_GestionarPlantillas(FE:ColocarPlantilla2)

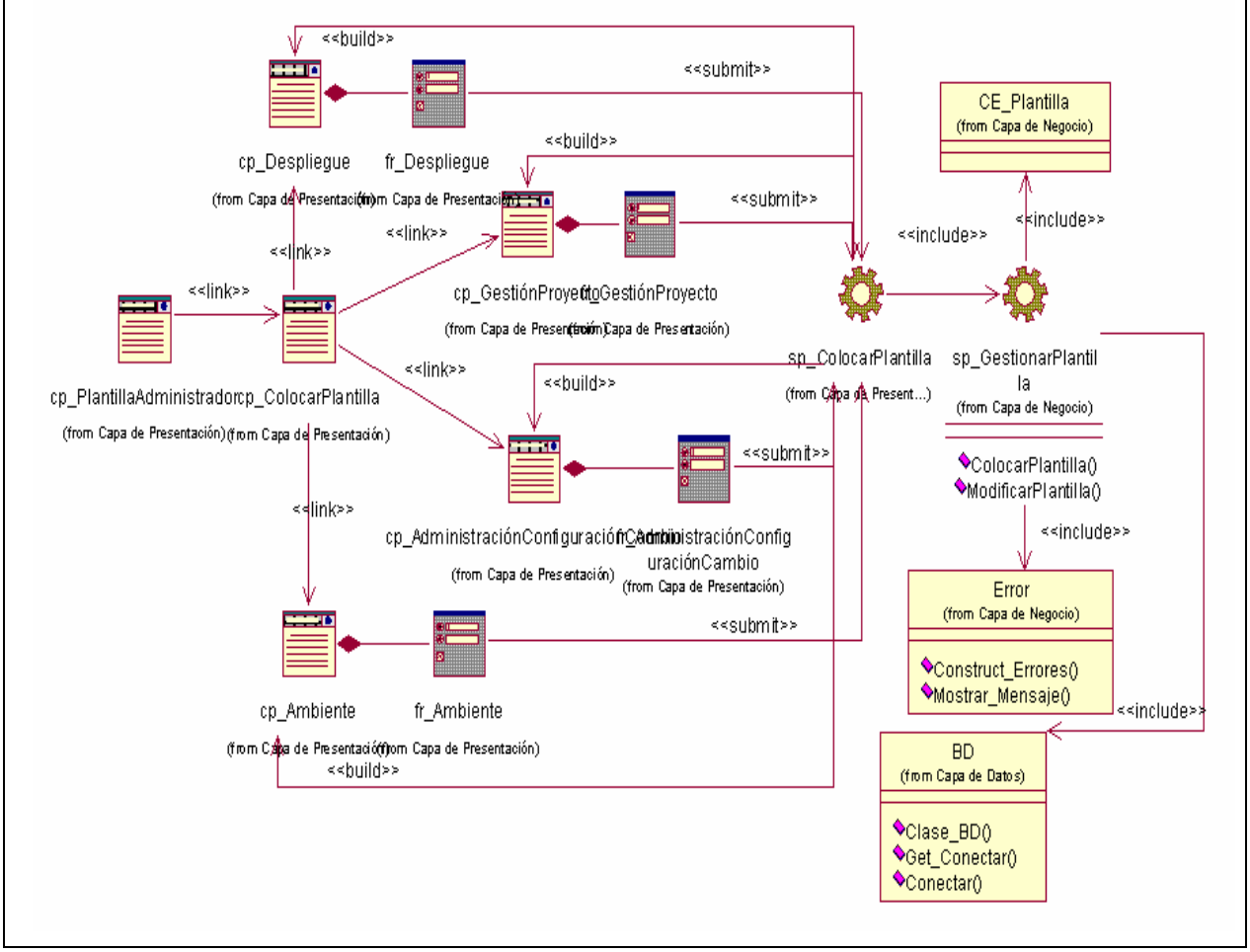


Tabla 3-9. Diagrama de clases. Caso de uso Gestionar plantillas. Parte 4

Diagrama de secuencia. CU Gestionar plantillas. Escenario "Colocar plantillas" (Anexos II, Figura 3-13).

Diagrama de secuencia. CU Gestionar plantillas. Escenario "Cambiar plantillas" (Anexos II, Figura 3-14).

DIAGRAMA DE CLASES: CU_Cambiar_Contraseña

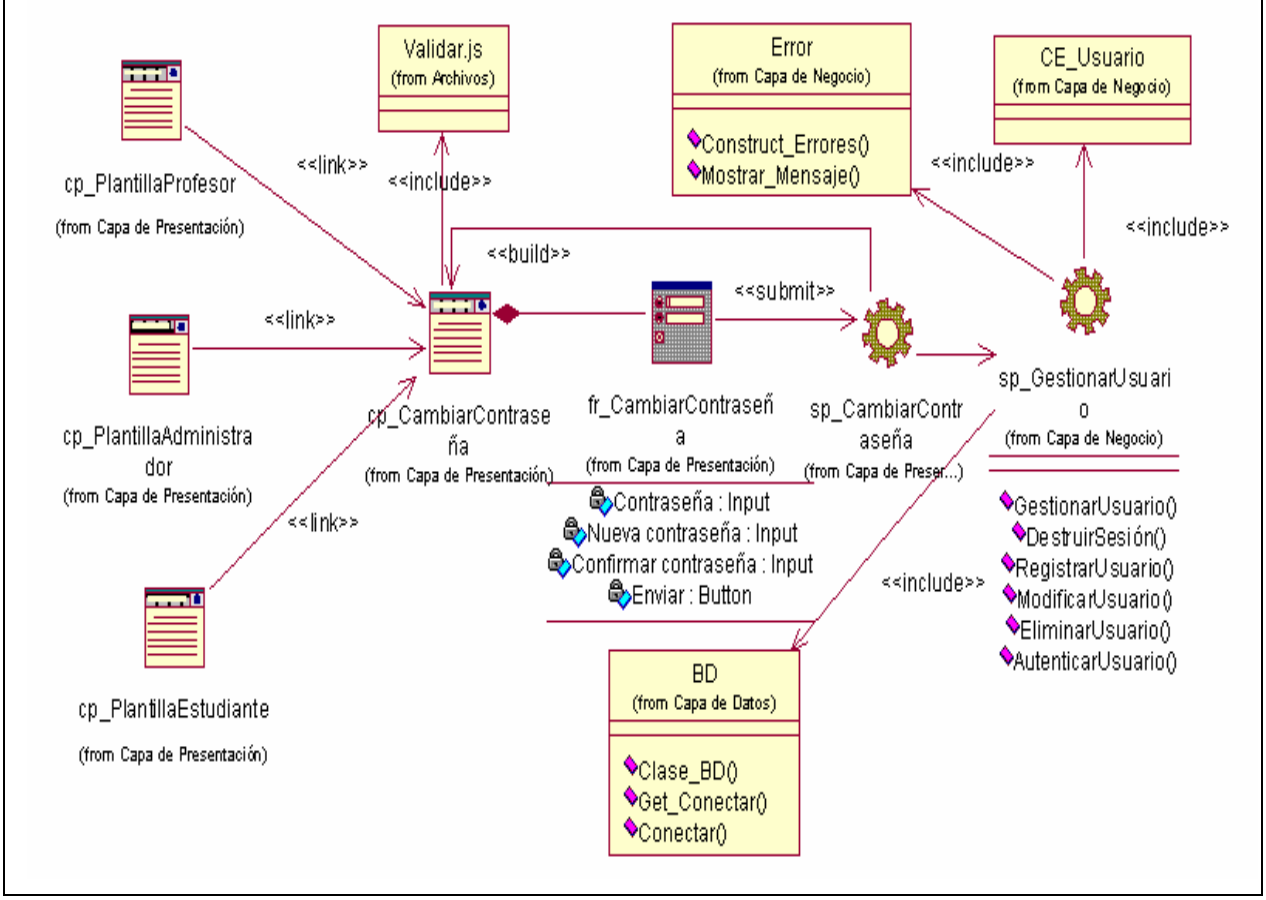


Tabla. 3-10. Diagrama de clases. Caso de uso Cambiar contraseña.

Diagrama de secuencia. CU Cambiar contraseña. Escenario "Cambiar contraseña" (Anexos II, Figura 3-15).

DIAGRAMA DE CLASES: CU_Gestionar_Artefactos

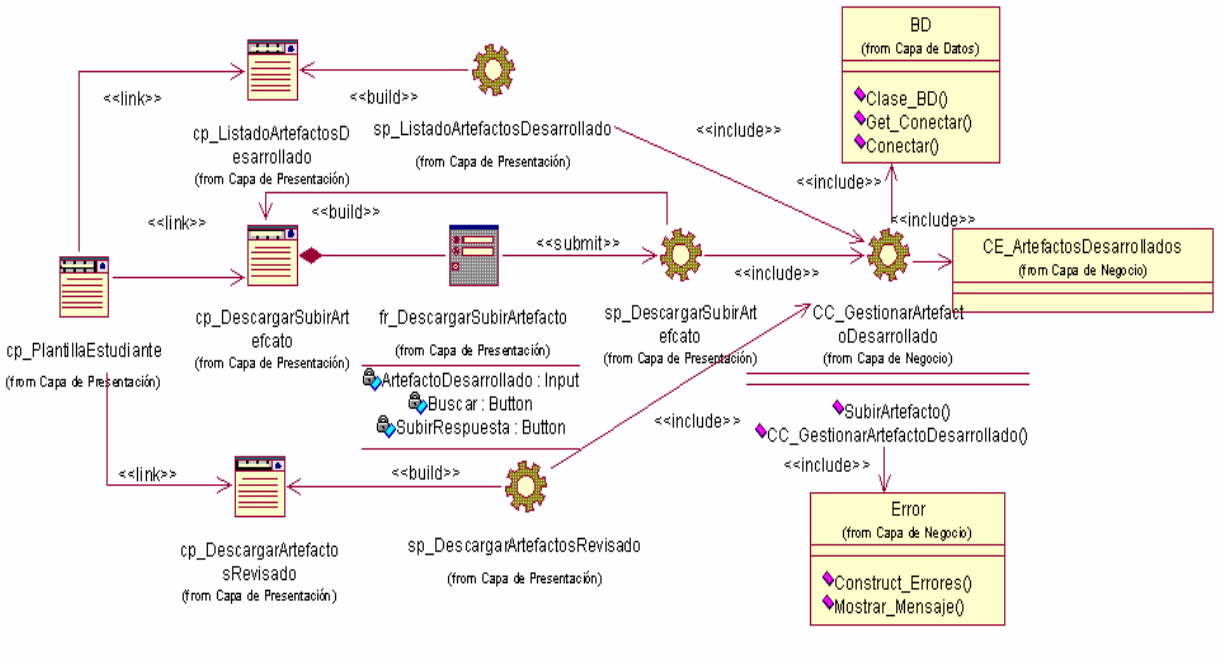


Tabla. 3-11. Diagrama de clases. Caso de uso Gestionar artefacto.

Diagrama de secuencia. CU Gestionar artefactos. Escenario "Descargar y subir artefacto" (Anexos II, Figura 3-16).

DIAGRAMA DE CLASES: CU:Controlar_Intercambios_Profesor_Estudiantes

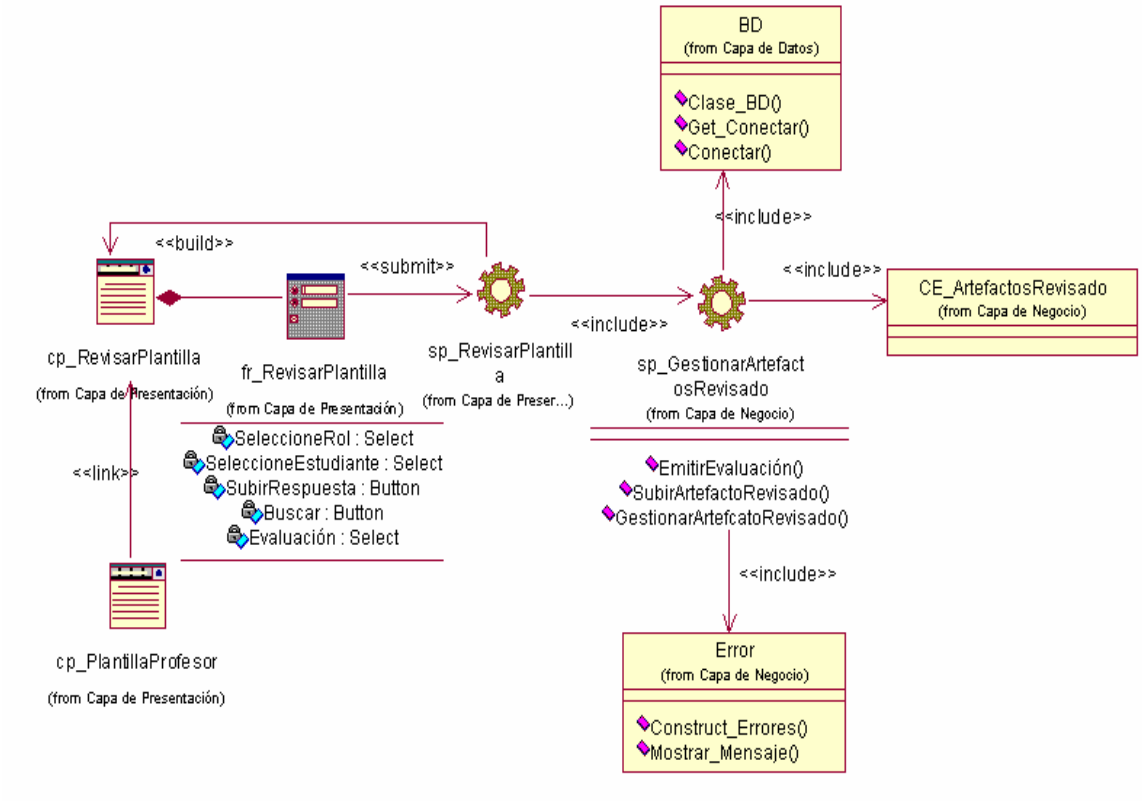


Tabla. 3-12. Diagrama de clases. Caso de uso Controlar intercambios entre el profesor y el estudiante.

Diagrama de secuencia. CU Controlar intercambios entre el profesor y el estudiante. Escenario "Revisar plantilla" (Anexos II, Figura 3-17).

DIAGRAMA DE CLASES: CU:Gestionar_Evaluación

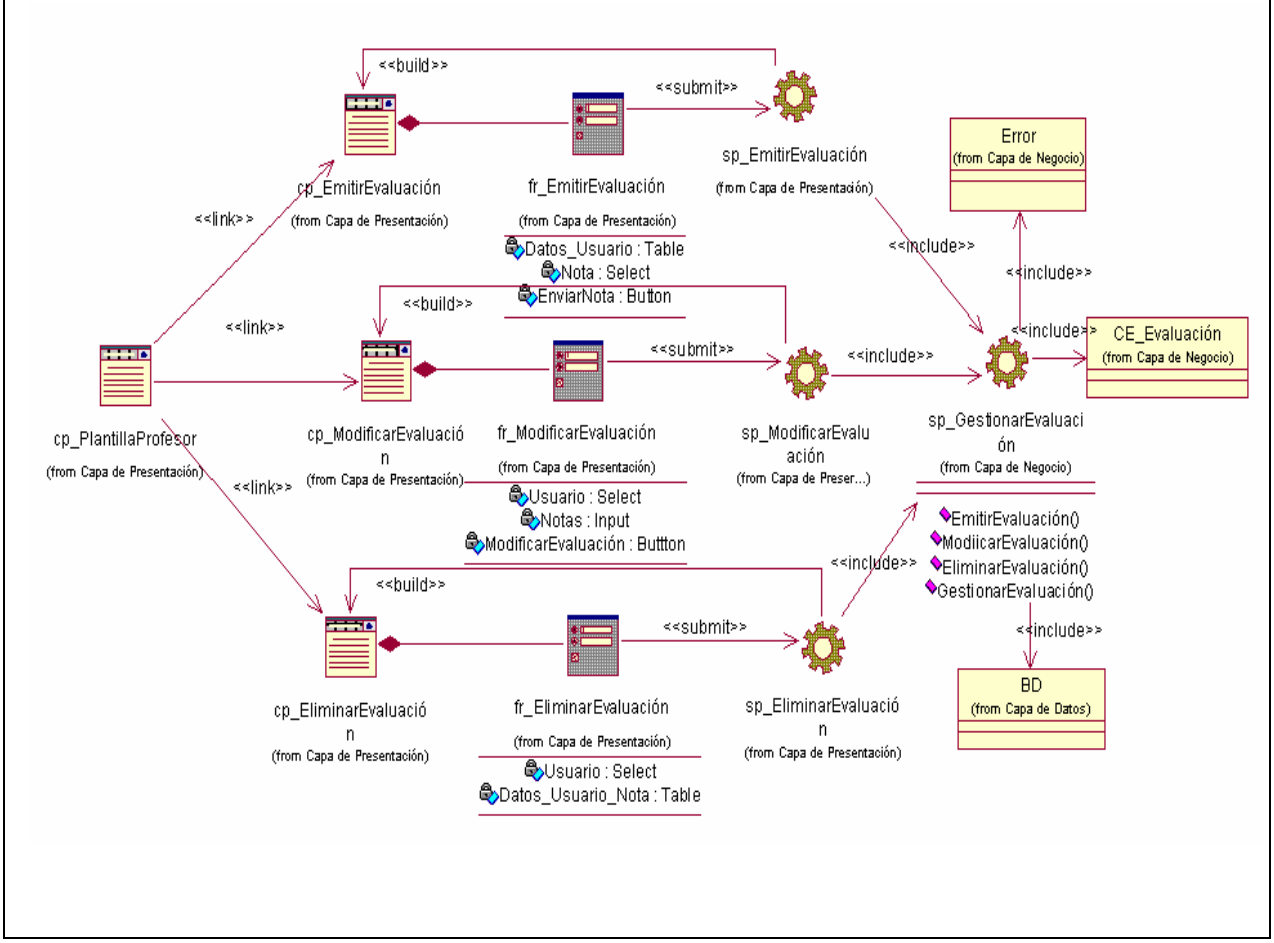


Tabla. 3-13. Diagrama de clases. Caso de uso Controlar intercambios entre el profesor y el estudiante.

Diagrama de secuencia. CU Gestionar evaluación. Escenario "Emitir evaluación" (Anexos II, Figura 3-18).

Diagrama de secuencia. CU Gestionar evaluación. Escenario "Modificar evaluación" (Anexos II, Figura 3-19).

Diagrama de secuencia. CU Gestionar evaluación. Escenario "Eliminar evaluación" (Anexos II, Figura 3-20).

3.2.3 Descripción de las clases del diseño.

En este epígrafe se describen las clases del diseño.

Nombre: CE_Proyecto	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Nombre_Proyecto	Varchar
Tipo_Proyecto	Varchar
Complejidad	Varchar
Descripción	Varchar
Usuario	Varchar
Para cada responsabilidad:	
Nombre:	CE_Proyecto($nombre_proyecto, $tipo_proyecto,$complejidad,$descripción)
Descripción:	Constructor de la clase proyecto.
Nombre:	Get_Nombre_Proyecto
Descripción:	Devuelve el nombre del proyecto.
Nombre:	Get_Tipo_Proyecto
Descripción:	Devuelve el tipò de proyecto.
Nombre:	Get_Complejidad
Descripción:	Devuelve la complejidad del proyecto.
Nombre:	Get_Descripción
Descripción:	Devuelve la descripción del proyecto.
Nombre:	Get_Usuario()
Descripción:	Devuelve el usuario que pertenece al proyecto.

Tipo de clase: CE_Usuario	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Nombre	Varchar
Primer_Apellido	Varchar
Segundo_Apellido	Varchar
Teléfono	Integer
Correo_Electrónico	Varchar
Usuario	Varchar
Contraseña	Varchar
Confirmar_Contraseña	

Para cada responsabilidad:	
Nombre:	CE_Usuario($usuario,$contrasenna,$nombre,$primer_apellido,$segundo_apellido, $telefono,$tipo,$correo,$confirmar_contrasena)
Descripción:	Constructor de la clase usuario.
Nombre:	Get_Nombre
Descripción:	Devuelve el nombre del usuario.
Nombre:	Get_Primer_Apellido
Descripción:	Devuelve el primer apellido del usuario.
Nombre:	Get_Segundo_Apellido
Descripción:	Devuelve el segudno apellido del usuario.
Nombre:	Get_Teléfono
Descripción:	Devuelve el teléfono del recurso humano.
Nombre:	Get_Correo_Electrónico
Descripción:	Devuelve el correo electrónico del usuario.
Nombre:	Get_Usuario
Descripción:	Devuelve el usuario del usuario.
Nombre:	Get_Contraseña
Descripción:	Devuelve la contraseña del usuario.
Nombre:	Get_Confirmar_Contraseña
Descripción:	Devuelve la confirmación de la contraseña del usuario.

Tipo de clase: CE_Roles	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Nombre_Rol	Varchar
Para cada responsabilidad:	
Nombre:	function CE_Roles($nombre_proyecto)
Descripción:	Constructor de la clase roles.
Nombre:	Get_Nombre_Roles()
Descripción:	Devuelve el nombre del rol.

Tipo de clase: CE_Recurso_Humano	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Usuario	Varchar
Nombre_Proyecto	Varchar
Para cada responsabilidad:	
Nombre:	CE_Recurso_Humano($usuario,$nombre_proyecto)
Descripción:	Constructor de la clase recurso humano.
Nombre:	Get_Usuario
Descripción:	Devuelve el usuario del recurso humano.
Nombre:	Get_Nombre_Proyecto()
Descripción:	Devuelve el proyecto en el que se encuentra el recurso humano.

Tipo de clase: CE_Fases	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Nombre_fase	Varchar
Nombre_proyecto	Varchar
Usuario	Varchar
Para cada responsabilidad:	
Nombre:	CE_Fases($nombre_fase,$nombre_proyecto,$usuario)
Descripción:	Constructor de la clase fase.
Nombre:	Get_Nombre_Fase()
Descripción:	Devuelve el nombre de la fase.
Nombre:	Get_Nombre_Proyecto()
Descripción:	Devuelve el nombre del proyecto.
Nombre:	Get_Usuario()
Descripción:	Devuelve el usuario del proyecto.

Tipo de clase: CE_FlujoTrabajo	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Nombre_FlujoTrabajo	Varchar
Nombre_Proyecto	Varchar
Nombre_Rol	Varchar

Para cada responsabilidad:	
Nombre:	CE_FlujosTrabajo($nombre_flujo,$nombre_proyecto,$nombre_rol)
Descripción:	Constructor de la clase CE_FlujoTrabajo.
Nombre:	Get_Nombre_FlujoTrabajo()
Descripción:	Devuelve el nombre del flujo de trabajo.
Nombre:	Get_Proyecto()
Descripción:	Devuelve el nombre del proyecto.
Nombre:	Get_Rol()
Descripción:	Devuelve el nombre el rol.

Tipo de clase: CE_Iteraciones	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Nombre_Iteraciones	Varchar
Para cada responsabilidad:	
Nombre:	CE_Iteraciones($nombre_iteraciones)
Descripción:	Constructor de la clase CE_Iteraciones.
Nombre:	Get_Nombre_Iteraciones()
Descripción:	Devuelve el nombre de las iteraciones.

Tipo de clase: CE_Artefactos	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Nombre_Artefacto	Varchar
Para cada responsabilidad:	
Nombre:	CE_Artefacto($nombre_artefacto)
Descripción:	Constructor de la clase artefacto
Nombre:	Get_Nombre_Artefacto()
Descripción:	Devuelve el nombre del artefacto.

Tipo de clase: CE_Actividades	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Nombre_Actividades	Varchar
Para cada responsabilidad:	

Nombre:	CE_Actividades($nombre_actividad)
Descripción:	Constructor de la clase actividades
Nombre:	Get_Nombre_Actividad()
Descripción:	Devuelve el nombre de la actividad.

Tipo de clase: CE_Profesor	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Usuario	Varchar
Especialidad	Varchar
Para cada responsabilidad:	
Nombre:	CE_Profesor($usuario, $especialidad)
Descripción:	Constructor de la clase profesor.
Nombre:	Get_Usuario()
Descripción:	Devuelve el usuario del profesor.
Nombre:	Get_Especialidad()
Descripción:	Devuelve la especialidad del profesor.

Tipo de clase: CE_Administrador	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Usuario	Varchar
Para cada responsabilidad:	
Nombre:	CE_Administrador($usuario)
Descripción:	Constructor de la clase administrador.

Tipo de clase: CE_Evaluación	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Usuario	Varchar
Usuario_Estudiante	Varchar
Evaluación	Integer
Para cada responsabilidad:	
Nombre:	CE_evaluacion($usuario_estudiante,$nota,$usuario)
Descripción:	Constructor de la clase evaluación.

Nombre:	Get_Usuario()
Descripción:	Devuelve el usuario del profesor.
Nombre:	Get_Usuario_Estudiante()
Descripción:	Devuelve el usuario del estudiante.
Nombre:	Get_Evaluación()
Descripción:	Devuelve la evaluación del estudiante.

Tipo de clase: CE_Plantilla	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Tamaño	Integer
Tipo	Varchar
tmp	Varchar
Tamaño_Máximo	Integer
Destino	Varchar
Título	Varchar
Usuario	Varchar
Para cada responsabilidad:	
Nombre:	CE_Plantilla($tamaño,$tipo,$tmp,$tamaño_máximo,$destino,$título,$usuario)
Descripción:	Constructor de la clase plantilla.
Nombre:	Get_Tamaño()
Descripción:	Devuelve el tamaño del fichero.
Nombre:	Get_Tipo()
Descripción:	Devuelve el tipo (si es un .doc, .exe, .ppt, etc) del fichero.
Nombre:	Get_tmp()
Descripción:	Devuelve la dirección del fichero.
Nombre:	Get_Tamaño_Máximo()
Descripción:	Devuelve el tamaño máximo del fichero (en el este caso el fichero no se puede exceder de los 5 Megas).
Nombre:	Get_Destino()
Descripción:	Devuelve la dirección hacia donde va a ir el fichero después de subido al sistema.
Nombre:	Get_Título()
Descripción:	Devuelve el título del fichero.
Nombre:	Get_Usuario()
Descripción:	Devuelve el usuario que sube el fichero.

Tipo de clase: CE_ArtefactoRevisado	
Tipo de clase: Entidad	
Atributo:	**Tipo:**
Usuario	Varchar
Tamaño	Integer
Tipo	Varchar
tmp	Varchar
Tamaño_Máximo	Integer
Destino	Varchar
Título	Varchar
Rol	Varchar
Para cada responsabilidad:	
Nombre:	CE_ArtefactosRevisados($usuario,$tamaño,$tipo,$tmp,$tamaño_máximo, $destino,$título,$rol,
Descripción:	Constructor de la clase artefacto revisado.
Nombre:	Get_Usuario()
Descripción:	Devuelve el usuario del profesor.
Nombre:	Get_Tamaño()
Descripción:	Devuelve el tamaño del fichero.
Nombre:	Get_Tipo()
Descripción:	Devuelve el tipo (si es un .doc, .exe, .ppt, etc) del fichero.
Nombre:	Get_tmp()
Descripción:	Devuelve la dirección del fichero.
Nombre:	Get_Tamaño_Máximo()
Descripción:	Devuelve el tamaño máximo del fichero (en el este caso el fichero no se puede exceder de los 5 Megas).
Nombre:	Get_Destino()
Descripción:	Devuelve la dirección hacia donde va a ir el fichero después de subido al sistema.
Nombre:	Get_Título()
Descripción:	Devuelve el título del fichero.
Nombre:	Get_Rol()
Descripción:	Devuelve el rol del estudiante.

Tipo de clase: CE_ArtefactoDesarrollado
Tipo de clase: Entidad

Atributo:	**Tipo:**
Usuario_Estudiante	Varchar
Tamaño	Integer
Tipo	Varchar
tmp	Varchar
Tamaño_Máximo	Integer
Destino	Varchar
Título	Varchar

Para cada responsabilidad:	
Nombre:	CE_ArtefactosRevisados($usuario_estudiante,$tamaño,$tipo,$tmp,$tamaño_máximo,$destino,$título)
Descripción:	Constructor de la clase artefacto desarrollado.
Nombre:	Get_Usuario_Estudiante()
Descripción:	Devuelve el usuario del estudiante.
Nombre:	Get_Tamaño()
Descripción:	Devuelve el tamaño del fichero.
Nombre:	Get_Tipo()
Descripción:	Devuelve el tipo (si es un .doc, .exe, .ppt, etc) del fichero.
Nombre:	Get_tmp()
Descripción:	Devuelve la dirección del fichero.
Nombre:	Get_Tamaño_Máximo()
Descripción:	Devuelve el tamaño máximo del fichero (en el este caso el fichero no se puede exceder de los 5 Megas).
Nombre:	Get_Destino()
Descripción:	Devuelve la dirección hacia donde va a ir el fichero después de subido al sistema.
Nombre:	Get_Título()
Descripción:	Devuelve el título del fichero.

Tipo de clase: CC_GestionarRecursoHumano	
Tipo de clase: Controladora	
Atributo:	**Tipo:**

Para cada responsabilidad:	
Nombre:	Constructor_Gestionar_RecursoHumano()

Descripción:	Construir la clase gestora de recurso humano.
Nombre:	Insertar_Recurso_Humano()
Descripción:	Inserta en la tabla recurso humano los datos que le entre el profesor.
Nombre:	Eliminar_Recurso_Humano ()
Descripción:	Elimina el recurso humano mediante su usuario que pertenece al proyecto que el profesor esté navegando en ese momento.
Nombre:	Modificar_Recurso_Humano ()
Descripción:	Modifica los datos del recurso humano seccionado por el usuario del mismo, que pertenece al proyecto que le profesor esté navegando.
Nombre:	Destruir_Session_Recurso_Humano()
Descripción:	Destruye la sesión del estudiante(usuario, contraseña)

Tipo de clase: CC_GestionarProyecto	
Tipo de clase: Controladora	
Atributo:	**Tipo:**
Para cada responsabilidad:	
Nombre:	Constructor_Gestionar_Proyecto()
Descripción:	Constructor de la clase.
Nombre:	Insertar_Proyecto()
Descripción:	Inserta en la tabla proyecto los datos que le entre el profesor.
Nombre:	Eliminar_Proyecto()
Descripción:	Elimina el proyecto.
Nombre:	Modificar_Proyecto()
Descripción:	Modifica los datos del proyecto por el cual el profesor esté navegando.
Nombre:	Destruir_Sessióm_Proyecto
Descripción:	Destruye la sesión del proyecto.

Tipo de clase: CC_Usuario	
Tipo de clase: Controladora	
Atributo:	**Tipo:**

Para cada responsabilidad:	
Nombre:	Constructor_Gestionar_Usuario()
Descripción:	Constructor de la clase.
Nombre:	Autenticar_Usuario()
Descripción:	Permite acceder a los privilegios del sistema según el rol (profesor, administrador, estudiante) que tenga el usuario.
Nombre:	Registrar_Usuario()
Descripción:	Registra en la tabla profesor y administrador los datos de los usuarios que el administrador entra.
Nombre:	Eliminar_Usuario()
Descripción:	Elimina el administrador del sistema a un usuario mediante la categoría (administrador, profesor) que tenga ese usuario.
Nombre:	Modificar_Usuario()
Descripción:	Modifica el administrador del sistema a un usuario mediante el usuario (sólo administrador y profesor) que tenga ese usuario.
Nombre:	Destruir_Sesión_Usuario()
Descripción:	Destruye la sesión del usuario.
Nombre:	Cambiar_Contraseña()
Descripción:	Permite cambiar la contraseña según el rol (profesor, administrador, estudiante) que tenga el usuario.

Tipo de clase: CC_GestionarProcesoDesarrollo	
Tipo de clase: Controladora	
Atributo:	**Tipo:**
Para cada responsabilidad:	
Nombre:	Constructor_Gestionar_Proceso_Desarrollo()
Descripción:	Constructor de la clase.
Nombre:	Insertar_Trabajo_Desarrollo($CE_fases, $CE_iteraciones, $CE_flujos, $CE_actividades, $CE_artefactos)
Descripción:	Inserta las fases con sus iteraciones, dentro de éstos los flujos de trabajo con sus actividades y artefactos.

Tipo de clase: CC_GestionarPlantilla	
Tipo de clase: Controladora	
Atributo:	**Tipo:**

Para cada responsabilidad:	
Nombre:	Constructor_Gestionar_Plantilla()
Descripción:	Constructor de la clase.
Nombre:	Colocar_Plantilla($CE_plantilla)
Descripción:	Inserta las plantillas el administrador en la tabla plantilla.
Nombre:	Modificar_Plantilla($CE_plantilla)
Descripción:	Modica la plantilla que el administrador desea reemplazar en la tabla plantilla.

Tipo de clase: CC_GestionarArtefactoRevisado	
Tipo de clase: Controladora	
Atributo:	**Tipo:**
Para cada responsabilidad:	
Nombre:	Constructor_Gestionar_Artefacto_Revisado()
Descripción:	Constructor de la clase.
Nombre:	Subir_Artefacto_Revisado($CE_artefactos_revisado)
Descripción:	El profesor sube los artefactos ya revisados del estudiante.
Nombre:	Emitir_Evaluacion($CE_evaluacion)
Descripción:	Inserta en la tabla evaluación la calificación que el profesor le asigna al estudiante.

Tipo de clase: CC_GestionarArtefactoDesarrollado	
Tipo de clase: Controladora	
Atributo:	**Tipo:**
Para cada responsabilidad:	
Nombre:	Constructor_Gestionar_Artefacto_Desarrollado()
Descripción:	Constructor de la clase.
Nombre:	Subir_Artefactos($CE_artefactos_desarrollados)
Descripción:	Sube el artefacto a la tabla artefactos desarrollados que el estudiante desarrolló.

Tipo de clase: CC_GestionarEvaluación	
Tipo de clase: Controladora	
Atributo:	**Tipo:**

Para cada responsabilidad:	
Nombre:	Constructor_Gestionar_Evaluación()
Descripción:	Constructor de la clase.
Nombre:	Emitir_Evaluación()
Descripción:	Inserta en la tabla evaluación la nota que le de el profesor al estudiante.
Nombre:	Eliminar_Evaluación()
Descripción:	Elimina una evaluación.
Nombre:	Modificar_Evaluación()
Descripción:	Modifica la evaluación de un estudiante.

3.2.4 Diagrama de clases persistentes

El diagrama de clases persistentes muestra las relaciones existentes entre las clases persistentes, aquellas que tienen la capacidad de mantener su valor en el espacio y el tiempo, más adelante convertido en el modelo de datos. El diagrama de clases persistentes realizado se define en la siguiente tabla.

DIAGRAMA DE CLASES PERSISTENTES

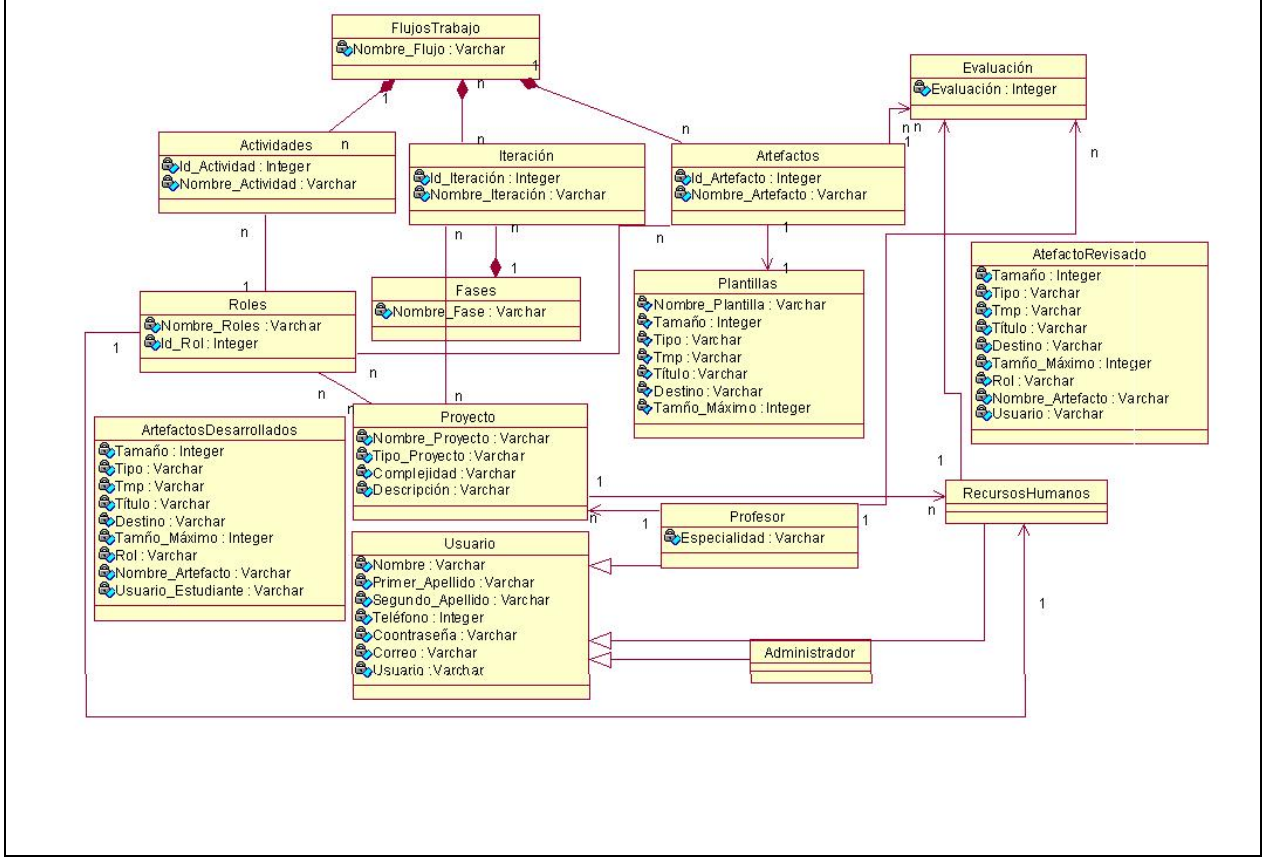


Tabla 3-14. Diagrama de clases persistentes.

3.2.5 Modelo de datos

El modelo de datos no es más que la representación física final de la base de datos obtenida a partir de las clases persistentes. El modelo de datos realizado se define en la siguiente tabla.

MODELO DE LA BASE DE DATOS

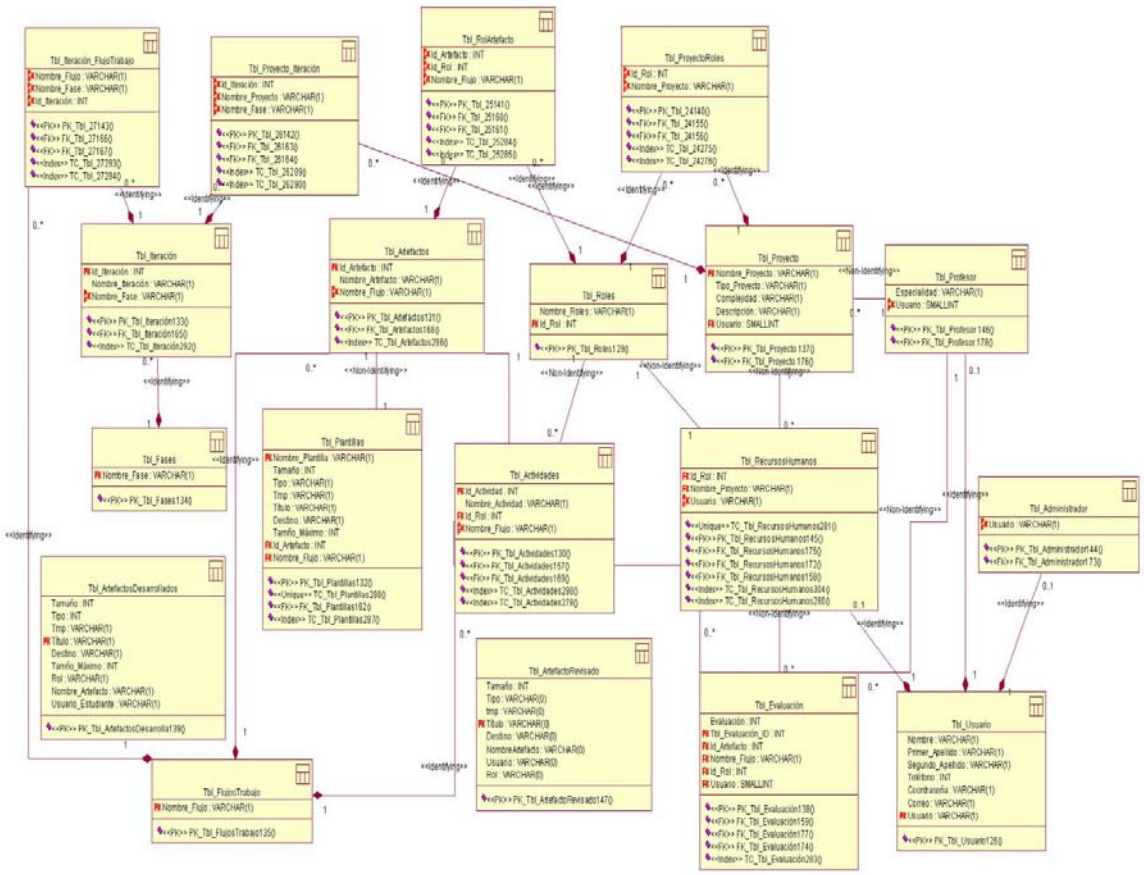


Tabla 3-15. Modelo de datos.

3.3 Principios de diseño

3.3.1 Estándares en la interfaz de la aplicación.

La interfaz de la aplicación es el medio mediante el cual los usuarios interactúan con el sistema, por lo que ésta debe ser lo más amigable posible y de fácil navegación para que el usuario se sienta identificado en la misma.

El producto presenta en todas las páginas el esquema Cabecera-Navegador-Contenido, uno de los más usados actualmente. La cabecera contiene el logotipo de la aplicación en la esquina superior izquierda, seguido con el nombre de la aplicación. En la parte lateral izquierda se encuentra una barra de navegación con enlaces de texto para cada uno de los módulos de la

aplicación. En el navegador, según el módulo activo, se incluyen los enlaces a las distintas secciones. En el área del contenido se muestran los formularios de entrada, tablas de contenido, etc.

Figura 3-21. Esquema de página.

Se utilizan para el diseño las tablas y plantillas, dado que son 100% compatibles con todos los navegadores, hasta en sus versiones más antiguas, a diferencia de los marcos.

Se utiliza también hojas de estilos para guardar la configuración del diseño de todas las páginas. Estas hojas de estilo establecen el tipo y tamaño de fuente de los distintos elementos de cada página. Se utiliza en general la familia de fuentes Verdana, Arial, Helvetica, sans-serif, de tamaño entre 11, 12 y 14 píxeles, según la información mostrada. También se establecen estilos en los select utilizando las familias de fuente Verdana, de tamaño de 10 píxel y con un color de #CCCCCC y en los input y tablas se utilizan los colores azul claro y azul oscuro con un color de letra #064293 y tamaño de 12 píxel. La hoja de estilos también establece el color de los vínculos, entre otros.

3.4 Estándares de codificación.

Para la implementación del sistema se estableció un estándar de codificación a utilizar, dividido en los siguientes aspectos:

Comentarios:

Los comentarios se definen comenzando con los caracteres ***/**** y terminando con ****/*** para los comentarios de varias líneas.

Para poner el nombre de los métodos y las consultas a la base de datos, y para una mejor delimitación de éstos se utiliza el siguiente comentario:

//----------NOMBRE METODO

//----------NOMBRE CONSULTA

Declaraciones:

Las variables utilizadas se encontraran en minúscula y separadas con un underscore. Ejemplo: ***private*** *$primer_apellido;*

Para declarar una clase se comenzará con una CE_ mayúscula seguido de un sustantivo en el caso de las clases entidades, en el caso de las clases controladoras por un CC_ mayúscula seguido de un verbo en infinitivo con un sustantivo y en el caso de las interfaces por verbo en infinitivo con un sustantivo; en caso de estar compuesto el nombre de cada una de las clases se separará con un underscore.

Ejemplo: ***class*** *CE_usuario,* ***class*** *CC_gestionar_usuario,* ***class*** registrar_usuario

Espacios en blanco:

Se colocarán espacios en blanco entre operadores lógicos-aritméticos y sus operandos. Ejemplo:

```
If( $iteraciones_definidas == 'modelo_de_casos_uso_del_negocio' )

{

$insert= mysql_query ("INSERT INTO tabla(a, b)

                    VALUES('$a','$b')");

}
```

Miscelánea:

Las llaves para abrir y cerrar un método o un bloque de control de flujo se encuentran al mismo nivel del bloque al que pertenecen. Ejemplo:

```
while($row!== true )
```

```
{

    for($i=0 ; $i<count($row) ; $i++)

      {

            $elemento =$row[$i];

       }

}
```

3.5 Modelo de despliegue.

El modelo de despliegue visualiza en nodos físicos la distribución de los componentes de software. Además representa una correspondencia entre la arquitectura software y la arquitectura del sistema.

El sistema estará estructurado según la metodología Web con un cliente y un servidor. El servidor de BD MySQL y el servidor Web Apache. La PC cliente se comunicará con la del servidor de la aplicación a través del protocolo HTTP y ésta a su vez con el servidor de la base de datos a través del protocolo TCP\IP. El cliente podrá visualizar la aplicación con el Internet Explore 4.0 o superior o cualquier browser.

El diagrama de despliegue realizado se define en la siguiente tabla.

DIAGRAMA DE DESPLIEGUE

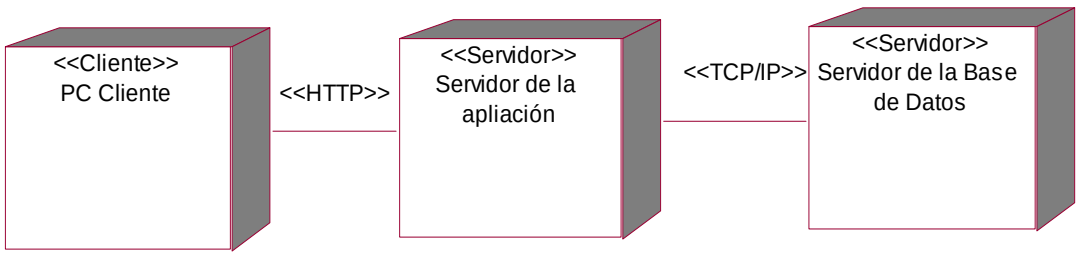


Tabla 3-16. Modelo de despliegue.

3.6 Modelo de implementación

"El modelo de implementación describe cómo los elementos del modelo de diseño, como las clases, se implementan en términos de componentes, como los ficheros de código fuente, ejecutables, etc. El modelo de implementación describe también cómo se organizan los

componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados, y cómo dependen los componentes unos de otros"(JACOBSON, IVAR *et al.* 2004).

Los diagramas de componentes se realizaron agrupándolos en tres paquetes; uno para los casos de uso que inicia el profesor, el otro para el caso de uso de iniciado por el estudiante y el último para los casos de uso iniciado por el administrador y el usuario del sistema (generalización de los actores profesor, estudiante y administrador); utilizando la arquitectura de tres capas definida en los inicios de este capítulo. Por último se realizó el diagrama de componentes de la base de datos de la aplicación.

Los tres diagramas de componentes agrupados por paquetes se definen en las tablas siguientes.

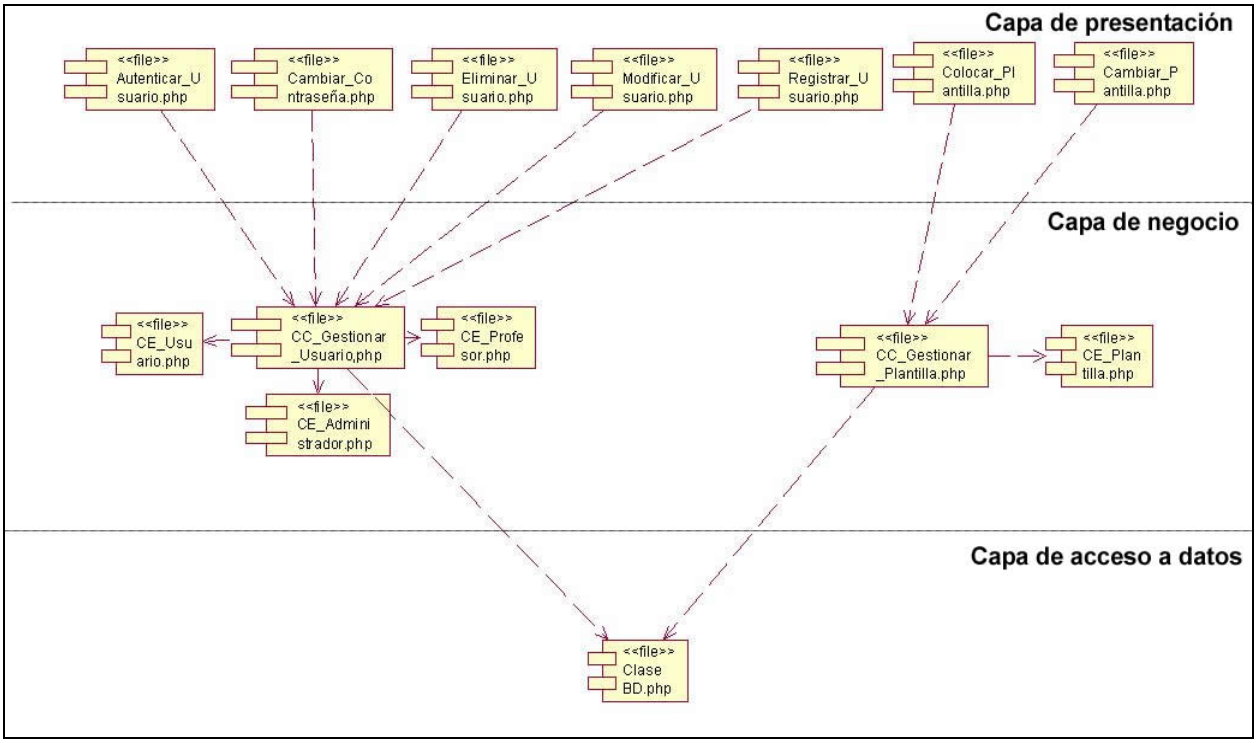


Tabla 3-17. Diagrama de componentes. Paquete administrador y usuario del sistema.

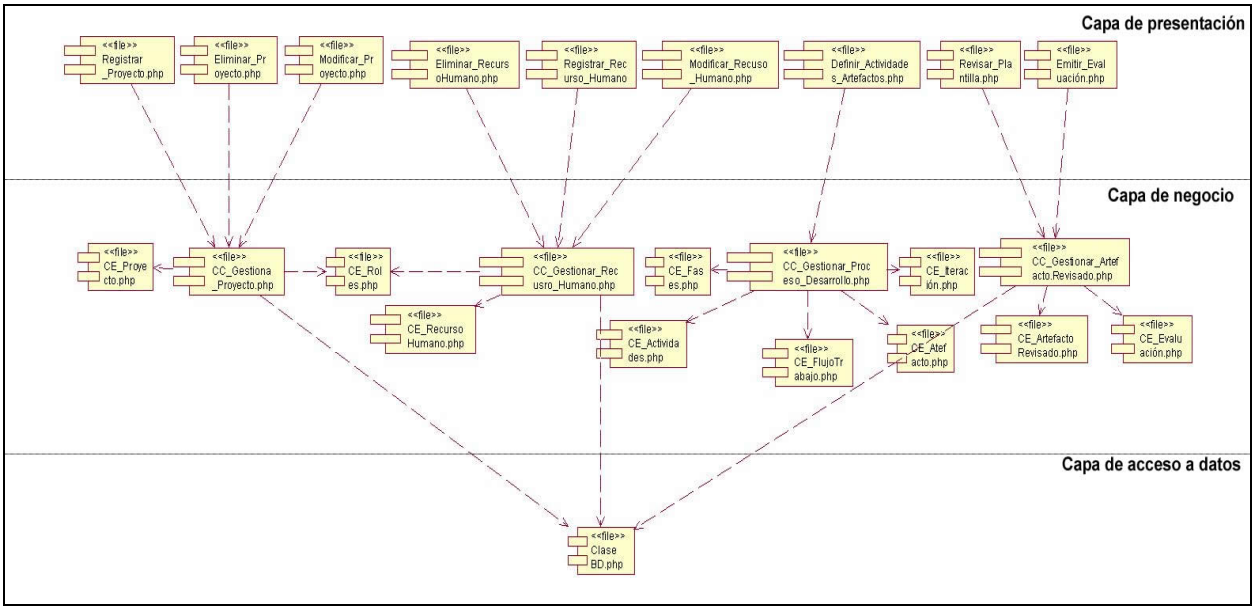


Tabla 3-18. Diagrama de componentes. Paquete profesor.

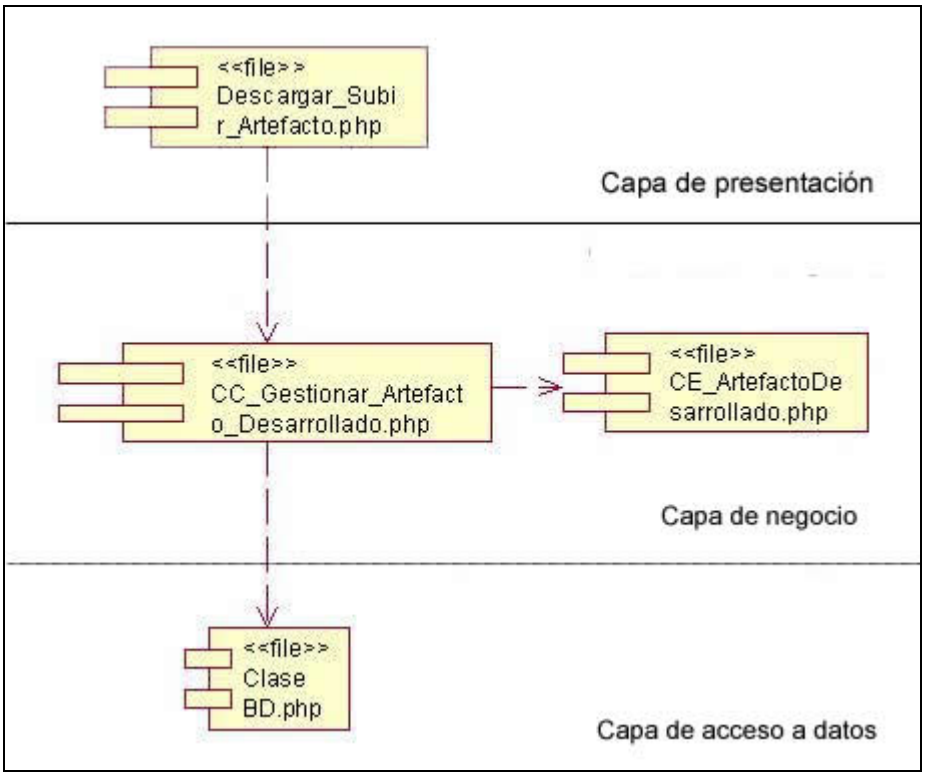


Tabla 3-19. Diagrama de componentes. Paquete estudiante.

Conclusiones

En la realización de este capítulo se hizo un estudio de los patrones de arquitectura seleccionado para el desarrollo de HADIS el patrón de arquitectura de tres capas, siguiendo este patrón se modeló el diseño con los diagramas de clases con extensiones Web y sus diagramas de secuencia por flujo de evento, a partir de los diagramas de clases se seleccionaron las entidades que iban a formar parte del diagrama de clases persistentes y éste darle entrada al modelo de la base de datos que soportará la aplicación. Luego se entró en el flujo de trabajo de implementación desarrollando los diagramas de componentes agrupando los casos de usos en términos de los privilegios que tienen los usuarios del sistema y siguiendo el estilo el patrón de arquitectura usado, y el modelo de despliegue a grandes rasgos como funciona la aplicación físicamente. Ya por último se analizaron los principios de diseño de la interfaz de la aplicación y los tratamientos de excepciones que se utilizaron en el desarrollo de HADIS.

CONCLUSIONES GENERALES

Con la realización de este trabajo se demostró la necesidad de diseñar e implementar una herramienta para la automatización de los procesos de administración, planificación y evaluación en los proyectos docentes de la asignatura de Ingeniería de Software.

Luego de un análisis de las tecnologías más usadas en la actualidad para la construcción de sistemas informáticos, se elaboró la propuesta de utilizar, como lenguaje de programación PHP, el gestor de bases de datos MySQL, el servidor Web Apache y la metodología de desarrollo de software RUP para la construcción de la solución propuesta, junto a otras tecnologías como son la técnica de desarrollo Web para crear aplicaciones interactivas Ajax y las hojas de estilo CSS.

Se modeló el negocio propuesto, identificándose los actores y trabajadores, así como las actividades que son objetos de automatización.

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 compresión de la funcionalidad que brindan.

Se diseñó el sistema, a través de diagramas de clases Web, el diagrama de clases persistentes y los diagramas de secuencia. 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 los diagramas de componentes.

Se plantearon los principios a seguir en el diseño de la interfaz de usuario y algunas convenciones a respetar durante la escritura del código fuente.

Luego de todo este proceso de trabajo se puede concluir que HADIS es un sistema que da solución a la situación problémica que lo originó y que su explotación significará una mejora considerable para el desarrollo de proyectos docentes de la asignatura de ingeniería de software, brindándoles a los profesores encargados de esta materia llevar a cabo un control estricto de todo el trabajo y la evaluación del estudiante.

RECOMENDACIONES

Se recomienda:

- Implementar el resto de las funcionalidades propuestas para lograr que la aplicación HADIS se encuentre en total funcionamiento para su uso en la universidad.
- Acceder a la aplicación mediante el dominio UCI, para que sólo tengan acceso los profesores, estudiantes de tercer año y cualquier otra persona vinculada a la misma.
- Integrar la herramienta HADIS a la plataforma que rige el proceso docente de la universidad: Entorno Virtual de Aprendizaje en la asignatura de Ingeniería de Software.
- Anexar la herramienta HADIS al sistema de control del proceso docente de la universidad Akademos para que la nota emitida se refleje directamente en el mismo.
- Realizar nuevas versiones de la herramienta HADIS adaptada a otras metodologías de desarrollo de software.

BIBLIOGRAFIA CONSULTADA

- ---.JACOBSON, Ivar; RUMBAUGH, James; BOOCH, Grady, El proceso unificado de desarrollo.2000. Addison Wesley.
- ---.WELICKI, L. *Patrones y Antipatrones: una Introducción - Parte II*, [Página Web]. 2005 [2006]. Disponible en:

  http://www.microsoft.com/spanish/msdn/comunidad/mtj.net/voices/MTJ_3317.asp
- ---.PRESSMAN, R. S. Ingeniería de Software. Un enfoque práctico. España, Concepción Fernández Madrid, 2002.
- ---.LARMAN, Craig UML y patrones 1999, Prentice Hall Iberoamericana.
- ---.Rational Unified Process (ayuda de la Suite de Rational Rose2003)

REFERENCIA BIBLIOGRAFICA

- ---.(DIS), D. D. I. D. S. *Introducción a la Ingeniería de Software.*: *Ingeniería de Software I*. http://teleformación.uci.cu, Universidad de las Ciencias Informáticas, 2005.

- ---. (DTP), D. D. T. D. L. P. *Programación cliente-servidor. Aplicaciones Web. Diferentes tipos de aplicaciones Web. Concepto de hipertexto y aplicaciones hipermedia.*: *Programación III* http://dprogramacion.uci.cu/p3/cgi-bin/admin/coments/files/10179.doc, Universidad de las Ciencias Informáticas, 2003.

- ---.CARO, P. S. and N. H. KAHLER. *Unified Modeling Language UML*, [Página Web]. Universidad de Chile: Departamento de Ciencias de la Computación 1996. [2006]. Disponible en: http://www.dcc.uchile.cl/~psalinas/uml

- ---.CONCEPCIÓN, P. *Planificación de Proyectos de Software*, 2005. 1.

- ---.DJOHNSON. *Mobile Marketing a new age strategy*, 2006. 1.

- ---.GONZÁLEZ, R. R. *La educación desde un enfoque histórico social: importancia para el desarrollo humano*, 2007. 1.

- ---.HUERTA, A. V. *Sistemas de detección de intrusos*, [PDF]. 2005. [2006]. Disponible en: http://andercheran.upv.es/~toni/personal/transpas-ids.pdf

- ---.JACOBSON. *Applying UML in The Unified Process*, [Presentación]. 1998. [2006]. Disponible en: http://www.rational.com/uml

- ---.JACOBSON, I.; G. BOOCH*, et al. El proceso unificado de desarrollo de software*. La Habana, Editorial Félix Varela 2004. 435 p. I-S-B-N

- ---.SCHUMACHER, R. and A. LENTZ. *Dispelling the Myths*, [Página Web]. MySQL AB, 2006. [2006]. Disponible en:

  http://dev.mysql.com/tech-resources/articles/dispelling-the-myths.html

- ---.SEPULVEDA, D. *Protocolos Seguros para el Web*, [Página Web]. 2006. [2006]. Disponible en: http://www.tejedoresdelweb.com/307/article-5670.html

- ---.WELICKI, L. *Patrones y Antipatrones: una Introducción - Parte II*, [Página Web]. 2005 [2006]. Disponible en:

  http://www.microsoft.com/spanish/msdn/comunidad/mtj.net/voices/MTJ_3317.asp

- ---.WIKIMEDIA FOUNDATION, I. *AJAX*, [Página Web]. Wikimedia Foundation, Inc., 2007. [2007]. Disponible en: http://es.wikipedia.org/wiki/AJAX

- ---. *Hojas de estilo en cascada*, [Página Web]. Wikimedia Foundation, Inc., 2006a. [2006]. Disponible en: http://es.wikipedia.org/wiki/Hojas_de_estilo_en_cascada

- ---. *Lenguaje de programación*, [Página Web]. Wikimedia Foundation, Inc.

  , 2006b. [2006]. Disponible en:

  http://es.wikipedia.org/wiki/Lenguaje_de_programacion

- ---. *MySQL*, [Página Web]. Wikimedia Foundation, Inc., 2006c. [2006]. Disponible en: http://es.wikipedia.org/wiki/MySQL

- ---. *Proyecto*, [Página Web]. Wikimedia Foundation, Inc., 2006d. [2006]. Disponible en: http://es.wikipedia.org/wiki/Proyecto

- ---. *Servidor HTTP Apache*, [Página Web]. Wikimedia Foundation, Inc.

  , 2006e. [2006]. Disponible en: http://es.wikipedia.org/wiki/Servidor_HTTP_Apache

- ---. *Servidor web*, [Página Web]. Wikimedia Foundation, Inc., 2006f. [2006]. Disponible en: http://es.wikipedia.org/wiki/Servidor_web

- ---. *Sistema de gestión de base de datos*, [Página Web]. Wikimedia Foundation, Inc., 2006g. [2006]. Disponible en: http://es.wikipedia.org/wiki/DBMS

ANEXOS

ANEXO I. MODELO DE NEGOCIO PROPUESTO.

Figura 2-1. Diagrama de actividades. CU del negocio Asignar proyecto de curso.

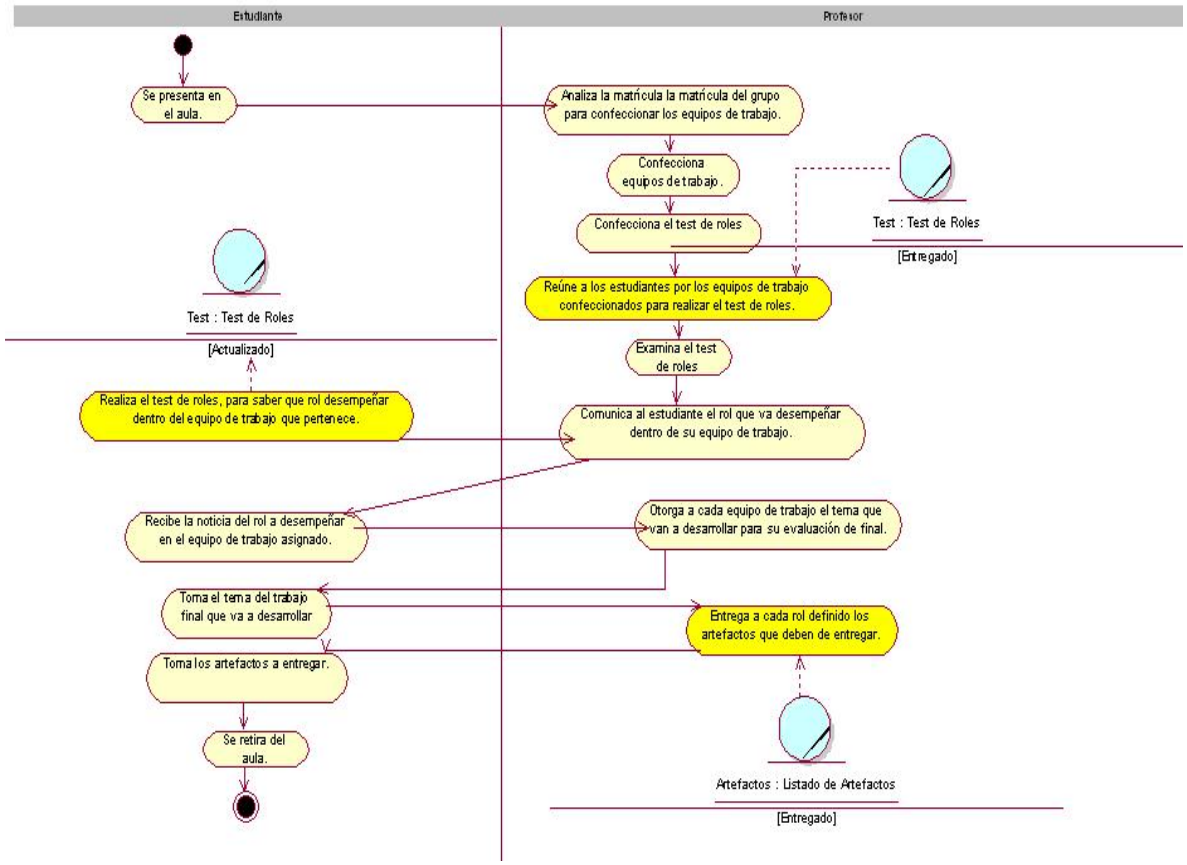
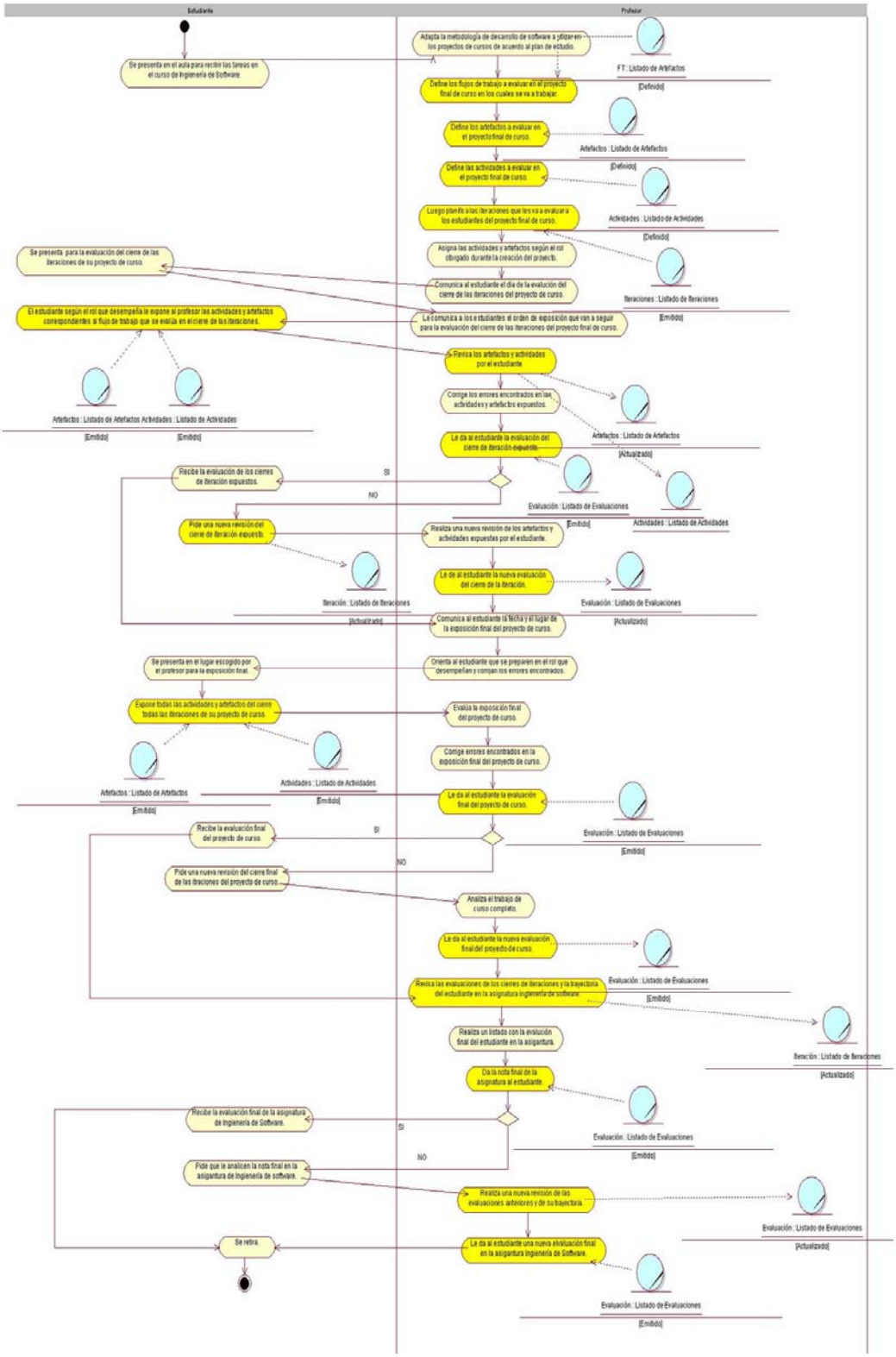


Fig.1-2. Diagrama de actividades. CU del negocio Desarrollar proyecto de curso.



ANEXO II. MODELO DEL DISEÑO PROPUESTO.

Figura.3-2. Diagrama de secuencia. Escenario Autenticar usuario.

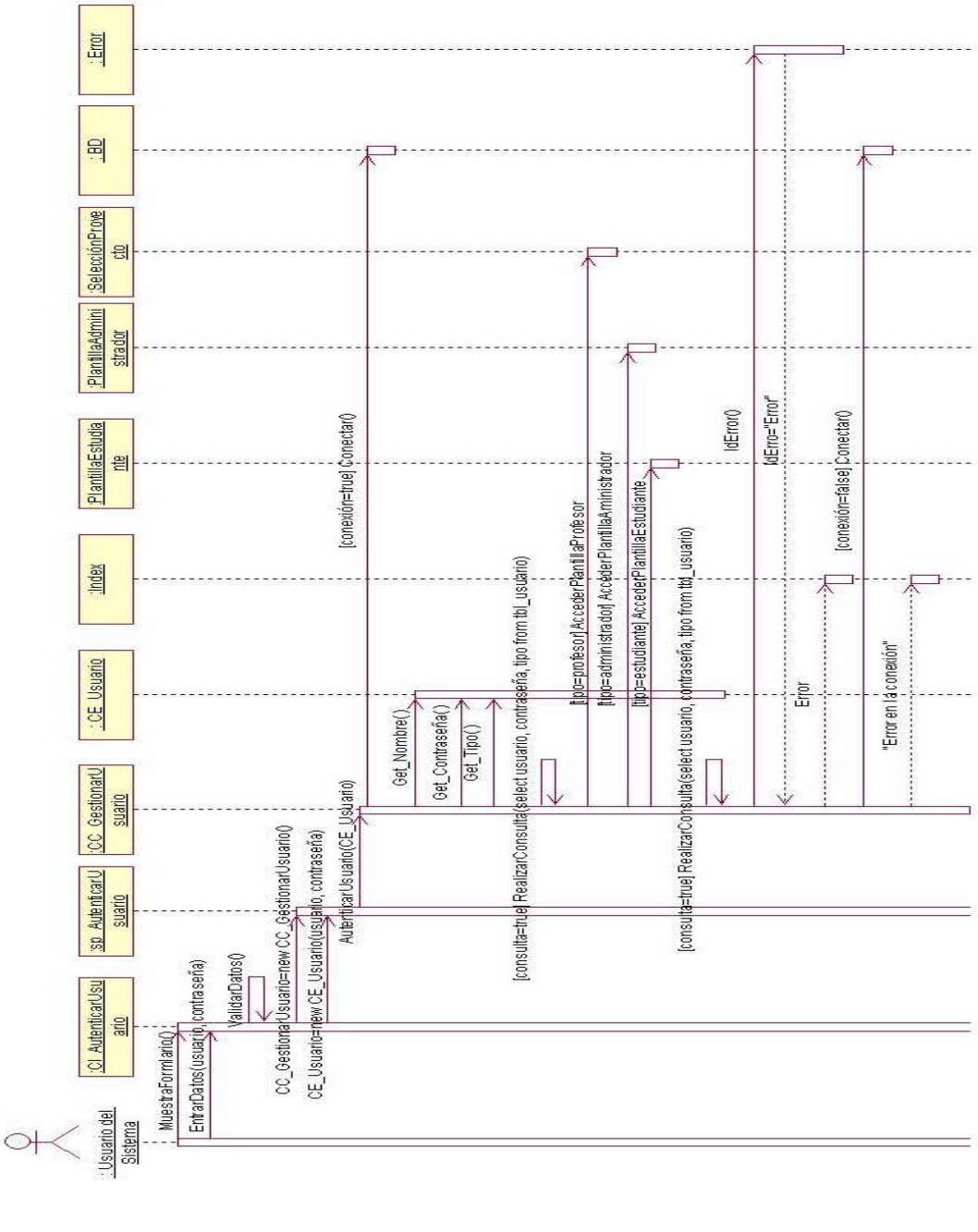


Figura.3-3. Diagrama de secuencia. Escenario Registrar recurso humano.

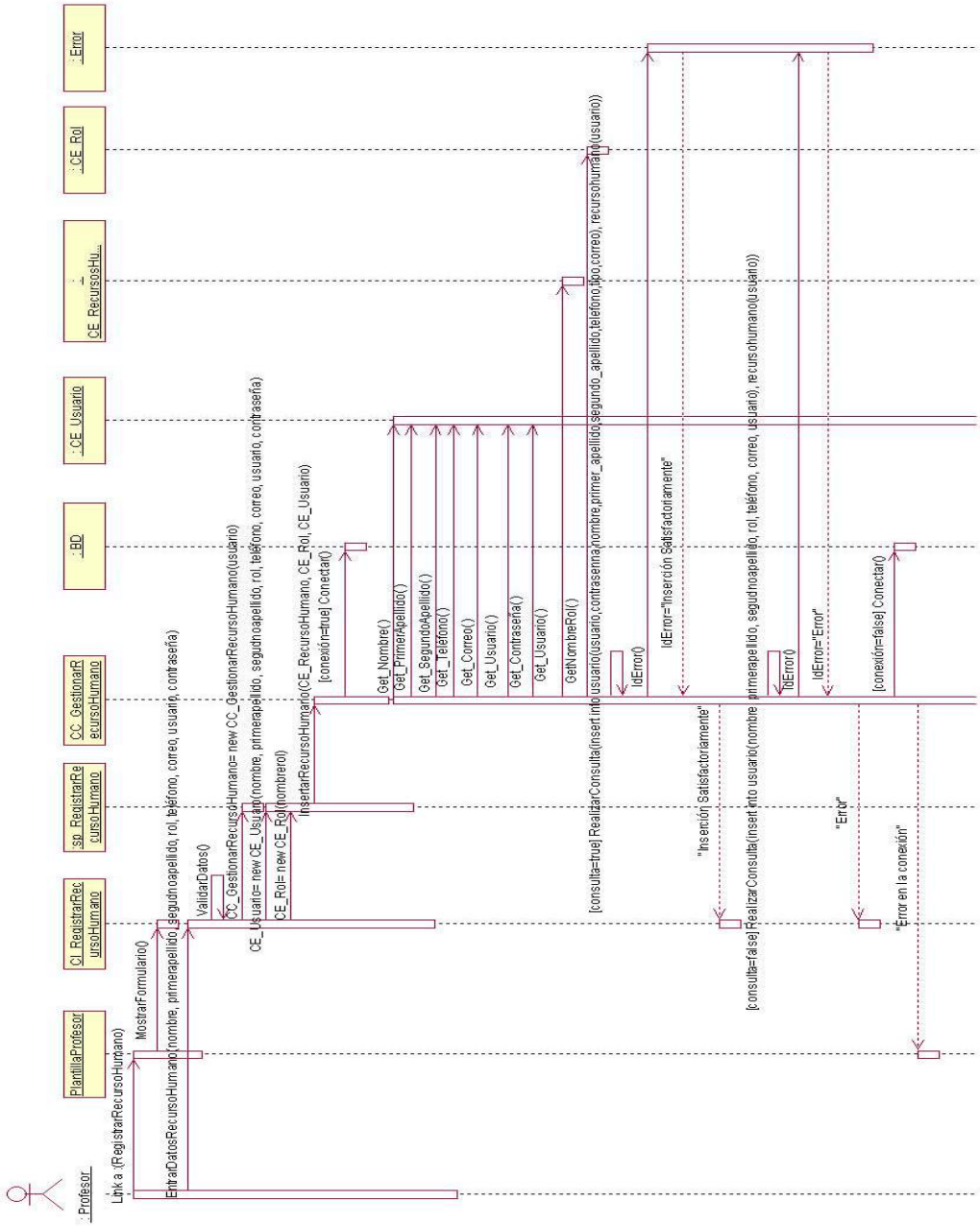


Figura.3-4. Diagrama de secuencia. Escenario Modificar recurso humano.

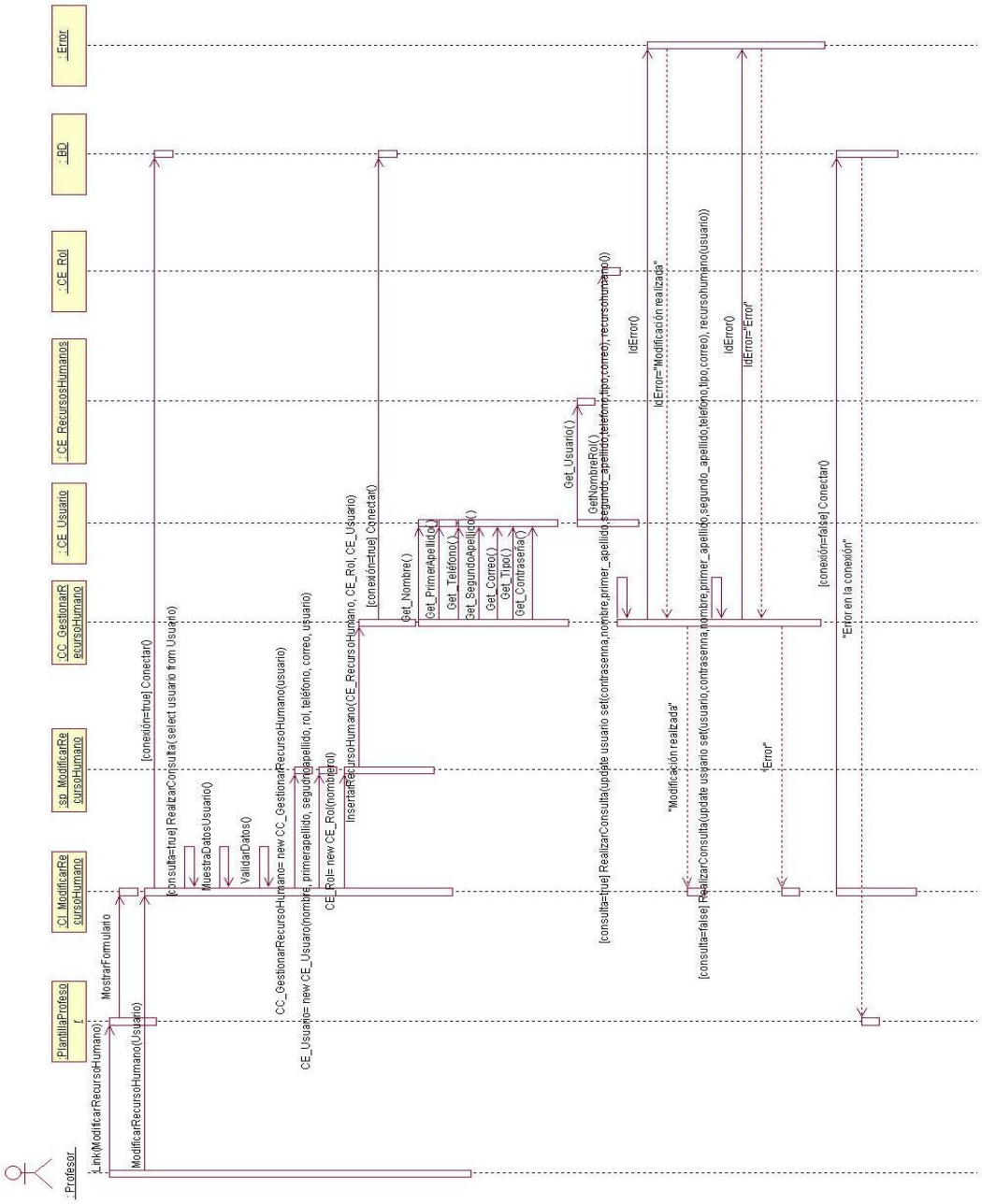


Figura.3-5. Diagrama de secuencia. Escenario Eliminar recurso humano.

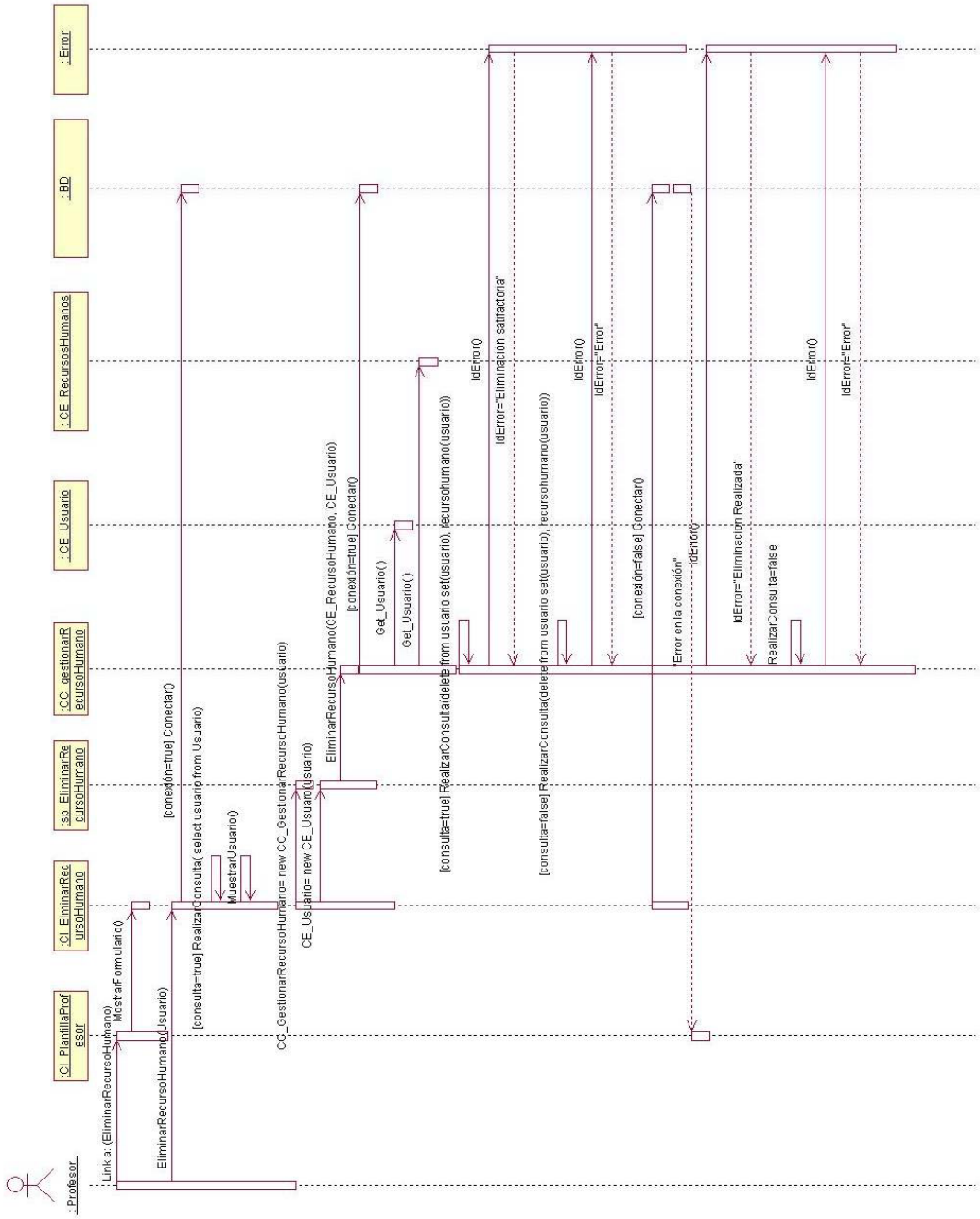


Figura.3-6. Diagrama de secuencia. Escenario Registrar usuario.

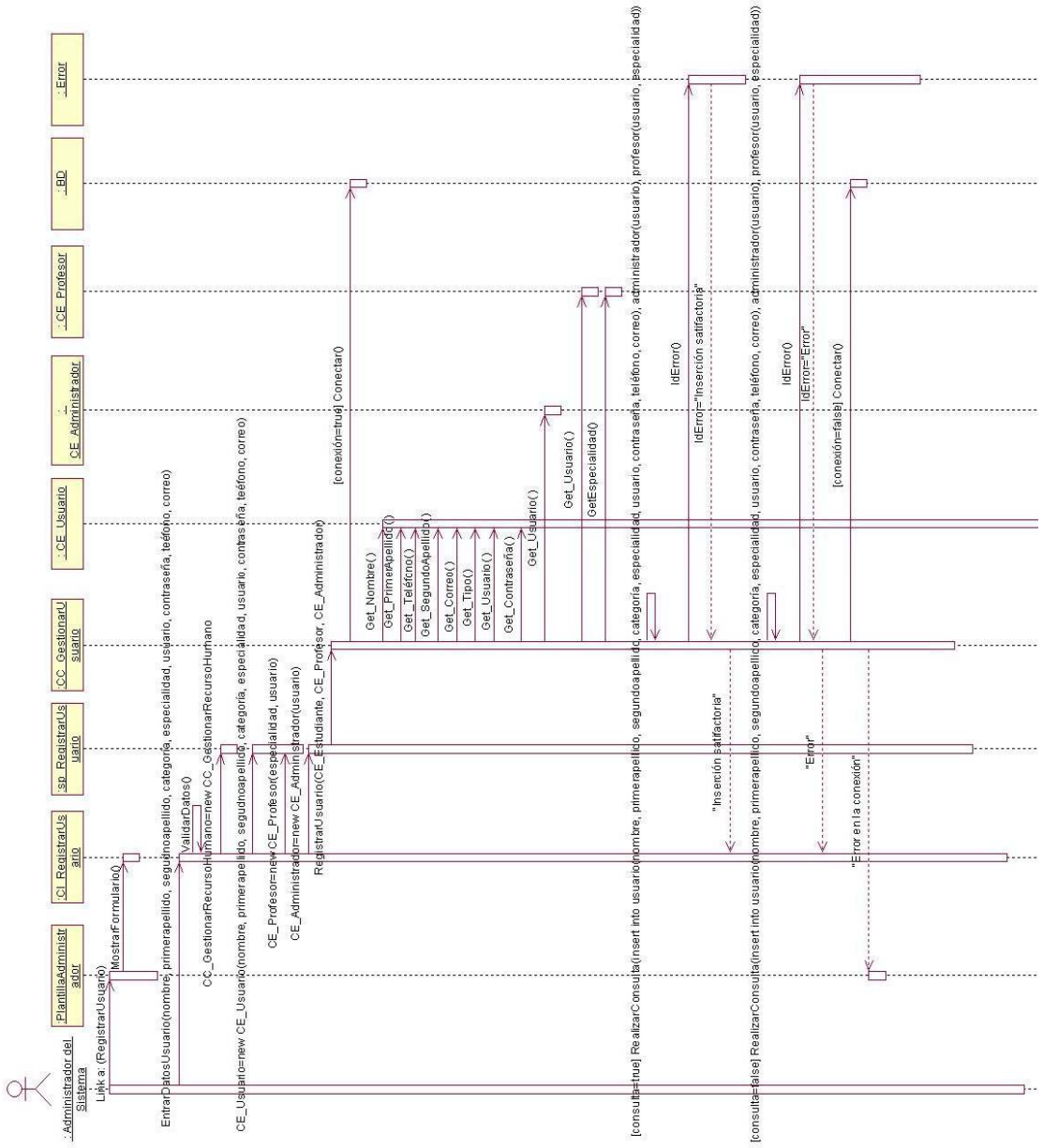


Figura.3-7. Diagrama de secuencia. Escenario Modificar usuario.

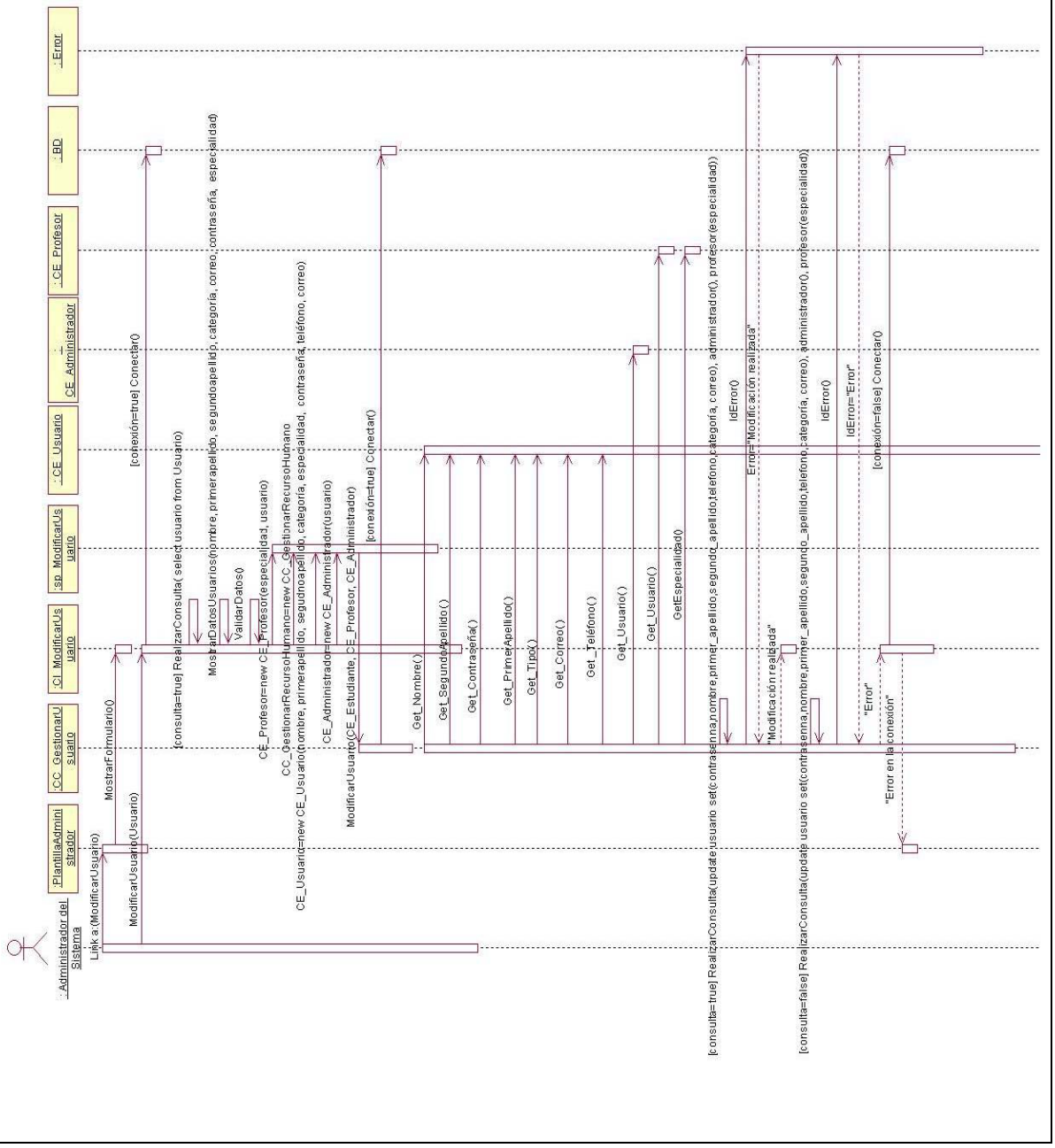


Figura.3-8. Diagrama de secuencia. Escenario Eliminar usuario.

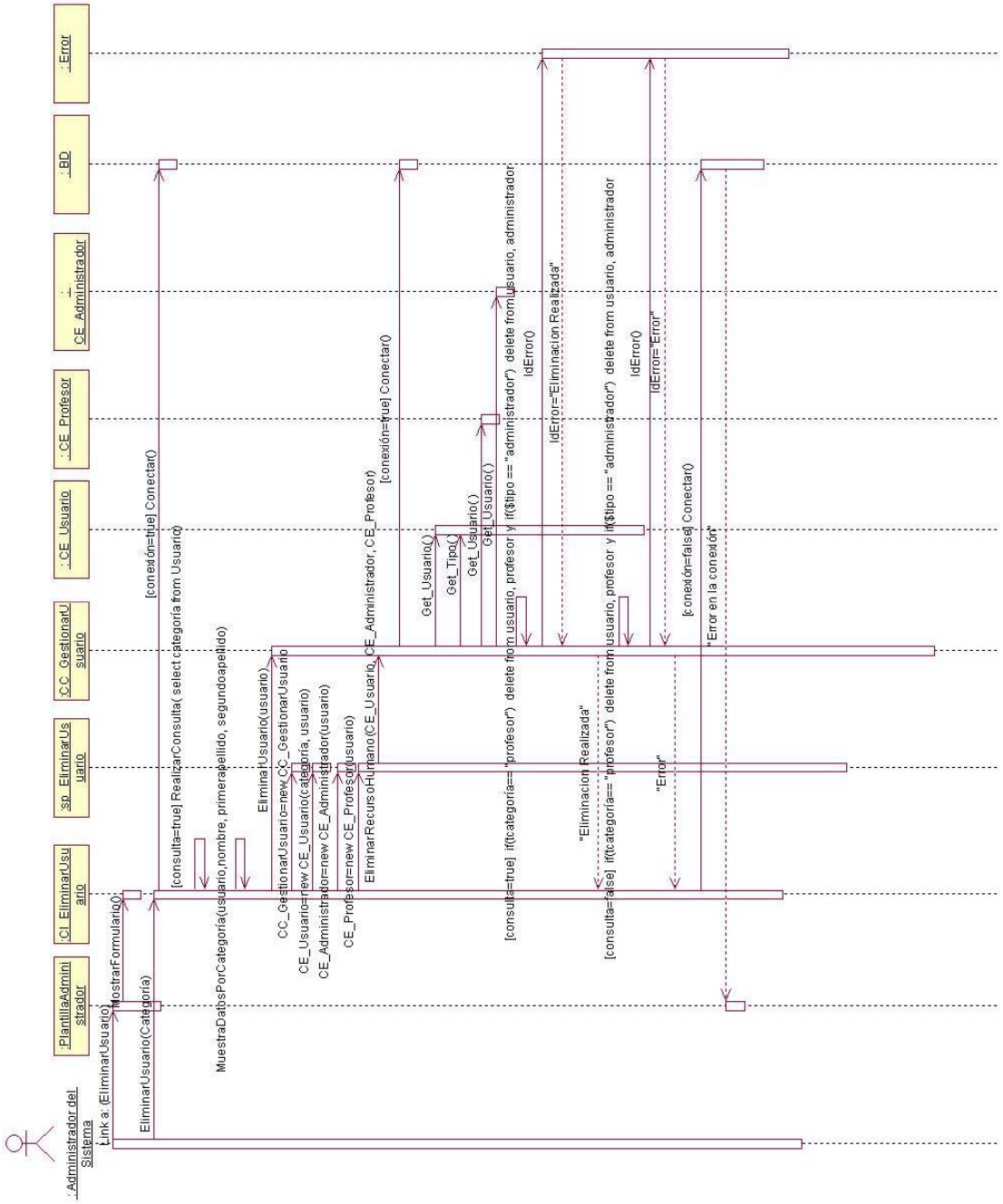


Figura.3-9. Diagrama de secuencia. Escenario Registrar proyecto.

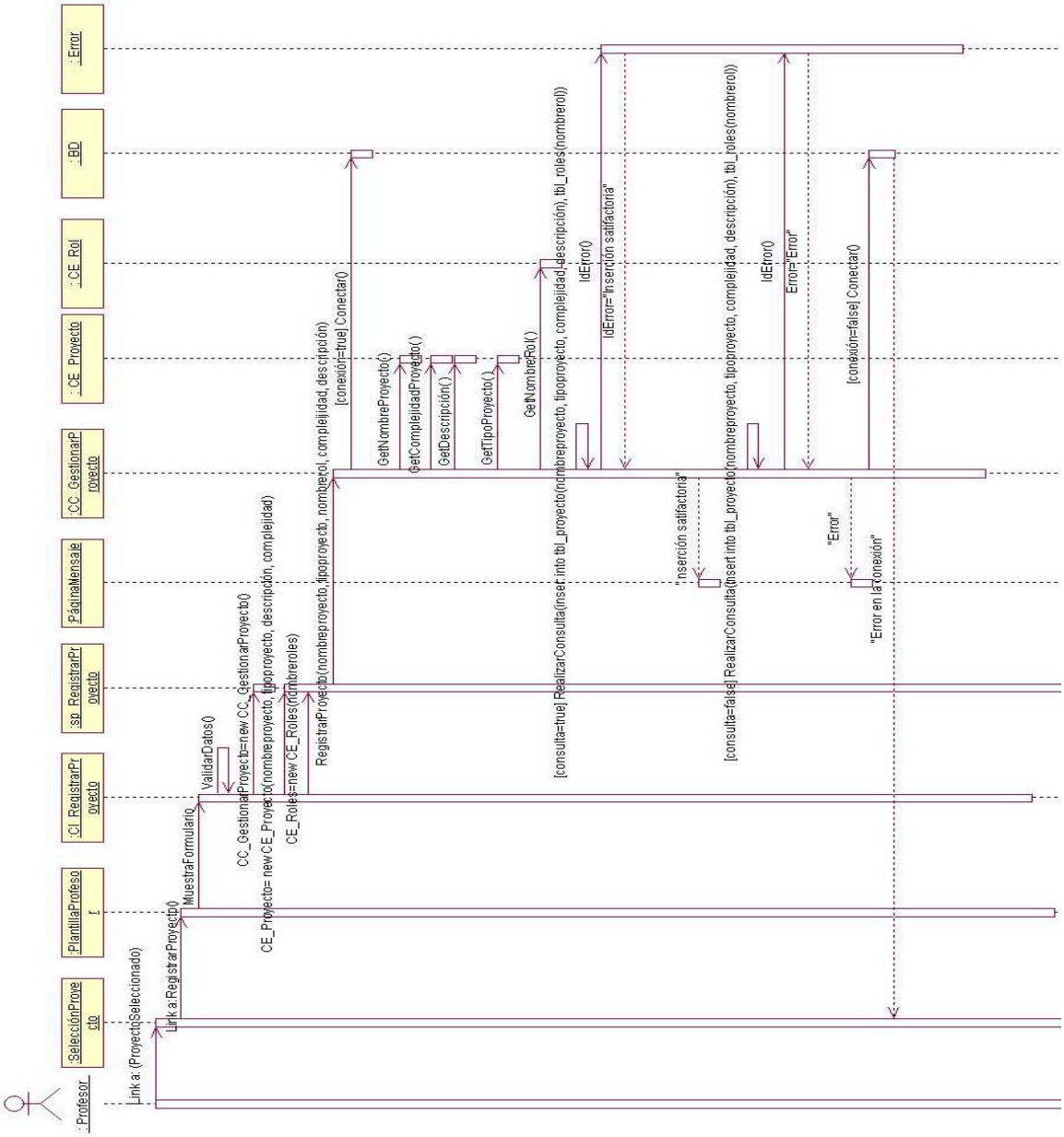


Figura.3-10. Diagrama de secuencia. Escenario Modificar proyecto.

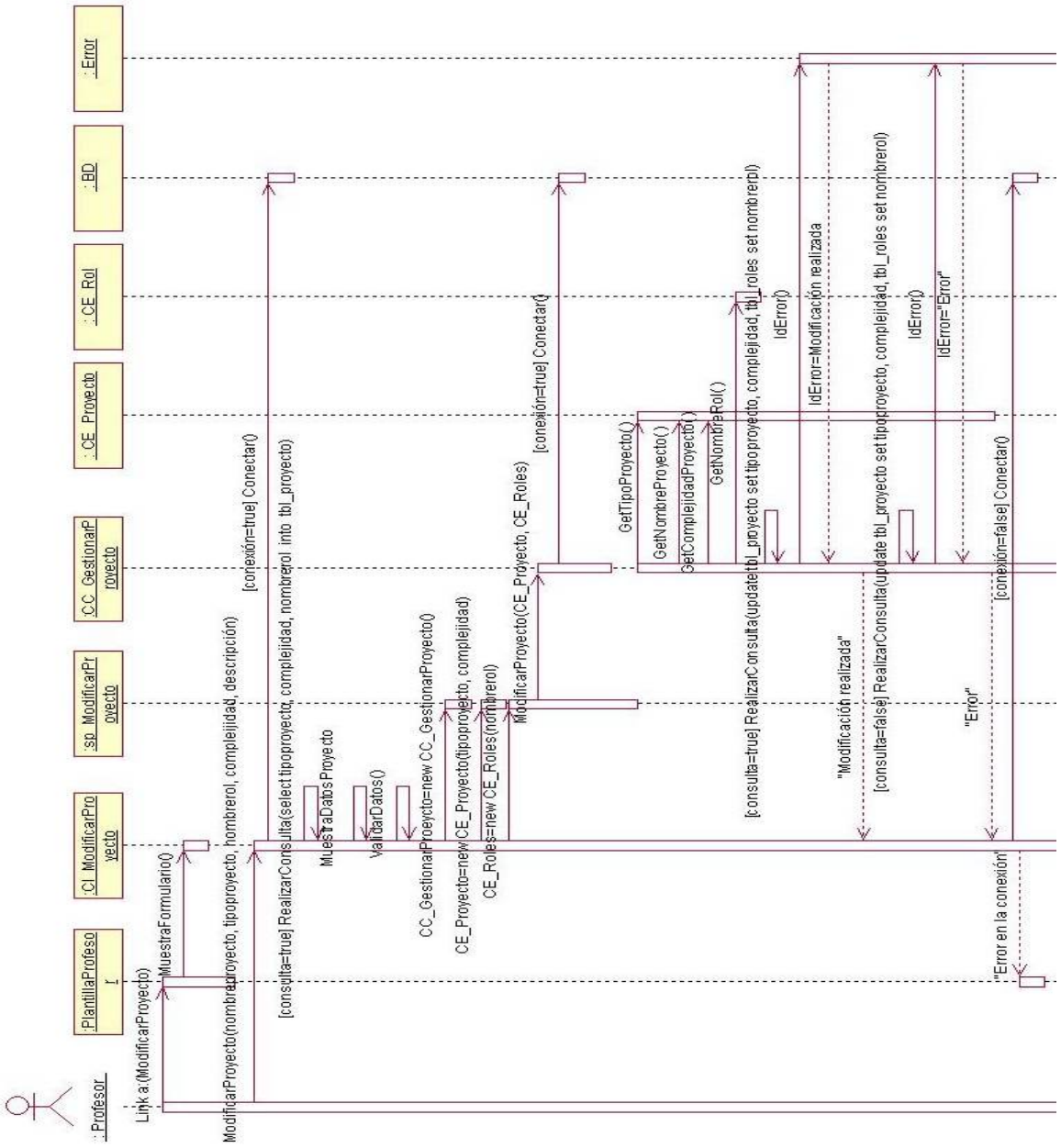


Figura.3-11. Diagrama de secuencia. Escenario Eliminar proyecto.

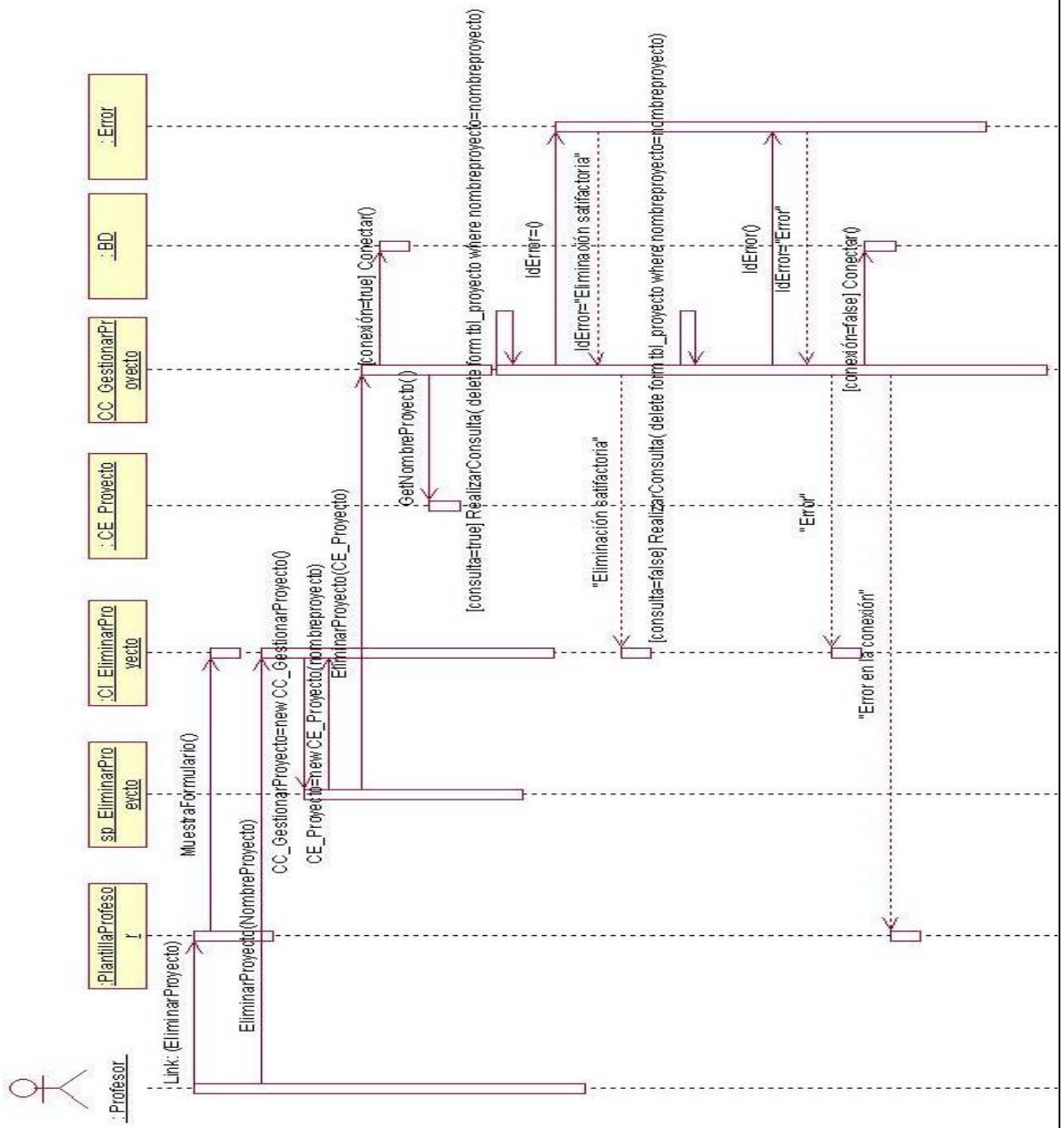


Figura.3-12. Diagrama de secuencia. Escenario Definir actividades y artefactos.

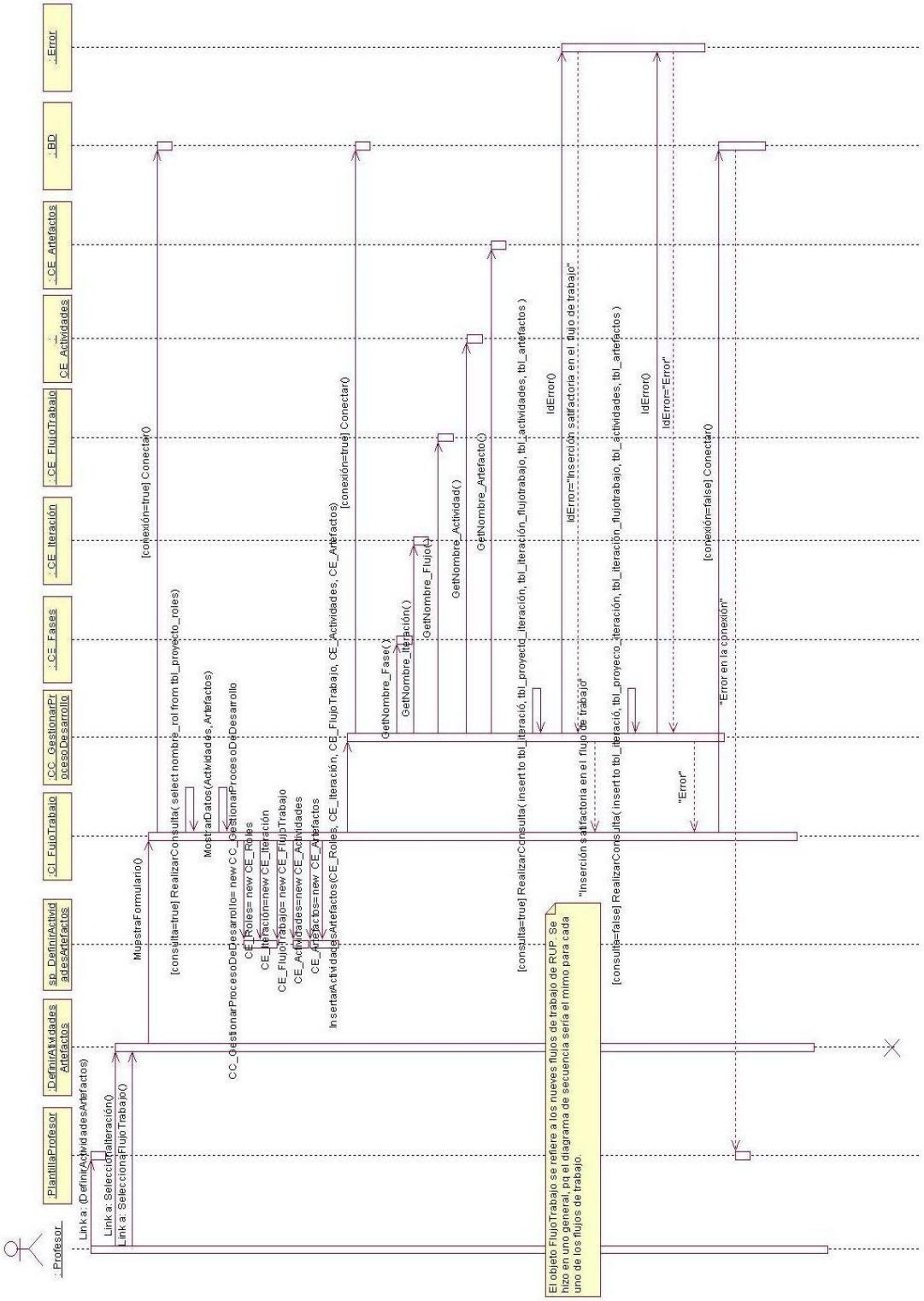
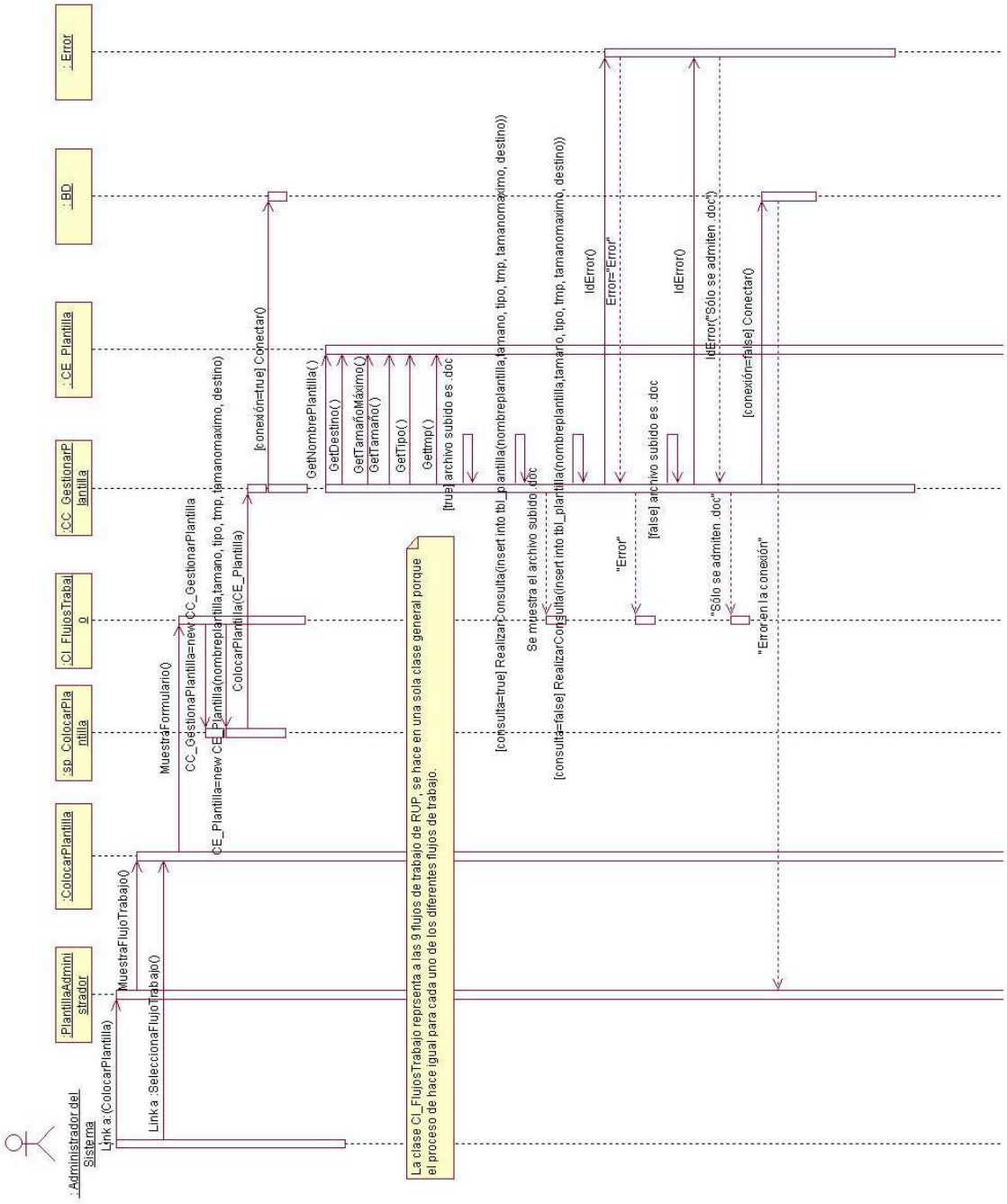


Figura.3-13. Diagrama de secuencia. Escenario Colocar plantilla.



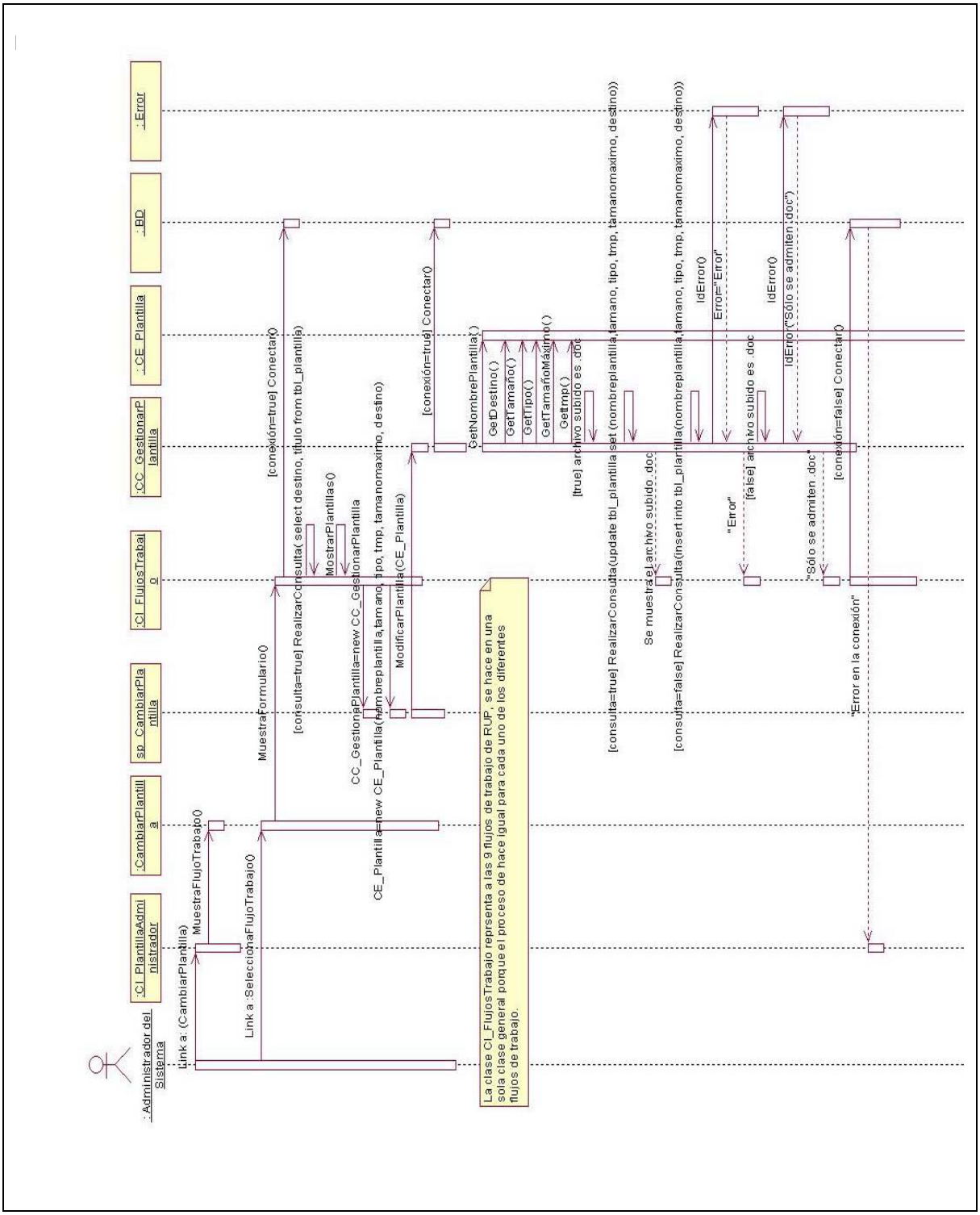

:Administrador del Sistema
:Cl_PlantillaAdministrador
:CambiarPlantilla
:sp_CambiarPlantilla
:Cl_FlujosTrabajo
:CC_GestionarPlantilla
:CE_Plantilla
:BD
:Error
Link a: (CambiarPlantilla)
MuestraFlujoTrabajo()
Link a :SeleccionaFlujoTrabajo()
MuestraFormulario()
[conexión=true] Conectar()
[consulta=true] RealizarConsulta(select destino, titulo from tbl_plantilla)
MostrarPlantillas()
CC_GestionarPlantilla=new CC_GestionarPlantilla
CE_Plantilla=new CE_Plantilla(nombreplantilla,tamano, tipo, tmp, tamanomaximo, destino)
ModificarPlantilla(CE_Plantilla)
[conexión=true] Conectar()
GetNombrePlantilla( )
GetDestino()
GetTamaño()
GetTipo()
GetTamañoMaximo()
GetImg()
[true] archivo subido es .doc
[consulta=true] RealizarConsulta(update tbl_plantilla set (nombreplantilla,tamano, tipo, tmp, tamanomaximo, destino))
Se muestra el archivo subido .doc
[consulta=false] RealizarConsulta(insert into tbl_plantilla(nombreplantilla,tamano, tipo, tmp, tamanomaximo, destino))
IdError()
Error="Error"
"Error"
[false] archivo subido es .doc
IdError()
IdError("Sólo se admiten .doc")
"Sólo se admiten .doc"
[conexión=false] Conectar()
"Error en la conexión"
La clase Cl_FlujosTrabajo representa a las 9 flujos de trabajo de RUP, se hace en una sola clase general porque el proceso de hace igual para cada uno de los diferentes flujos de trabajo.

Figura.3-15. Diagrama de secuencia. Escenario Cambiar contraseña.

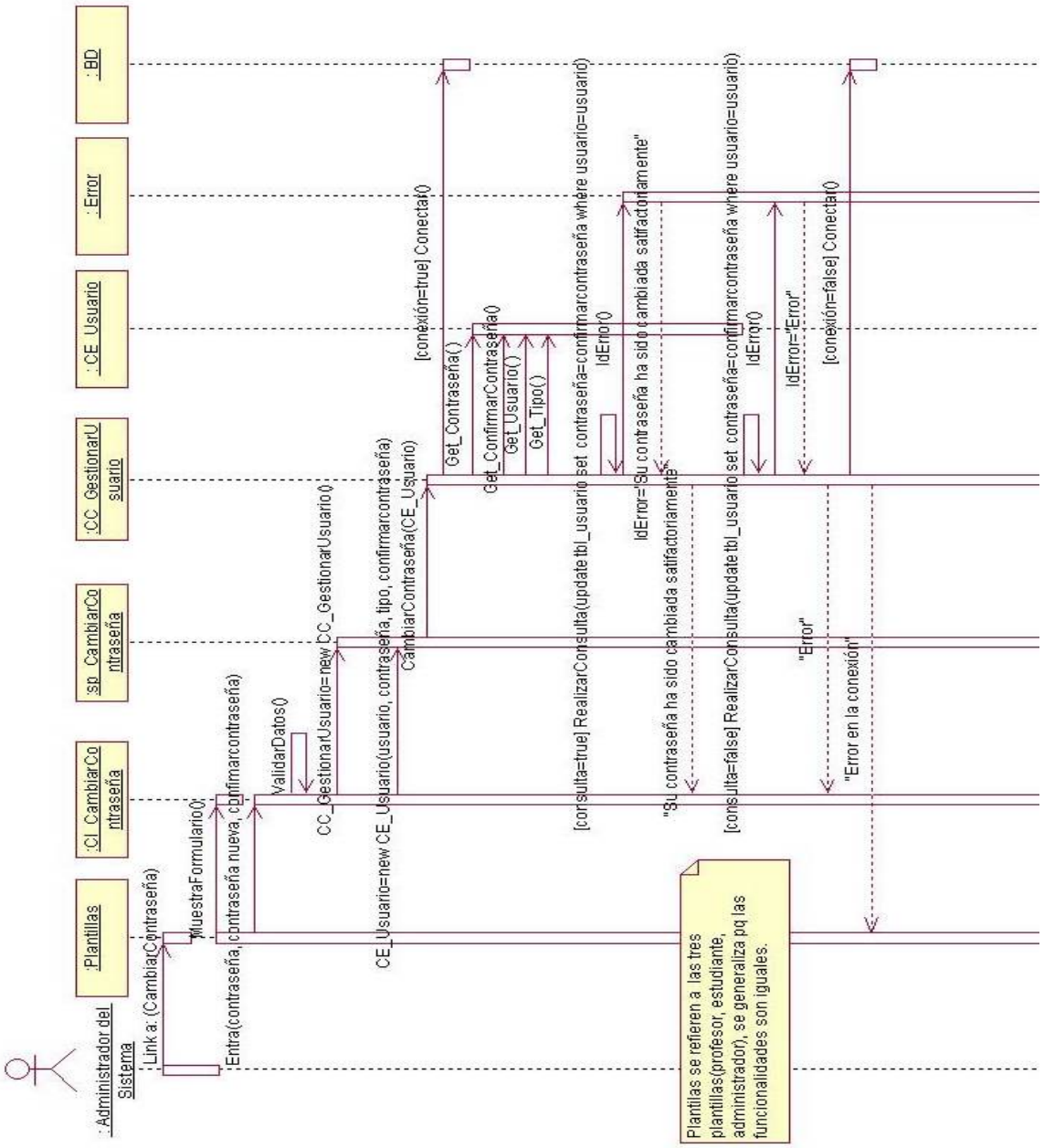


Figura.3-16. Diagrama de secuencia. Escenario Descargar y subir artefacto.

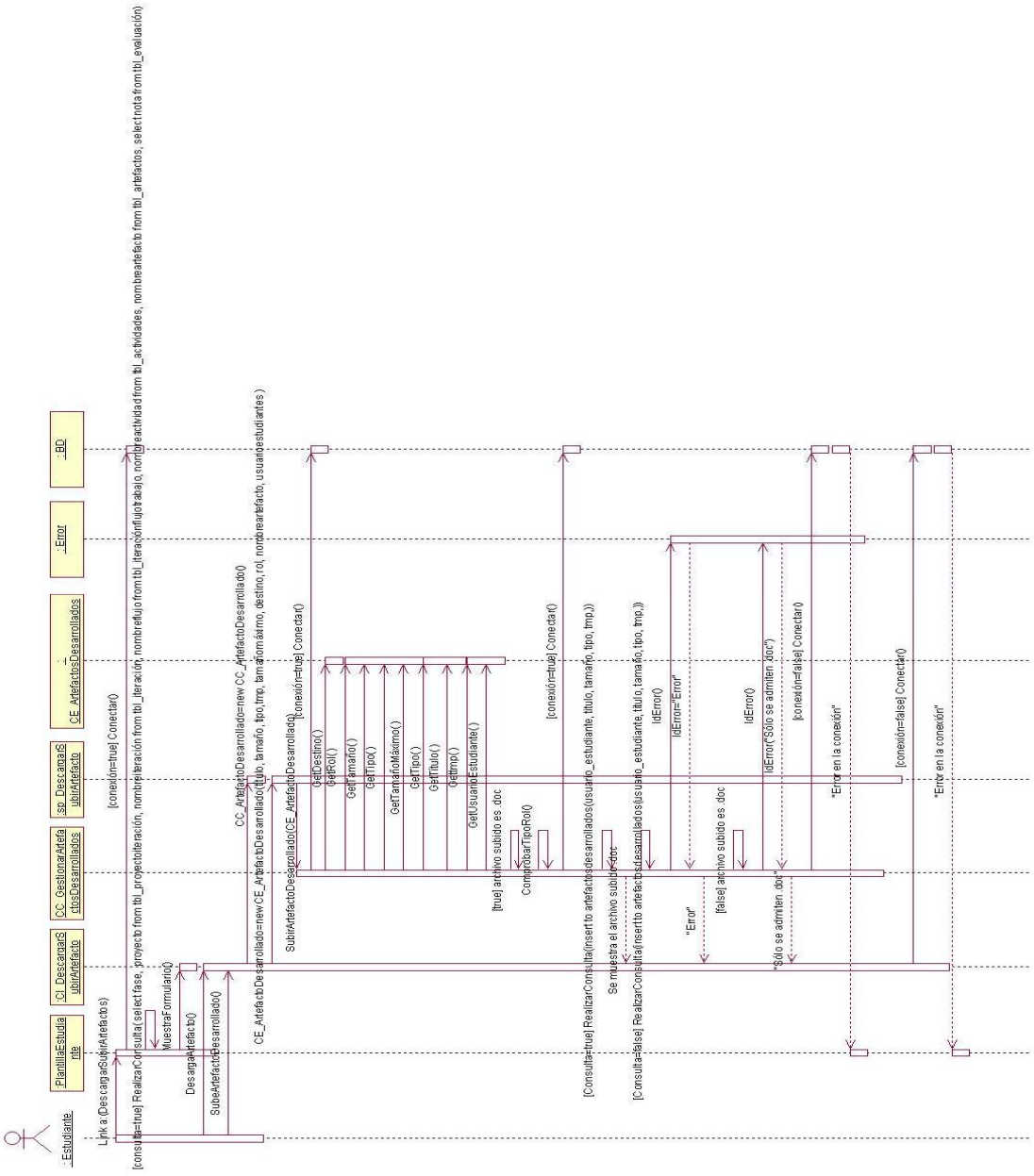


Figura.3-17. Diagrama de secuencia. Escenario Revisar plantilla.

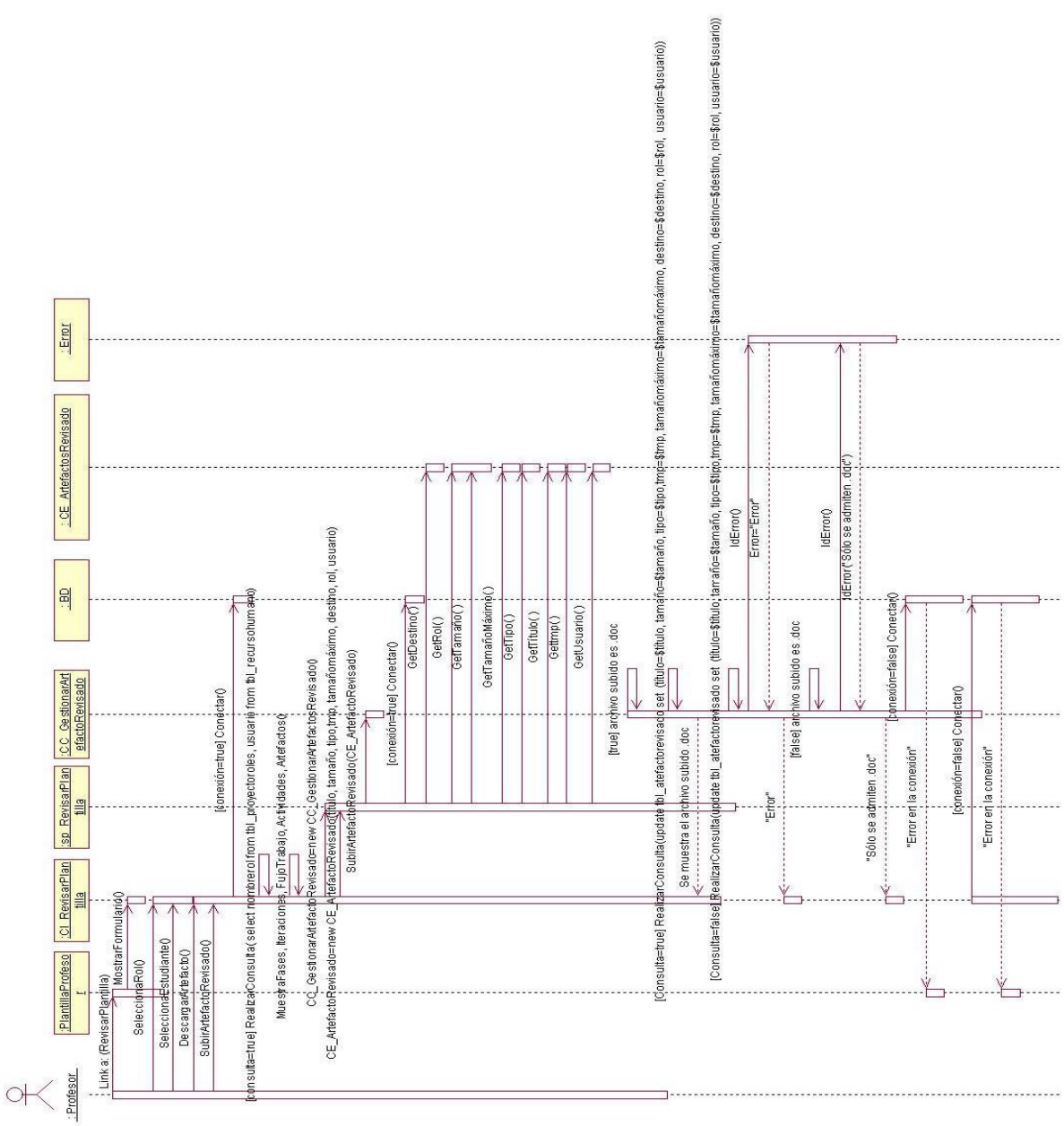


Figura.3-18. Diagrama de secuencia. Escenario Emitir evaluación.

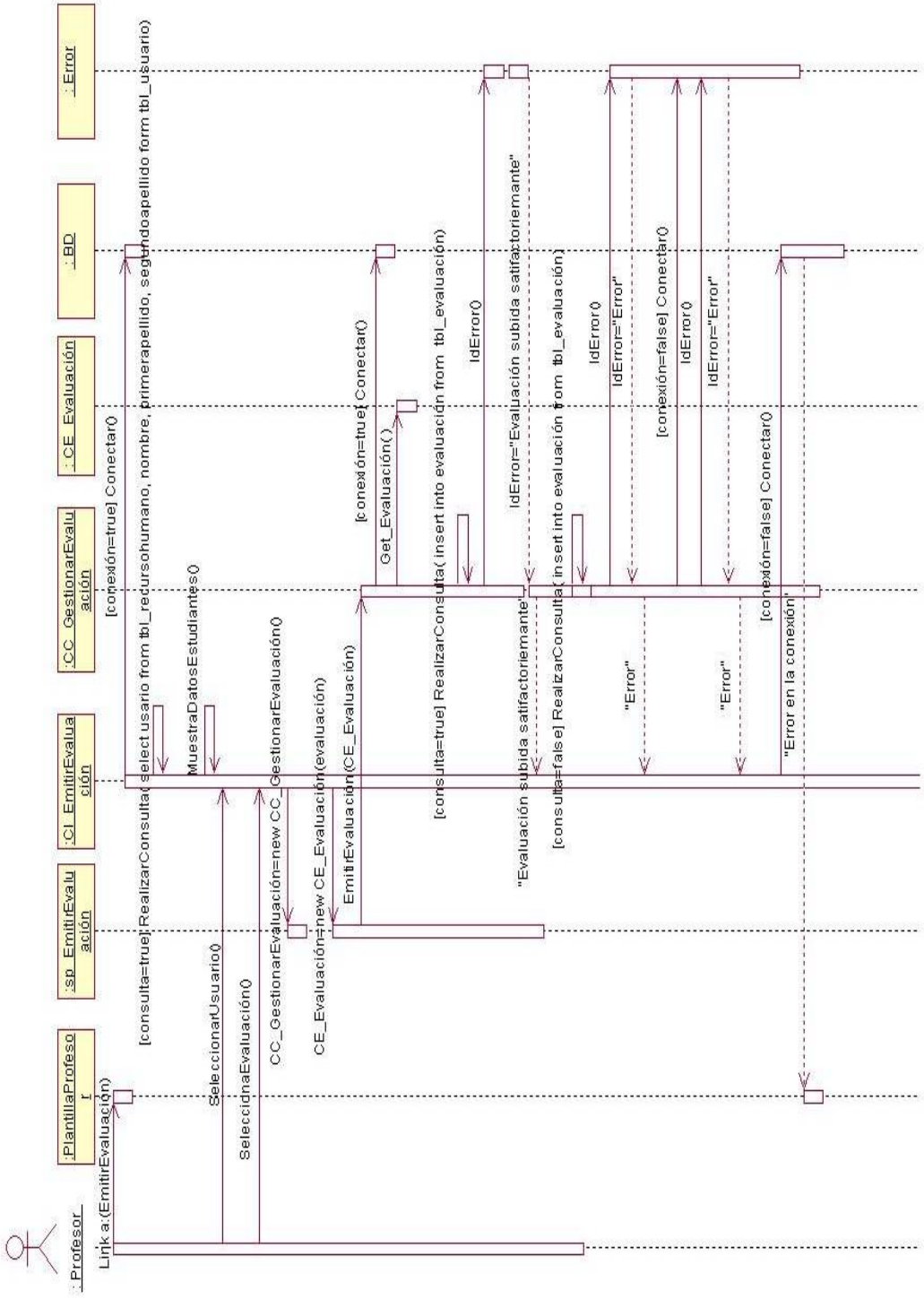


Figura.3-19. Diagrama de secuencia. Escenario Modificar evaluación.

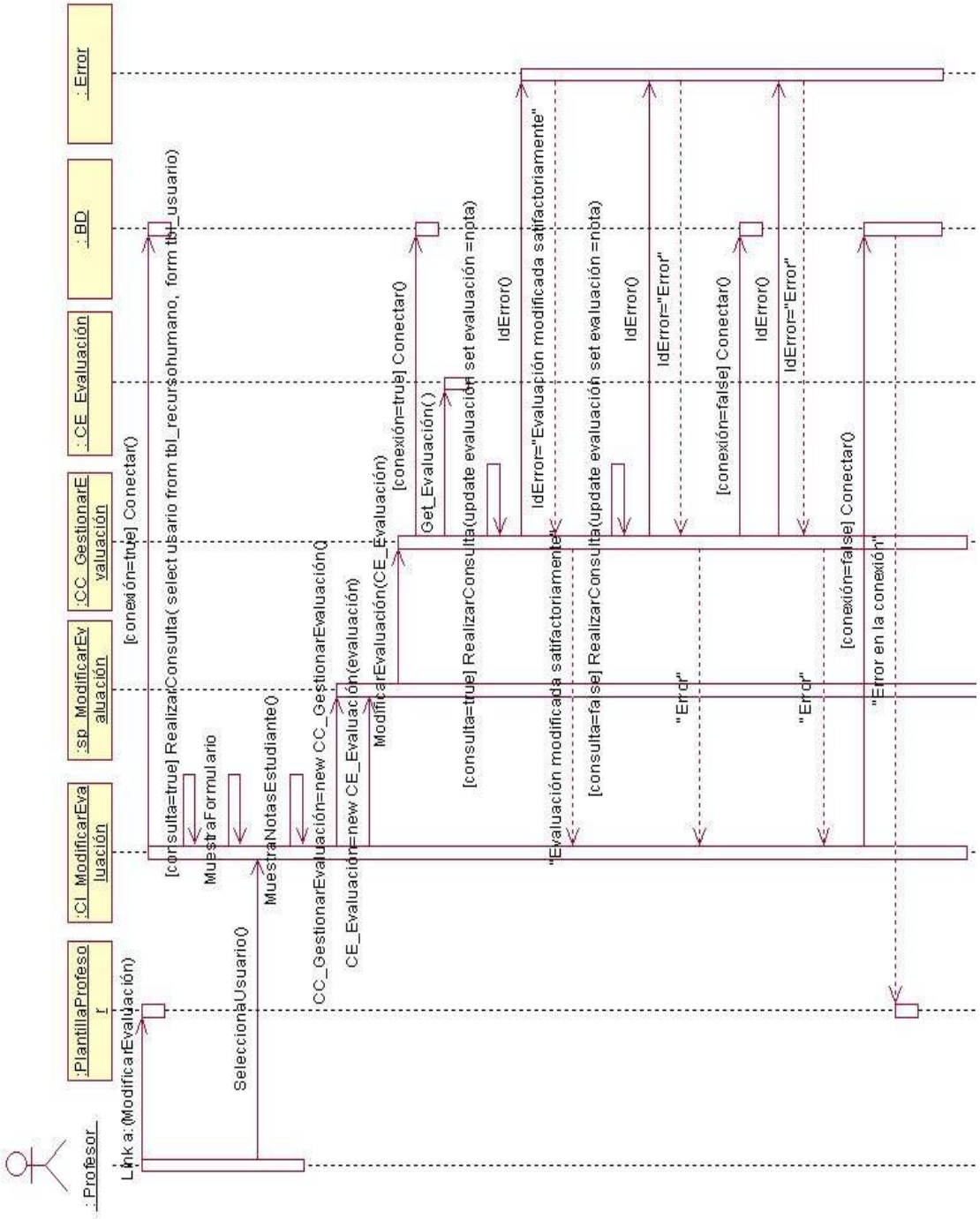
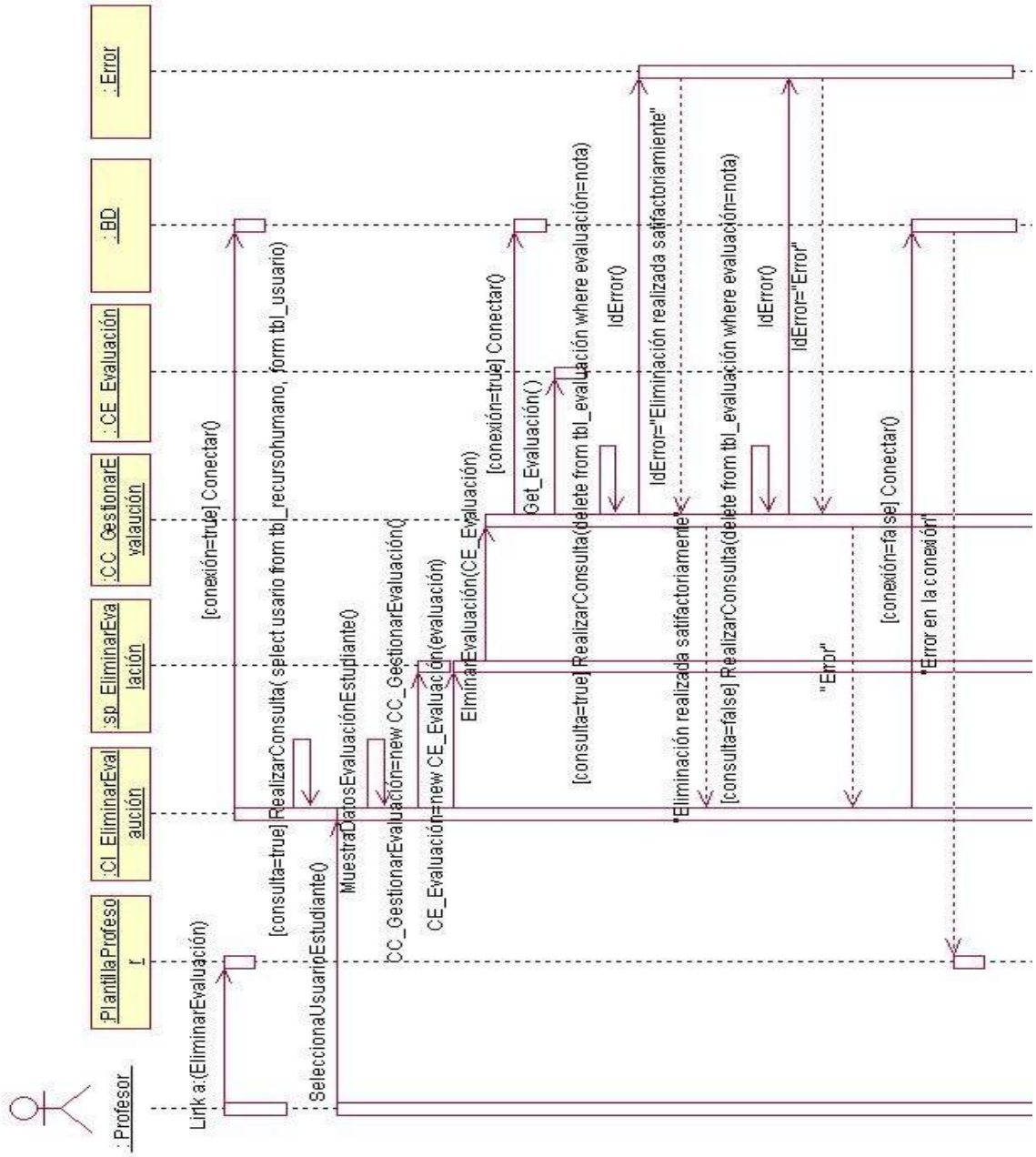


Figura.3-20. Diagrama de secuencia. Escenario Eliminar evaluación.



GLOSARIO DE TERMINOS

A continuación, en orden alfabético, se muestra el significado de algunos términos usados en este documento cuyo uso no es común y que pueden dificultar la compresión del mismo:

Actividad: Unidad tangible de trabajo realizada por un trabajador en un flujo de trabajo, de forma que implica una responsabilidad bien definida para el trabajador, produce un resultado bien definido (un conjunto de artefacto) basado en una entrada bien definida (otro conjunto de artefactos), y representa una unidad de trabajo con límites bien definidos a la que, probablemente se refiera el plan de proyecto al asignar tareas a los individuos. También puede verse como la ejecución de una operación por un trabajador.

Artefacto: Pieza de información tangible que es creada, modificada y usada por los trabajadores al realizar actividades; representa un área de responsabilidad, y es candidata a ser tenida en cuenta para el control de la configuración. Un artefacto puede ser un modelo puede ser un modelo, o un documento.

CU: Caso de uso.

Extranet: Es una red privada virtual resultante de la interconexión de dos o más intranets que utiliza Internet como medio de transporte de la información entre sus nodos.

Fase: Período de tiempo entre dos hitos principales de un proceso de desarrollo.

Fase de construcción: Tercera fase del ciclo de vida del software, en la que el software es desarrollado a partir de una línea base de la arquitectura ejecutable, hasta el punto en el que está listo para ser transmitido a la comunidad de los usuarios.

Fase de elaboración: Segunda fase del ciclo de vida, en la que se define la arquitectura.

Fase de inicio: Primera fase del ciclo de vida del software, en la que la idea inicial para el desarrollo es refinada hasta el punto de quedar lo suficientemente bien establecida como para garantizar la entrada en la fase de elaboración.

FE: Flujo de evento.

Flujo de trabajo: Realización de un caso de uso de negocio o parte de él. Puede describirse en términos de diagramas de actividad, que incluyen a los trabajadores participantes, las actividades que se realizan y los artefactos que se producen.

Flujo de trabajo ambiente: La finalidad de este flujo de trabajo es dar soporte al proyecto con las adecuadas herramientas, procesos y métodos. Brinda una especificación de las herramientas que se van a necesitar en cada momento, así como definir la instancia concreta del proceso que se va a seguir.

Flujo de trabajo análisis y diseño: El objetivo de este flujo es traducir es traducir los requisitos a una especificación que describe como implementar el sistema.

Flujo de trabajo captura de requisitos: Este es uno de los flujos de trabajo más importantes, porque en él se establece qué tiene que hacer exactamente el sistema que construyamos. En esta línea los requisitos son el contrato que se debe cumplir, de modo que los usuarios finales tienen que comprender y aceptar los requisitos que especificamos.

Flujo de trabajo configuración y administración del cambio: La finalidad de este flujo de trabajo es mantener la integridad de todos los artefactos que se crean en el proceso, así como de mantener información del proceso evolutivo que han seguido.

Flujo de trabajo gestión de proyectos: Es el arte de lograr un balance al gestionar objetivos, riesgos, restricciones para desarrollar un producto que sea acorde a los requisitos de los clientes y los usuarios.

Flujo de trabajo implementación: En este flujo de trabajo se implementan las clases y objetos en ficheros fuente, binarios, ejecutables y demás. Además se deben hacer las pruebas de unidad: cada implementador es responsable de probar las unidades que produzca. El resultado final de este flujo de trabajo es un sistema ejecutable.

Flujo de trabajo instalación y distribución: El objetivo de este flujo de trabajo es producir con éxito distribuciones del producto y distribuido a los usuarios.

Flujo de trabajo modelo del negocio: Con este flujo de trabajo pretendemos llegar a un entendimiento de la organización donde se va a implantar el producto.

Flujo de trabajo prueba: Este flujo de trabajo es el encargado de evaluar la calidad del producto que estamos desarrollando, pero no para aceptar o rechazar e producto final del desarrollo del proceso, sino que debe ir integrado en todo el ciclo de vida.

FTP: Es un protocolo de transferencia de ficheros entre sistemas conectados a una red TCP basado en la arquitectura cliente-servidor.

Hash: Se refiere a una función o método para generar claves o llaves que representen de manera casi unívoca a un documento, registro, archivo, etc.

Hito: Es un momento significativo en el proyecto que señala el haber conseguido un logro importante.

HTTP: Protocolo de transferencia de hipertexto

Iteración: Conjunto de actividades llevadas a cabo de acuerdo a un plan (de iteración) y unos criterios de evaluación, que lleva a cabo una versión, ya sea interna o externa.

MD5: Es un algoritmo de reducción criptográfico de 128 bits ampliamente usado.

Patrón: Solución común a un problema común de un determinado contexto.

RC4: Es un sistema de cifrado de flujo para proteger el tráfico de información.

Socket: Es un método para la comunicación entre un programa del cliente y un programa del servidor en una red, se define como el punto final en una conexión.

SMTP: Protocolo simple de transferencia de correo electrónico.

TCP/IP: Protocolo de Control de Transmisión (TCP) y Protocolo de Internet (IP),

TELNET: Es un protocolo (y del programa informático que implementa el cliente) que sirve para acceder mediante una red a otra máquina, para manejarla como si estuviéramos sentados delante de ella.