
Universidad de las Ciencias Informáticas

Facultad 15

**Trabajo de Diploma para optar por el título de**

**Ingeniero en Ciencias Informáticas**

# **Título:** Herramienta para la gestión de requisitos en los proyectos productivos de la UCI.

**AUTORES**

Yoel Osorio Turruelles

Yusbel Pérez Pérez

**TUTOR**

Ing. Leevan Abón Cepeda

**Ciudad de la Habana**

**Junio 2010**

**Declaración de Autoría**

Declaro que somos los únicos autores del trabajo titulado: Herramienta para la gestión de requisitos en los proyectos productivos de la UCI, y autorizo a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____ _____

Yoel Osorio Turruelles Yusbel Pérez Pérez

_____

Ing. Leevan Abón Cepeda

*“La grandeza de un hombre no se mide por el terreno que ocupan sus pies, sino por el horizonte que descubren sus ojos”.*

***José Martí***

## Agradecimientos

Agradecer en primer lugar a la Revolución y a la Universidad de las Ciencias Informáticas por darme la oportunidad de estudiar en una gran escuela.

A mis padres y hermano por su constante dedicación y apoyo durante los cinco años de la carrera.

A Yadián y Liset por brindarme su apoyo cuando más lo necesité.

A Yosvany por ser mi amigo y mi gran maestro.

A Kariné por su especial atención, su gran ayuda en el trabajo, a ti te deseo muchos éxitos en la vida.

A Luis y Fernando por ser mis mejores amigos.

A Mabel por ser mi mejor compañera de mesa.

A todos los compañeros de aula.

A Miguel, Alexis, Ricardo, Ivian, Adrián, Yusniel, Bernardo, Sergio, Yadira y Lissuan por aportar al presente trabajo de diploma.

A mi compañero de tesis Yusbel por aguantarme durante todo el trabajo.

A Rafael por siempre enseñarme dentro y fuera del aula.

A Hugo y Anelys por ser buenos profesores y amigos a la vez.

A nuestro tutor Leevan por decir siempre presente y por su apoyo durante todo el desarrollo del trabajo.

A todos los que de una forma u otra han compartido conmigo en los buenos y malos momentos de mi vida, a ustedes gracias.

Yoel

## Agradecimientos

Agradecer a nuestro Comandante en Jefe Fidel Castro Ruz, a La Revolución y a la Universidad de las Ciencias Informáticas por darme la posibilidad de estudiar en una escuela como esta.

Agradecer a mi mamá por ser guía, mi ejemplo y aportar tantos valores a mi formación, sin su apoyo mi sueño no hubiese sido cumplido.

Agradecer a mi novia Greter, por su apoyo y tanta compresión durante todo este tiempo.

Agradecer a mi tía y a su esposo por brindarme siempre su apoyo incondicional.

Agradecer a Yiyi y Juan Carlos por su constante preocupación sobre mis estudios y por brindarme siempre su apoyo.

Agradecer a los padres de mi novia por el apoyo que me han dado, por su cariño, respeto y admiración.

Agradecer a mis dos grandes amigos Yohairo y Pepe, por estar siempre a mi lado y apoyarme en las cosas buenas.

Agradecer a Irislaidis, por su cariño, apoyo y ser mi hermanita adoptada.

Agradecer a Yosvany, por su apoyo brindado y por su disposición de ayudarnos siempre.

Agradecer a Liset y Yadian, por su apoyo y constante preocupación.

Agradecer a mi compañero de tesis Yoel, por ser el mejor de los compañeros y por su apoyo en todo este tiempo, y sobre todo por soportarme a su lado.

Agradecer a Kariné por todo el tiempo que dedicó a nuestro trabajo, por su disposición de ayudarnos siempre: A ti, que la vida te llene de alegría y felicidad.

Agradecer a Rafael por su apoyo, ejemplo y enseñanza.

A Leevan nuestro tutor, por su disposición de ayudarnos y apoyarnos en todo este tiempo.

A todos los que de una forma u otra han compartido conmigo en los buenos y malos momentos de mi vida, a ustedes muchas gracias.

Yusbel

## Dedicatoria

Quiero dedicar la tesis a tres personas que han sido mi apoyo durante toda mi vida, primero a mi mamá que sin el cariño de ella no fuera quien soy, a mi padre por darme el ejemplo y ser mi guía en los estudios, y a mi gran hermano que es sin dudas mi amigo y compañero del alma.

Yoel

## Dedicatoria

A la memoria de mi mamá, que aunque ya no esté conmigo seguirá siendo la inspiración de mi vida.

A mi tía y abuela por darme el cariño de madre cuando más lo necesite.

Yusbel

## Resumen

En la actualidad la gestión de los requisitos se ha convertido en un proceso de mucha importancia para todas las empresas que desarrollan software, hoy en día un mal cumplimiento de estos, puede llevar al fracaso de un proyecto. El presente trabajo está dividido en 4 capítulos que recogen los artefactos fundamentales para el apoyo en la gestión de requisitos, utilizando la metodología de desarrollo RUP por sus siglas en inglés.

Para el desarrollo de esta herramienta se hizo un estudio de las tecnologías más adecuadas para su implementación, concluyendo que PHP sería el lenguaje de programación a utilizar, Postgresql como Sistema Gestor de Bases de Datos y servidor web Apache. Además se seleccionaron tres framework para una mejor eficiencia en el desarrollo de la aplicación, para la interfaz se usaron las librerías javascript de Ext, en la lógica del negocio Symfony y Propel para el mapeo a la Base de Datos. También se usaron otras herramientas de complemento como son:

- ✓ Netbeans 6.8 como IDE de desarrollo.
- ✓ Visual Paradigm 6.4 como herramienta CASE para el modelado de los artefactos.
- ✓ Photoshop 10 para el diseño de los banner.

**Palabras claves:**

Requisitos, software, artefactos, metodología, implementación, gestor, interfaz.

## Introducción

En los últimos tiempos, el desarrollo del software ha tenido un auge considerable. Se conoce como la célula fundamental de muchos sistemas productivos. Las compañías y empresas mundiales, dependen cada día más de un buen software que le facilite su producto con mejores resultados, menos tiempo y menor costo. Uno de los principales problemas por lo que la mayoría de los proyectos no cumplen con las fechas establecidas y con sus objetivos es porque no se realiza de manera correcta la ingeniería de requisitos, los errores más comunes y más costosos de reparar, así como los que más tiempo consumen se deben a una inadecuada Gestión de Requisitos. Una buena gestión de requisitos es el factor común relacionado con el éxito de los proyectos que se realizan habitualmente en cualquier sector, saber lo que se va a hacer, cómo se va a hacer y dejar constancia de ello, trae como resultado un proyecto con mejor calidad y entrega en el tiempo establecido. Para hacer la ingeniería de requisitos en La Universidad de las Ciencias Informáticas se utilizan herramientas privativas de un alto costo en cuanto a licencias, aunque también se apoya en algunos modelos manuales. Las herramientas libres existentes carecen de funcionalidades, que ayuden a realizar de forma adecuada la gestión de requisitos.

Debido a lo antes expuesto surge el siguiente **problema a resolver**: Las herramientas libres para la gestión de requisitos no permiten administrar correctamente los requisitos de los proyectos productivos de la UCI.

Se decide realizar un estudio de los requisitos más importantes que debe tener una herramienta de administración de requisitos, planteándonos como **objetivo**: desarrollar una herramienta para la gestión de requisitos en los proyectos productivos de la UCI, para esto será necesario realizar los **objetivos específicos** del presente trabajo:

- ✓ Identificar los requisitos funcionales y no funcionales para la herramienta.
- ✓ Obtener el modelo de análisis y el modelo de diseño.
- ✓ Implementar un prototipo web que responda al diseño.

Como **objeto de estudio**: Los procesos para la gestión de requisitos en el desarrollo del software, y el **campo de acción** estará enmarcado en: Los procesos para la gestión de requisitos en los proyectos productivos de la UCI.

# ÍNDICE

# CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

# CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

## 1.1 Introducción.

En este capítulo se presentan conceptos fundamentales sobre Ingeniería de Requisitos (IR), se realiza un estudio de las herramientas de administración de requisitos más usadas en el mundo de desarrollo de software. Además se realiza un estudio de los lenguajes de programación, los gestores de base de datos, herramientas de modelado, así como la metodología de desarrollo utilizadas.

## 1.2 ¿Qué es un requisito?

Circunstancia o condición necesaria para algo. (RAE, 2009)

Para un mayor entendimiento, se define por requisito, como la cualidad necesaria de un sistema, una declaración que identifique una capacidad, una característica, o un factor de calidad de un sistema que proporcione valor y utilidad al cliente o usuario. Los requisitos son las descripciones de los servicios y las restricciones que se generan durante el proceso de ingeniería de requisitos. (Larman, 1999)

## 1.3 Herramientas de administración de requisitos.

Para la selección de las herramientas se tuvo en cuenta que proporcionaran casi todas las necesidades básicas que exige tener una herramienta de este tipo. Estas herramientas están ampliamente difundidas y son reconocidas internacionalmente. En general, todas se basan en el uso de base de datos para almacenar la información correspondiente a los requisitos. Todas asumen que la estructura de los requisitos es jerárquica, de forma que un requisito puede estar formado o tener asociados otros requisitos de nivel inferior.

### Las herramientas que seleccionamos para su análisis son:

- ✓ OSRMT v1_5
- ✓ GatherSpace
- ✓ Requisite Pro
- ✓ Caliber RM

- ✓ DOORS
- ✓ IRqA 3.0

### 1.3.1 Características generales de las herramientas.

**OSRMT v1_5:** Es una herramienta de software libre, el nombre completo es Open Source Requeriment Management Tool. Fue diseñada para servir el ciclo de vida completo del desarrollo del software. Es independiente de la plataforma que se vaya a utilizar, pues se ejecuta sobre Java. Permite la descripción de los requisitos, además de los documentos relacionados con la Ingeniería de Requisitos, también comprende las distintas matrices de trazabilidad, funcionalidades, casos de usos y casos de pruebas.

**GatherSpace:** Es una herramienta de gran alcance, con toda la gerencia y el uso en línea simple de los requisitos que permite centralizar, modelar y compartir requisitos del software. Es capaz de proporcionar una intuitiva solución para pequeñas y grandes organizaciones. Es capaz de gestionar los requisitos del software utilizando las especificaciones funcionales, además de los casos de usos así como el modelado. Es un software no instalable, está en línea 100% de las funcionalidades, solo se necesita un navegador.

**RequisitePro:** Es una herramienta centrada en documentos, se almacenan los requisitos asociándolos a documentos (aunque también permite guardarlos directamente en la base de datos). Auxilia especialmente el control de cambio de requisitos, con trazabilidad para especificaciones de software y pruebas. Está muy unido a MS Word[1]. La herramienta permite el uso de Oracle[1] sobre Unix[2] o Windows[3] como “back-end database” y también soporta SQL Server sobre Windows.

**CaliberRM:** Es para sistemas grandes y complejos y proporciona una base de datos de requisitos con trazabilidad. La compañía ve a los requisitos como parte del proceso de gestión de la calidad del software al igual que las pruebas y el trazado de defectos. CaliberRM está basado en Internet y maneja referencias de documentos, responsabilidad de usuario, trazabilidad, prioridad y estado entre algunas otras características.

[1] MS Work: Es una suite ofimática de Microsoft, es un procesador de texto.

**DOORS:** A diferencia del resto de las herramientas, considera los requisitos como objetos y los documentos como módulos. Tiene una orientación basada en objetos, frente a RequisitePro y Caliber-RM, que manejan solamente requisitos y sus atributos. Es una herramienta para organizaciones grandes que necesitan controlar complejos conjuntos de usuarios y requisitos de sistemas con una completa trazabilidad. Proporciona buena visualización de tales documentos como jerárquicos y su lenguaje de extensión, permite una gran variedad de soporte de herramientas a ser construidas.

**IRqA 3.0:** Herramienta CASE de Ingeniería de Requisitos, diseñada para soportar las actividades realizadas en el proceso de especificación de sistemas. Proporciona y formaliza la comunicación entre cliente, proveedor y entre los distintos miembros del equipo de desarrollo. Facilita la captura, organización y análisis de los requisitos y la especificación de la solución mediante un apoyo metodológico adaptable a cada cliente.

### 1.3.2 Comparación entre las herramientas.

#### 1.3.2.1 Captura e identificación de requisitos.

**OSRMT v1_5:** Se gestionan los requisitos completamente, características específicas, diseño, puesta en práctica, casos de prueba, y cualquier otro artefacto. Los requisitos se pueden importar directamente desde un XML[4]. Los artefactos pueden tener una mejor clasificación si los usuarios especifican una lista de categorías.

**GatherSpace:** Es una herramienta de gran alcance en la gestión y para manejar y compartir los requisitos del software. En la parte inicial se puede comenzar agregando las características del proyecto y luego continuar con otros artefactos.

**RequisitePro:** Los tipos de requisitos pueden ser configurados fácilmente, así como atributos y tipo de documentos. Definir consultas y filtros para encontrar rápidamente la información de interés. Presenta un potente motor de requerimientos Adaptable a los procesos personalizados.

**CaliberRM:** Permite la discusión en línea, captura y graba la información para que no se pierda en juntas y correos electrónicos. Liga el contexto al requisito específico para promover el compartir del conocimiento.

**DOORS:** Se integra con Report Miner (RM) suite, para capturar, tracear, manejar y analizar una amplia información, para llevar a cabo una buena gestión de requisitos y mantener los estándares especificados. Se combina con base datos RM siendo una poderosa combinación, y un intuitivo documento de estilo.

**IRqA 3.0:** Se realiza la Captura de requisitos, recopilación de la información facilitada por los usuarios, de forma sencilla y organizada. IRqA permite al usuario definir distintas vistas para cada uno de los elementos de la especificación, de forma que la información a visualizar sea la que más le interesa en función de las operaciones que desee realizar en cada momento.

#### 1.3.2.2 Captura de la estructura de los elementos del sistema.

**OSRMT v1_5:** Se pueden incorporar y mantener organizados los artefactos del diseño y la puesta en práctica de los mismos. Puede tener textos y unir documentos o cualquier accesorio de un archivo.

**GatherSpace:** Se pueden agregar e incorporar paquetes, se tienen en cuenta la característica de los requisitos.

**RequisitePro:** Las opciones gráficas son variadas, por ejemplo la matriz demuestra las relaciones entre los requisitos, existe un gran acoplamiento entre otras herramientas de modelado que permite un acoplamiento entre los elementos gráficos del modelo del análisis o del diseño y sus requisitos relacionados.

**CaliberRM:** El diseño es intuitivo y fácil de empleo. Proporciona un gestor central y asegura el la base de datos para los requisitos del proyecto. Se capturan los casos de usos para que los analistas, probadores, los encargados de la comercialización estén consciente del ciclo de vida.

**DOORS:** Puede capturar gráficamente la implementación del sistema como (la arquitectura, descomposición funcional) y mostrarlas de tal manera que los requisitos puedan asociarse a esta captura grafica.

**IRqA 3.0:** Se capturan los elementos estructurales genéricos siguientes: Diagramas de bloque:

Representan bloques y las relaciones entre ellos, y se utilizan para navegar a través de la especificación y para demostrar la trazabilidad entre los elementos contenidos en bloques relacionados.

#### 1.3.2.3 Análisis de trazabilidad.

**OSRMT v1_5:** Mantiene las matrices de trazabilidad, acoplamientos individuales de los artefactos y estos se pueden importar. Mediante los filtros o los reportes de los usuarios se puede identificar la terminación de los requisitos.

**GatherSpace:** Genera un informe de trazabilidad que demuestra el acoplamiento bidireccional entre las características y casos de usos, además de las características y los requisitos.

**RequisitePro:** Proporciona un fácil mecanismo de preguntas que permiten un eficiente acoplamiento en la trazabilidad. El árbol de trazabilidad permite darle un seguimiento completo a los requisitos y sus relaciones.

**CaliberRM:** Permite acceder a la base de datos y marcar la trazabilidad de los requisitos durante el ciclo de vida. Permite que los requisitos sean asociados a los artefactos. Analiza el impacto a través de los casos de usos y mediante la trazabilidad se muestra el alcance de los casos de usos.

**DOORS:** Una vez que las asociaciones estén completas los usuarios desearán verlas y estas pueden ser mostradas. Los requisitos pueden establecerse por los usuarios y establecer una trazabilidad para ver su origen (de donde vienen) y hacia donde van.

**IRqA 3.0:** La matriz de trazabilidad permite que los usuarios examinen las relaciones entre los requisitos y otros elementos relacionados con el sistema (conceptos, panoramas de la prueba, servicios y diagramas).

#### 1.3.2.4 Gestión de la configuración.

**OSRMT v1_5:** Se pueden establecer líneas bases, se guarda el historial de gestión de cambios y las versiones realizadas. Mantiene la seguridad mediante contraseñas para evitar que se cambien los artefactos por personas no autorizadas.

**GatherSpace:** Existen flujos básicos establecidos por el usuario y estos terminan al especificar cuál es la respuesta del sistema.

**RequisitePro:** Se lleva un historial completo de cada requisito. Se almacena fecha, autor, tiempo y el contenido existente y los cambios realizados por el autor. Todo el historial se pone al día cuando un requisito cambia. Se incluyen líneas base de desarrollo.

**CaliberRM:** Se proporcionan capacidades de los requisitos que ayudan a los líderes de proyectos a planear el alcance, horario y recursos del proyecto. Captura versiones de los requisitos y se puede establecer líneas bases para el desarrollo.

**DOORS:** Una vez que los requisitos hayan sido capturados se necesita conocer cuando estos han sido modificados y ¿por qué es que se cambian estas funciones? En algún momento se necesitará establecer las líneas bases para los requisitos. Las líneas bases se muestran pero no se pueden modificar después de ser establecidas.

**IRqA 3.0:** Mantiene un historial de los cambios realizados a los requisitos de quien, cuando, que, donde y como fueron realizados. El historial se puede mantener también con las versiones guardadas (registrando los cambios realizados). Esto permite que el usuario pueda comparar las diferentes versiones de los requisitos y ver las diferencias entre ellas.

#### 1.3.2.5 Ambiente del sistema.

**OSRMT v1_5:** Permite varios usuarios conectados, la herramienta es multiplataforma, tiene soporte para múltiples gestores de base datos comerciales como MS Access, Oracle, o SQL Server y otras libres como MySQL Server, PostgresSQL y los requisitos del hardware son mínimos.

**GatherSpace:** Se permite que un mismo proyecto tenga varios usuarios, el ambiente de trabajo es vía web y es muy intuitivo.

**RequisitePro:** Soporta múltiples usuarios y el uso para varias bases de datos comerciales como MS Access, Oracle, o SQL Server. Los requisitos del hardware son medios.

**CaliberRM:** Soporta múltiples usuarios. Provee un repositorio central, seguro para administrar requisitos de proyectos a través del ciclo de vida de las aplicaciones, mejorando la comunicación entre todos los miembros del equipo. Los requisitos del hardware son medios.

**DOORS:** Muestra una ayuda múltiple (textos y visuales) para los usuarios. Los requisitos de hardware dependen de la cantidad de usuarios y el tamaño de la base de datos establecida.

**IRqA 3.0:** Soporta la conexión de múltiples usuarios siempre y cuando el gestor de base datos lo permita. Se puede utilizar en múltiples sistemas operativos.

#### 1.3.2.6 Interfaz del usuario.

**OSRMT v1_5:** Se pueden abrir múltiples artefactos, las actualizaciones se reflejan en las vistas, todos los artefactos se pueden exportar y la interfaz de trabajo es un simple navegador web o una aplicación de escritorio.

**GatherSpace:** Por lo menos para comenzar se debe agregar un usuario para usar el GatherSpace; se puede administrar todo el proyecto mediante este usuario que es un administrador. Para el uso de la herramienta se puede utilizar un navegador web.

**RequisitePro:** La interfaz es muy intuitiva en lo que se debe hacer, se puede realizar reportes, trabajar en los documentos de requisitos, estos se pueden editar vía web y todo es posible realizarlo al mismo tiempo.

**CaliberRM:** Está arquitectónicamente diseñado para comunicarse eficientemente vía TCP/IP[5], donde se da la posibilidad a los usuarios de una comunicación instantánea. Tiene un entorno multiusuario basado en web manejando un versionado automático de requisitos.

**DOORS:** Los usuarios tienen la posibilidad de generar informes y trabajar con los requisitos al mismo tiempo. Cualquier cambio se refleja si afecta a otros componentes. Tiene una vista web disponible que se puede utilizar con una conexión a la base de datos.

**IRqA 3.0:** La herramienta fue diseñada para que tenga una vista de ventanas independientes, esto permite que se puedan abrir al mismo tiempo diversas vistas del mismo proyecto y que estas se puedan manejar en paralelo.

## 1.4 Lenguajes del lado del servidor para desarrollo Web.

El desarrollo de aplicaciones web ha tenido un auge en todo el mundo gracias a las ventajas que las mismas ofrecen a empresas e instituciones. Un lenguaje del lado del servidor es un lenguaje de programación que es reconocido, ejecutado e interpretado en el servidor. Para definir un lenguaje del lado del servidor a utilizar se deben tener en cuenta la complejidad del mismo, si el lenguaje es capaz de resolver el problema, la explotación que hagan de la base de datos, los recursos que requiera, la velocidad de solución de problemas, entre otras características. Para la selección de los lenguajes se tuvo en cuenta la utilización en las empresas, además la política de software libre que plantea la UCI[6], y otros aspectos de rendimiento, robustez y seguridad.

### 1.4.1 Java.

Java es un lenguaje de programación sencillo, de propósito general, independiente de la plataforma de desarrollo, orientado a objetos, con una sintaxis muy parecida a la de C o C++. Fue creado por la compañía Sun Microsystems a principios de los 90. Su principal ventaja es la independencia de la plataforma. Como principales desventajas se encuentran, que el rendimiento de una aplicación echa en este lenguaje depende de la eficiencia del compilador, o de la Máquina Virtual de Java (JVM). Java necesita que este programa esté instalado en la computadora, además, sus entornos de desarrollo consumen una gran cantidad de sus recursos.

**Características:**

- ✓ Indiferente a la arquitectura.
- ✓ Robusto.
- ✓ Interpretado y compilado a la vez.
- ✓ Seguro.

- ✓ Distribuido.

### 1.4.2 Hypertext Preprocessor (PHP)

PHP fue ideado en el año 1994 por Rasmus Lerdorf. Sus primeras versiones, que no eran distribuidas a los clientes. La primera versión que estuvo disponible para el público en 1995, se llamaba "Herramientas para páginas web personales"(Personal Home Page Tools).

PHP es un lenguaje de programación interpretado, que es ejecutado del lado del servidor, es utilizado para generar páginas web dinámicas, embebido en páginas HTML[7], y ejecutado en el servidor, aunque puede ser utilizado para realizar aplicaciones con una interfaz gráfica, utilizando las debidas extensiones del lenguaje. La mayoría de su sintaxis ha sido obtenida del lenguaje de programación C, Java y Perl, aunque tiene algunas características específicas de sí mismo. Es un lenguaje que tiene como objetivo principal permitir a los desarrolladores la generación dinámica de páginas web. Al ser ejecutado en el servidor web, nos brinda la posibilidad de acceder a los recursos que se encuentren en el servidor, como por ejemplo, una base de datos.
PHP es un lenguaje independiente del navegador, o sea, el navegador no tiene que obligatoriamente soportar el lenguaje; para que las páginas PHP funcionen, el servidor donde se encuentran las páginas debe soportarlo. Debido a su condición de ser Software Libre posee una gran cantidad de características, que lo hacen un potente lenguaje para la creación de páginas web dinámicas y que lo convierten en uno de los lenguajes más usado a nivel mundial.

**Dentro de las múltiples características de PHP se pueden definir como fundamentales las que siguen:**

- ✓ Soporta una gran cantidad de bases de datos: MySQL, PostgreSQL, Oracle, entre otras.
- ✓ Integrado con disímiles bibliotecas externas, brinda la posibilidad de generar documentos en formato PDF, hasta el análisis de código XML.

- ✓ Provee al usuario una solución simple y universal para las paginaciones dinámicas fácilmente programable.
- ✓ Su código es más fácil de mantener y poner al día que en otros lenguajes.
- ✓ Es soportado por una gran comunidad de desarrolladores, como producto de código abierto que es, teniendo la ayuda de un grupo grande de programadores, permitiendo esto encontrar y reparar los fallos de funcionamiento.
- ✓ El código se actualiza constantemente con extensiones y mejoras continuas del lenguaje.
- ✓ A través del uso de PHP, se pueden realizar diferentes funcionalidades, como por ejemplo el procesamiento de información en formularios, foros de discusión, manipulación de cookies[8] y páginas dinámicas.
- ✓ Ofrece gran seguridad, debido a que puede acceder a ficheros, ejecutar comandos y abrir conexiones de red en el servidor.
- ✓ Existen diferentes opciones de configuración para controlar su funcionamiento.
- ✓ Muy fácil de aprender.
- ✓ Soporte multiplataforma de Sistemas Operativos, así como servidores y de base de datos.
- ✓ Se distribuye de forma gratuita bajo una licencia gratuita.
- ✓ Posee la capacidad de expandir su potencial, a través de la gran cantidad de módulos, llamados ext's o extends.
- ✓ Posee una gran documentación en su sitio oficial, y en otros sitios de la comunidad de desarrollo.
- ✓ Soporta programación orientada a objetos.
- ✓ Posee una biblioteca nativa de funciones sumamente amplia e incluida.

Una vez analizados los lenguajes de programación del lado del servidor, es necesario hacer una selección de cuál utilizar en el desarrollo del sistema. Para la implementación del sistema se usará PHP ya que posee una gran comunidad de desarrollo, la cual implementa constantemente mejoras en su código, en La Universidad de las Ciencias Informáticas este lenguaje es muy utilizado por un gran número de programadores, además de que el mismo es software libre, permite técnicas de programación orientada a objetos, puede ser ejecutado en la mayoría de los sistemas operativos, permite la combinación con un gran número de servidores de bases de datos entre los cuales se encuentran los posibles a utilizar, que le es de gran beneficio para los programadores, y fundamentalmente, porque el servidor apache, en el cual es interpretado el código de PHP es uno de los más rápidos del mundo y porque PHP es un lenguaje pensado para dar solución rápida a los problemas.

Por otro lado, Java es un lenguaje más pesado (más lento en su ejecución), pensado para realizar proyectos complejos y que no requieran de una solución inmediata, además, el código de Java puede llegar a ser redundante, debido a las frecuentes declaraciones de tipos y conversiones de tipo manual, sus entornos de desarrollo consumen menos recursos que los de Java, y es necesario tener la máquina virtual instalada para que corra una determinada aplicación.

## 1.5 Framework para desarrollo web con PHP

Un Marco de trabajo o Framework en inglés, es un modelo con una estructura definida, capaz de agilizar y organizar el desarrollo de un proyecto, mantiene una coherencia con el diseño y plantea el uso de una arquitectura, modela las relaciones generales de las entidades, además de contar con una estructura y una metodología de trabajo. Ayuda a la hora de realizar el mantenimiento del sitio gracias a la organización durante el desarrollo de la aplicación. Son desarrollados con el objetivo de brindarles a los programadores y diseñadores una mejor organización y estructura a sus proyectos.

En el presente trabajo se presentan como los más relevantes KumbiaPHP, Zend Framework, Symfony y CakePHP. (**Ver Anexo I:** Comparación entre los marcos de trabajo de PHP). A continuación se

caracterizan cada uno de ellos, y concluyendo con la selección de uno de ellos para el desarrollo del trabajo.

### 1.5.1 KumbiaPHP

**KumbiaPHP:** es un Framework para aplicaciones web libre escrito en PHP5. Fomenta la velocidad y eficiencia en la creación y mantenimiento de aplicaciones web, reemplazando tareas de codificación repetitivas por poder, control y placer. Brinda la posibilidad de conectarse a gestores de bases de datos como: PostgreSQL y Oracle en fase BETA[9]. Está patentizado con la licencia GNU/GPL[10]. Combina buenas prácticas de programación y patrones de diseño, tiene un respaldo de una gran comunidad en español, lo que podemos contar con una documentación actualizada. Implementa algunas prácticas de otros framework, estas son: Generación de reportes en PDF, WORD, EXCEL, formularios web y la utilización de helpers[11].

**Sus principales características son:**

- ✓ Administración de Caché[12]
- ✓ Listas de control de acceso (ACL) para la seguridad
- ✓ Soporte para AJAX
- ✓ Sistema de Plantillas sencillo
- ✓ Modelo de Objetos
- ✓ Arquitectura de tres capas MVC[13]
- ✓ Generación de Formularios
- ✓ Componentes Gráficos

### 1.5.2 Zend Framework

**Zend Framework (ZF)** es un marco de trabajo de código abierto para desarrollar aplicaciones web y servicios web con PHP 5, está bajo la filosofía de programación orientado a objetos. La estructura de los componentes de ZF es algo único; cada componente está construido con una baja dependencia de otros componentes. Esta arquitectura débilmente acoplada permite a los desarrolladores utilizar los componentes por separado.

**Algunas características de Zend Framework**

- ✓ Arquitectura de tres capas (MVC)
- ✓ Gran robustez y rendimiento
- ✓ Abstracción de base de datos fácil de usar
- ✓ Interfaz orientada a objetos
- ✓ Seguridad
- ✓ Administración de caché
- ✓ Herramientas y desarrollo rápido de aplicaciones (RAD)
- ✓ Programación orientada a objetos
- ✓ Soporte para internacionalización y localización de contenidos
- ✓ Soporte para envío de correo electrónico.
- ✓ Validación de Formularios
- ✓ Manejo propio de sesiones por usuarios.

### 1.5.3 CakePHP

CakePHP es un marco de trabajo para la creación de aplicaciones web, escrito sobre PHP, facilita al usuario la interacción con la base de datos, pueden implementarse aplicaciones orientadas a objetos, es compatible con PHP 5, tiene un conjunto de plantillas fáciles de utilizar, está destinado a proyectos medianos, su flexibilidad, seguridad y rapidez lo hacen un marco de trabajo líder en el mercado, tiene integrado un CRUD de la base de datos, utiliza los patrones de diseño MVC y ORM[14].

**Algunas características del marco de trabajo**

- ✓ Es ligero, no tiene sobre peso de código fuente, solo el necesario para la aplicación.
- ✓ Corre en PHP4 y PHP5.
- ✓ No necesita configuración, solo algunas pequeñas configuraciones en los archivos de configuración de la base de datos y algunas pocas constantes las cuales pueden ser modificadas, se puede comenzar la implementación de la aplicación literalmente en menos de cinco minutos luego de su instalación.

- ✓ Soporte para asociaciones entre tablas extendidas, permitiendo la creación de una arquitectura de la base de datos bastante compleja.
- ✓ Una estructura de directorio sumamente lógico y funcional.
- ✓ Mejorado soporte para AJAX[15], a través de la vista de helpers de AJAX y Javascript.
- ✓ Cuenta con una comunidad muy activa y bastantes sitios de ayuda.
- ✓ Presenta sistemas de validación a los largo de todo el marco de trabajo, verificación de ingreso de datos permitidos.
- ✓ Generación de plantillas de manera rápida y flexible.
- ✓ Componentes de seguridad, manejo de sesiones y de peticiones.
- ✓ Listas de control de acceso flexibles.
- ✓ Almacenamiento en caché de las vistas.
- ✓ Trabaja en cualquier subdirectorio de un servidor web.

### 1.5.4 Symfony

El marco de trabajo Symfony es un proyecto de software libre con más de una década de experiencia y se ha convertido en uno de los marcos de trabajo de PHP más populares gracias a sus características y su gran documentación.

Symfony es un marco de trabajo para PHP5 patentado bajo licencia MIT[16], está orientado a la creación de aplicaciones web de un gran tamaño, es compatible con la mayoría de gestores de bases de datos más utilizados en el mundo, PostgreSQL, Oracle y SQL Server de Microsoft, entre otros en dependencia del tipo de abstracción de la base de datos que se utilice.

**Algunas características de Symfony**

- ✓ Su instalación es sencilla en cualquier plataforma (Windows, Unix) y su curva de aprendizaje es alta gracias al trabajo de su comunidad y de los desarrolladores.

- ✓ Permite la integración con otros marcos de trabajo.
- ✓ Buena integración con Propel[17]
- ✓ Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- ✓ Utiliza la arquitectura en tres capas (MVC).
- ✓ Código fácil de leer que incluye comentarios y que permite un mantenimiento muy sencillo.
- ✓ Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros además de la utilización de ´´*helpers*´´ que permiten una integración con código para HTML, Javascript[18], AJAX y JSON[19].
- ✓ Excelente mecanismo de configuración mediante ficheros YML y XML.
- ✓ Administración de la caché.
- ✓ Manipula la gestión de credenciales.
- ✓ Independiente del gestor de base de datos, lo que hace un marco de trabajo con una utilidad superior a otros de su tipo.
- ✓ Mantiene la mayoría de mejores prácticas y patrones de diseño para la web.
- ✓ La capa de presentación utiliza plantillas que pueden ser creados por diseñadores HTML sin ningún tipo de conocimiento del marco de trabajo. Los ´´*helpers*´´ incluidos permiten minimizar el código utilizado en la presentación.
- ✓ Los formularios incluyen validación automatizada lo que posibilita insertar datos correctos en la base de datos.
- ✓ Tiene incluido el CRUD[20] de la base de datos.

Teniendo en cuenta que el marco de trabajo Symfony es una potente herramienta para el desarrollo rápido de aplicaciones web, soporta php5+, se puede integrar a Propel para hacer un buen uso del mapeo de la base de datos, además de la gran comunidad de desarrolladores hispanos que tiene y su buena aceptación en la Universidad de las Ciencias Informáticas, se decide para este trabajo utilizar *Symfony_1.2.12.*

## 1.6 Framework para interfaz de aplicaciones web

### 1.6.1 JQuery

Es un marco de trabajo de Javascript, usa licencias como MIT y GPL para poder hacer distribuciones libre y aplicaciones privadas, permite simplificar la manera de interactuar con los documentos HTML. Su eficiencia lo ha llevado a ser uno de los marcos de trabajo más utilizados en el mundo para el desarrollo de interfaz, ofrece una serie de funciones que reducen el código javascript en la aplicación.

**Algunas características del marco de trabajo**

- ✓ Selección de eventos y elementos DOM[21].
- ✓ Es compatible con navegadores como: Firefox 2.0+, Internet Explorer 6+, Safari 2.0.2+ y Opera 9+.
- ✓ Incluye soporte para CSS[22].
- ✓ Manipula las hojas con estilo CSS.
- ✓ Incluye efectos y animaciones para la aplicación.

La librería o marco de trabajo consiste en un único fichero JavaScript que contiene las funcionalidades comunes de DOM, eventos, efectos y AJAX. La característica principal de la biblioteca es que permite cambiar el contenido de una página web sin necesidad de recargarla, mediante la manipulación del árbol DOM y peticiones AJAX.

### 1.6.2 Ext JS

**ExtJS** es un excelente marco de trabajo para realizar interfaces web, proporciona un rendimiento excelente. El marco está totalmente orientado a objetos, está escrito en el lenguaje JavaScript. ExtJS se implementó con el uso de patrones.

- ✓ Patrones de creación: Singleton.
- ✓ Patrones estructurales: Peso Mosca

- ✓ Patrones de comportamiento: Patrón de observador

Es compatible con los principales navegadores web existentes, a continuación se listan algunos de los navegadores:

- ✓ Windows Internet Explorer versión 6 y posteriores
- ✓ Mozilla Firefox versión 1.5 y posteriores
- ✓ Apple Safari versión 2 y posteriores
- ✓ Opera versión 9 y posterior

Dispone de un conjunto de componentes para el desarrollo de aplicaciones web, como son:

- ✓ cuadros y áreas de texto
- ✓ campos para fechas
- ✓ campos numéricos
- ✓ radiobuttons y checkboxs.
- ✓ árboles de datos y pestañas
- ✓ paneles divisibles en secciones

**¿Por qué ExtJS?**

Este marco de trabajo es compatible con una gran cantidad de navegadores web, está disponible bajo la licencia GPL versión 3 Comercial. Se integra a marcos de trabajo como Symfony y CakePHP, tiene una gran estabilidad, sus interfaces son muy amigables para el usuario. Tiene una gran comunidad de desarrollo. Cuenta además con una gran documentación.

## 1.7 Metodologías de desarrollo del software

Un Proceso de Desarrollo de Software es la definición del conjunto de actividades que guían los esfuerzos de las personas implicadas en el proyecto, a modo de plantilla que explica los pasos necesarios para terminar el proyecto. El proceso de desarrollo del software, define el conjunto de actividades precisas para convertir los requisitos de los usuarios en el conjunto seguro y resistente de artefactos que componen un producto de software. (Ivar Jacobson, 2000)

Existen dos vertientes fundamentales de acuerdo con el método de desarrollo, los cuales están dirigidos a beneficiar la labor de los desarrolladores de software: los métodos ágiles y los métodos pesados. Las metodologías ágiles proponen mejorar la calidad del producto software a través de la comunicación inmediata y directa, mientras que las metodologías pesadas proponen que sea a través del orden y la documentación.

Entre las metodologías de desarrollo de software más utilizadas a nivel mundial, en el país, y específicamente en la UCI, se pueden encontrar:

- ✓ Extreme Programing (XP).
- ✓ Rational Unified Process (RUP).
- ✓ SCRUM

Para realizar el sistema que se quiere desarrollar es necesario definir la metodología a utilizar, y es por ello que hay que analizar cada una de ellas, teniendo en cuenta sus características, ventajas y desventajas, y su adecuación al desarrollo del sistema.

### 1.7.1 Programación Extrema (XP)

La metodología XP, Extreme Programing en inglés, es una metodología ágil, que se centra en el aumento de las relaciones interpersonales para lograr un desempeño exitoso en el desarrollo del software, donde el trabajo en equipo es muy importante, preocupándose por la existencia de un ambiente de trabajo óptimo, y porque los desarrolladores realicen un buen aprendizaje. Se basa en el trabajo orientado directamente al objetivo, teniendo en cuenta para esto la velocidad de reacción para la implementación y para los cambios que puedan surgir durante el desarrollo del proceso, además se basa en la simplicidad, la comunicación y la realimentación o reutilización del código desarrollado. Es el más destacado de los procesos ágiles de desarrollo de software. XP minimiza el riesgo de fallo del proceso manteniendo dentro del equipo a un representante competente del cliente, este representante es quién responderá a todas las preguntas y dudas que surjan por parte del equipo de desarrollo durante el proceso, de forma que no se retrase la toma de decisiones.

XP tiene cuatro variables principales, que son: Costo, Tiempo, Calidad y Alcance, y como fases se definen: Planificación, Diseño, Desarrollo y Pruebas. Para su uso, XP define varios roles, como son: Programador, Cliente, Encargado de Pruebas, Encargado de seguimiento, Entrenador, Consultor y Gestor.

XP se basa en historias de uso (UseStories), estas historias las escribe el cliente o su representante dentro de equipo y describen los escenarios claves del funcionamiento del software, a partir de estas se generan las entregas (releases) entre el equipo y el cliente.

Una característica fundamental de XP, es que el código siempre se produce en parejas, parejas que van cambiando constantemente para lograr así que todo el equipo sepa y pueda modificar según necesidades el código generado, esto logra en el equipo que los integrantes aprendan entre sí y compartan todo el código.

### 1.7.2 SCRUM

**SCRUM**: define un marco para la gestión de proyectos, que se ha utilizado exitosamente durante los últimos años.

**Dentro de sus principales características se encuentran:**

- ✓ Equipos auto dirigidos.
- ✓ Utiliza reglas para crear un entorno ágil de administración de proyectos.
- ✓ No prescribe prácticas específicas de ingeniería.
- ✓ Los requisitos se capturan como ítems de la lista reserva del producto.
- ✓ El producto se construye en una serie de sprints de un mes de duración.
- ✓ Usado para proyectos complejos con requisitos cambiantes.
- ✓ Basado en un control de proceso empírico.

### 1.7.3 Proceso Unificado del Software (RUP)

El Proceso Unificado del Software (RUP por sus siglas en inglés), es una metodología de desarrollo de software, pesada y orientada a objetos, es un marco de trabajo genérico que puede ser especializado para una gran variedad de software para distintas áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos.

RUP está basado en componentes interconectados a través de interfaces y utiliza UML[23] como lenguaje de modelado de procesos, es dirigido por casos de uso, centrado en la arquitectura e iterativo e incremental.

Dirigido por casos de uso, porque los casos de uso reflejan lo que los usuarios futuros desean y necesitan, lo cual se capta cuando se modela el negocio y se representa a través de los requisitos, a partir de ahí, los casos de uso guían el proceso de desarrollo.

Centrado en la arquitectura, porque la arquitectura muestra una visión común del sistema completo, en la que el equipo del proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes producirlos económicamente.

Iterativo e incremental, porque RUP propone que cada fase se desarrolle en iteraciones, y cada iteración tiene que proponerse un incremento en el proceso de desarrollo del software.

**RUP define como sus principales elementos:**

**Trabajadores ("quién"):** Define el comportamiento y responsabilidades (rol) de un individuo, grupo de individuos, sistema automatizado o máquina, que trabajan en conjunto con un equipo.

**Actividades ("cómo"):** Es una tarea que tiene un propósito claro, es realizada por un trabajador y manipula elementos.

**Artefactos ("qué"):** Productos tangibles del proyecto que son producidos, modificados y usados por las actividades. Pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables.

**Flujo de Actividades ("cuándo"):** Secuencia de actividades realizadas por los trabajadores y que produce un resultado de valor observable. Las actividades son agrupadas en grupos lógicos, donde se

definen 9 flujos de trabajo fundamentales, de los cuales solo los primeros 6 son considerados flujos de ingeniería, y los 3 restantes y últimos como flujos de apoyo. Estos flujos son:

- ✓ Modelado de Negocio
- ✓ Requisitos
- ✓ Análisis y Diseño
- ✓ Implementación
- ✓ Pruebas
- ✓ Despliegue
- ✓ Administración de Proyecto
- ✓ Administración de Configuración y Cambios
- ✓ Entorno

**Además, RUP define cuatro fases importantes en el desarrollo, las mismas son:**

- ✓ **Conceptualización (Concepción o Inicio):** La cual describe el negocio y delimita el proyecto, se definen sus alcances con la identificación de los casos de uso del sistema.
- ✓ **Elaboración:** Donde se define la arquitectura que tendrá el sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen.
- ✓ **Construcción:** Donde se obtiene un producto listo para su utilización que está documentado y tiene un manual de usuario, se obtienen una o más entregas del producto que han pasado las pruebas y por último, pero no menos importante.
- ✓ **Transición:** En la cual el release (versión) ya está listo para su instalación en condiciones reales.

**En RUP existen algunos principios fundamentales de desarrollo de software que son:**

- ✓ **Adaptar el proceso:** Lo que se refiere a que los procesos deben de adaptarse al tamaño de los proyectos o de la organización.
- ✓ **Balancear prioridades:** Se debe de encontrar un balance que satisfaga los deseos de todos.
- ✓ **Colaboración entre equipos:** Los proyectos de desarrollo de software no son llevados a cabo por una sola persona, sino varias o varios equipos de desarrollo, los cuales deben de contar con una buena comunicación para que esto les permita coordinar esfuerzos.

- ✓ **Demostrar valor iterativamente:** Los proyectos aunque sea de manera interna deben de entregarse de manera iterada en cada iteración se analiza el avance, estabilidad, calidad del producto.
- ✓ **Elevar el nivel de abstracción:** Esto previene a los ingenieros de software ir directamente de los requisitos del cliente a la codificación, un nivel alto de abstracción permite discusiones sobre diversos niveles de arquitectura, los cuales se pueden acompañar por representaciones visuales de la arquitectura como por ejemplo utilizando UML.
- ✓ **Enfocarse a la calidad:** El control de calidad deberá de ser llevado a cabo no solo al final de cada iteración sino a todo lo largo de toda la producción.

En el desarrollo del sistema no se utilizará la metodología XP debido a que no se cuenta con la participación directa del cliente dentro del equipo de desarrollo, además, porque no se cuenta con el personal necesario para realizar la programación en pareja, dos de los principales principios de XP.

SCRUM no se utilizará debido a que no prescribe prácticas específicas de ingeniería, lo cual se interpone a la organización de los procesos a automatizar, a la hora de implementar, además, porque no brinda la organización del trabajo que se requiere, ni la documentación ideal una vez terminado el producto.

Para el desarrollo del sistema se utilizará la metodología RUP, debido a la organización que brinda, así como la documentación que genera en cada iteración. Esta metodología, brinda una excelente organización y planificación, al dividir cada etapa por flujos de trabajo, donde los casos de uso y la arquitectura son los pilares fundamentales, realizando diferentes iteraciones que mejoran constantemente el producto.

## 1.8 Herramientas CASE

Las herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Computadora) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo y de dinero. Estas herramientas sirven de ayuda en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, cálculo de costos, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras.

### 1.8.1 Rational Rose

**Racional Rose** es una herramienta CASE utilizada para el modelado visual. Permite completar los flujos fundamentales de RUP, incluye además herramientas de ingeniería inversa y generación de código. Está entre las mejores herramientas para traducir requisitos de alto nivel a una arquitectura basada en componentes. Se ha convertido en una de las mejores opciones por la notación estándar que brinda para especificar, visualizar y construir productos software y sistemas, debido a que está en la avanzada en cuanto al desarrollo de UML.

### 1.8.2 Visual Paradigm

**Visual Paradigm** es una herramienta CASE gratuita para el modelado UML. Presenta características gráficas muy cómodas para el usuario, haciendo más fácil el modelado de diagramas que sigue el estándar de UML, los cuales son: Diagramas de clase, Casos de Uso, Comunicación, Secuencia, Estado, Actividad, Componentes.

**Ventajas de Visual Paradigm:**

- ✓ Entorno de creación de diagramas para UML 2.0.
- ✓ Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.
- ✓ Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- ✓ Capacidades de ingeniería directa e inversa.
- ✓ Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- ✓ Disponibilidad de múltiples versiones, para cada necesidad.
- ✓ Disponibilidad en múltiples plataformas.

- ✓ Producto de calidad.
- ✓ Soporta aplicaciones Web.
- ✓ Varios idiomas.
- ✓ Generación de código para Java y exportación como HTML.
- ✓ Fácil de instalar y actualizar.
- ✓ Compatibilidad entre ediciones.

Teniendo en cuenta la política de desarrollo de la universidad, se define como herramienta CASE para el modelado del sistema, Visual Paradigm, debido a las características antes mencionadas, ya que una de sus características principales es ser un software libre para el modelado.

## 1.9 Sistemas Gestores de Base de Datos (SGBD)

Los sistemas gestores de bases de datos son un tipo de software muy específico que sirve de interfaz entre el usuario, las aplicaciones y la base de datos. Están compuestos por un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Su propósito general es manejar de forma sencilla y ordenada los datos que posteriormente se convertirán en información relevante para el manejo correcto de información. Los objetivos fundamentales de los Sistemas Gestores de Bases de Datos (SGBD) son: independencia de los datos y los programas de aplicación, minimización de la redundancia, integración y sincronización de las bases de datos, integridad de los datos, seguridad y protección de los datos, facilidad de manipulación de la información y control centralizado, entre los más usados mundialmente están: Oracle y PostgreSQL.

### 1.9.1 Oracle

La compañía es la líder mundial como distribuidora de programas para administración de la información, y la segunda compañía de software independiente del mundo. Oracle es uno de los Sistemas Gestores de Bases de Datos (SGBD) más comunes y poseedor de las características más avanzadas. Oracle es una base de datos relacional para entornos cliente/servidor. Es un sistema gestor de base de datos robusto,

tiene muchas características que nos garantizan la seguridad e integridad de los datos; que las transacciones se ejecuten de forma rápida y correcta, sin causar inconsistencias en la base de datos; ayuda a administrar y almacenar grandes volúmenes de datos; estabilidad, escalabilidad. Oracle está disponible en múltiples plataformas como Windows, Linux. Su desarrollo constante lo convierte en una potente solución empresarial. Aunque su dominio en el mercado de servidores empresariales ha sido casi total, actualmente existe la competencia otros con licencia de Software Libre, el más utilizado actualmente es PostgreSQL.

### 1.9.2 PostgreSQL

Es un SGBD relacional de software libre y orientado a objetos, liberado bajo la licencia BSD[24]. Posee las características de los más potentes sistemas comerciales como Oracle o SQL Server, con la ventaja de que su licencia es gratuita. Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una sola compañía sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales. Tiene más de 17 años de desarrollo activo y se ha ganado la reputación de ser confiable y mantener la integridad de los datos. Se ejecuta en la mayoría de los sistemas operativos más utilizados en el mundo como Linux, varias versiones de UNIX y Windows.

**Algunas de sus características.**

- ✓ Alta concurrencia
- ✓ Amplia variedad de tipos nativos
- ✓ Cuenta con disparadores o *triggers*
- ✓ Restauración de base de datos en procesos paralelos, que acelera la recuperación de un respaldo hasta 8 veces.
- ✓ Tiene un motor de búsqueda semántica

Se decide utilizar PostgreSQL por las razones siguientes: posee una instalación ilimitada, es decir, nadie puede demandarlo por violar acuerdos de licencia, puesto que no hay costo asociado a la licencia del software. Además posee ahorros considerables en costos de operación puesto que ha sido diseñado y creado para tener un mantenimiento y ajuste mucho menor que otros productos, conservando todas las características, estabilidad y rendimiento. Es extensible ya que el código fuente está disponible para todos

sin costo. Es multiplataforma, además la universidad tiene una política de desarrollo con el gestor de base de datos PostgreSQL.

## 1.10 Conclusiones Parciales

En el presente capítulo se ha hecho un estudio profundo de las herramientas de administración de requisitos más usadas en el mundo, analizando sus características, teniendo en cuenta que el análisis nos llevó a la conclusión de que las herramientas más poderosas son de software privativo. De todas las herramientas RequisitePro es la más completa y usada en el mundo, analizamos también herramientas de software libre, evidenciando que OSRMT es la más usada, pero aún no cumple con todos los aspectos que se desean para una buena administración de requisitos. Partiendo de que la UCI fomenta el desarrollo de software libre y que no podemos pagar grandes sumas de dinero para la adquisición de herramientas para gestionar los requisitos, es necesario realizar una aplicación que cumpla con los requisitos que se necesitan para el desarrollo de software en los proyectos de la UCI. Se tuvo en cuenta los lenguajes más factibles para su implementación, se utilizará como lenguaje de programación php5+, el marco de trabajo Symfony_1.2.12, apache 2.0 como servidor web. Como herramienta CASE para el modelado, Visual Paradigm y PostgreSQL 8.4 para gestionar los datos que serán almacenados.

# CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

## 2.1Introducción

En este capítulo se propone una solución al problema, presentando la propuesta del sistema, y describiéndose además los artefactos generados en la modelación del negocio, levantamiento de requerimientos, específicamente: modelo de dominio o conceptual, descripción de los requerimientos funcionales y no funcionales, y definición de actores y casos de uso del sistema. El resultado del presente capítulo es de gran importancia para el posterior análisis y el diseño del sistema.

## 2.2Información que se maneja

Cada proyecto que se vaya a realizar debe cumplir ciertos requisitos, estos pueden ser funcionales o no funcionales. Un proyecto tiene varios aspectos que deben tenerse en cuenta:

- ✓ Objetivo
- ✓ Negocio
- ✓ Requisitos
- ✓ Casos de Uso
- ✓ Implementación
- ✓ Prueba

La aplicación que se realizará podrá gestionar los aspectos fundamentales de los requisitos en un determinado proyecto a lo largo de todo el ciclo de vida.

## 2.3Información de un requisito

- ✓ Nombre
- ✓ Descripción
- ✓ Trazabilidad
- ✓ Ubicación
- ✓ Estado
- ✓ Asignado
- ✓ Aceptado
- ✓ Cerrado

- ✓ Aprobado

## 2.4 Propuesta del Sistema

El sistema propuesto debe manejar los requisitos de un proyecto de manera eficiente, permitiendo a un usuario crear un proyecto y a este especificarle sus requisitos, y actualizarlo cada vez que lo estime conveniente. Debe garantizar la seguridad de la información manejada, ya que esta puede servir para tomar decisiones importantes en los proyectos de La Universidad.

El sistema presentará al usuario una interfaz amigable y fácil de usar. Además de administrar los requisitos de software a lo largo del ciclo de vida, permitirá al usuario definir distintas vistas para cada uno de los elementos de la especificación. Debe ser una herramienta en línea para centralizar la gestión de los requerimientos por las diferentes áreas de la universidad (polos, facultades, centros de producción)

## 2.5 Modelo de Dominio

Debido a que no existe un proceso de negocio bien definido, se lleva a cabo el modelo de dominio o modelo conceptual, donde se explican los principales conceptos y eventos que suceden en el entorno en el que se trabaja.

### 2.5.1 Glosario de términos del modelo de dominio

- ✓ **Universidad**: Institución donde se instalará la aplicación.
- ✓ **Proyectos**: Entidad que será gestionada con la aplicación.
- ✓ **Usuario**: Persona con cierto acceso a la aplicación.
- ✓ **Administrador de Sistema**: Persona con todos los privilegios dentro de la aplicación.
- ✓ **Requisitos**: Aspectos en general que debe cumplir un software.

### 2.5.2 Diagrama del Modelo del Dominio

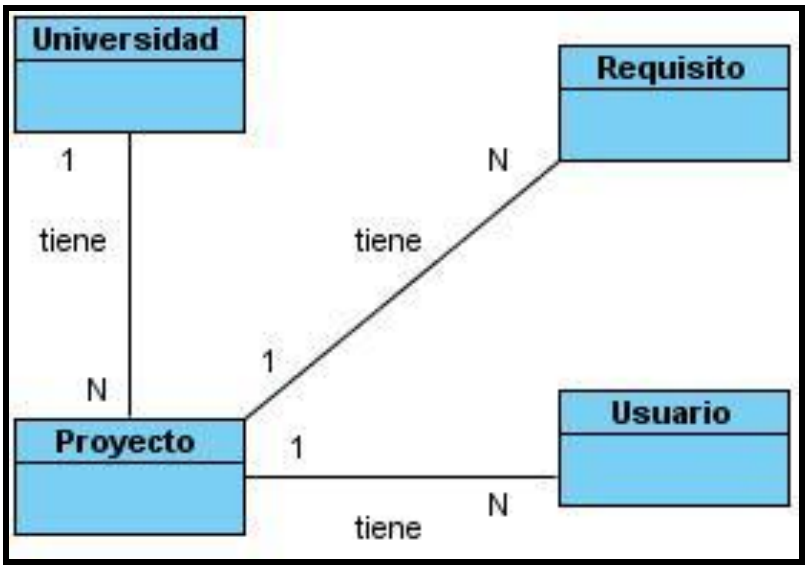


Figura 1: Diagrama del Modelo de Dominio.

## 2.6 Especificación de Requerimientos de Software

La IEEE ( Institute of Electrical and Electronics Engineers, en español: Instituto de Ingenieros Eléctricos y Electrónicos) define un requisito como una condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo.

### 2.6.1 Requisitos Funcionales (RF)

RF1. Autenticación.

RF1.1. Permitir el acceso a los usuarios autorizados.

RF1.2. Denegar el acceso a los usuarios no autorizados.

RF2. Gestionar Usuarios.

RF2.1. Adicionar un nuevo usuario.

RF2.2. Modificar un usuario existente.

RF2.3. Eliminar un usuario.

RF3. Gestionar Objetivos.

RF3.1. Adicionar Objetivo.

RF3.2. Modificar Objetivo.

RF3.3. Eliminar Objetivo.

RF4. Gestionar Negocio.

RF4.1. Gestionar Casos de Uso del Negocio (CUN).

RF4.1.1. Adicionar CUN.

RF4.1.2. Modificar CUN.

RF4.1.3. Eliminar CUN.

RF4.2. Gestionar Procesos.

RF4.2.1. Adicionar Procesos.

RF4.2.2. Modificar Procesos.

RF4.2.3. Eliminar Procesos.

RF4.3. Gestionar Clases del Dominio.

RF4.3.1. Adicionar clases del dominio.

RF4.3.2. Modificar clases del dominio.

RF4.3.3. Eliminar clases del dominio.

RF5. Gestionar Requisitos del Sistema (RS).

RF5.1. Adicionar RS.

RF5.2. Modificar RS.

RF5.3. Eliminar RS.

RF6. Gestionar Casos de Uso del Sistema (CUS).

RF6.1. Adicionar CUS.

RF6.2. Modificar CUS.

RF6.3. Eliminar CUS.

RF7. Gestionar Diseño.

RF7.1. Gestionar Diagrama de Clases (DC).

RF7.1.1. Adicionar DC.

RF7.1.2. Modificar DC.

RF7.1.3. Eliminar DC.

RF7.2. Gestionar Entidades de la Base de Datos (EBD).

RF7.2.1. Adicionar EBD.

RF7.2.2. Modificar EBD.

RF7.2.3. Eliminar EBD.

RF7.3. Gestionar Diagrama de iteración (DI).

RF7.3.1. Adicionar DI.

RF7.3.2. Modificar DI.

RF7.3.3. Eliminar DI.

RF8. Gestionar Implementación.

RF8.1. Gestionar Diagramas de Clases.

RF8.1.1. Adicionar Diagramas de Clases.

RF8.1.2. Modificar Diagramas de Clases.

RF8.1.3. Eliminar Diagramas de Clases.

RF8.2. Gestionar Diagrama de Despliegue.

RF8.2.1. Adicionar Diagrama de Despliegue.

RF8.2.2. Modificar Diagrama de Despliegue.

RF8.2.3. Eliminar Diagrama de Despliegue.

RF9. Gestionar Casos de Prueba.

RF9.1 Adicionar Casos de Prueba.

RF9.2 Modificar Casos de Prueba.

RF9.3. Eliminar Casos de Prueba.

RF10. Gestionar Proyecto

RF10.1 Adicionar Proyecto

RF10.2 Modificar Proyecto

RF10.3 Eliminar Proyecto

### 2.6.2 Requerimientos No Funcionales

- ✓ **Apariencia o Interfaz:**

El sistema debe proveer una interfaz sencilla y fácil de usar para la interacción con el usuario. Las imágenes y colores de las interfaces de usuario deben identificar el negocio donde se implantará el

sistema.

- ✓ **Usabilidad:**

Un usuario avanzado, con conocimientos de informática demorará 1 semana en su adiestramiento para el uso de la aplicación.

Un usuario básico demorará 3 semanas en su adiestramiento para el uso de la aplicación.

El sistema debe presentar una ayuda para el asesoramiento en el uso del sistema.

- ✓ **Seguridad**:

El sistema debe garantizar que todos sus servicios sean usados por el personal autorizado y que la información sea modificada y vista solamente por el personal con permiso para ello.

Un administrador del sistema tendrá la responsabilidad de gestionar toda la información de las cuentas de usuario.

- ✓ **Disponibilidad**:

Permitir a los usuarios conectarse desde cualquier computadora dentro de La Universidad.

El sistema tendrá todos sus servicios disponibles las 24 horas durante todos los días de la semana.

**Portabilidad**:

La implementación de la aplicación en PHP posibilita que sea portable y pueda ejecutarse en plataformas Windows y *nix.

- ✓ **Rendimiento**:

El tiempo que demorará el sistema en brindar una respuesta a las solicitudes que realicen los usuarios estará en el intervalo de 3 segundos a 5 segundos.

El sistema debe presentar pocas imágenes en las páginas para garantizar una respuesta más rápida.

- ✓ **Hardware**:

Las características de hardware mínimas para los servidores de bases de datos y de aplicación son: Servidor P4, Velocidad del procesador: 2.40Hgz, Memoria RAM de 256 MB y 10 GB de espacio en el disco duro.

- ✓ **Restricciones de Diseño e Implementación:**

Para implementar el sistema se usará el lenguaje de programación PHP5.

El Sistema Gestor de Base de Datos será Postgresql-8.4.

- ✓ **Software:**

En las estaciones de trabajo clientes se necesita la instalación de un navegador web que puede ser Internet Explorer, Mozilla Firefox o Opera.

El navegador debe tener el intérprete de JavaScript habilitado.

## 2.7 Descripción del los Actores del Sistema

Los actores del sistema son aquellas personas que interactuarán directamente con el mismo. Para el desarrollo de la aplicación se definen dos tipos de usuarios que son los que tendrán acceso al sistema, los cuales se describen a continuación.

| Actores | Descripción |
|---|---|
| Administrador | Tiene todos los privilegios del sistema, encargado de gestionar los usuarios, proyectos y darles los privilegios necesarios a cada usuario. |
| Analista | Es el encargado de gestionar los requisitos de un software. |

**Tabla 1:** Descripción del los Actores del Sistema

## 2.8 Descripción Breve de los Casos de Uso del Sistema

Con el objetivo de crear un modelo de las funcionalidades deseadas para el sistema y su entorno, y debido a que se utilizan como entrada en los flujos de análisis, diseño, implementación y prueba, se

definen una serie de casos de uso del sistema, realizándose el diagrama de casos de uso del sistema y una descripción detallada de cada uno de ellos. La descripción ampliada de los casos de uso del sistema muestra una descripción detallada de los mismos (**Ver Anexo II:** Descripción ampliada de los casos de uso del sistema).

| **CU-1** | **Autenticar** |
|---|---|
| Actor | Usuario |
| Descripción | Permite o limita el acceso a los usuarios y se inicia cuando un usuario quiere acceder al sistema. |
| Referencia | RF1, RF1.1, RF1.2 |

**Tabla 2: Breve descripción del caso de uso "Autenticar"**

| **CU-2** | **Gestionar Usuario** |
|---|---|
| Actor | Administrador |
| Descripción | Permite al administrador del sistema crear, modificar y eliminar usuarios en el sistema. |
| Referencia | RF 2, RF 2.1, RF 2.2, RF 2.3 |

**Tabla 3: Breve descripción del Caso de Uso "Gestionar Usuario"**

| **CU-3** | **Gestionar Objetivo** |
|---|---|
| Actor | Analista |
| Descripción | Permite crear, modificar y eliminar objetivos en el sistema. |
| Referencia | RF 3, RF 3.1, RF 3.2, RF 3.3 |

**Tabla 4: Breve descripción del Caso de Uso "Gestionar Objetivo"**

| **CU-4** | **Gestionar Casos de Uso del Negocio (CUN)** |
|---|---|
| Actor | Analista |
| Descripción | Permite crear, modificar y eliminar información de los casos de uso del negocio. |

| | |
|---|---|
| Referencia | RF 4.1, RF4.1.1, RF4.1.2, RF4.1.3 |

**Tabla 5: Breve descripción del Caso de Uso "Gestionar CUN"**

| **CU-5** | **Gestionar Proceso** |
|---|---|
| Actor | Analista |
| Descripción | Permite crear, modificar y eliminar información de los procesos. |
| Referencia | RF 4.2, RF4.2.1, RF4.2.2, RF4.2.3 |

**Tabla 6: Breve descripción del Caso de Uso "Gestionar Proceso"**

| **CU-6** | **Gestionar Clases del Dominio** |
|---|---|
| Actor | Analista |
| Descripción | Permite crear, modificar y eliminar información de las clases del dominio. |
| Referencia | RF 4.3, RF4.3.1, RF4.3.2, RF4.3.3 |

**Tabla 7: Breve descripción del Caso de Uso "Gestionar Clase del Dominio"**

| **CU-7** | **Gestionar Requisitos del Sistema** |
|---|---|
| Actor | Analista |
| Descripción | Permite crear, modificar y eliminar requisitos en el sistema. |
| Referencia | RF5, RF 5.1, RF 5.2, RF 5.3 |

**Tabla 8: Breve descripción del Caso de Uso "Gestionar Requisitos del Sistema"**

| **CU-8** | **Gestionar Casos de Uso** |
|---|---|
| Actor | Analista |
| Descripción | Permite crear, modificar y eliminar casos de uso en el sistema. |
| Referencia | RF6, RF6.1, RF6.2, RF6.3 |

**Tabla 9: Breve descripción del Caso de Uso "Gestionar Casos de Uso"**

| CU-9 | Gestionar Diagrama de Clases |
|---|---|
| Actor | Analista |
| Descripción | Permite crear, modificar y eliminar diagrama de clase de diseño en el sistema. |
| Referencia | RF 7.1, RF7.1.1, RF7.1.2, RF7.1.3 |

**Tabla 10: Breve descripción del Caso de Uso "Gestiona Diagrama de Clase"**

| CU-10 | Gestionar Entidades de la Base de Datos |
|---|---|
| Actor | Analista |
| Descripción | Permite crear, modificar y eliminar entidades de la base de datos en el sistema. |
| Referencia | RF7.2, RF7.2.1, RF7.2.2, RF7.2.3 |

**Tabla 11: Breve descripción del Caso de Uso "Gestionar Entidades de la Base de Datos"**

| CU-11 | Gestionar Diagrama de Iteración |
|---|---|
| Actor | Analista |
| Descripción | Permite crear, modificar y eliminar diagramas de iteración en el sistema. |
| Referencia | RF7.3, RF7.3.1, RF7.3.2, RF7.3.3 |

**Tabla 12: Breve descripción del Caso de Uso "Gestionar Diagrama de Iteración"**

| CU-12 | Gestionar Diagrama de Clase |
|---|---|
| Actor | Analista |
| Descripción | Permite crear, modificar y eliminar diagramas de clase de implementación en el sistema. |
| Referencia | RF8.1, RF8.1.1, RF8.1.2, RF8.1.3 |

**Tabla 13: Breve descripción del Caso de Uso "Gestionar Diagrama de Clase"**

| CU-13 | Gestionar Diagrama de Despliegue |
|---|---|
| Actor | Analista |
| Descripción | Permite crear, modificar y eliminar diagramas de despliegue en el sistema. |
| Referencia | RF8.2, RF8.2.1, RF8.2.2, RF8.2.3 |

**Tabla 14: Breve descripción del Caso de Uso "Gestionar Diagrama de Despliegue"**

| CU-14 | Gestionar Casos de Prueba |
|---|---|
| Actor | Analista |
| Descripción | Permite crear, modificar y eliminar información sobre los casos de pruebas. |
| Referencia | RF9, RF9.1, RF9.2, RF9.3 |

**Tabla 15: Breve descripción del Caso de Uso "Gestionar Casos de Prueba"**

| CU-15 | Gestionar Proyecto |
|---|---|
| Actor | Administrador |
| Descripción | Permite insertar, eliminar y modificar un proyecto de la aplicación. |
| Referencia | RF10, RF10.1, RF10.2, RF10.3 |

**Tabla 16: Breve descripción del Caso de Uso "Gestionar Proyecto"**

## 2.9Diagrama de Casos de Uso del Sistema

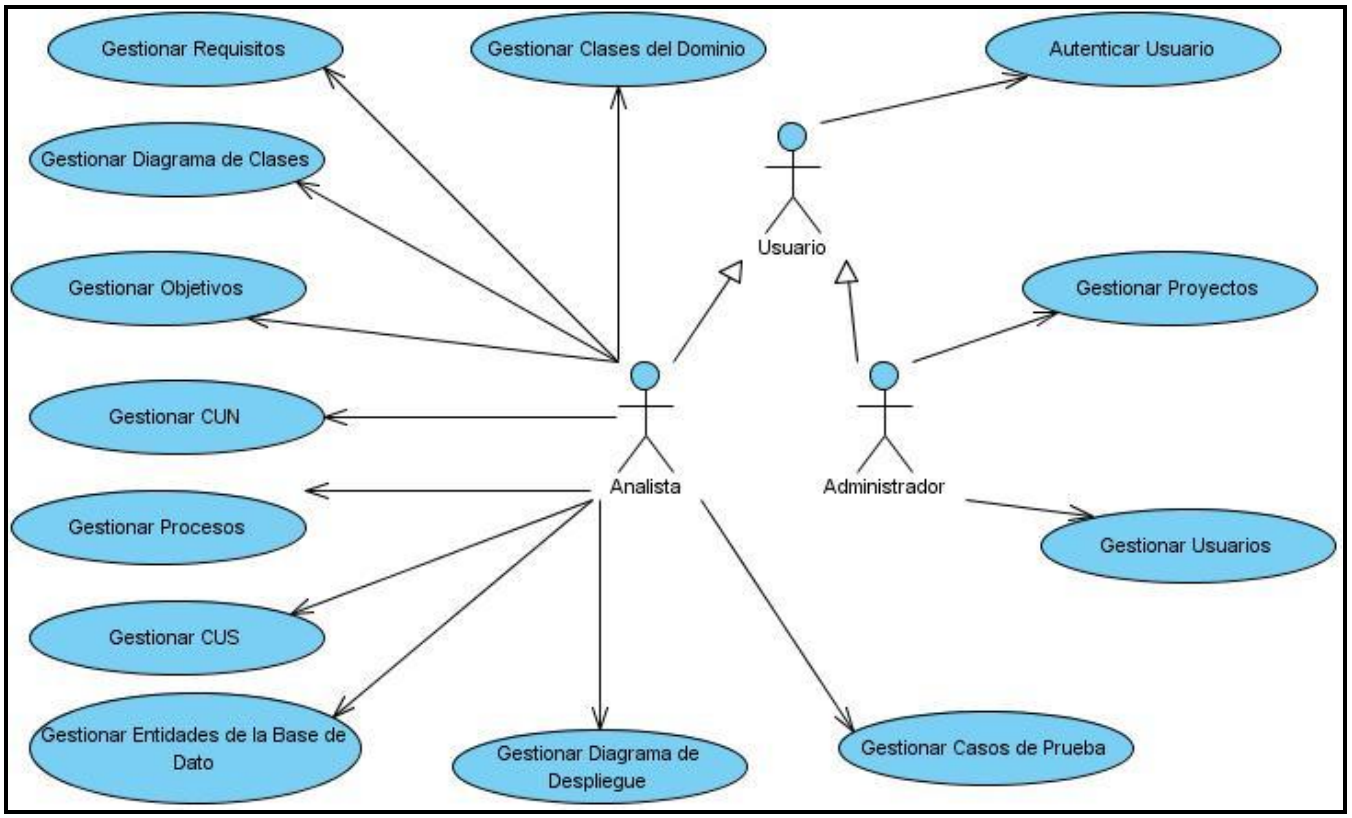


Figura 1: Diagrama de Casos de Uso del Sistema

## 2.10 Conclusiones Parciales

En este capítulo se identificaron los principales conceptos que se usarán para desarrollar el sistema, así como los tipos más importantes de objetos que existen y los eventos que suceden en el entorno en el que estará el sistema. Se obtuvieron los requerimientos funcionales y los no funcionales de la aplicación. Partiendo de los requisitos del sistema, posteriormente fueron presentados los casos de uso del sistema y sus relaciones con los actores del sistema. Realizando una detallada descripción de los casos de uso. Los artefactos generados en este capítulo son la base para comenzar el análisis y el diseño.

CAPÍTULO 3: ANÁLISIS Y DISEÑO

## 3.1 Introducción

En este capítulo se abordan los temas referentes al análisis y el diseño del sistema. A partir del resultado obtenido en el capítulo anterior se realiza el análisis, donde se consigue una comprensión más precisa de los requisitos, de manera que puedan ser refinados y estructurados, proporcionando una visión general del sistema, describiéndose en el diagrama de clases del análisis. Además, se obtiene el modelo de diseño, el cual incluye el diseño de la base de datos, el diagrama de clases del diseño, con sus respectivos diagramas de interacción. Se describen detalladamente las clases del diseño y las tablas de la base de datos. El resultado de este capítulo es muy importante porque sirve de entrada para la implementación.

## 3.2 Diagrama de Clases del Análisis

Con el objetivo de refinar y estructurar los requisitos obtenidos con anterioridad, profundizándose en el dominio de la aplicación, y para obtener una mayor comprensión del problema para modelar la solución, se realiza el análisis, cuyo principal resultado es el diagrama de clases del análisis (DCA). Para desarrollar el sistema, fue necesario hacer este diagrama, el cual representa los conceptos en el dominio del problema. Representa las cosas del mundo real, no de la implementación automatizada de estas cosas.

**Los diagramas de clases del análisis resultante se muestran a continuación:**

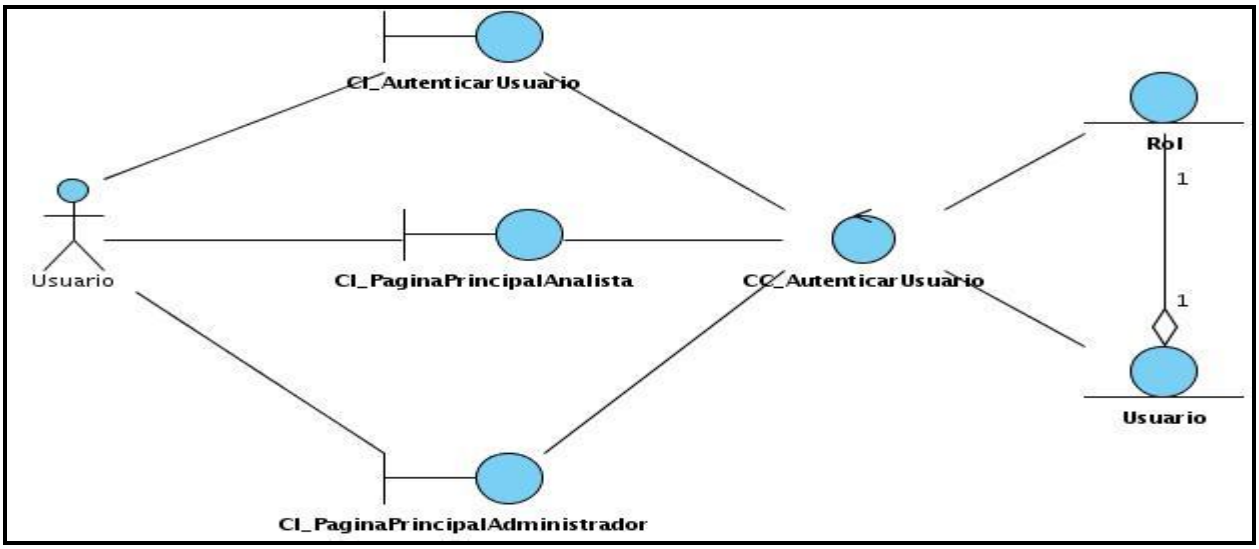


Figura 3: DCA Autenticar Usuario.

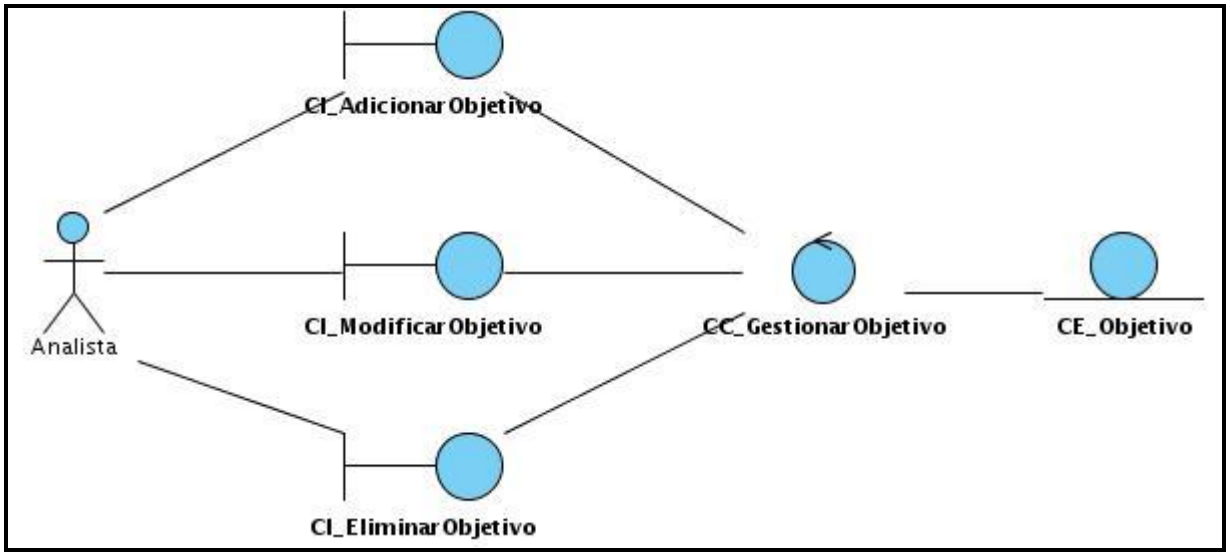


Figura 4: DCA Gestionar Objetivo.

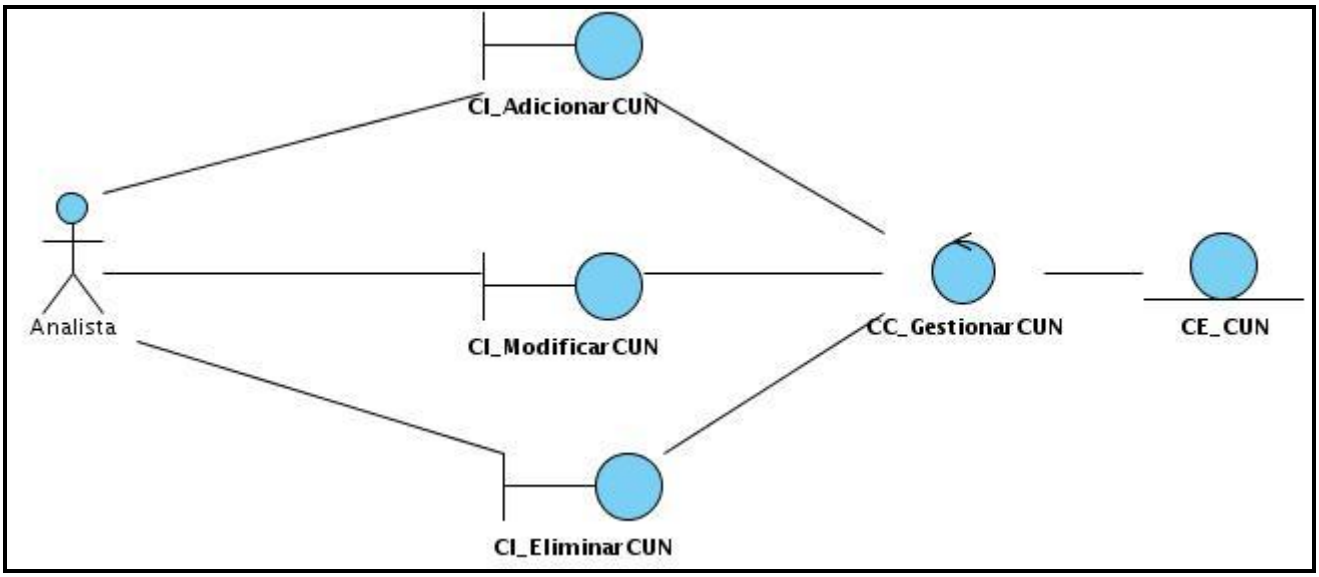


Figura 5: DCA Gestionar Caso de Uso del Negocio.

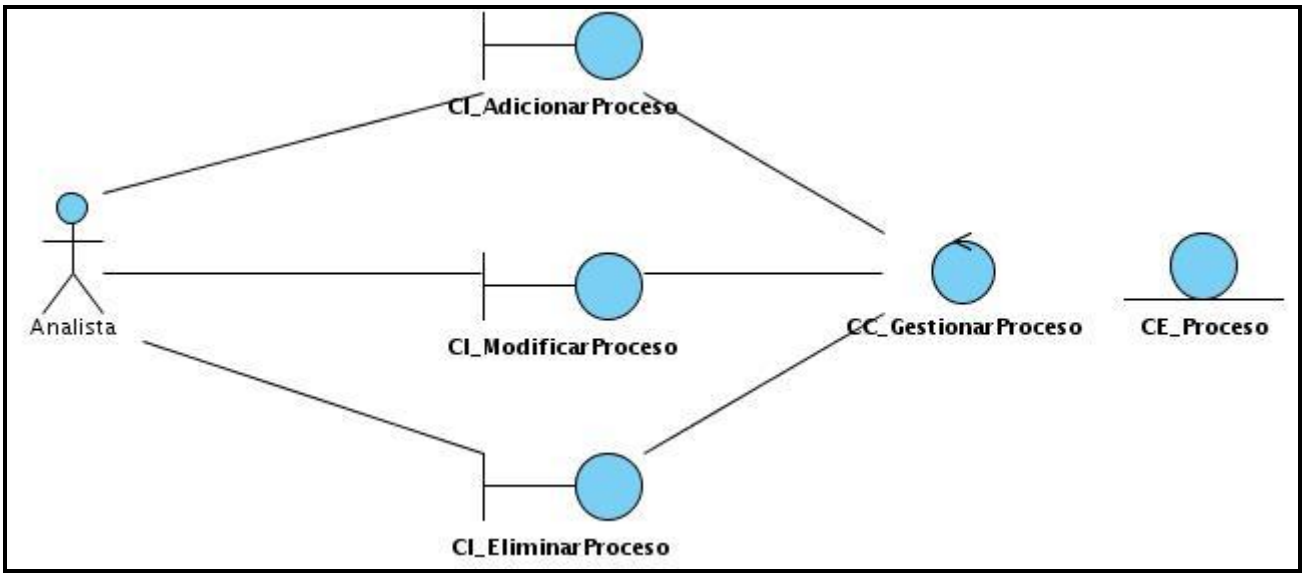


Figura 6: DCA Gestionar Proceso.

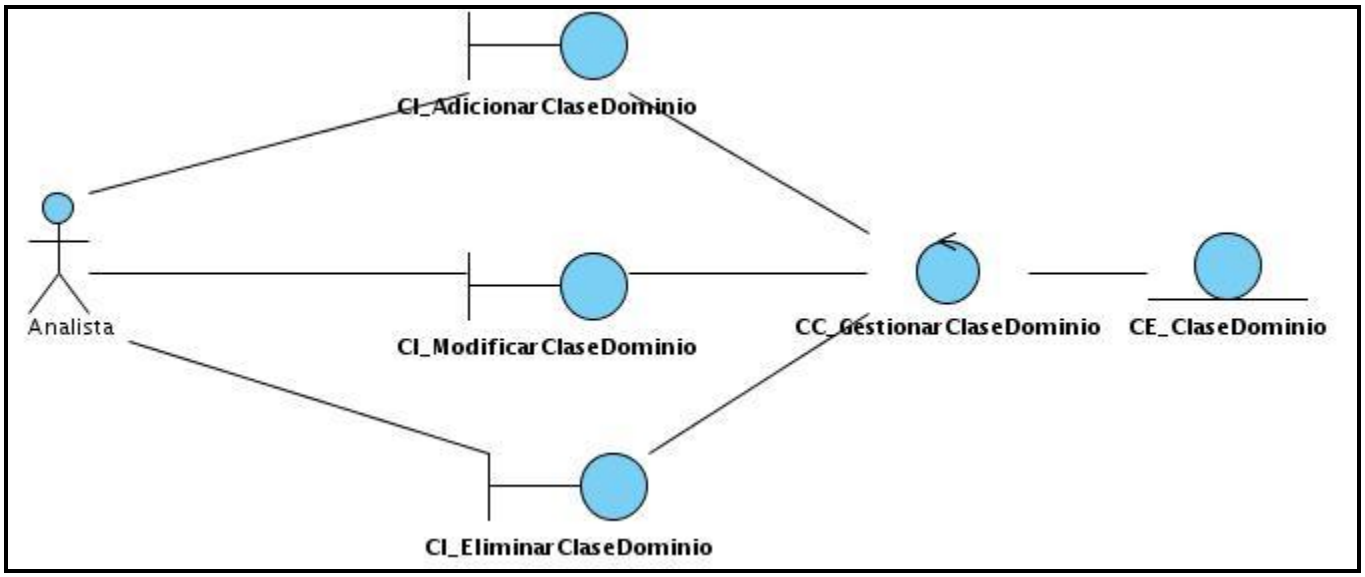


Figura 7: DCA Gestionar Clase del Dominio.

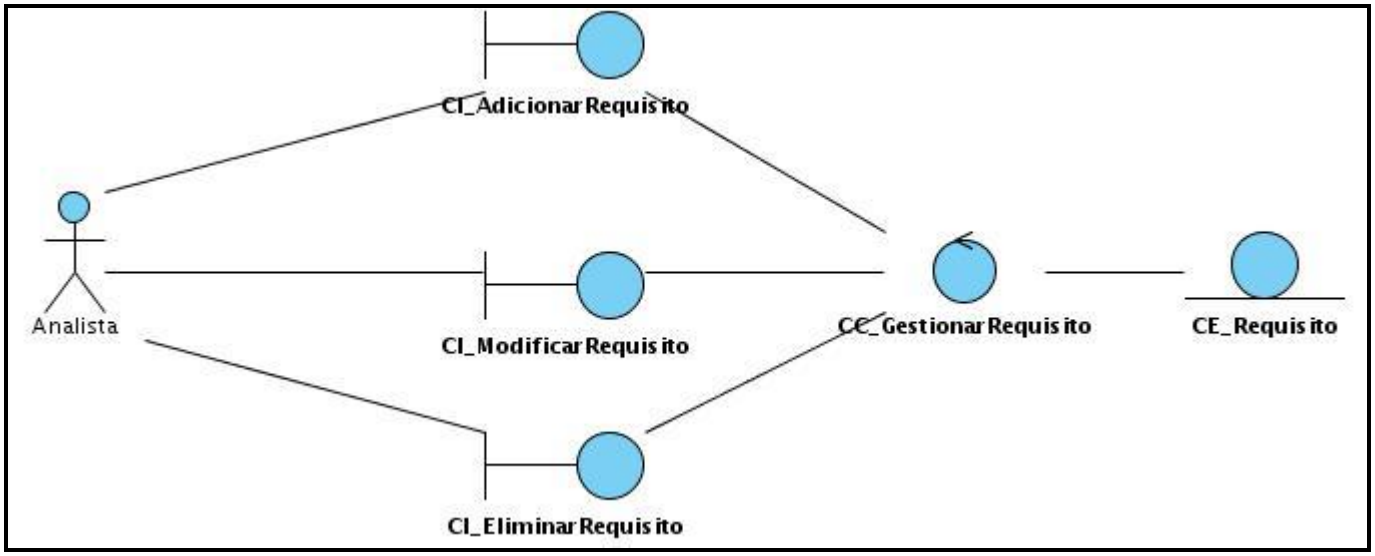


Figura 8: DCA Gestionar Requisito.

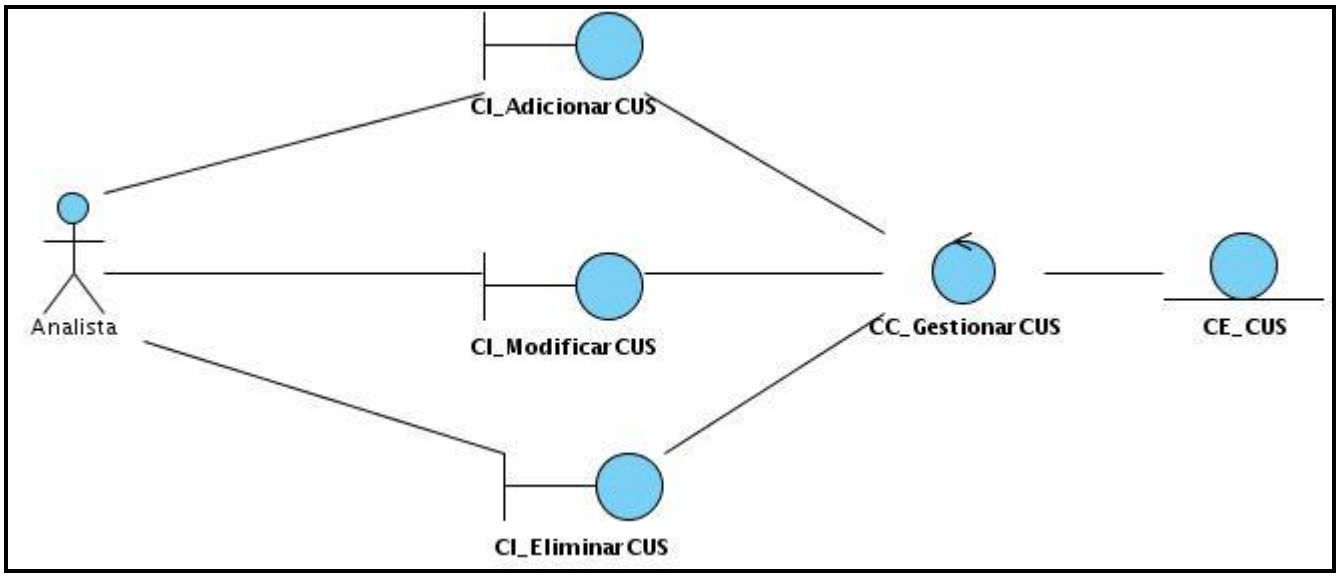


Figura 9: DCA Gestionar Caso de Uso del Sistema.

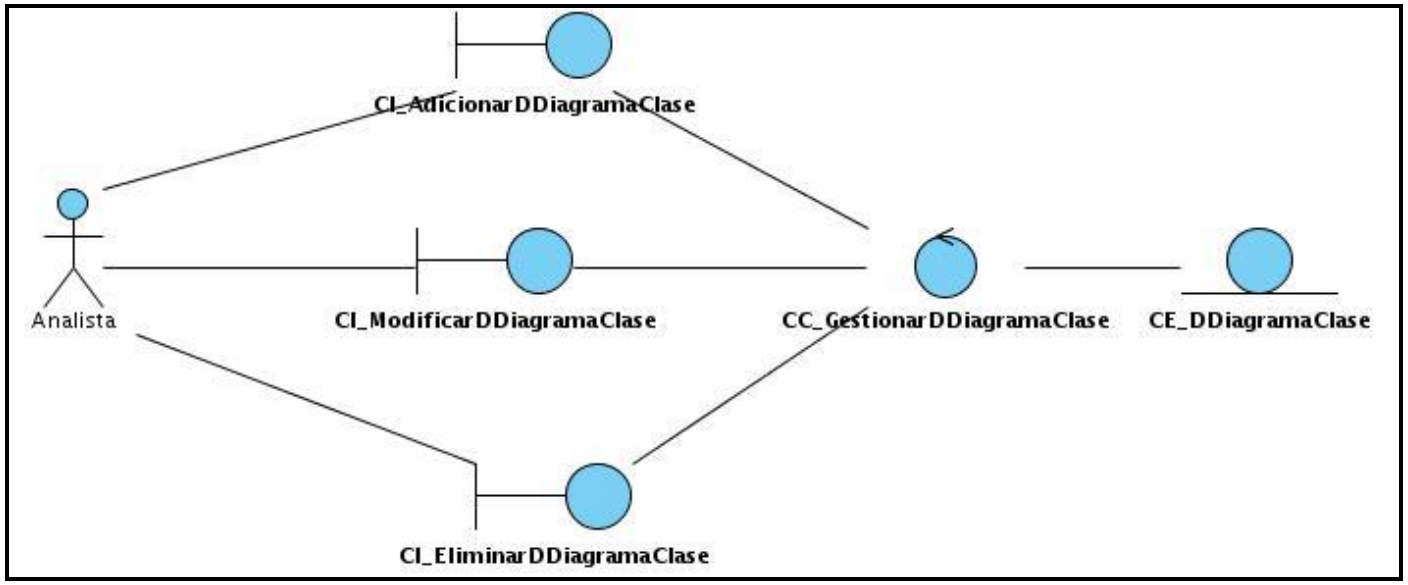


Figura 10: DCA Gestionar Diagrama de Clase del Diseño.

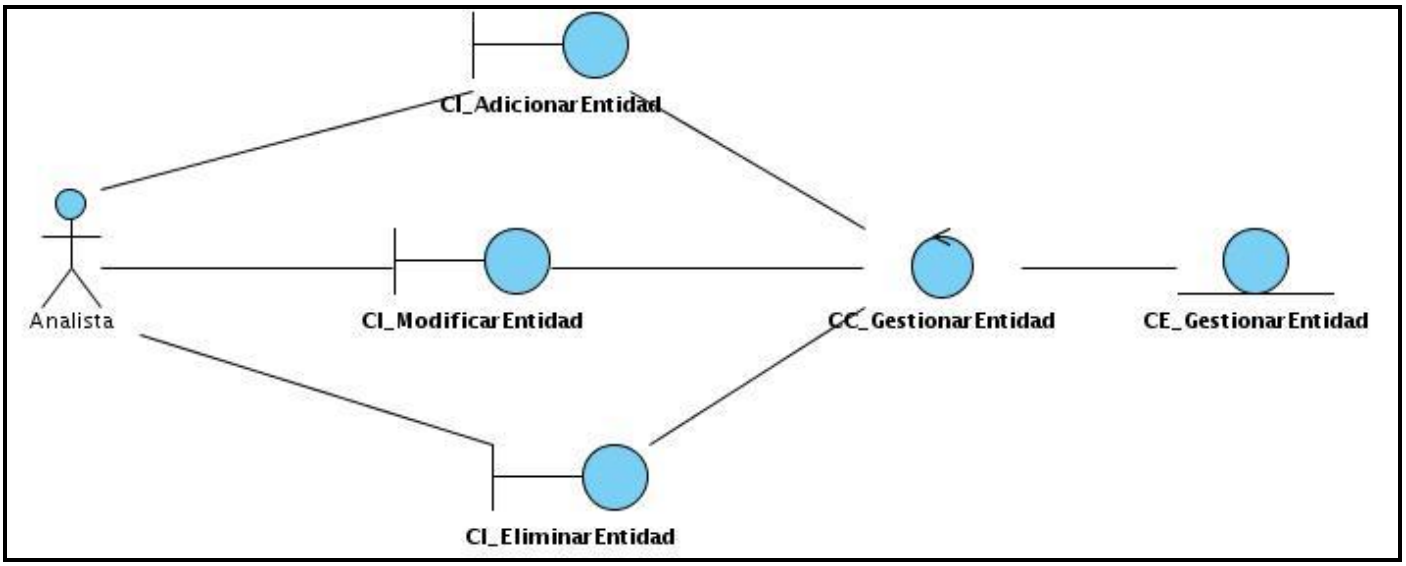


Figura 11: DCA Gestionar Entidades de la Base de Dato.

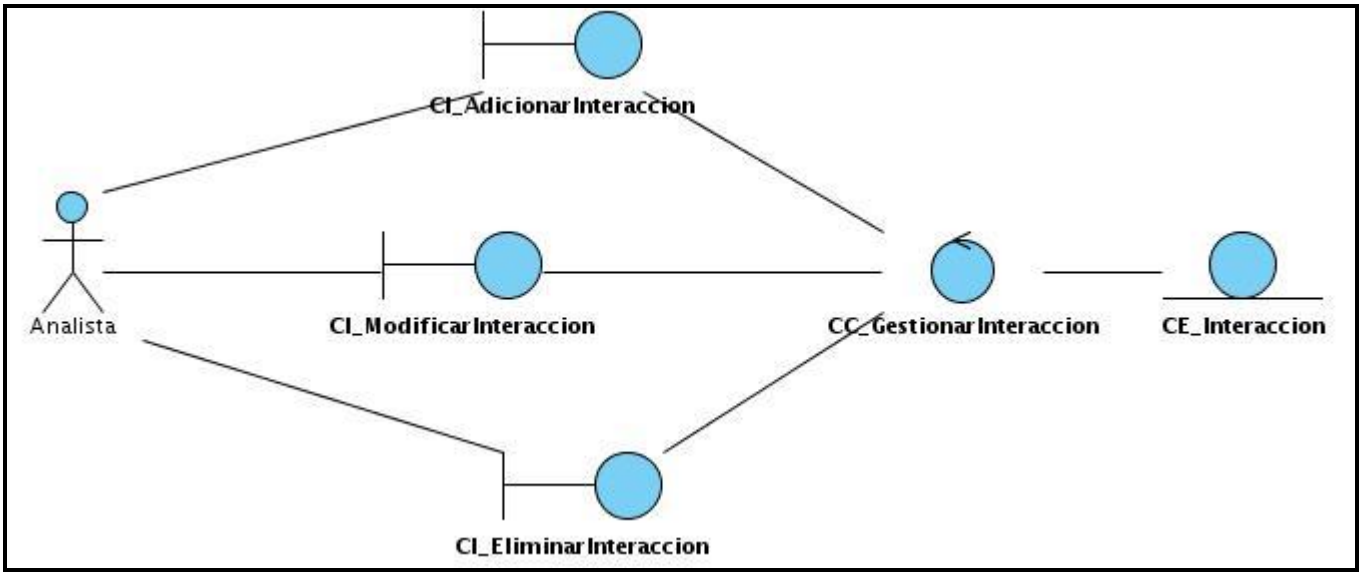


Figura 12: DCA Gestionar Diagrama de Interacción.

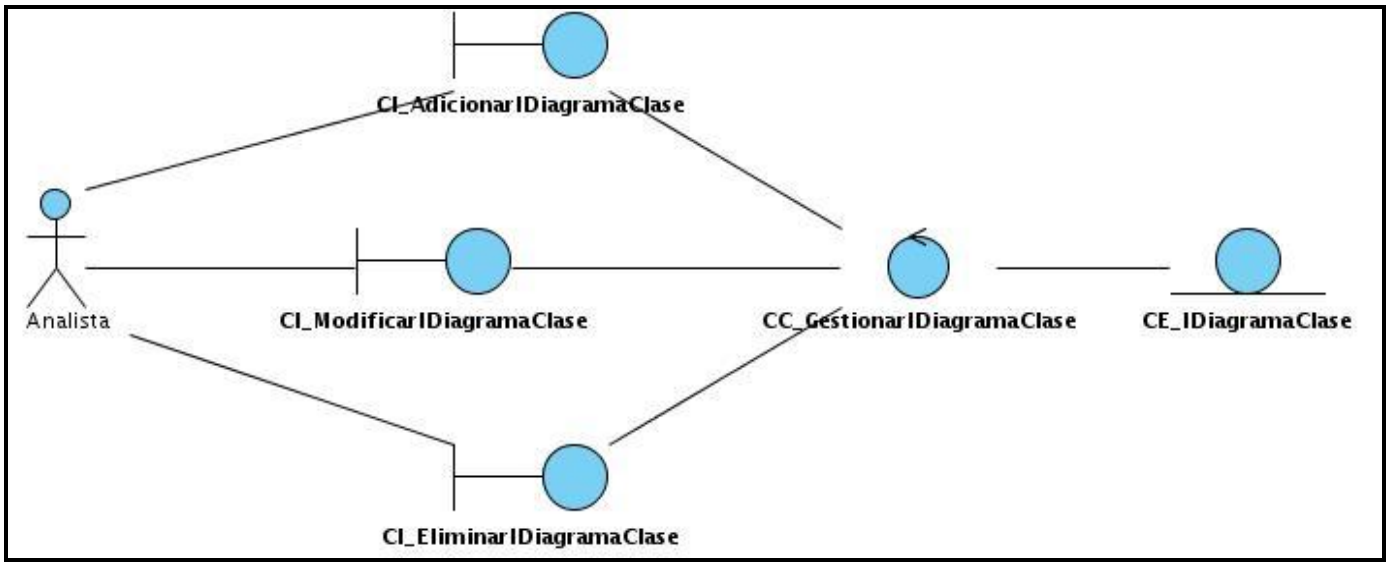


Figura 13: DCA Gestionar Diagrama de Clase de Implementación.

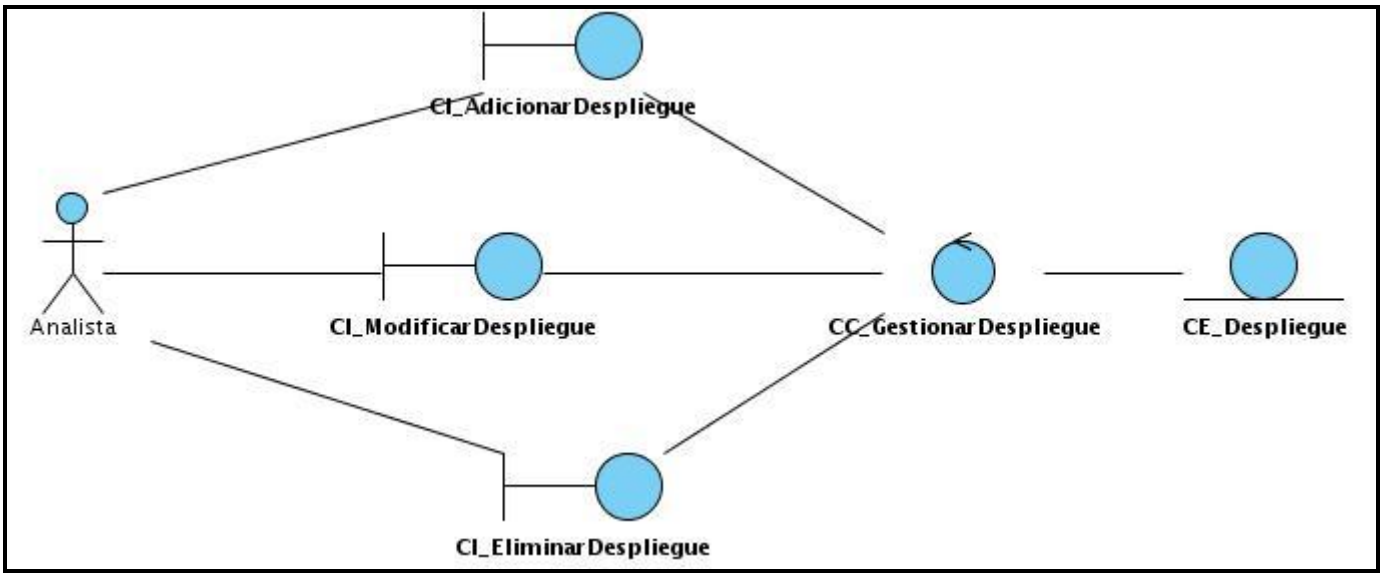


Figura 14: DCA Gestionar Diagrama de Despliegue.

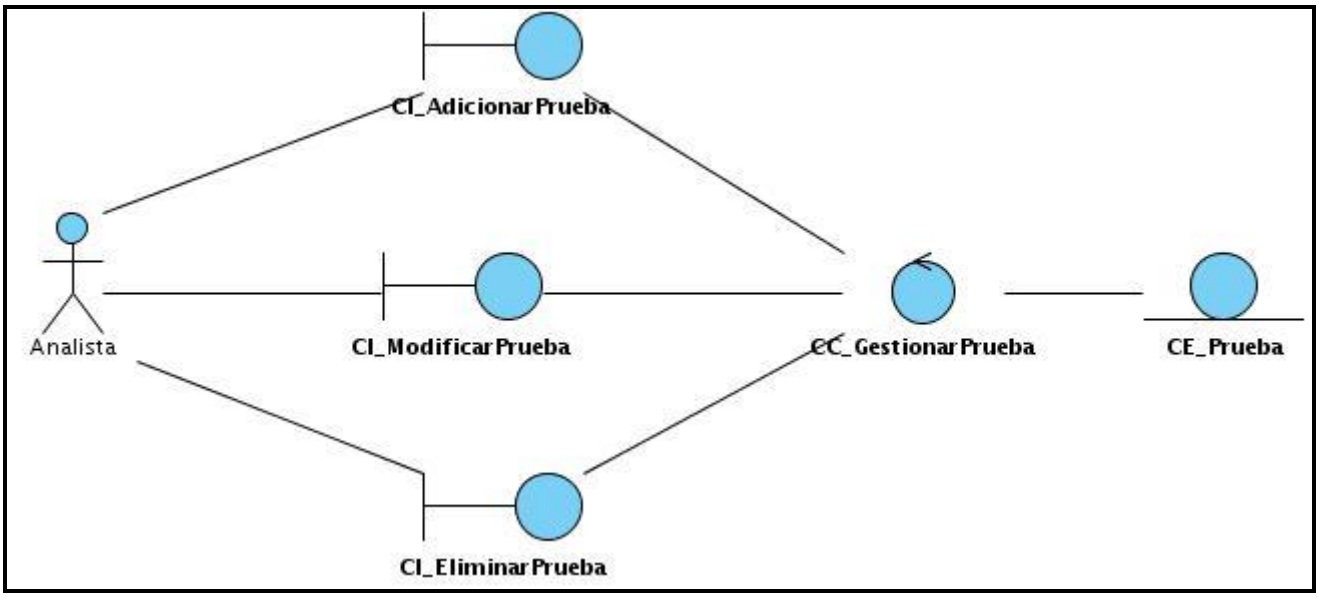


Figura 15: DCA Gestionar Prueba.

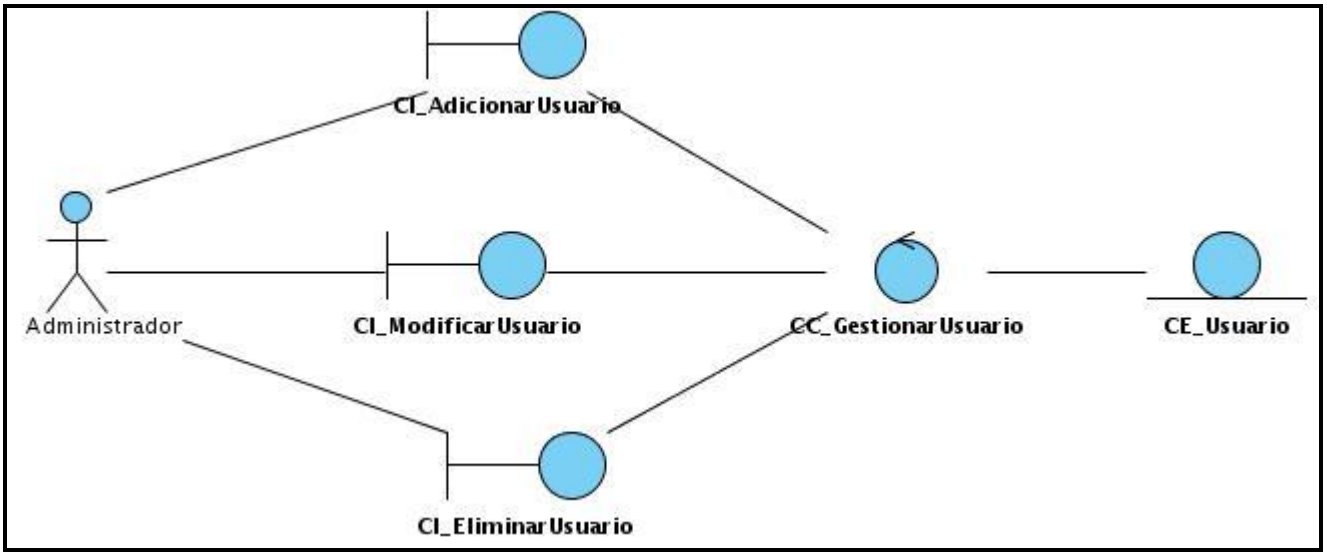


Figura 16: DCA Gestionar Usuario.

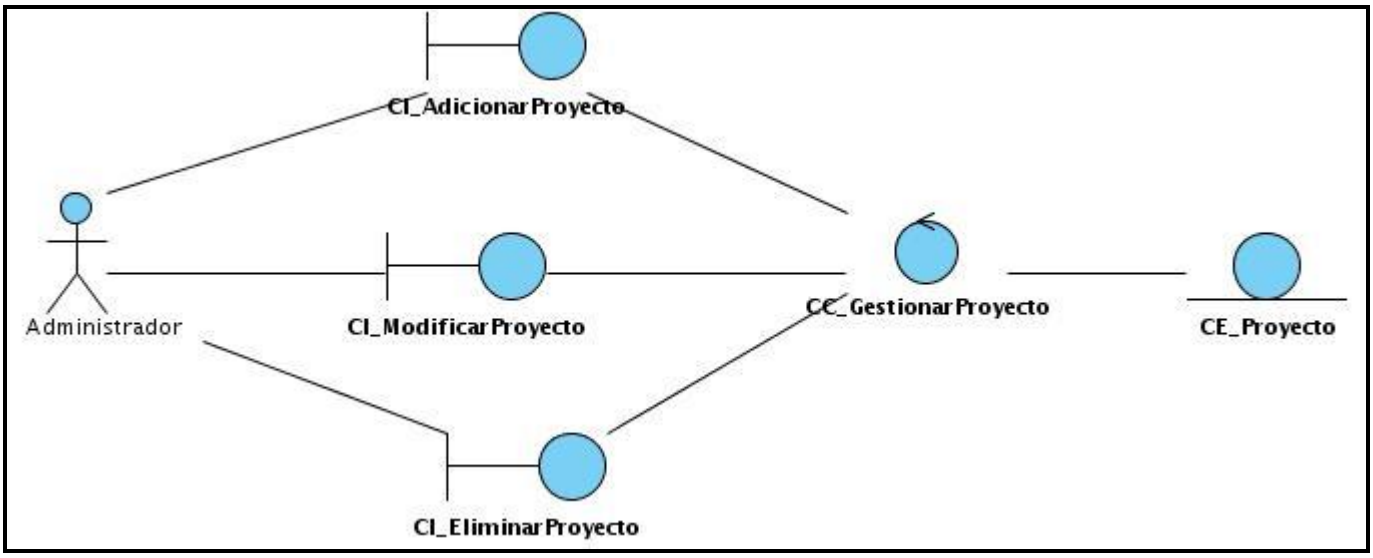


Figura 17: DCA Gestionar Proyecto.

## 3.3 Modelo de Diseño.

En el diseño se modela el sistema y se encuentra su forma (incluida la arquitectura) para que soporte todos los requerimientos, incluyendo los no funcionales y sus restricciones. Una entrada esencial en el diseño es el resultado del análisis, que proporciona una comprensión detallada de los requisitos.

### 3.3.1 Diagrama de Secuencia

Los diagramas de secuencia son importantes en el diseño de cualquier sistema, estos muestran gráficamente las interacciones del actor y de las operaciones a que dan origen. Siguiendo esta idea, se realizaron los diagramas de secuencia del sistema, a continuación se mostrará de el diagrama secuencia correspondiente al de adicionar un proyecto, los demás diagramas de secuencia se encuentran en los anexos. (Ver **Anexo III:** Diagramas de secuencia)

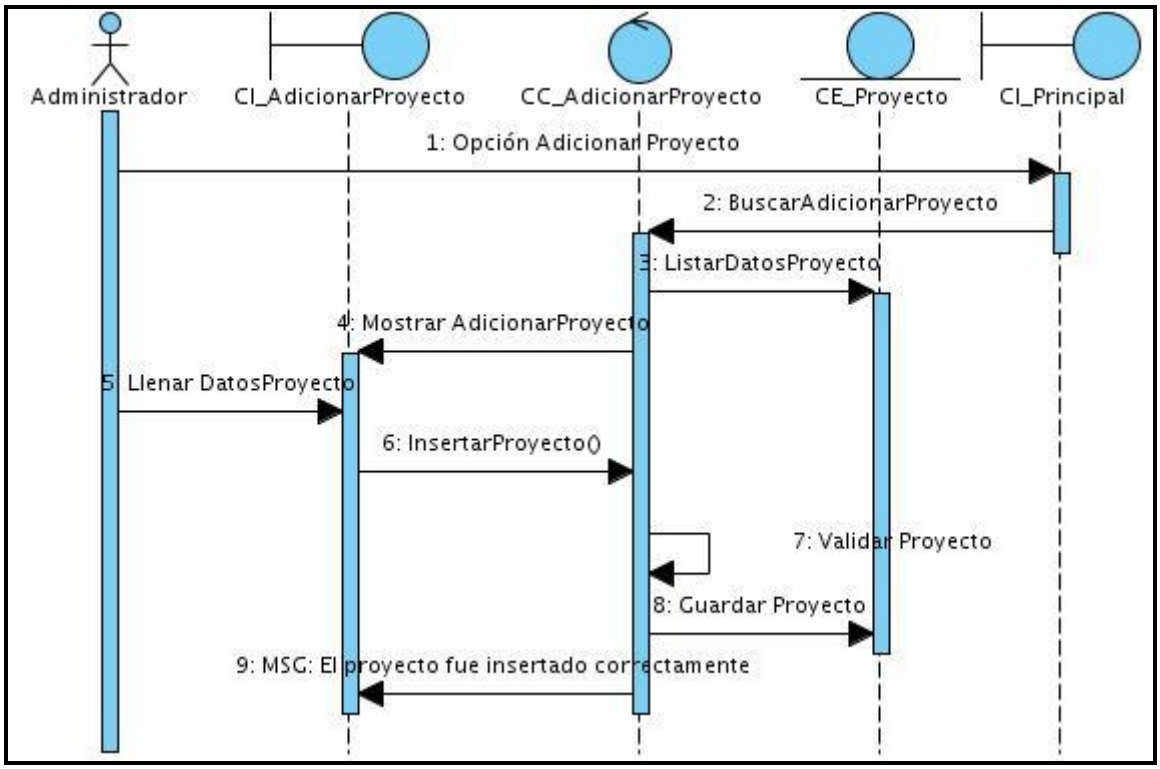


Figura 18: Diagrama de secuencia “Adicionar Proyecto”.

### 3.3.2 Diagrama de Clases Web.

El Diagrama de Clases Web (DCW) es donde se muestran las relaciones entre las clases del modelo, las controladoras y las vistas, así como sus atributos y operaciones y las relaciones entre las mismas. Para el presente trabajo solo se mostrará en DCW “Autenticar”, los demás diagramas de estereotipos web están en los anexos. (**Ver Anexo IV:** Diagramas de Clase Web).

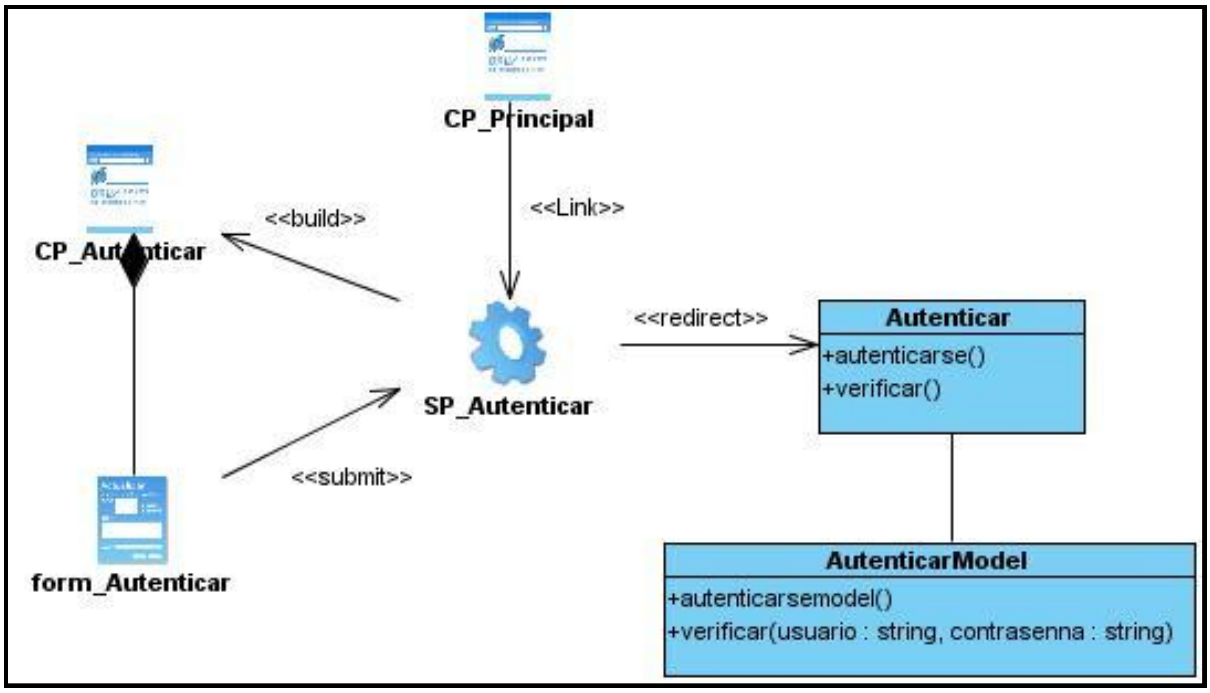


Figura 19: DCW “Autenticar”.

### 3.3.3 Diseño de la Base de Datos.

En la base de datos es donde se almacenarán todos los datos con los cuales trabaja el sistema. En el diseño de la base de datos se deben tener en cuenta las tablas, índices, restricciones, y otro elemento que se necesite para almacenar, recuperar y eliminar objetos persistentes, siendo de gran importancia elaborar el diagrama de entidad-relación, el cual muestra las relaciones que existen entre las tablas de la base de datos. Además, se realizó una descripción de las tablas de la base de datos para su mejor entendimiento. (**Ver Anexo V:** Descripción de las Tablas de la Base de Datos).

## CAPÍTULO 3: ANÁLISIS Y DISEÑO

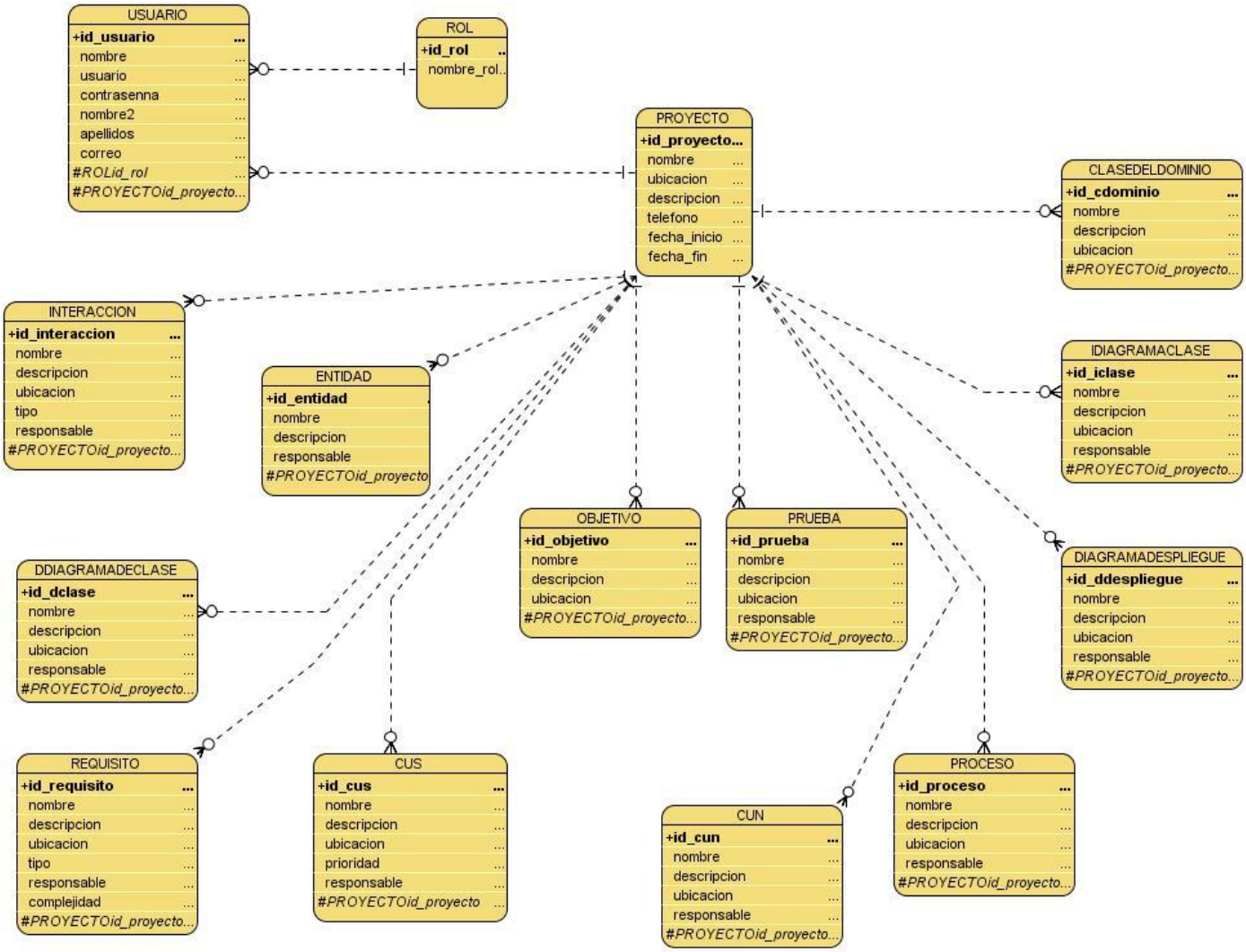


Figura 20: Diagrama Entidad-Relación de la Base de Datos.

## 3.4 Conclusiones Parciales

En el presente capítulo se desarrolló la modelación del sistema, realizándose los diagramas de clases del análisis y el diseño, así como el diseño de la base de datos. Además, se describieron detalladamente las clases del diseño y las tablas de la base de datos, así como los aspectos de seguridad, interfaz, patrones de diseño, definiciones de diseño. Este capítulo es de suma importancia para el desarrollo del sistema, los artefactos generados constituyen la entrada para el flujo de trabajo de implementación, debido a que brinda una vista de la arquitectura del sistema, y describe detalladamente las acciones a tomar en el siguiente flujo de trabajo.

# CAPÍTULO 4: IMPLEMENTACIÓN

## 4.1 INTRODUCCIÓN

El capítulo actual parte del resultado obtenido en el flujo de trabajo Análisis y Diseño servirá de entrada para la implementación de la herramienta en términos de componentes. En este capítulo se muestra el diagrama de despliegue, donde se representan los nodos de la herramienta, basado en las características y funcionalidades que debe cumplir. En el modelo de implementación también se ilustran los diagramas de componentes que contiene la herramienta.

## 4.2 Diagrama de Despliegue

El modelo de despliegue describe como una aplicación se despliega a través de una infraestructura. Muestra la configuración de los tipos de nodo de la herramienta, en los cuales se realizará el despliegue de los componentes. En el mismo se muestra una computadora (PC) cliente, mediante la cual el usuario accederá a dicha herramienta. Una PC servidor Web donde estará emplazada la herramienta. Una PC Servidor de Bases de Datos, donde estará la base de datos a consultar por la aplicación. La aplicación además tendrá que hacer uso del servidor de dominio UCI para la autenticación de sus usuarios. Para un mejor entendimiento ver la Figura 21: Diagrama de Despliegue.

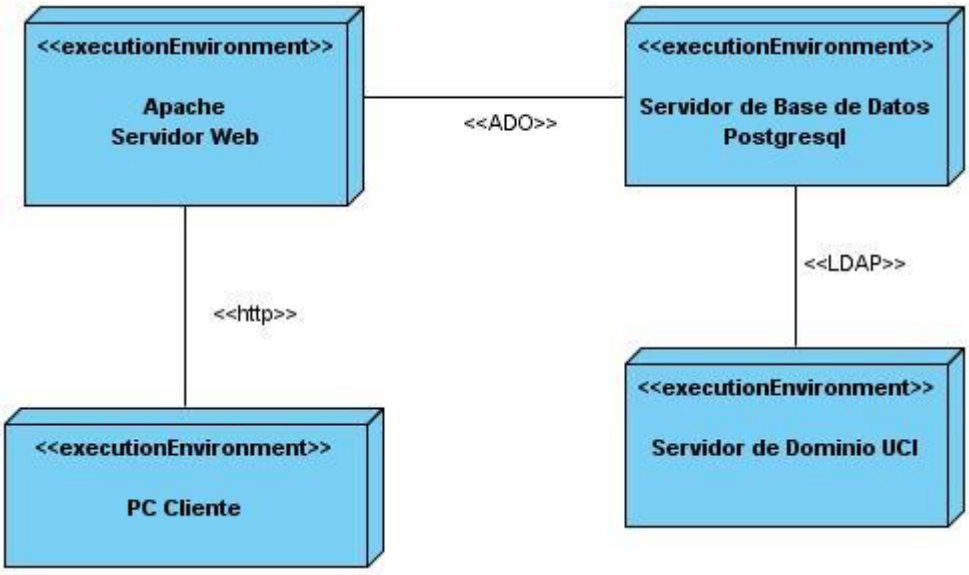


Figura 21: Diagrama de Despliegue.

## 4.3 Diagrama de Implementación

Los Diagramas de Implementación se usan para modelar la configuración de los elementos de procesado en tiempo de ejecución y de los componentes, procesos y objetos de software que viven en ellos. En el diagrama despliegue, empiezas modelando nodos físicos y las asociaciones de comunicación que existen entre ellos. Para cada nodo, puedes indicar las instancias de componentes que se ejecutan en el nodo. También puedes modelar los objetos que contiene el componente. Los Diagramas de Implementación se usan para modelar sólo componentes que existen como entidades en tiempo de ejecución; no se usan para modelar componentes solo en tiempo de compilación.

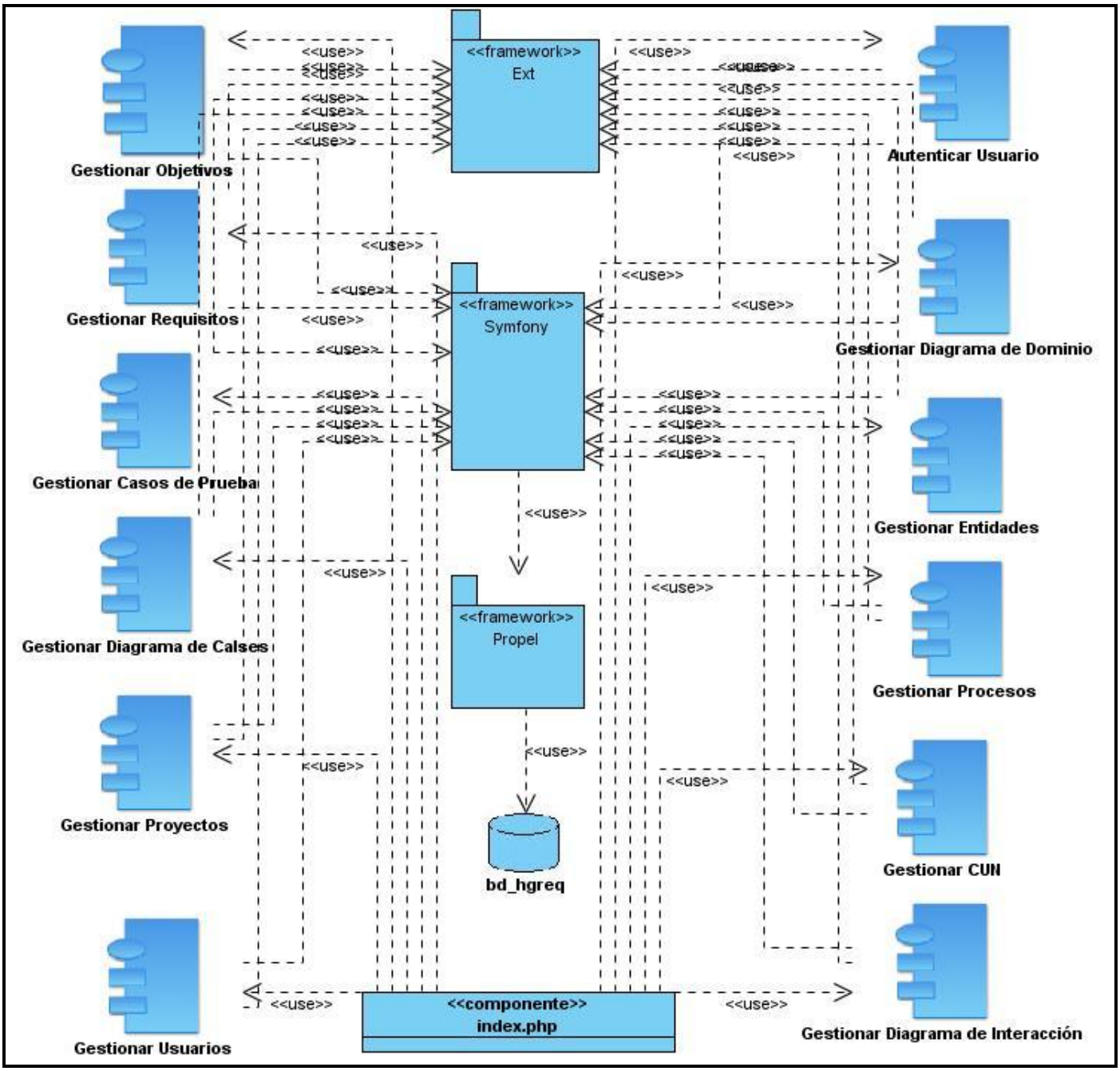


Figura 22: Diagramas de Implementación.

## 4.4 Diagramas de Componentes.

El diagrama de componentes ilustra los componentes del software que serán usados para construir el sistema. Muestra la relación entre los componentes del software, sus dependencias, comunicaciones, localización y otras condiciones. Los Diagramas de Componentes son usados para estructurar los componentes en los sistemas del software. Ellos examinan y controlan las dependencias entre componentes o interfaces de los componentes. Un componente representa una parte modular, desplegable y reutilizables de un sistema.

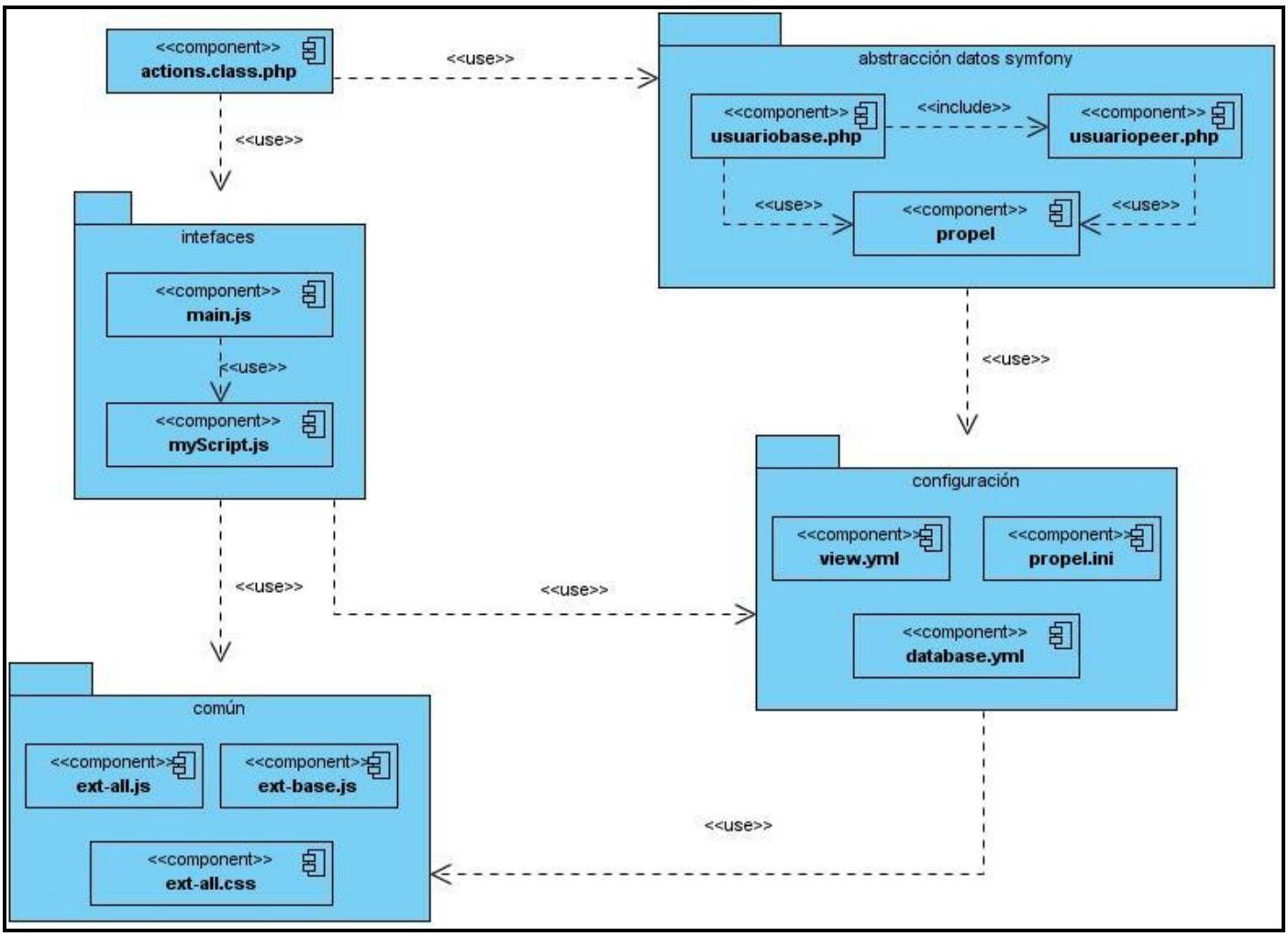


Figura 23: Diagrama de Componente “Inicio”.

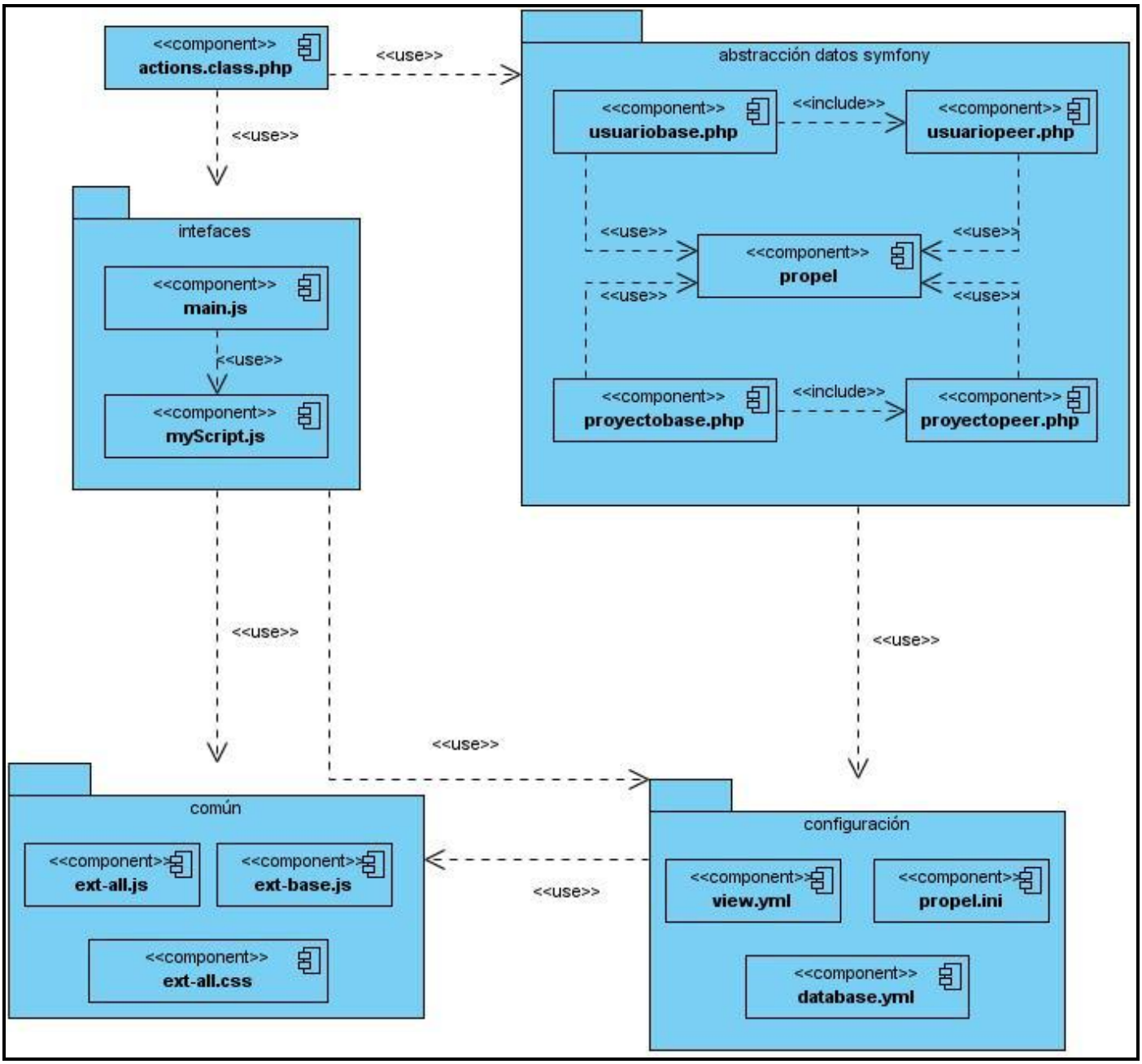


Figura 24: Diagrama de Componente “Administración”.

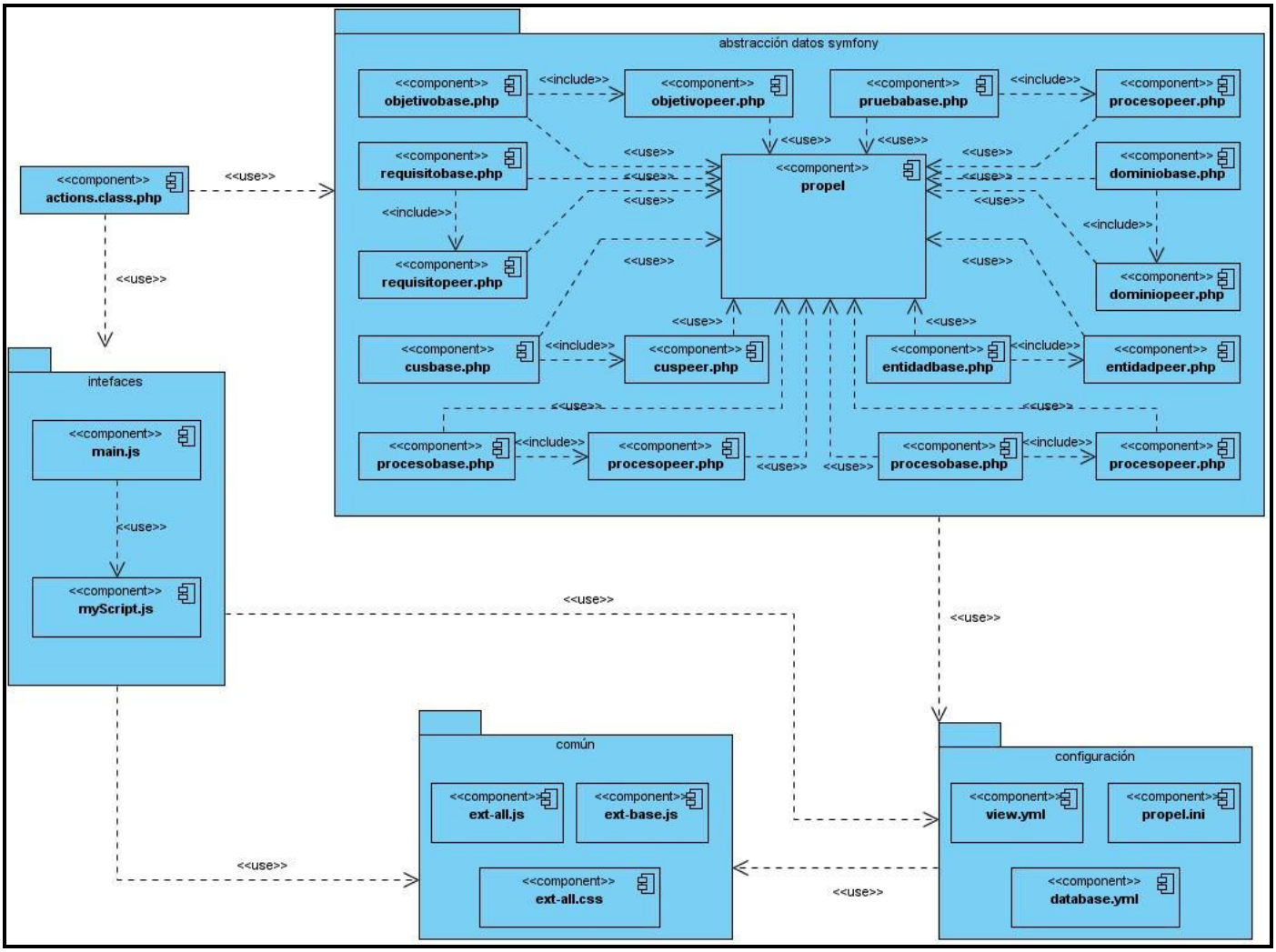


Figura 25: Diagrama de Componente “Analista”.

## 4.5 Conclusiones Parciales

En el presente capítulo se ha mostrado mediante el diagrama de despliegue como quedaría instalada la aplicación, se necesitaría una computadora servidora para la base de datos que estaría sobre PostgreSQL8.4, por otro lado se tendrá un servidor web para la aplicación corriendo con Apache2.0, los servidores tendrían 256 MB de RAM, 10 MB de disco duro y 2.40 Hgz de velocidad; estos son los requisitos mínimos que harán falta junto a una computadora cliente para que funcione la aplicación. Además, se realizó el diagrama de componentes para evidenciar las relaciones entre cada uno y como se divide modularmente la aplicación.

## Conclusiones Generales

Al concluir con el trabajo de diploma podemos decir que se realizó con éxito la herramienta propuesta para la gestión de requisitos en los proyectos productivos de la UCI, teniendo encuentra el alcance inicial del presente trabajo podemos afirmar que se obtuvo una aplicación web capaz de administrar los requisitos de cada proyecto dándole cumplimiento a los objetivos trazados, evidenciándose en los resultados siguientes:

- ✓ Se identificaron los requisitos funcionales y no funcionales que debe cumplir una herramienta para la gestión de los requerimientos de los proyectos productivos de la UCI.
- ✓ Se obtuvo el modelo de análisis y diseño para la aplicación utilizando la metodología de desarrollo RUP.
- ✓ Se implementó prototipo web.

## Recomendaciones

- ✓ Incluir la trazabilidad en forma matricial y en forma de árbol teniendo en cuenta las metodologías de desarrollo.
- ✓ Brindar la posibilidad de que la aplicación pueda adjuntar documentos al servidor.
- ✓ Continuar con el desarrollo de la aplicación.

**Bibliografías**

Apache. [En línea] [Citado el: 14 de febrero de 2010.] http://www.apache.org/.

**Armando Sosa.** CakePHP. [En línea] 2010. [Citado el: 8 de febrero de 2010.] http://cakephp.org/.

**Ivar Jacobson, Grady Booch, James Rumbaugh.** El Proceso Unificado de Desarrollo de Software. s.l. : Addison Wesley, 2000. ISBN: 84-7829-036-2.

KumbiaPHP Framework. [En línea] [Citado el: 13 de febrero de 2010.] http://www.kumbiaphp.com/blog/about/.

Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia. Rational Unified Process (RUP). [Disponible en: https://pid.dsic.upv.es/C1/Material/Documentos%20Disponibles/Introducci%C3%B3n%20a%20RUP.doc].

**larman, Craig.** UML y Patrones: Introducción al análisis y diseño orientado a objeto. Mexico : Universitarios, 1999. ISBN: 970-17-0261-1.

Oracle 11g. [En línea] [Citado el: 13 de febrero de 2010.] http://www.oracle.com/index.html.
Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia. Rational

Unified Process (RUP). [Disponible en: https://pid.dsic.upv.es/C1/Material/Documentos%20Disponibles/Introducci%C3%B3n%20a%20RUP.doc].

**RAE**. Diccionario de la lengua española. [En línea] 2009. [Citado el: 114 de febrero de 2010.] http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=requisito.

**Rolando Alfredo Hernández León, Sayda Coello González.** EL PARADIGMA CUANTITATIVO DE LA INVESTIGACIÓN CIENTIFICA. Ciudad de la Habana : EDUNIV, 2002. ISBN: 959-16-0343-6.

**Sharp, Remy.** jQuery for Designers. 2009. ISBN: 9781935182221.

**Shea Frederick, Colin Ramsay, Steve 'Cutter' Blades.** Learning Ext JS. s.l. : PACKT, 2008. ISBN: 978-1-847195-14-2.

Symfony. [En línea] [Citado el: 13 de febrero de 2010.] http://www.symfony.es/.

**Visual Paradigm.** Visual Paradigm for UML - UML tool for software application development. [En línea] [Citado el: 9 de febrero de 2010.] http://www.visual-paradigm.com/product/vpuml/.

Zend Framework. [En línea] [Citado el: 13 de febrero de 2010.] http://framework.zend.com/.

**PRESSMAN, R. S.** "Ingeniería del Software. Un Enfoque Práctico". 1998. p.

**PRESSMAN, R. S.** "Ingeniería del Software. Un Enfoque Práctico". 2000. p.

**Bañeres, Juan Palacio.** Compendio de Ingeniería de Software I. 2006.

**Bañeres, Juan Palacio.** Compendio de Ingeniería de Software II. 2006.

Symfony1.2, la guía definitiva. Libros Web. [En línea] [Citado el: 12 de 5 de 2010.] http://www.librosweb.es/symfony_1_2/.

**Potencier, François Zaninotto y Fabien**. Symfony 1.2, la guía definitiva. s.l. : Apress , 2008. ISBN-13: 978-1590597866.

**Costilla, Carmen.** Características Objeto-Relacionales del Sistema de Gestión de Bases de Datos Oracle. [En línea] 2002.