Universidad de las Ciencias Informáticas

Facultad 4.

Título: Sistema para la Gestión de Recursos Informáticos en la Universidad de las Ciencias Informáticas.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor: Daineris Dinza Garrido

Tutores: Ing. Yulio Seriocha García Gallardo.

Ing. Robert Alberto Peña Yantá.

La Habana, Cuba

Junio de 2009

.

DATOS DE CONTACTO

Ing. Yulio Seriocha García Gallardo

Email: ygarciag@uci.cu

Graduado en el 2007 de la Universidad de la Ciencias Informáticas (UCI).

Ing. Robert Alberto Peña Yantá.

Email: rpenay@uci.cu

Graduado en el 2007 de la Universidad de la Ciencias Informáticas (UCI).

PENSAMIENTOS

Nuestra Recompensa se encuentra en el esfuerzo y no en el resultado, un esfuerzo total es una victoria completa."

Mohandas Gandhi.

El futuro tiene muchos nombres. Para los débiles es lo inalcanzable. Para los temerosos, lo desconocido. Para los valientes es la oportunidad.

Víctor Hugo

AGRADECIMIENTOS

A Dios por ponerme en este camino, por oír siempre mis súplicas, protegerme y ayudarme.

A toda mi familia porque son mi razón de ser. En especial a mis abuelos Reyna y Jesús por guiarme siempre por el buen camino, por mostrarme lo maravillosa que es la vida y enseñarme a ver luz en la oscuridad.

A mis padres por darme la vida, por confiar en mí y ser mis guías en todo momento.

A mis hermanas por existir, por ser lo más lindo que tengo en mi vida.

A mis primas por ser como hermanas para mí.

A mis tíos y tías por acompañarme siempre en los malos y buenos momentos.

A Yulio por su incondicional ayuda, por haber dedicado gran parte de su tiempo a orientarme y apoyarme.

A Daylen por su amistad , por estar conmigo cuando la necesito..

A Mary, Iri, Mailis, Dey, Maggy, Yilian, Liudmila por ser mi segunda familia.

A Hino, Joek, Jorge, Edwing, Yankiel, Lilian, Yile por ser amigos incondicionales.

A todos mis compañeros, que han transitado conmigo a lo largo de estos años, los que llegaron y los que se quedaron en el camino.

A mis vecinos y amigos del barrio por siempre estar pendiente de mí.

A todos mis profesores, que han sido pilares fundamentales en mi formación durante toda mi vida de estudiante, les agradezco en gran medida ser quien soy

A Fidel y a la Revolución por darme la oportunidad de convertirme en una profesional

En fin a todos los que de una forma u otra han hecho posible que lo que un día fue una utopía en mi vida, hoy se convierta en realidad. A todos gracias….

DEDICATORIA

Con todo el amor del mundo dedico este trabajo a mi abuela Reyna por darme la fuerza para llegar a este momento, por ser una madre para mí, por enseñarme de la vida todo lo que sé, porque siempre ha estado ahí cuando la necesito porque de todas las cosas que tengo en el mundo, tu amor es lo mas grande, por eso este trabajo es también tuyo

RESUMEN

Entre los procesos que se llevan a cabo en la Infraestructura Productiva de la Universidad de las Ciencias Informáticas se encuentra el Proceso de Gestión de Recursos Informáticos, que tiene como objetivo inventariar los recursos informáticos con los que se cuenta para la producción. Actualmente este proceso se realiza de forma manual y es considerado una tarea complicada debido a la gran cantidad de información que se maneja y la importancia de esta.

Este trabajo consiste en la creación de un sistema informatizado para la gestión de los recursos informáticos de la UCI, en respuesta a la necesidad de mejorar y perfeccionar la manera en que actualmente se lleva a cabo esta tarea. La utilización de un sistema informatizado para la gestión de Recursos de los proyectos productivos significaría una mejora considerable de este proceso en cuanto a eficiencia, tiempo y organización, contribuyendo favorablemente al desarrollo de software en la universidad.

En este documento se presenta la fundamentación teórica que sustenta la realización de este trabajo. Se valora el estado del arte de los sistemas de gestión, así como el estudio de algunas de las tecnologías, herramientas y lenguajes de programación más utilizados en la actualidad para el desarrollo de aplicaciones web. Se analiza el negocio, el sistema, se desarrolla un modelo del diseño y se obtiene la aplicación final. Para conocer la factibilidad de la aplicación se realiza un estudio basado en actores y casos de uso que arroja beneficios tangibles e intangibles.

PALABRAS CLAVES: Recursos Informáticos, Solución Informática, Proyecto Productivo, Gestión, Sistema

.

Índice de contenido

Índice de figuras

Índice de tablas

INTRODUCCIÓN

Actualmente la Industria de Desarrollo del Software es una de las más importantes que existe en el mundo, esto se debe a que se ha convertido en una de las más rentables. El desarrollo se debe en gran medida al hecho de que la informatización se ha convertido en una necesidad para el progreso de la humanidad, ayudando a elevar la productividad y efectividad en cualquier rama que se aplica.

La Industria Cubana del Software esta encaminada a ser uno de los eslabones principales en la economía del país, por esta razón es por lo que se han trazado varias estrategias con el objetivo de lograr la inserción del software cubano en el mercado mundial (Castro, 2006).

La Universidad de las Ciencias Informáticas no se exonera de esta situación, pues fue creada como una de las estrategias del país para la producción del software, esta universidad única en su tipo, combina la docencia con la producción y tiene como principal fin formar especialistas de alto nivel profesional, además de crear productos, servicios informáticos y soluciones tecnológicas integrales tanto para la informatización nacional como para la exportación, todo esto mediante un grupo de proyectos productivos. La universidad pese a que no tiene mucho tiempo de creada y que por la tanto no tiene demasiada experiencia en la rama, ha tenido excelentes resultados en la exportación, con productos de alta calidad.

Para alcanzar estos resultados satisfactorios y mantener el prestigio con el que cuenta ya la Universidad en la producción de software, es necesario garantizar el correcto funcionamiento de los proyectos, sin permitir que estos se detengan durante su desarrollo y se retrasen sus cronogramas de entrega.

De garantizar el funcionamiento adecuado y el cumplimiento de los proyectos productivos se encargan: la Vicerrectoría de Inversiones y Tecnología, la Dirección General de la Infraestructura Productiva (IP), la Dirección Técnica de la IP, así como el jefe y especialistas que integran el Grupo de Soporte al Desarrollo de la IP. Estos junto a las Facultades tienen que ser capaces de detectar las deficiencias que existan en los proyectos productivos, que entorpezcan el adecuado desarrollo de los mismos.

En ocasiones estas deficiencias están dadas porque no se conoce la existencia real de recursos en los proyectos productivos, al no tener esta información fiable se posibilita la situación de que se tengan más o menos recursos para la producción, que los que realmente necesita el proyecto productivo. Esto se debe a que la mayoría de los recursos informáticos con los que cuentan los proyectos, pese a que están inventariados, no se encuentran en un registro en el cual esté especificado la cantidad real de los mismos

y si se encuentran o no en funcionamiento. Además de que se hace bastante engorroso el proceso de asignación de maquinas nuevas a un proyecto, debido a que todos estos procesos se realizan de forma manual.

Actualmente la Infraestructura Productiva de la Universidad de las Ciencias Informáticas no posee un sistema informatizado, que permita llevar el control y seguimiento de forma efectiva y ágil de los recursos disponibles para la producción, lo que provoca que en ocasiones puedan alterarse las planillas y no estar en ellas la situación real de los recursos disponibles en los proyectos.

Las razones expuestas anteriormente dan lugar a que el ***problema científico*** a solucionar sea: ¿Cómo lograr una solución informática para la gestión de Recursos que garantice el ágil desarrollo de este proceso en la Universidad?

Donde se tiene como ***objeto de estudio*** el Proceso de Gestión de Recursos y el ***campo de acción*** se enmarca en la informatización del Proceso de Gestión de Recursos Informáticos en la Universidad de Ciencias Informáticas.

Para dar respuesta al problema planteado se define como ***objetivo general:*** Desarrollar un sistema que permita la informatización de los procesos que se llevan a cabo en la Gestión de Recursos en la Universidad.

Las tareas de la investigación trazadas para dar cumplimiento a este objetivo son:

- ✓ Estudio del estado del arte sobre las herramientas y tecnologías actuales para el desarrollo de software.
- ✓ Estudio y análisis de los procesos que se llevan a cabo en la Gestión de Recursos de la Universidad.
- ✓ Confección del diseño de la solución para los procesos de la Gestión de Recursos de la universidad, así como implementación de la solución diseñada.
- ✓ Estudio y análisis de la factibilidad de la solución obtenida.

Inicialmente se parte de la siguiente ***hipótesis:*** Si se desarrolla una aplicación que informatice los procesos de gestión de recursos informáticos de la UCI entonces se logrará una mayor gestión, control y eficiencia de los recursos informáticos disponibles para la producción.

El presente trabajo está estructurado por cuatro capítulos, tal y como se describe a continuación:

Capítulo 1. Fundamentación Teórica: Se analizan algunas soluciones de software existentes y se hace referencia a la metodología de desarrollo de software y las herramientas seleccionadas para realizar el trabajo.

Capítulo 2. Características del sistema: Se realiza la modelación del negocio, se describe formalmente el proceso de gestión de recursos, especificando para cada actividad el responsable y las entidades del negocio que se utilizan y generan. Se enumeran las funcionalidades y los atributos del sistema. Por último se definen los casos de uso del sistema.

Capítulo 3. Construcción de la solución propuesta: Se presenta el modelo de análisis, que incluye los diagramas de clases del análisis, los diagramas de colaboración para cada caso de uso y diagrama de clases de diseño web por CU. Por último, presenta el modelo de diseño de la base de datos con el diagrama entidad de la BD y los diagramas de despliegue e implementación, concluyendo la programación del sistema.

Capítulo 4. Estudio de Factibilidad: Contiene el estudio de factibilidad realizado para el sistema, se enumeran los beneficios y se analizan los costos que representa la elaboración de la propuesta de solución.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA DEL TEMA.

1.1 Introducción

En este capítulo se abordan características de soluciones informáticas que gestionan recursos, así como aspectos principales de cómo gestionará recursos la solución informática. Además se analizan las herramientas, lenguajes de modelado, metodologías de desarrollo de software y se justifican las usadas para el cumplimiento de las tareas trazadas.

En la actualidad las tecnologías de la información juegan un papel importante en las estrategias de negocios, ya que están cambiando la forma en que las empresas realizan sus procesos. Los sistemas de información permiten a las compañías lograr ventajas competitivas de diferentes maneras: coordinando actividades de valor en localidades que se encuentran en una amplia geografía, o también mediante la creación de nuevas interrelaciones entre los negocios, ampliando el alcance de las industrias. Asimismo le sirve a las empresas para soportar sus estrategias competitivas, ya sea para ir un paso delante de la competencia o reducir las ventajas que la misma pueda presentar. Desde hace ya varios años, se ha dado mayor importancia a las tecnologías de la información y su alineación con las estrategias del negocio para mejorar sus procesos claves del negocio. Hoy más que nunca las empresas requieren de herramientas que les proporcionen control y centralización de su información, esto con el fin de tomar las mejores decisiones para sus procesos y estrategias de negocios.

Un ERP (Planificador de Recursos Empresariales) es un sistema de planificación de recursos empresariales que integra e informatiza la mayoría de los procesos que están asociados con las operaciones de producción y los aspectos de distribución de una empresa. Los ERP cubren una serie de funciones tales como: contabilidad, finanzas, logística, producción y recursos humanos. Entre las principales características que distinguen a un **software ERP** de cualquier otro software de gestión, es que deben de ser sistemas integrales, con modularidad y adaptables.

Integrales: porque permiten controlar los diferentes procesos de la compañía entendiendo que todos los departamentos de una empresa se relacionan entre sí, es decir, que el resultado de un proceso es punto de inicio del siguiente.

Modulares: ya que los **software ERP** entienden que una empresa es un conjunto de departamentos que se encuentran interrelacionados por la información que comparten y que se genera a partir de sus procesos. Las funcionalidades se encuentran divididas en módulos, los cuales pueden instalarse de acuerdo con los requerimientos del cliente.

Los principales objetivos de un ERP son:

- Optimización de los procesos empresariales.
- Acceso a toda la información de forma confiable, precisa y oportuna.
- Sus componentes interactúan entre si, fortaleciendo y facilitando sus operaciones.
- Los datos deben ser consistentes, completos y lo más común posible.
- Las empresas deben modificar algunos de sus procesos para alinearlos con los del ERP.
- Son sistemas modulares.
- Mayormente hay un software para cada funcionalidad.

Como todo sistema tiene sus ventajas y desventajas. Dentro de los beneficios de los ERP se pueden mencionar que:

Se utiliza solo un sistema para manejar la mayoría de los procesos comerciales.

- Hay una integración entre las funciones de las aplicaciones.
- Reduce los costos de la gerencia.
- Incrementa el retorno de inversión.
- Son una fuente de infraestructura abierta.

Como desventajas se tienen que:

- Son muy caros.
- Requieren cambios en la compañía y procesos en la instalación.
- Son complejos y muchas compañías no se pueden ajustar a ellos.
- Hay pocos expertos en ERP.

- Su implementación es larga cara y difícil, muchas veces más cara que adquirir la licencia.
- Dependencia de un solo proveedor.

Por lo anteriormente explicado se decide desarrollar un sistema para la Gestión de Recursos Informáticos en la Universidad de las Ciencias Informáticas.

1.2 Soluciones informáticas existentes vinculadas al campo de acción

1.2.1 Soluciones Informáticas en el Mundo

Los sistemas ERP en el mundo contemporáneo tienen una alta popularidad y gran demanda por lo que se han creado varios de ellos, unos por las grandes empresas de software como la Oracle, Microsoft, otros por empresas de no tan alto nivel pero sin dudas son sistemas sumamente caros. A continuación se presentan varios ejemplos:

SAP ERP es un sistema de gestión que permite analizar el negocio, optimizar las finanzas, gestionar los recursos humanos, operaciones y servicios corporativos. Además también proporciona soporte para cuestiones referentes a la gestión de sistemas de administración de usuarios, gestión de configuración, gestión de datos centralizada y gestión de servicios de Web. El paquete de Enterprise Resource Planning integra todas las anteriores funcionalidades dentro de la plataforma SAP NetWeaver, lo que permite a los usuarios medir y gestionar recursos empresariales en tiempo real.

Una gestión eficiente de los recursos de producción y operacionales, puede también tener un coste asociado, especialmente si los procesos de producción son complejos y requieren de una constante monitorización por parte de los empleados. No obstante, SAP se encarga de gestionar y planificar la utilización de los recursos mediante sistemas de gestión Enterprise Resource Planning (ERP). Las aplicaciones ERP y sistemas de gestión de recursos permiten un completo control de la utilización de todos los recursos disponibles en una empresa y además permiten cuantificar las capacidades y planear respectivamente la expansión de las bases utilizando estimados fiables y realistas.

Es un sistema flexible, amigable, con ayuda en línea que puede ser instalado en una microcomputadora o sobre varias, funcionando en ambiente multiusuario incluidas estaciones remotas. Asimismo, proporciona opciones de seguridad que le permiten limitar el acceso a los diferentes procesos del sistema de acuerdo con el perfil de cada usuario. Está diseñado para Multi-Compañía, con una estructura organizativa a varios

niveles, en la que podrán existir: Grupo Corporativo, Corporativo, Grupo de Agrupaciones, Agrupación, Almacenes y Centros de Costos. Para entidades con esta estructura se brinda un Módulo de Comunicaciones que facilita poder intercambiar información entre ellas.

Este sistema informático cuenta con un modulo para la gestión de inventarios el cual posibilita registrar y realizar un seguimiento de los materiales, según la cantidad y el valor. Puede reducir los costes de almacenamiento, transporte, cumplimiento de pedidos y manipulación de materiales y, a la vez, mejorar el servicio al cliente. Puede mejorar significativamente la rotación de inventario, optimizar el flujo de mercancías y acortar las rutas en su almacén o centro de distribución.

El ERP español Spyro es un software integral dirigido a la pequeña y mediana empresa española. Su adaptabilidad como herramienta de gestión ha permitido que la solución esté implantada en empresas de diversos sectores, de industria, comerciales y servicios.

Spyro es en la actualidad un moderno, potente y eficaz **E.R.P.** (Enterprise Ressources Planning), sistema para la gestión de los recursos de la empresa, con un interface de usuario muy intuitivo, que abarca todas las áreas de la empresa, (Gestión Económico Financiera, Gestión de Compras, Gestión de Ventas, Gestión de Almacén, Control de Producción, Control de Calidad, Gestión de Recursos Humanos y Organizador Empresarial), y que a la vez gestiona todos y cada uno de los datos de sus departamentos.

Spyro ya ha sido implantado en numerosas empresas con un altísimo grado de satisfacción por parte de quienes han pretendido un producto sólido, de rápida y segura instalación, elaborado con herramientas y tecnologías de punta para entornos abiertos, y totalmente amigable.

Cuenta con servicios de asistencia remota en permanente atención, que proporcionan un eficiente soporte a todos los clientes. Spyro garantiza tiempos y costes mínimos y bajo control.

1.2.2 Solución Informática en Latinoamérica.

El Sistema ISIS ERP Manager:

El Sistema ISIS ERP Manager (sistema de planificación de recursos), producto diseñado, producido, comercializado y distribuido por Quality Soft Argentina s.a., es un sistema de gestión de información que integra y automatiza las prácticas de negocio asociadas con los aspectos comerciales, productivos, financieros, contables, entre otros, de las pequeñas, medianas o grandes empresas. El ISIS ERP, es un

producto integral de gestión global de la empresa. Se caracteriza por estar compuesto por diferentes partes o módulos integrados en una única aplicación. Estos módulos comprenden por ejemplo circuitos productivos, compras, contables, ventas, inventarios, impositivos, etc. Su Arquitectura utiliza Microsoft SQL Server 2000 o superior como base de datos, Business Object Crystal Reports XI es su generador de reportes y está programado en Microsoft Visual Studio.

1.2.3 Solución Informática en Cuba.

Debido a la posible integración de Cuba a la Comunidad Open Source, en el campo de los sistemas ERP se tiene que trabajar pues los sistemas de este tipo que son software libre conocidos internacionalmente no se ajustan a los parámetros y los métodos de trabajo de las entidades cubanas que potencialmente podrían ser sus usuarios, aun cuando dada la facilidad de modificación de estos, tengan que ser adaptados al medio cubano. Se impone entonces la creación de un, o varios productos nacionales que suplan esta necesidad, ERP/CRM Cubano.

Cedrux surge por esta necesidad, se le solicita la creación a la Universidad de Ciencias Informáticas en apoyo de varias entidades que hasta el momento llevaban este tema, como Desoft y TEICO Villa Clara, como todo ERP este se encarga de la organización de las empresas y cuenta con módulos para la gestión de sistemas de administración de usuarios, gestión de configuración, gestión de datos centralizada y gestión de servicios Web, así como la gestión de Recursos Humanos, este sistema se encuentra aun en fase de prueba, aunque existen algunas empresas dentro de las cuales se han realizados pruebas pilotos y los resultados han sido óptimos.

Muchos de estos sistemas pueden resultar muy eficientes en varias empresas que lo emplean, pero como parte de la solución que se desea obtener no son factibles, pues son sistemas construidos a la medida y su usabilidad esta limitada por las especificidades propias del software, por lo que sería muy poco factible adquirir un sistema ya desarrollado que habría que personalizar totalmente a las características de la institución. Además de que los sistemas mencionados anteriormente no cumplen con la estrategia tecnológica que sigue el país en la actualidad, que es lograr la independencia tecnológica a través del software libre, la mayoría de ellos son software propietarios por la tanto obtener las licencias demandaría un gasto excesivo de dinero, y de tiempo para Cuba debido a las imposiciones del embargo económico que se ha extendido ya al campo de la informática. Todo lo antes mencionado, demuestra que a pesar de

existir diversos sistemas eficientes y con alta capacidad de reutilización, su uso no es extensible al campo de acción de esta investigación.

1.2.4 Propuesta de Solución Informática para la Gestión de Recursos Informáticos en la UCI.

La solución informática para la gestión de recursos informáticos en la UCI sistematizará, digitalizará y centralizará todos los recursos informáticos de la Universidad con los que cuentan los proyectos para la producción. Permitirá mantener un control exacto y real de los medios con el fin de poder planificar correctamente los cronogramas de entrega y que estos no se retrasen.

La utilización de un sistema para la gestión de recursos en la Universidad de las Ciencias Informáticas implantará una forma de crear valor para la institución. Con ella se persiguen objetivos como lograr la optimización de la disponibilidad de los equipos productivos. Con esto se favorecerá en gran medida a la protección de los medios destinados a la producción y se mitigarán los procesos tardíos para la emisión de reportes encaminados a la gestión de información referente a la existencia de los recursos, se realizarán de forma inmediata las consultas, se atenuarán los errores humanos, se podrá además optimizar los recursos humanos que se dedican a esta labor.

1.3 Tendencias y Tecnologías Actuales.

A continuación se analizan las tendencias y tecnologías más utilizadas actualmente a nivel mundial en el desarrollo de aplicaciones web, lenguajes, herramientas de modelado, y metodologías para el desarrollo del software.

1.3.1 ¿Qué es software libre?

Software libre: se refiere a la libertad de los usuarios para poder copiar, ejecutar, distribuir, estudiar, cambiar y mejorar el software. Suele estar disponible gratuitamente (no significa necesariamente que sean gratis), o a precio del costo de la distribución de éste. Aunque conserve su carácter libre, puede ser vendido comercialmente. Existen cuatro libertades para los usuarios del software libre, ellas son:

- ✓ Libertad de usar el programa, con cualquier propósito.
- ✓ Libertad de estudiar cómo funciona el programa, y adaptarlo a sus necesidades.
- ✓ Libertad de distribuir copias, con lo que se puede ayudar a los compañeros.

- ✓ Libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que puedan beneficiarse más personas.

1.3.2 Web.

World Wide Web, o sencillamente Web, es el universo de información accesible a través de internet. Se caracteriza por:

- ✓ Información por hipertexto: Diversos elementos (texto o imágenes) de la información que se nos muestran en la pantalla están vinculados con otras informaciones que pueden ser de otras fuentes. Para mostrar en pantalla esta otra información bastara con hacer clic sobre ellos.
- ✓ Gráfico: En la pantalla aparece simultáneamente texto, imágenes e incluso sonidos.
- ✓ Global: Se puede acceder a el desde cualquier tipo de plataforma, usando cualquier navegador y desde cualquier parte del mundo.
- ✓ Pública: Toda su información esta distribuida en miles de ordenadores que ofrecen su espacio para almacenarla. Toda esta información es pública y toda puede ser obtenida por el usuario.
- ✓ Dinámica: La información, aunque esta almacenada, puede ser actualizada por el que la publicó sin que el usuario deba actualizar su soporte técnico.
- ✓ Independiente: Dada la inmensa cantidad de fuentes, es independiente y libre.

Mediante un software especial denominado Browser o Explorador un usuario visualiza páginas web que pueden contener texto, imágenes, videos u otros contenidos multimedia, y navega a través de ellas usando hipervínculos. Internet por su parte es un conjunto de ordenadores o servidores conectados a una red de redes mundiales, que prestan servicios a otros ordenadores que se conectan a dicha red y que comparten un mismo protocolo de comunicación; en la actualidad es una fuente inagotable de conocimiento para el hombre.

1.3.3 Aplicación web

Una aplicación web no es más que un sistema web, dígase servidor web, red, protocolo o navegador, donde la entrada del usuario (entrada de datos y navegación) afecta el estado del negocio. La arquitectura general de estas aplicaciones es la de un sistema cliente/servidor. Las mismas implementan lógica de

negocios y su uso varía el estado del negocio. Instalar una aplicación web consiste en configurar los componentes del lado del servidor en la red y es innecesaria una instalación o configuración en el lado cliente. El protocolo principal de comunicación es HTTP (Protocolo de Transferencia de Hipertexto), y habitualmente funciona desconectado.

1.3.4 Servidor web.

El Servidor Web es el programa encargado de gestionar las peticiones de los usuarios que visitan su página Web. Es el programa encargado de mostrar sus páginas cuando un visitante realiza una petición en su dominio.

Teniendo en cuenta que el 80% de las operaciones que realiza un usuario en el servidor es visionar paginas Web, la elección del programa encargado para tal cometido es clave para el buen funcionamiento general del servidor. A continuación se exponen algunas de las características del Servidor Apache y el IIS con vista a seleccionar el que se utilizará para el desarrollo de la aplicación.

Servidor web Apache.

Apache es considerado el Servidor Web por excelencia, no solo por su gran aceptación, puesto que casi el 70% de los servidores Internet confía en él, sino porque desde su nacimiento ha demostrado con creces su estabilidad, solidez y su mayor rendimiento ante sus competidores.

Principales ventajas:

Fiabilidad: Alrededor del 90% de los servidores con más alta disponibilidad funcionan bajo servidores Apache.

Software Libre: Apache es totalmente gratuito, y se distribuye bajo la licencia Apache Software License, que permite la modificación del código.

Extensibilidad: se pueden añadir módulos para ampliar las capacidades de Apache.

También dispone de una amplia variedad de módulos, que permiten desde generar contenido dinámico (con PHP, Java, Perl) monitorizar el rendimiento del servidor, crear servidores virtuales por IP o por nombre (varias direcciones web son manejadas en un mismo servidor) y limitar el ancho de banda para

cada uno de ellos. Dichos módulos están disponibles junto con su código fuente, por lo cual pueden ser incluso modificados por cualquier persona con conocimientos de programación.

Apache es un servidor Web que permite acceder a las páginas Web que están alojadas en una computadora. Es de código abierto y actualmente es el servidor Web que más se utiliza en el mundo, encontrándose por encima de sus competidores, tanto gratuitos como comerciales. Funciona sobre cualquier plataforma. Hoy en día es uno de los mejores servidores en términos de eficiencia, funcionalidad y velocidad.

Este servidor tiene capacidad para servir tanto páginas estáticas como dinámicas a través de otras herramientas soportadas que facilitan la actualización de los contenidos usando bases de datos, ficheros u otras fuentes de información.

Apache posee una estructura de módulos, es decir, está dividido en muchas porciones de código que hacen referencia a diferentes aspectos o funcionalidades. Esta modularidad es intencionada ya que la configuración de cada módulo se hace mediante la configuración de las directivas que están contenidas dentro del módulo.

Las funcionalidades más elementales se encuentran en el módulo base, siendo necesario un módulo multiproceso para manejar las peticiones. Se han diseñado varios módulos multiprocesos para cada uno de los sistemas operativos sobre los que se ejecuta el Apache, optimizando el rendimiento y rapidez del código.

El resto de funcionalidades del servidor se consigue por medio de módulos adicionales que se pueden cargar. Para añadir un conjunto de utilidades al servidor, simplemente hay que añadirle un módulo, de forma que no sea necesario volver a instalar el software.

Servidor Web Internet Information Server (IIS)

Internet Information Server, IIS, es una serie de servicios para los ordenadores que funcionan con Windows.

Este servicio convierte a un ordenador en un servidor de Internet o Intranet, es decir, que en las computadoras que tienen este servicio instalado se pueden publicar páginas web tanto local como remotamente (servidor web).

El servidor web se basa en varios módulos que le dan capacidad para procesar distintos tipos de páginas.

Una de las ventajas es la autenticación implícita que permite a los administradores autenticar a los usuarios de forma segura a través de servidores de seguridad y proxy.

IIS también es capaz de impedir que aquellos usuarios con direcciones IP conocidas obtengan acceso no autorizado al servidor, permitiendo especificar la información apropiada en una lista de restricciones.

Microsoft ha mejorado sustancialmente su software estrella en el campo de los servicios Web. Los avances vienen motivados sobre todo por la seguridad y el rendimiento pero todavía adolece de algunos agujeros de seguridad además de ser código cerrado, IIS solo puede operar en plataforma Windows.

Justificación de la selección realizada

Luego de lo analizado anteriormente se decide seleccionar Apache como servidor web porque es soportado en múltiples Sistemas Operativos, lo que lo hace prácticamente universal. Es una tecnología gratuita de código fuente abierto. El hecho de ser gratuita es importante pero no tanto como que se trate de código fuente abierto. Esto le da una transparencia a este software de manera que si se quiere ver que es lo que se esta instalando como servidor, se puede saber, sin ningún secreto, sin ninguna puerta trasera. Es un servidor altamente configurable de diseño modular. Es muy sencillo ampliar sus capacidades. Trabaja con gran cantidad de lenguajes como Perl, PHP y otros lenguajes de script. Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto. Tiene una alta configurabilidad en la creación y gestión de logs. Permite la creación de ficheros log para tener un mayor control sobre lo que sucede en el servidor.

1.3.5 Metodologías para el Desarrollo de Software

Las metodologías de desarrollo de software son muy usadas, estas llegaron para quedarse, evolucionar y revolucionar las prácticas de ingeniería de software.

Para tener un proceso de producción de software con el menor número de fallos, adecuado a las necesidades del cliente y entregar a tiempo el producto, la producción de software debe convertirse en un proceso disciplinado.

En los últimos tiempos la cantidad y variedad de los procesos de desarrollo ha aumentado de forma impresionante, se han desarrollado dos corrientes en lo referente a los procesos de desarrollo: los llamados métodos pesados y los métodos ligeros. La diferencia fundamental entre ambos es que mientras los métodos pesados intentan conseguir el objetivo común por medio de orden y documentación, los métodos ligeros (también llamados métodos ágiles) tratan de mejorar la calidad del software por medio de una comunicación directa e inmediata entre las personas que intervienen en el proceso.

A continuación se detallarán algunas características de las Metodologías RUP y XP, a razón de que hoy día son de las más populares, en vista de una mayor comprensión de la selección realizada.

RUP

El Proceso Unificado de Rational (Rational Unified Process en inglés, habitualmente resumido como RUP) 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.

El RUP está basado en 5 principios clave que son: adaptar el proceso, balancear prioridades, demostrar valor iterativamente, elevar el nivel de abstracción y enfocarse en la calidad.

El ciclo de vida RUP es una implementación del desarrollo en espiral. Fue creado ensamblando los elementos en secuencias semi-ordenadas. El ciclo de vida organiza las tareas en fases e iteraciones.

RUP divide el proceso en cuatro fases: Inicio, Elaboración, Construcción y Transición, dentro de las cuales se realizan varias iteraciones en número variable según el proyecto y en las que se hace un mayor o menor hincapié en los distintas actividades. Las primeras iteraciones (en las fases de Inicio y Elaboración) se enfocan hacia la comprensión del problema y la tecnología, la delimitación del ámbito del proyecto, la eliminación de los riesgos críticos, y al establecimiento de una línea base de la arquitectura. En la fase de construcción, se lleva a cabo la construcción del producto por medio de una serie de iteraciones. En la fase de transición se pretende garantizar que se tiene un producto preparado para su entrega a la comunidad de usuarios.

Principales características de RUP:

- ✓ Forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo)
- ✓ Pretende implementar las mejores prácticas en Ingeniería de Software
- ✓ Desarrollo iterativo
- ✓ Administración de requisitos
- ✓ Uso de arquitectura basada en componentes
- ✓ Control de cambios
- ✓ Modelado visual del software
- ✓ Verificación de la calidad del software

El RUP se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Incluye artefactos (que son los productos tangibles del proceso como por ejemplo, el modelo de casos de uso, el código fuente, etc.) y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso).

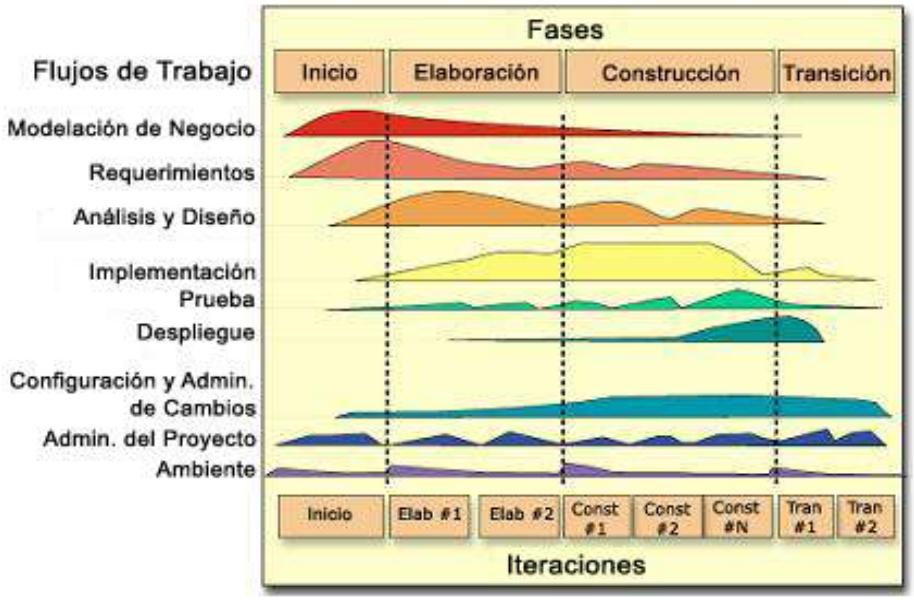


Fig. # 1: Fases e iteraciones de la Metodología

Programación extrema (XP).

La programación extrema (XP) es el más destacado de los procesos ágiles de desarrollo de software. Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software.

Los principios básicos de la programación extrema son: Simplicidad, Comunicación, Retroalimentación.

Las características fundamentales del método son:

- ✓ Desarrollo iterativo e incremental.
- ✓ Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión.
- ✓ Programación en parejas.
- ✓ Frecuente integración del equipo de programación con el cliente o usuario.
- ✓ Corrección de todos los errores antes de añadir nueva funcionalidad. Hacer entregas frecuentes.
- ✓ Refactorización del código, es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento.
- ✓ Propiedad del código compartida.
- ✓ Simplicidad en el código

  La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer. Mientras más simple es el sistema, menos tendrá que comunicar sobre este, lo que lleva a una comunicación más completa, especialmente si se puede reducir el equipo de programadores.

Justificación de la selección realizada.

Luego de analizar dos de las metodologías mas utilizadas se decide utilizar RUP pues a pesar de que este pudiera parecer un proyecto de corto plazo, representa un subsistema de un proyecto de larga duración

por la que necesita una metodología robusta para su desarrollo, además de que debe existir uniformidad entre todos los subsistemas, por lo que no puede utilizarse una metodología ligera en algunos módulos y una pesada en otros.

1.3.6 Lenguajes de Modelado

El constante avance en el nivel de complejidad de las soluciones informáticas ha hecho de la utilización de los modelos, un mecanismo para facilitar la comprensión de estos. “Los modelos proporcionan un mayor nivel de abstracción, permitiendo trabajar con sistemas mayores y más complejos, y facilitando el proceso de codificación e implementación del sistema de forma distribuida y en distintas plataformas.” (Lidia y Vallecillo Fuentes, 2003.)

“Un modelo es una descripción de (parte de) un sistema, descrito en un lenguaje bien definido.” (Lidia y Vallecillo Fuentes, 2003.).

A continuación se detallarán algunas características de dos de los Lenguajes de Modelado más usados en la actualidad, para una mayor comprensión de la selección realizada.

Lenguaje Unificado de Modelado (UML)

UML “es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software “(Jacobson, Booch, & Rumbaugh, 2000).No define un proceso de desarrollo específico, tan solo se trata de una notación. “Permite modelar sistemas de información, y su objetivo es lograr modelos que, además de describir con cierto grado de formalismo tales sistemas, puedan ser entendidos por los clientes o usuarios de aquello que se modela” (Jacobson, y otros, 2000).

Uno de los fines principales de la creación de UML fue posibilitar el intercambio de modelos entre las distintas herramientas CASE orientadas a objetos del mercado, para ello era necesario definir una notación y semántica común, es decir, un estándar. Se logró con este lenguaje “un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyecto” (Jacobson, y otros, 2000). Algunas de las propiedades de UML como lenguaje de modelado son las que se exponen en (Jacobson, y otros, 2000) son:

- ✓ Concurrencia, es un lenguaje distribuido y adecuado a las necesidades de conectividad actuales y futuras.
- ✓ Reemplaza a decenas de notaciones empleadas con otros lenguajes.
- ✓ Modela estructuras complejas.
- ✓ Las estructuras más importantes que soportan tienen su fundamento en las tecnologías orientadas a objetos, tales como objetos, clase, componentes y nodos.
- ✓ Emplea operaciones abstractas como guía para variaciones futuras, añadiendo variables si es necesario.
- ✓ Comportamiento del sistema: casos de uso, diagramas de secuencia y de colaboraciones, que sirven para evaluar el estado de las máquinas.

UML ofrece una amplia variedad de diagramas que permiten visualizar el sistema desde varias perspectivas, estos son: Diagrama de Casos de Uso, Diagrama de Clases, Diagrama de Objetos, Diagrama de Secuencia, Diagrama de Colaboración, Diagrama de Estados, Diagrama de Actividad, Diagrama de Componentes, Diagrama de Despliegue. Para modelar el comportamiento dinámico del sistema son utilizados los diagramas de actividad, secuencia y colaboración.

En relación con los Diagramas de Actividad de UML, a pesar de que su uso para la modelación de los procesos del negocio ha sido criticado debido a la limitada expresividad de sus versiones anteriores, razón por la cual “en el paso de la versión 1.5 a la versión 2.0 la parte más modificada es precisamente la referente a los Diagramas de Actividad” (Pérez J. D., 2007), lográndose que los mismos fueran una notación con la expresividad adecuada para modelar todo tipo de procesos de negocio, se debe tener presente que:

- ✓ Existen muchas herramientas para trabajar con ellos como por ejemplo: IBM Rational Software Architect, Enterprise Architect, Visual Paradigm.
- ✓ Existe gran cantidad de procesos de negocio que están modelados usando esta notación.
- ✓ Los desarrolladores tienen gran experiencia utilizando tanto UML como sus Diagramas de Actividad (J.B.López, 2005.).

Notación para el Modelado de Procesos del Negocio (BPMN).

BPMN es un estándar cuyo principal objetivo es proporcionar una notación fácilmente comprensible por todos los usuarios del negocio, desde los analistas, los desarrolladores técnicos, hasta aquellos que monitorizarán y gestionarán los procesos.

La creación de BPMN tuvo como base la recopilación de experiencias de varios estándares como UML. Es importante tener en cuenta que BPMN abarca únicamente los procesos de negocio, lo que significa que otro tipo de modelos relacionados (estructura de la organización, recursos, modelos de datos, estrategias, reglas de negocio) quedan fuera de la especificación (J.B.López, 2005.).

Presenta una gran expresividad a la hora de especificar procesos, mucho más expresivo que los diagramas de actividad de UML, es gráficamente más rico, con menos símbolos fundamentales, pero con más variaciones de estos, lo que facilita su comprensión por parte de gente no experta.

Justificación de la selección realizada.

De los lenguajes de modelados antes explicados se decide utilizar UML, debido a que es un lenguaje estándar, fácil de aprender, además permite una comunicación fluida entre los desarrolladores de software y ofrece una amplia variedad de diagramas para visualizar el sistema desde varias perspectivas.

1.3.7 Herramientas CASE

Cuando se habla de herramientas de modelado en la disciplina de Ingeniería del Software, es importante mencionar las herramientas CASE. Estas están tomando cada vez más relevancia en la planeación y ejecución de proyectos que involucren sistemas de información, pues suelen inducir a sus usuarios a la correcta utilización de metodologías que le ayuden a llegar con facilidad a los productos de software construidos.

CASE proporciona al ingeniero la posibilidad de automatizar actividades manuales y de mejorar su visión general de la ingeniería [...] Las herramientas CASE ayudan a garantizar que la calidad se diseñe antes de llegar a construir el producto" (Pressman, 2005).

Actualmente existen varias herramientas CASE(Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) y varían con respecto a las capacidades de modelado con UML, el soporte del ciclo de vida del proyecto, las ingenierías directa e inversa, el modelado de datos, el precio, el

soporte, la facilidad de uso, etc. A continuación se mencionan algunas de estas herramientas analizadas para finalmente llegar a la comprensión de la herramienta seleccionada.

Rational Rose

Esta herramienta CASE da soporte al modelado visual con UML cubriendo todo el ciclo de vida de un proyecto. Se enmarca dentro del desarrollo de modelado para fines académicos, investigativos y comerciales. Permite la autogeneración de código a partir de modelos y viceversa para lenguajes como Java y realizar ingeniería inversa en: Visual Basic, C++, etc.

Está dotado de un navegador que muestra información de forma Jerárquica de todos los elementos de los modelos de un proyecto. Presenta varias opciones para el manejo de los diagramas en el momento de su visualización, exportación o impresión.

Visual Paradigm

Esta herramienta tiene unas características gráficas muy cómodas, que facilitan la realización de los diagramas de modelado que sigue el estándar de UML, los mismo son: Diagramas de clase, Casos de Uso, Comunicación, Secuencia, Estado, Actividad, Componentes, etc. Ofrece:

- ✓ Diseño centrado en casos de uso y enfocado al negocio que generan 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 (versión profesional) e inversa.
- ✓ Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.

Umbrello

Umbrello es una herramienta CASE libre para crear y editar diagramas UML, que ayuda en el proceso del desarrollo de software, facilitará la creación de un producto de alta calidad, especialmente durante fases de análisis y diseño del proyecto. Genera código automáticamente en los lenguajes C++, Java, entre otros, y se puede importar de C++. Presenta varias opciones para el manejo de los diagramas en el momento de su visualización, exportación o impresión.

Justificación de la selección realizada.

Luego de hacer un estudio detallado de algunas de las herramientas CASE, se decide utilizar Visual

Paradigm ya que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Este software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La principal razón de esta selección es que es una herramienta libre, multiplataforma, y en calidad de la migración que pretende llevar a cabo Cuba unánimemente se está adoptando como herramienta de modelado en la universidad

1.3.8 Sistemas gestores de bases de datos.

PostgreSQL.

PostgreSQL ofrece muchas ventajas para su compañía o negocio respecto a otros sistemas de bases de datos:

- ✓ Este software ha sido diseñado y creado para tener un mantenimiento y ajuste mucho menor que los productos de los proveedores comerciales, conservando todas las características, estabilidad y rendimiento.
- ✓ En contraste a muchos sistemas de bases de datos comerciales, es extremadamente común que compañías reporten que PostgreSQL nunca ha presentado caídas en varios años de operación de alta actividad.
- ✓ El código fuente está disponible para todos sin costo.
- ✓ PostgreSQL está disponible en casi cualquier Unix (34 plataformas en la última versión estable), y una versión nativa de Windows está actualmente en estado beta de pruebas.
- ✓ Diseñado para ambientes de alto volumen

PostgreSQL usa una estrategia de almacenamiento de filas llamada MVCC para conseguir una mucha mejor respuesta en ambientes de grandes volúmenes. Los principales proveedores de sistemas de bases de datos comerciales usan también esta tecnología, por las mismas razones.

Manejador de bases de datos. PgAdmin.

Todas las bases de datos construidas necesitan ser administradas y que mejor para cumplir con dicho objetivo que un programa con interfaz gráfica. Para PostgreSQL existen muchas herramientas de propósito general que administran sus bases de datos entre las que se encuentra PgAdmin una de las

más utilizadas por la comunidad de desarrollo del software libre. PgAdmin diseña, mantiene y administra fácilmente las bases de datos construidas en PostgreSQL, funciona en distintos sistemas operativos como Windows 95/98, NT, Xp, así como en plataformas libres. Está liberada bajo la licencia Open Source. Es la herramienta más popular y completa, diseñada para responder a las necesidades de los usuarios permitiéndole escribir desde simples consultas y sentencias SQL hasta diseñar complejas bases de datos. Algunas de las características de PgAdmin son:

- ✓ Entradas SQL aleatorias.
- ✓ Pantallas de información y ayudas para bases de datos, tablas, índices, secuencias, vistas, programas de arranque, funciones y lenguajes.
- ✓ Preguntas y respuestas para configurar usuarios, grupos y privilegios.
- ✓ Control de revisión con mejora de la generación de script.
- ✓ Ayudas para importar y exportar datos.
- ✓ Ayuda para migrar bases de datos.

Oracle

Oracle es sin duda una de las mejores bases de datos que se tiene en el mercado, 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 correcta, sin causar inconsistencias; ayuda a administrar y almacenar grandes volúmenes de datos; estabilidad, escalabilidad.

Oracle está disponible en múltiples plataformas como Windows, Linux. La naturaleza multiplataforma de Oracle, lo convierte en una verdadera solución empresarial.

Aunque su dominio en el mercado de servidores empresariales ha sido casi total, recientemente sufre la competencia de gestores de bases de datos comerciales y de la oferta de otros con licencia Software Libre como PostgreSql, Mysql. Las últimas versiones de Oracle han sido certificadas para poder trabajar bajo Linux.

Justificación de la selección realizada

Se decide utilizar Postgres SQL 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 puesto que está disponible en casi cualquier Unix (34 plataformas en la última versión estable), y ahora en versión nativa para Windows.

1.3.9 Hojas de estilo en cascada CSS

Hojas de Estilo en Cascada (CSS) (Cascading Style Sheets), es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos. Las ventajas que se obtienen cuando se trabaja con CSS son:

- ✓ **Separación del contenido y presentación:** Las hojas de estilo generalmente se encuentran en archivos separados del código principal. Esto permitirá que en un equipo de trabajo, programador y diseñador puedan realizar sus tareas de forma independiente aunque paralela, sin correr el riesgo de que haya interferencias entre ambos, y ello no alterará el resultado final.
- ✓ **Flexibilidad:** Utilizando las hojas de estilos se puede cambiar en cualquier momento alguna parte o la totalidad del diseño de las páginas web con sólo modificar la hoja de estilo, sin que ello suponga modificar el contenido.
- ✓ **Unificación del diseño de las páginas del sitio:** Mantener una misma apariencia de las páginas de un sitio web se puede volver una tarea pesada y tediosa si se tiene que copiar y pegar código cada vez que cree una página nueva, o que se desee modificar una misma cosa en todas sin embargo enlazando las páginas web con las hojas de estilo, se agiliza este proceso y se minimiza el trabajo.
- ✓ **Optimización de los tiempos de carga y de tráfico en el servidor:** Al dividir contenido y apariencia se obtienen archivos más ligeros, y esto nos reporta dos beneficios: Por un lado, se reducen notablemente los tiempos de carga del sitio en el navegador. A esto se le une la capacidad de éste para mantener la hoja de estilo en caché. Por otro lado, se reduce el volumen de tráfico del servidor.

- ✓ **Precisión o elasticidad:** Desde el momento en que se use CSS, el tamaño y posicionamiento de los elementos que formen las páginas web podrá ser exacto. Se le indicará al navegador donde debe colocarse dichos elementos o aquellas imágenes que se empleen así como el ancho y alto de los mismos. También se emplean medidas variables o relativas que permitan expandir el contenido hasta ocupar la totalidad de la ventana de navegación como se desee, o contraerla a sólo una parte de la misma, con independencia de la resolución de pantalla del usuario.
- ✓ **Limpieza del código fuente:** Si se escribe una hoja de estilos independiente, el código fuente de la web va a resultar menos pesado y se agilizarán las tareas de localización de las líneas que se buscan.
- ✓ **Compatibilidad y continuidad:** Las reglas establecidas por la especificación CSS-1 fijaron los estándares del diseño, y se mantienen y respetan en la CSS-2.

1.3.10 Lenguajes de Programación del lado del servidor

PHP

PHP (Procesador de hipertexto) es un lenguaje de programación interpretado usado normalmente para la creación de páginas web dinámicas. Es un lenguaje de programación del lado del servidor muy potente que, junto con HTML, permite crear sitios web dinámicos. La forma de usar PHP es insertando código PHP dentro del código HTML de un sitio web. Cuando un cliente (cualquier persona en la web) visita la página web que contiene éste código, el servidor lo ejecuta y el cliente sólo recibe el resultado. Su ejecución, es por tanto en el servidor, a diferencia de otros lenguajes de programación que se ejecutan en el navegador.

Ventajas

- ✓ Es un lenguaje multiplataforma.
- ✓ Posee capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad
- ✓ Posee capacidad de expandir su potencial utilizando la enorme cantidad de módulos
- ✓ Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.

- ✓ Es un lenguaje libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- ✓ Permite las técnicas de Programación Orientada a Objetos.
- ✓ Posee una biblioteca nativa de funciones sumamente amplia e incluida
- ✓ No requiere definición de tipos de variables.
- ✓ Tiene manejo de excepciones.

ASP.NET

ASP.NET es un marco de trabajo de programación generado en Common Language Runtime que puede utilizarse en un servidor para generar eficaces aplicaciones Web. ASP.NET ofrece varias ventajas importantes acerca de los modelos de programación Web anteriores:

Mejor rendimiento. ASP.NET es un código compilado que se ejecuta en el servidor. A diferencia de sus predecesores, puede aprovechar las ventajas del enlace anticipado, la compilación en tiempo de ejecución (just-in-time), la optimización nativa y los servicios de caché desde el primer momento. Esto supone un incremento espectacular del rendimiento antes de siquiera escribir una línea de código.

Compatibilidad con herramientas de primer nivel. Se complementa con un diseñador y un paquete de herramientas excelentemente diseñado en el entorno integrado de programación. Una de las características que proporciona esta herramienta es que presenta controles de servidor de arrastrar y colocar, además de la implementación automática.

Justificación de la selección realizada

Se decide utilizar PHP como lenguaje de programación del lado del servidor, debido a que es un leguaje multiplataforma, tiene una capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y Postgres, posee una amplia documentación en Internet, incluyendo una gran variedad de ejemplos y de ayudas, además se adapta a las técnicas de Programación Orientada a Objetos.

1.3.11 Lenguajes del lado del cliente

Java Script

Lenguaje que permite interactuar con el navegador de manera dinámica y eficaz, proporcionando a las páginas web mayor dinamismo. Es el navegador el que soporta la carga de procesamiento. Su uso se basa fundamentalmente en la creación de efectos especiales en las páginas y la definición de interactividades con el usuario.

El advenimiento de Java Script ha resuelto de manera fácil y elegante la mayoría de los problemas con que se enfrenta el diseñador de páginas Web referidos a la programación. Sus requerimientos son relativamente sencillos, es un lenguaje cuyos códigos se interpretan en el navegador del cliente, sin tener que ir y venir del cliente al servidor actualizando la información.

Visual Basic Script

Es un lenguaje de programación de scripts del lado del cliente, pero sólo compatible con Internet Explorer. Está basado en Visual Basic, un popular lenguaje para crear aplicaciones Windows. Tanto su sintaxis como la manera de trabajar están muy inspiradas en él. Sin embargo, no todo lo que se puede hacer en Visual Basic se puede hacer en Visual Basic Script, pues este último es una versión reducida del primero.
El modo de funcionamiento de Visual Basic Script para construir efectos especiales en páginas web es muy similar al utilizado en JavaScript y los recursos a los que se puede acceder también son los mismos: el navegador.

Justificación de la selección realizada

Se decide emplear como lenguaje de programación del lado del cliente JavaScript, debido a que es un leguaje seguro y fiable, no requiere un tiempo de compilación, además los scripts se pueden desarrollar en un periodo de tiempo relativamente corto, este no incluye la sintaxis y reglas complejas de Java, no requieren mucha memoria ni tiempo adicional de transmisión.

1.3.12 Entorno Integrado de Desarrollo (IDE).

A continuación se expondrán, las características fundamentales de los entornos de desarrollo integrado: Zend Studio y Eclipse, quienes presentan mayor potencialidad para el trabajo con la Web que otros entornos o IDEs, como por sus siglas se conoce.

De manera general un entorno de desarrollo integrado o en inglés Integrated Development Environment, es un programa compuesto por un conjunto de herramientas para un programador (MaestrosdelWeb, 2008). Puede dedicarse en exclusiva a un sólo lenguaje de programación o bien, puede utilizarse para varios.

Zend Studio.

Zend Estudio es un programa de la compañía Zend, impulsores de la tecnología de servidor PHP, orientada a desarrollar aplicaciones web, en lenguaje PHP. El programa, además de servir de editor de texto para páginas PHP, proporciona una serie de ayudas que pasan desde la creación y gestión de proyectos hasta la depuración de código.

El programa entero está escrito en Java, lo que a veces supone que no funcione tan rápido como otras aplicaciones de uso diario. Sin embargo, esto ha permitido a Zend lanzar con relativa facilidad y rapidez versiones del producto para Windows y Linux.

Zend Studio consta de dos partes en las que se dividen las funcionalidades de parte del cliente y las del servidor. Las dos partes se instalan por separado, la del cliente contiene el interfaz de edición y la ayuda, permite además hacer depuraciones simples de scripts, aunque para disfrutar de toda la potencia de la herramienta de depuración habrá que disponer de la parte del servidor que instala Apache y el módulo PHP o, en caso de que estén instalados, los configura para trabajar juntos en depuración (MaestrosdelWeb, 2008).

Si se desea aumentar la productividad en los desarrollos PHP no cabe duda que este programa puede resultar útil y prestar gran ayuda en la realización de los mismos ofreciendo un entorno agradable y funcionalidades que simplifican el trabajo y optimizan el resultado. Todas las opciones que dispone están pensadas con acierto por personas capacitadas que conocen a fondo la tecnología. Zend Studio incorpora suficientes ayudas como para que su utilización sea confiable y resulte idóneo para desarrollar aplicaciones Web.

Eclipse

En la web oficial de eclipse, se define como "An IDE for everything and nothing in particular"(un IDE para todo y para nada en particular). Eclipse es en el fondo, únicamente un armazón sobre el que se puede montar herramientas de desarrollo para cualquier lenguaje, mediante la implementación de los plugins adecuados.

La arquitectura de plugins de Eclipse permite además de integrar diversos lenguajes sobre un mismo IDE, introducir otras aplicaciones accesorias que pueden resultar útiles durante el proceso de desarrollo como: herramientas UML, editores visuales de interfaces, ayuda en línea para librerías, entre otros. Existen versiones de Eclipse instalables para cualquier plataforma que incluyen el código fuente y los plugins más habituales.

Una de las características más curiosas del IDE Eclipse es el modo en que se compilan los proyectos. No existe en Eclipse ningún botón que permita compilar individualmente un fichero concreto. La compilación es una tarea que se lanza automáticamente al guardar los cambios realizados en el código.

La principal diferencia entre un simple editor y un buen entorno de desarrollo es que este se integre con una buena herramienta visual para depurar los programas escritos. Eclipse incluye un depurador potente, sencillo y muy cómodo de utilizar. Permite de una forma muy simple, generar automáticamente la documentación del propio proyecto.

Eclipse ofrece servicios que permiten el trabajo entre desarrolladores de forma más eficiente y sincronizada, pues cuenta con las Taks (tareas), que posibilitan una comunicación directa entre los miembros del equipo, permitiendo dar órdenes de trabajo de manera online en forma de correo electrónico. Esta opción hace de Eclipse un IDE ideal para el trabajo en equipo.

Justificación de la selección realizada

Se decide seleccionar Eclipse como editor de PHP. La elección se ha basado fundamentalmente en su soporte, posibilidades de depuración y pruebas de PHP con el set más completo de herramientas para la creación de aplicaciones altamente fiables como lo requiere el mercado; en su alto rendimiento y escalabilidad; la seguridad mejorada sobre otras tecnologías web existentes y la posibilidad de hacer despliegue de sistemas desarrollados con esta tecnología en ambiente tanto Windows como Linux, además de ser de código abierto, objetivos específicos de la presente investigación.

1.3.13 Framework

Un framework simplifica el desarrollo de una aplicación web, esto se logra a través de la automatización de algunos patrones utilizados para resolver tareas comunes, además proporciona estructura al código fuente y facilita la programación de aplicaciones. En este epígrafe se estudiara 2 de estos framework con el fin de escoger uno de ellos para utilizar en la construcción de la solución propuesta.

Symfony

Symfony es uno de los framework PHP más populares entre los usuarios y las empresas, ya que permite que los programadores sean mucho más productivos a la vez que crean código de más calidad y más fácil de mantener. Symfony es maduro, estable, profesional y está muy bien documentado.

Symfony es un framework para construir aplicaciones Web con PHP. En otras palabras, Symfony es un enorme conjunto de herramientas y utilidades que simplifican el desarrollo de las aplicaciones Web.

Symfony emplea el tradicional patrón de diseño MVC (modelo-vista-controlador) para separar las distintas partes que forman una aplicación Web. El modelo representa la información con la que trabaja la aplicación y se encarga de acceder a los datos.

La vista transforma la información obtenida por el modelo en las páginas Web a las que acceden los usuarios. El controlador es el encargado de coordinar todos los demás elementos y transformar las peticiones del usuario en operaciones sobre el modelo y la vista.

Algunas Características de Symfony:

Symfony se diseñó para que se ajustara a los siguientes requisitos:

- ✓ Fácil de instalar y configurar en la mayoría de plataformas
- ✓ Independiente del sistema gestor de bases de datos
- ✓ Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos
- ✓ Sigue la mayoría de *mejores prácticas* y patrones de diseño para la Web

- ✓ Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo
- ✓ Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo
- ✓ Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

Prado

Prado es un Framework bastante orientado a objetos con similitudes con lenguajes como Microsoft ASP .NET. Entre sus principales características están:

- ✓ Componentes comunes en el diseño web con lógica implementada como TButton, TCheckBox, TTable, TTabPanel, entre otros.
- ✓ Desarrollo orientado a componentes con la posibilidad de que los mismos programadores/usuarios los perfeccionen.
- ✓ Programación dirigida a eventos.
- ✓ Servicio SOAP propio (TSoapService) para aplicaciones que se comunican vía Web Services.
- ✓ Manual online en la misma Web con la explicación de cada componente, ejemplos, y sintaxis de los métodos.

Prado es un framework para PHP basado en componentes y en eventos. Inicialmente inspirado en Apache Tapestry, la primera versión se realizó para PHP4, pero se reescribió completamente para PHP5.

Entre las características que ofrece se pueden encontrar la separación entre la presentación y la lógica, su arquitectura modular configurable, componentes web, internalización y localización, manejo de errores y mucho más.

Justificación de la selección realizada.

Luego de un análisis detallados de ambos framework se decide utilizar Symfony deviene candidato ideal porque, entre otras razones, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web; proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo

de una aplicación web compleja; así como informatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación.

1.3.14 Propuestas de herramientas seleccionadas.

Con lo explicado y analizado en este capítulo se ha decidido que es necesario realizar el análisis, diseño, implementación de una aplicación web que permita gestionar los recursos informáticos de los proyectos productivos de la UCI, la aplicación debe manejar gran cantidad de datos y con una interfaz amigable que responda a las necesidades de los clientes.

Después de realizar un análisis detallado de algunas herramientas CASE, metodologías de desarrollo de software, lenguaje de modelado, gestores de base de datos, lenguajes de programación web, se propone como metodología a utilizar RUP, UML como lenguaje de modelado, herramienta CASE Visual Paradigm, Sistema gestor de base de datos PostgreSQL, Symfony como Framework, como IDE Eclipse.PHP para la implementación del lado del servidor y JavaScript como lenguaje de programación del lado del cliente. También se empleará como material de apoyo CSS para los estilos Además de Apache como servidor web.

1.4 Conclusiones

El estudio de las diferentes metodologías de desarrollo de software, lenguaje de modelado herramientas CASE, lenguajes de programación web y sistemas gestores de base de datos permitió conocer cuales son las que se ajustan para el desarrollo de la solución informática para la Gestión de Recursos Informáticos. A través del análisis de las diferentes estrategias de captura de requisitos se conocieron las más adecuadas a utilizar teniendo en cuenta las características y condiciones del cliente y del equipo de desarrollo.

El análisis de algunas soluciones informáticas existentes, que gestionan recursos demuestra la necesidad de la solución informática para la gestión de recursos en la UCI.

CAPITULO 2 CARACTERÍSTICAS DEL SISTEMA

2.1 Introducción

En el presente capítulo se describen los procesos del negocio que tiene que ver con el objeto de estudio, primeramente se modela el negocio propuesto, se identifican los actores, trabajadores, los casos de uso correspondientes y la descripción de los mismos. Se especifican las reglas del negocio. También se enumeran los requisitos; tanto funcionales como no funcionales que debe cumplir el sistema que se propone, se identificará mediante un diagrama de casos de uso las relaciones de los actores que interactúan con el sistema.

El desarrollo de la aplicación web que se propone se centra en la puesta en práctica del Proceso Unificado de Desarrollo de Software (RUP), con el apoyo del Lenguaje Unificado de Modelado (UML) para la modelación de los distintos artefactos que involucran el desarrollo del trabajo y, haciendo uso de la herramienta Case Visual Paradigm para darle cumplimiento a la solución propuesta.

2.2 ¿Qué es un Modelo de Negocio?

Para describir los procesos del negocio que se relacionan con el campo de acción de este trabajo es necesario centrarse en los procesos de gestión de recursos de los proyectos productivos. Para un mayor y mejor entendimiento de cómo se lleva actualmente a cabo la gestión de recursos en los proyectos productivos es mejor hacer uso de las técnicas de modelado que propone UML. El paso más importante dentro de dicho modelado es la identificación de los procesos del negocio de la producción de la UCI. La obtención de los procesos del negocio es crucial ya que establece las metas y límites de modelado del negocio.

Dada la estructuración del negocio que se está estudiando se propone un Modelo del Negocio ya que este permite de forma visual mostrar a los usuarios los principales conceptos que se manejan. Esto se hace con el objetivo de ayudar a los usuarios, clientes, desarrolladores y otros a tener un mejor entendimiento de los procesos de negocio y, a utilizar un vocabulario común entre todos para un mejor entendimiento del contexto en que se pone el sistema. Para una mejor captura de los requisitos y así construir un sistema correcto se necesita tener un seguro conocimiento de las actividades y del funcionamiento del objeto de estudio.

2.2.1. Reglas generales del negocio

A continuación se definirán las políticas que deben cumplirse o condiciones que deben satisfacerse para que se ejecuten los casos de uso, de manera que regulen los procesos de negocio de acuerdo a las especificidades de cada uno.

- ✓ El Solicitante debe llevar todos los documentos necesarios y correctos al Especialista del Grupo de Soporte para el Desarrollo (GSD) de la IP para la solicitud de gestión de recursos que desee realizar.
- ✓ El Solicitante es la persona autorizada a solicitar la gestión de recursos
- ✓ El Especialista GSD solo puede tener acceso a la información que se maneja para la gestión de los recursos de la UCI.
- ✓ El Especialista GSD es el encargado de revisar las solicitudes que le hace un Solicitante y es quien elabora la solicitud a nivel UCI.

2.2.2 Definición Actores y Trabajadores del Negocio.

Los Actores son aquellas personas que se beneficiarán directamente con el Negocio, para los que la realización de Casos de Uso tiene un resultado visible. Los Trabajadores serán aquellas personas o sistemas automatizados o semi-automatizados que llevan a cabo las diversas tareas o actividades para dar cumplimiento al Negocio.

Actor del negocio	**Descripción**
 Solicitante (Asesor de Arquitectura, Especialista y Líder de Proyecto Productivo)	Se encarga de realizar el levantamiento de los recursos informáticos en el proyecto productivo.
Trabajador	**Descripción**

 Especialista Grupo de Soporte al Desarrollo(GSD)	Se encarga de revisar todas las planillas recibidas de las facultades, elabora las planillas UCI y procesa toda la información proveniente de las facultades.

Tabla # 1 Descripción de actores y trabajadores del negocio

2.2.3 Diagrama de Caso de Uso del Negocio.

Los Casos de Uso del Negocio son un conjunto de procesos que serán iniciados por un Actor y que tendrán resultados observables para este. Se modelan en un Diagrama de Casos de Uso y de cada uno de ellos se crea una especificación con su descripción detallada.

El diagrama de casos de uso del negocio que a continuación se muestra, representa gráficamente a los procesos que ocurren para la gestión de recursos de los proyectos productivos

Fig. # 2: Modelado de Caso de Uso del Negocio

2.3 Descripción de los procesos del negocio propuesto.

La descripción de los Casos de Uso del Negocio es el complemento del Diagrama de Casos de Uso. Permite conocer en detalle como se realizan los procesos, las acciones de los Actores y Trabajadores y los resultados de esto. En ella se especifican el nombre del Caso de Uso, los Actores y Trabajadores involucrados, las condiciones previas y posteriores a la ejecución de las acciones, el flujo de eventos y los flujos alternos que se puedan generar, además de un breve resumen que permita conocer el contenido del Caso de Uso. A continuación se expone la descripción del caso de uso, en vista a una mayor comprensión del desarrollo del proceso de gestión de recursos informáticos

Descripción textual Caso de Uso del Negocio: Gestionar Recursos Informáticos

Caso de Uso:	**Gestionar Recursos Informáticos**
Actores:	Solicitante (Asesor de Arquitectura, Especialista, Líder Proyecto Productivo).
Trabajadores:	Especialistas del Grupo de Soporte para el Desarrollo de la IP(GSD)
Resumen:	Este Caso de Uso se inicia cuando el actor Solicitante llena la planilla con el levantamiento de los recursos informáticos con que cuentan el proyecto productivo. Estas planillas son enviadas a los Especialistas del GSD de la IP, los cuales las procesan, terminando así este Caso de uso.
Precondiciones:	

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Negocio**
1. El Solicitante realiza un levantamiento de los recursos informáticos con que cuenta el proyecto productivo a través de la planilla de Levantamiento de Recursos Informáticos	
2. Envía la planilla con el levantamiento de los recursos a un Especialista del GSD de la IP.	3. El Especialista del GSD de la IP recibe la planilla

	4. Verifica que todos los datos de la planilla de Levantamientos de Recursos Informáticos del proyecto productivo estén correctos.
	5. Se crea la planilla UCI con toda la información del levantamiento de recursos
	6. Se le informa al Solicitante de la aprobación del registro de recursos.
7. Recibe la Información	

Flujos Alternos	
Acción del Actor	**Respuesta del Negocio**
	4.1. La solicitud no es aprobada por el Especialista del GSD de la IP.
8. Recibe la Información	
Prioridad:	Crítico

Tabla # 2: Descripción textual del Caso de Uso: Gestionar Recursos Informáticos

2.3.1 Diagrama de actividades

Un diagrama de actividad describe un proceso que explora el orden de las tareas o actividades que logran los objetivos del negocio. Es similar a un diagrama de estados en el cual todos o la mayoría de los estados son estados de actividad y en la cual todas o la mayoría de las transiciones se disparan al completarse las acciones en los estados fuentes precedentes.

Diagrama de Actividad CU Gestión de Recursos Informáticos

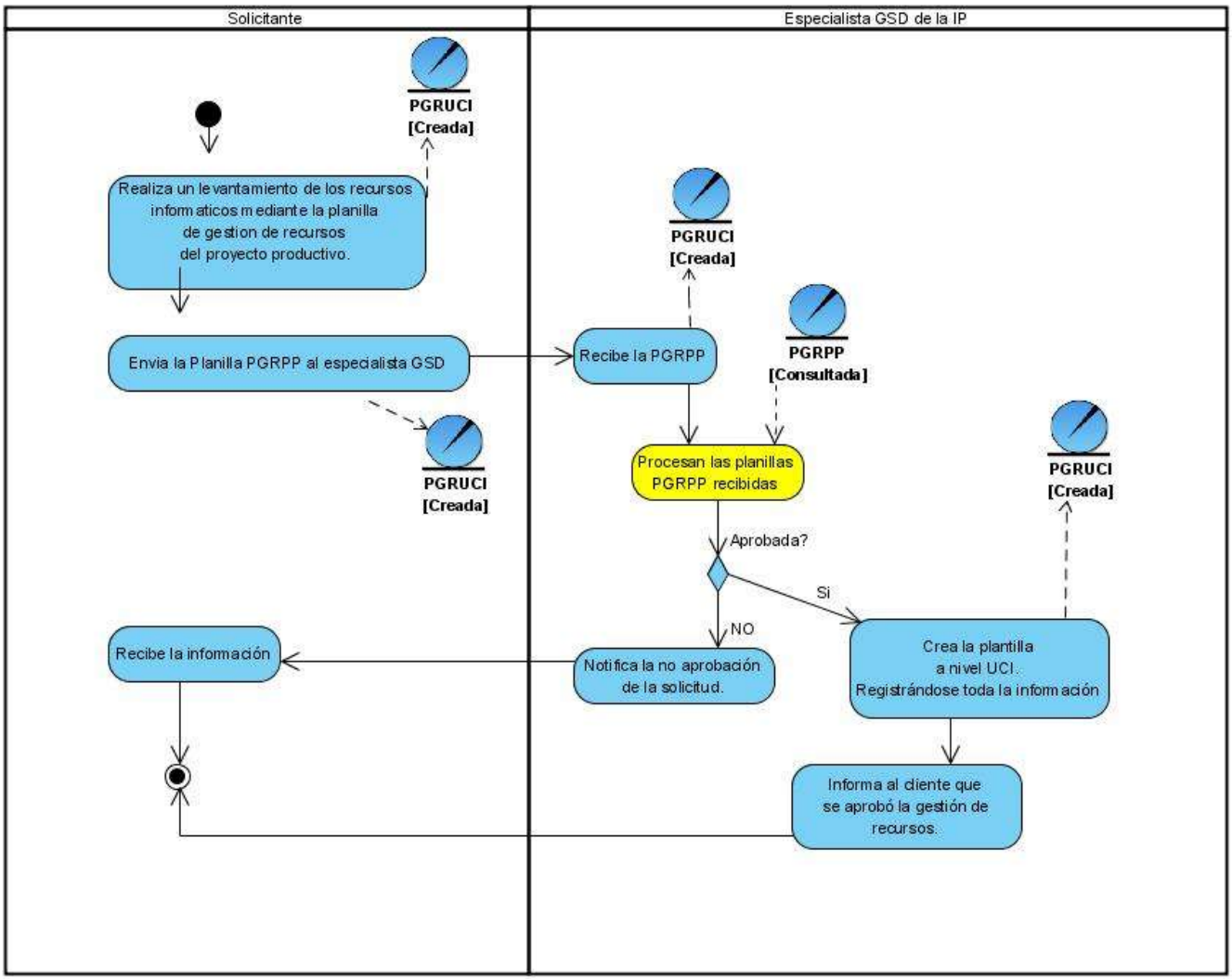


Fig. # 3: Diagrama de actividades CU Gestionar Recursos Informáticos

2.4 Modelo de Objetos del Negocio.

El modelo de objetos del negocio describe cómo colaboran los trabajadores y las entidades del negocio dentro del flujo de trabajo de cada uno de los procesos del negocio.

Diagrama de Clases del Modelo de Objetos.

Fig. # 4: Modelo de Objetos del Negocio

2.5 Requisitos del Sistema

Desde ahora, ya conocido los conceptos asociados al objeto de estudio del problema se comienza a modelar el sistema que se va construir, y de paso se analizará ¿Qué debe hacer el sistema para que se cumplan los objetivos planteados al inicio de este trabajo? Para lo mismo se identifican los requisitos Funcionales (RF) y No Funcionales (RNF), y se modelan los RF en representaciones de Casos de Uso del sistema. De acuerdo con los objetivos planteados el sistema debe ser capaz de:

2.5.1. Requerimientos funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir, por tal motivo a continuación se presenta un conjunto de ellos:

RF1. El sistema debe ser capaz de: Autenticar Usuario.

RF2. El sistema debe ser capaz de: Gestionar Usuario.

2.1 Insertar usuario.

2.2 Eliminar Usuario.

2.3 Buscar Usuario.

2.4 Modificar Usuario

RF3. El sistema debe ser capaz de: Gestionar Proyectos.

3.1 Insertar Proyectos Productivos.

3.2 Eliminar Proyectos Productivos.

3.3 Modificar Proyectos Productivos.

3.4 Permutar computadora de un Proyecto a Otro

3.4 Mostrar cantidad de Proyectos Productivos UCI.

RF4. El sistema debe ser capaz de: Gestionar Facultad.

4.1 Insertar Facultades.

4.2 Eliminar Facultades

4.3 Modificar Facultades

4.4 Mostrar cantidad de facultades de la UCI.

RF5. El sistema debe ser capaz de: Gestionar Laboratorios

5.1 Insertar Laboratorios.

5.2 Eliminar Laboratorios.

5.3 Modificar Laboratorios.

5.4 Mostrar cantidad de Laboratorios de la UCI.

RF6. El sistema debe ser capaz de: Gestionar Docentes.

6.1 Insertar Docente.

6.2 Eliminar Docente.

6.3 Modificar Docente

6.4 Mostrar cantidad de docentes de la UCI.

RF7. El sistema debe ser capaz de: Gestionar Computadoras.

7.1 Insertar Computadoras.

7.2 Eliminar Computadoras.

7.3 Modificar datos de las Computadoras.

7.4 Mostrar datos de las Computadoras existentes UCI.

7.5 Buscar computadoras por facultades y proyectos.

RF8. El sistema debe ser capaz de: Gestionar Componentes.

8.1 Insertar Componentes.

8.2 Eliminar Componentes.

8.3 Modificar Componentes

8.4 Mostrar cantidad de componentes existentes en la UCI.

RF9. El sistema debe ser capaz de: Gestionar Características de los Componentes.

9.1 Insertar Características de los Componentes.

9.2 Eliminar Características de los Componentes.

9.3 Modificar Características de los Componentes.

9.4 Mostrar datos de las Características de los Componentes

RF10. El sistema debe ser capaz de: Gestionar Medidas.

10.1 Insertar Capacidades de Medidas.

10.2 Eliminar Capacidades de Medidas.

10.3 Modificar Capacidades de Medidas.

10.4 Mostrar datos de las Capacidades de Medidas existentes en la UCI.

RF11. El sistema debe ser capaz de Generar reportes.

RF12. El sistema debe ser capaz de: Realizar búsquedas de Datos a nivel de Universidad.

12.1 Dada una facultad buscar todos los datos de hardware de sus proyectos.

12.2 Dado una facultad y un proyecto buscar todos los datos de hardware del mismo.

12.3 Dado una facultad, un proyecto y una computadora buscar los datos de la misma

2.5.2 Requisitos no funcionales.

Para que el sistema que se propone logre ser óptimo y eficaz en el campo de acción en que se utilice, no solo es importante definir las capacidades que debe cumplir sino también aquellas cualidades que lo harán más atractivo al cliente, usable, rápido y confiable.

Los requisitos no funcionales pueden ser más críticos que los funcionales, puesto que si un requisito funcional no se cumple, el sistema se degrada, pierde eficacia, y puede no responder a la totalidad de los requerimientos del usuario, pero en cambio si un requisito no funcional no se cumple, el sistema puede inutilizarse.

A continuación se expondrán los requisitos no funcionales que el sistema propuesto debe cumplir.

Usabilidad

RNF 1. El sistema debe estar disponible las veinticuatro horas del día, sin ninguna interrupción.

RNF 2. El sistema debe ser accesible desde todos los puntos donde exista una máquina conectada a la red.

RNF 3. El sistema debe tener una interfaz que le sea familiar al usuario para aprovechar sus conocimientos en el manejo de herramientas de software.

RNF 4. El sistema debe ser una interfaz de fácil aprendizaje para que usuarios inexpertos puedan familiarizarse lo más pronto posible y le sea cómodo el manejo del software.

RNF 5. El sistema debe diferenciar las interfaces gráficas y opciones para los usuarios que accedan al sistema con diferentes roles.

Confiabilidad y Seguridad

RNF 6. El sistema debe tener la capacidad de identificar con certeza a los diversos usuarios o entidades que interactúan con el.

RNF 7. El sistema debe tener la capacidad que tiene el sistema de darle seguridad al usuario, de que las informaciones solo serán vistas por quien este capacitado para esto.

RNF 8. El servicio del sistema debe tener una disponibilidad aceptable (99%).

RNF 9. El tiempo entre fallos debe ser breve o cero, haciendo lo posible para que esto no ocurra.

RNF 10. La base de datos debe estar fraccionada sobre varios esquemas, dividiendo así de una forma lógica las funcionalidades, evitando así la pérdida total de la información en caso de algún accidente o ataque.

RNF 11. Permitir autentificarse a través de un Servidor de Dominio.

Rendimiento

RFN 12. El sistema debe ser capaz de mantener el mismo rendimiento y estabilidad a medida que aumenta la cantidad de datos a gestionar.

RNF 13. El sistema debe ser capaz de correr al ser montado en una misma PC todos sus componentes tales como Gestor de bases de Datos, Aplicación Web, etc.

Soportabilidad y Operatividad

RNF 14. El software podrá operar en cualquier sistema operativo debido a que se desarrollará en PHP y herramientas de software libre como es el caso de PostgreSQL. En cuanto al soporte debe tenerse en cuenta en el sistema operativo en que se este trabajando y buscar la tecnología necesaria para las herramientas que requiere el software en el mismo.

Mantenimiento y actualización

RNF 15. El ambiente de desarrollo donde se implementará dicho software será: Eclipse. Gestor de Base de Datos: PostgreSQL. Lenguaje de Programación: PHP. El sistema debe ser construido en un código estándar, cada procedimiento debe estar comentado.

Funcionalidad

RNF 16. Capacidades de búsqueda con velocidad apropiada. El sistema debe ser Multiplataforma.

Desempeño y escalabilidad

RNF 17. El tiempo de respuestas debe ser corto con respuestas rápidas y eficientes, 3.0 segundos.

RNF 18. El sistema podrá soportar un gran número de clientes online, 150 clientes.

RNF 19. El sistema debe tener un rendimiento óptimo debido a que presta servicios a un gran número de usuarios.

Requerimientos de software del Sistema.

RNF 20. Las computadoras clientes deben contar con Mozilla Firefox .

RNF 21. El sistema operativo debe ser Linux Ubuntu.

RNF 22. La computadora servidora debe tener instalado el SGBD PostgreSQL8.2.

RNF 23. La computadora servidora debe tener instalado el servidor Apache2.2.

RNF 24. La computadora servidora debe tener instalado el Framework Symfony y PHP5.

Requerimientos de hardware del sistema

RNF 25. La computadora servidor debe tener como mínimo 13 GB de HDD y 2GB de RAM.

RNF 26. Las computadoras clientes deben estar conectadas en red.

RNF 27. Las computadoras clientes deben ser Pentium III o superior.

RNF 28. El servidor debe ser Pentium III o superior.

RNF 29. El servidor debe tener Intel(R) PRO/1000 PM Network Connection.

Requerimiento de ayuda y documentación

RNF 31. El proyecto debe brindar un manual de usuario para el correcto uso de sus funcionalidades y brindarles una mayor experiencia a los usuarios del sistema.

2.6 Propuesta de solución

Haciendo uso de las habilidades y facilidades que brinda UML, se capturan los requisitos funcionales (RF)

del sistema y se representan mediante un Diagrama de Casos de Uso. Se define anteriormente cuales serán los actores que van a interactuar con el sistema, y a la vez los Casos de Uso que me van a representar las diferentes funcionalidades.

Los casos de uso son artefactos narrativos que describen, bajo la forma de acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario. Por lo tanto, establece un acuerdo entre clientes y desarrolladores sobre las condiciones y posibilidades (requisitos) que debe cumplir el sistema. Por otra parte Los actores del sistema pueden representar un rol que juega una o varias personas, un equipo o un sistema automatizado, pueden ser un recipiente pasivo de información.

Los actores del sistema pueden representar un rol que juega una o varias personas, un equipo o un sistema automatizado, pueden ser un recipiente pasivo de información e intercambiar información con él pero no son parte del sistema.

2.6.1 Actores del sistema

Actores	**Descripción**
Usuario(Líder del Proyecto Productivo, Asesor de tecnología del vicedecano de producción de la facultad)	Se encarga de interactuar con el sistema con el objetivo de gestionar los datos referentes al proyecto productivo y a la facultad, generaliza el acceso al sistema y se debe autenticar antes de entrar.
Especialistas del GSD de la IP	Es el encargado de configurar los perfiles de los usuarios del sistema

Tabla # 3: Descripción de los Actores del sistema a informatizar

2.6.2 Diagramas casos de uso del sistema

Un DCUS representa gráficamente a los procesos y su interacción con los actores, permite comprender qué actor interviene en qué proceso

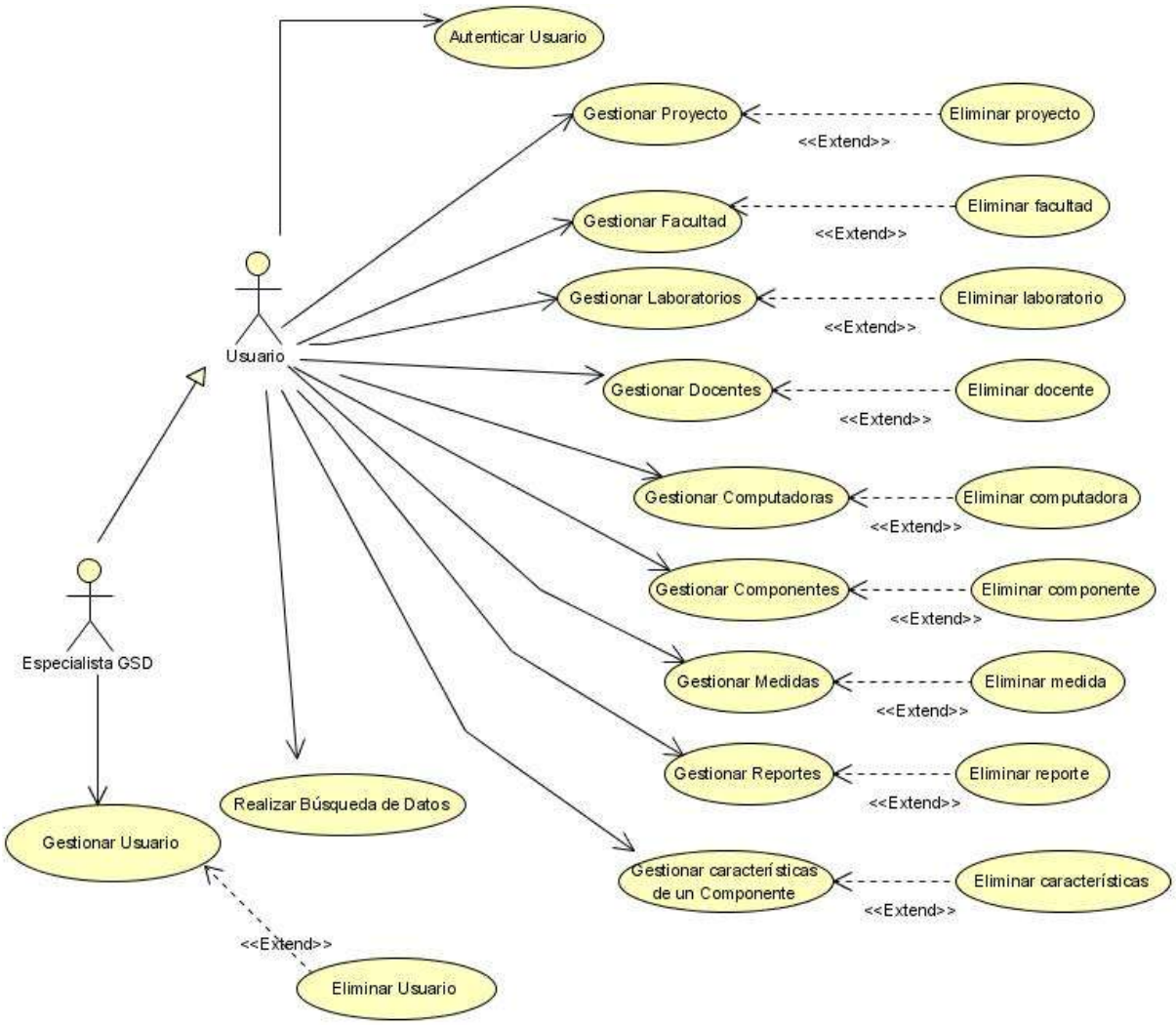


Fig. # 5: Diagrama de Caso de Uso del Sistema

2.7. Descripción de los Casos de Uso del Sistema

Para una mayor comprensión de los casos de uso se explicara en detalle el flujo que describen.

2.7.1. Descripción del Caso de Uso: Autenticar Usuario

Caso de Uso:	Autenticar Usuario.
Actores:	Usuario(inicia)
Resumen:	El caso de uso permite que los usuarios con determinadas responsabilidades en el sistema introduzcan sus credenciales (usuario y contraseña) locales para que el sistema las verifique y ejecuten las funcionalidades que puedan según su rol.
Referencia:	RF1
Precondiciones:	-
Poscondiciones:	El sistema permite que el usuario se autentifique y acceda a la zona de trabajo que tiene acceso.

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario inicia la aplicación.	2. El sistema le muestra una interfaz con un formulario para que se autentifique.
3. El usuario ingresa los datos en el formulario (usuario y contraseña).	4. El sistema verifica que los campos del formulario no estén vacio.
	5. Verifica la existencia del usuario en el sistema.
	6. El sistema le da acceso al usuario de interactuar con la aplicación

Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	4.1. Muestra un mensaje de error indicando que debe llenar todos los campos. (“Campos requeridos”).
	5.1. Muestra un mensaje de error indicando que la cuenta existe. (“Usuario no autenticado o contraseña incorrecta”).
Prioridad:	Secundario
Prototipo de Interfaz:	

Tabla # 4: Descripción textual del Caso de Uso: Autenticar Usuario

2.7.2. Descripción del Caso de Uso: Gestionar Usuario

Caso de Uso:	**Gestionar Usuario**
Caso de Uso Asociado	**Eliminar Usuario**
Actores:	Especialista Grupo de soporte para el desarrollo (GSD)(inicia)
Resumen:	El caso de uso comienza cuando un Especialista GSD accede a la opción de Insertar, Buscar o Modificar Usuario del menú de operaciones. El caso de uso termina cuando el sistema devuelve la información solicitada, y el usuario accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF2

Precondiciones:	El usuario con el rol de Especialista GSD se debe haber autenticado con anterioridad.
Poscondiciones:	1. Información de usuario adicionada a la Base de Datos. 2. Información de usuario modificada en la base de Datos. 3. Mostrar la información que contiene un usuario determinado.

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Especialista GSD Selecciona la opción de Adicionar Usuario.	2. El sistema le muestra una interfaz con los datos a completar. (Usuario y contraseña).
3. El Especialista GSD ingresa los datos solicitados por el sistema.	4. El sistema verifica que los campos del formulario no estén vacios.
	5. El sistema verifica que el usuario no este registrado en la base de datos.
	6. El sistema procede al registro del usuario en la base de datos correspondiente
Flujos Alternos	
	4.1. Muestra un mensaje de error indicando que debe llenar todos los campos. (“Campos requeridos”). Retorna a la acción 2.
	5.1. Muestra un mensaje de error indicando que

	la cuenta ya existe. (“El nombre de usuario especificado ya existe”).
Sección Modificar Usuario	
1. El Especialista GSD selecciona la opción Modificar Usuario	2. El sistema busca todos los usuarios existentes y los muestra en una pantalla para que el Especialista GSD elija el usuario a modificar.
3. El Especialista GSD elige el usuario y pincha en el botón Modificar Usuario.	4. El sistema muestra una pantalla con todos los datos del usuario elegido de forma editable.
5. El Especialista GSD modifica los datos necesarios y pincha el botón Aceptar.	6. El sistema verifica que los campos modificados estén correctos.
	7. El sistema modifica todos lo datos del usuario seleccionado.
Flujos Alternos de los Eventos	
	6.1. El sistema muestra un mensaje de error indicando el problema.
Sección Buscar Usuario	
1. El Especialista GSD selecciona la opción Buscar Usuario.	2. El sistema muestra una pantalla con los identificadores (nombre usuario) de los usuarios existentes.
3. El Especialista GSD elige el usuario que desea buscar.	4. El sistema realiza la búsqueda y muestra los datos del mismo.

Prioridad:	Crítico
Prototipo de Interfaz:	

Tabla # 5: Descripción textual del Caso de Uso: Gestionar Usuario

2.7.3. Descripción del Caso de Uso: Gestionar Proyectos

Caso de Uso:	**Gestionar Proyectos**
Caso de Uso Asociado	**Eliminar Proyecto**
Actores:	Usuario (inicia)
Resumen:	El caso de uso comienza cuando un Usuario accede a la opción de Adicionar, Modificar y Mostrar proyectos productivos del menú de operaciones. El caso de uso termina cuando el sistema devuelve la información solicitada, y el usuario accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF3
Precondiciones:	El usuario con el rol de Usuario se debe haber autenticado con anterioridad.
Poscondiciones:	1. Información del proyecto adicionada a la Base de Datos. 2. Información del proyecto modificada en la base de Datos. 3. Mostrar los proyectos productivos existentes.
Flujo Normal de Eventos	
Sección Adicionar Proyectos Productivos.	

Acción del Actor	Respuesta del Sistema
1. El Usuario selecciona la opción de Adicionar Proyecto.	2. El sistema muestra la interfaz para que el Usuario introduzca los datos del nuevo proyecto productivo.
3. El Usuario ingresa los datos solicitados por el sistema y presiona el botón aceptar.	4. El sistema verifica que los campos del proyecto no estén vacio.
	5. El sistema verifica que ese proyecto no este registrado en la base de datos.
	6. El sistema registra los datos del nuevo proyecto productivo y muestra el nuevo proyecto productivo ya insertado.
Flujos Alternos	
	4.1 Muestra un mensaje de error indicando que debe llenar todos los campos. (“Llene todos los campos”).
	5.1 El sistema muestra un mensaje de error indicando que ese proyecto ya existe en la base de datos.
Sección Modificar Proyectos Productivos.	
1. El Usuario selecciona la opción Modificar Proyecto Productivo.	2. El sistema busca todos los Proyectos Productivos de la UCI y las muestra en una pantalla para que el Usuario elija el

	Proyecto Productivo a Modificar.
3. El Usuario elije el Proyecto Productivo a Modificar y presiona el botón Modificar.	4. El sistema muestra una pantalla con todos los datos del proyecto productivo de forma editable.
5. El Usuario modifica los datos necesarios y pincha el botón Aceptar.	6. El sistema verifica que todos lo datos modificados estén correctos.
	7. El sistema modifica todos lo datos del proyecto productivo seleccionado.
Flujos Alternos de los Eventos	
	6.1. El sistema muestra un mensaje de error indicando el problema.
Sección Mostrar Proyecto Productivos.	
1. El Usuario selecciona la opción Listar	2. El sistema busca los proyectos Productivos existentes y los muestra en una nueva pantalla.
Prioridad:	Crítico
Prototipo de Interfaz:	

Tabla # 6: Descripción textual del Caso de Uso: Gestionar Proyectos

2.7.4. Descripción del Caso de Uso: Gestionar Facultades

Caso de Uso:	**Gestionar Facultades**
Caso de Uso Asociado	**Eliminar Facultad**

Actores:	Usuario(inicia)
Resumen:	El caso de uso comienza cuando un Usuario accede a las opciones de Adicionar, Modificar, o Mostrar las Facultades de la UCI. El caso de uso termina cuando el sistema realiza la operación elegida, y el Usuario accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF4
Precondiciones:	El usuario con el rol de Usuario se debe haber autenticado con anterioridad.
Poscondiciones:	1. Información de la facultad adicionada a la Base de Datos. 2. Información de la facultad modificada en la base de Datos. 3. Mostrar las facultades existentes.

Flujo Normal de Eventos	
Sección Adicionar Facultad.	
Acción del Actor	**Respuesta del Sistema**
1. El Usuario selecciona la opción de Adicionar Facultad.	2. El sistema muestra la interfaz para que el Usuario introduzca los datos de la nueva Facultad.
3. El Usuario ingresa los datos solicitados por el sistema y presiona el botón aceptar.	4. El sistema verifica que los campos del formulario no estén vacios.
	5. El sistema verifica que esa facultad no este registrada en la base de datos.

	6. El sistema registra los datos de la nueva facultad y los muestra
Flujos Alternos	
	4.1. Muestra un mensaje de error indicando que debe llenar todos los campos. (“Llene todos los campos”).
	5.1. El sistema muestra un mensaje indicando que esa facultad ya existe en la base de datos.
Sección Modificar Facultad.	
1. El Usuario selecciona la opción Modificar Facultad.	2. El sistema busca todas las Facultades de la UCI y las muestra en una pantalla para que el Usuario elija la Facultad a Modificar.
3. El Usuario elije la facultad a Modificar y presiona el botón Modificar.	4. El sistema muestra una pantalla con todos los datos de la facultad de forma editable.
5. El Usuario modifica los datos necesarios y pincha el botón Aceptar.	6. El sistema verifica que los campos modificados estén correctos.
	7. El sistema modifica todos lo datos de la facultad seleccionada.
Flujos Alternos de los Eventos	
	6.1. El sistema muestra un mensaje de error indicando el problema.
Sección Mostrar Facultades.	

1. El Usuario selecciona la opción Listar	2. El sistema busca las facultades existentes y las muestra en una nueva pantalla
Prioridad:	Crítico
Prototipo de Interfaz:	

Tabla # 7: Descripción textual del Caso de Uso: Gestionar Facultades

2.7.5. Descripción del Caso de Uso: Gestionar Laboratorios

Caso de Uso:	**Gestionar Laboratorios**
Caso de Uso Asociado	**Eliminar Laboratorio**
Actores:	Usuario(inicia)
Resumen:	El caso de uso comienza cuando un Usuario accede a la opción de Adicionar, Modificar y Mostrar laboratorios del menú de operaciones. El caso de uso termina cuando el sistema devuelve la información solicitada, y el usuario accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF5
Precondiciones:	El usuario con el rol de Usuario se debe haber autenticado con anterioridad.
Poscondiciones:	1. Información del laboratorio adicionada a la Base de Datos. 2. Información del laboratorio modificada en la base de Datos. 3. Mostrar los laboratorios existentes.

Flujo Normal de Eventos	
Sección Adicionar Laboratorio.	
Acción del Actor	**Respuesta del Sistema**
1. El Usuario selecciona la opción de Adicionar Laboratorio.	2. El sistema muestra la interfaz para que el Usuario introduzca los datos del nuevo laboratorio.
3. El Usuario ingresa los datos solicitados por el sistema y presiona el botón aceptar.	4. El sistema verifica que los campos del formulario no estén vacios.
	5. El sistema verifica que ese laboratorio no este registrado en la base de datos.
	6. El sistema registra los datos del nuevo laboratorio y muestra el nuevo laboratorio ya insertado.
Flujos Alternos	
	4.1. Muestra un mensaje de error indicando que debe llenar todos los campos. (“Llene todos los campos”).
	5.1. El sistema muestra un mensaje de error indicando que ese laboratorio ya existe en la base de datos.

Sección Modificar Laboratorio.	
1. El Usuario selecciona la opción Modificar Laboratorio.	2. El sistema busca todos los Laboratorios existentes UCI y los muestra en una pantalla para que el Usuario elija el Laboratorio a Modificar.
3. El Usuario elije el Laboratorio a Modificar y presiona el botón Modificar.	4. El sistema muestra una pantalla con todos los datos del laboratorio de forma editable.
5. El Usuario modifica los datos necesarios y pincha el botón Aceptar.	6. El sistema verifica que los campos modificados estén correctos.
	7. El sistema modifica todos lo datos del laboratorio seleccionado.
Flujos Alternos de los Eventos	
	6.1 El sistema muestra un mensaje de error indicando el problema.
Sección Mostrar Laboratorios.	
1. El Usuario selecciona la opción Listar	2. El sistema busca los laboratorios existentes y los muestra en una nueva pantalla
Prioridad:	Crítico
Prototipo de Interfaz:	

Tabla # 8: Descripción textual del Caso de Uso: Gestionar Laboratorios

2.7.6. Descripción del Caso de Uso: Gestionar Docentes

Caso de Uso:	**Gestionar Docentes**
Caso de Uso Asociado	**Eliminar Docente**
Actores:	Usuario (inicia)
Resumen:	El caso de uso comienza cuando un Usuario accede a las opciones de Adicionar, Modificar, o Mostrar los Docentes de la UCI. El caso de uso termina cuando el sistema realiza la operación elegida, y el Usuario accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF6
Precondiciones:	El usuario con el rol de Usuario se debe haber autenticado con anterioridad.
Poscondiciones:	1. Información del docente adicionada a la Base de Datos. 2. Información del docente modificada en la base de Datos. 3. Mostrar los docentes existentes.
Flujo Normal de Eventos	
Sección Adicionar Docente.	
Acción del Actor	**Respuesta del Sistema**
1. El Usuario selecciona la opción de Adicionar Docente.	2. El sistema muestra la interfaz para que el Usuario introduzca los datos del nuevo Docente.

3. El Usuario ingresa los datos solicitados por el sistema y presiona el botón aceptar.	4. El sistema verifica que los campos del formulario no estén vacios.
	5. El sistema verifica que ese docente no este registrado en la base de datos.
	6. El sistema registra los datos del nuevo docente y muestra el nuevo docente ya insertado.
Flujos Alternos	
	4.1 Muestra un mensaje de error indicando que debe llenar todos los campos. (“Llene todos los campos”).
	5.1. El sistema muestra un mensaje de error indicando que ya existe ese docente en la base de datos.
Sección Modificar Docente.	
1. El Usuario selecciona la opción Modificar Docente.	2. El sistema busca todos los Docentes de la UCI y las muestra en una pantalla para que el Usuario elija el Docente a Modificar.
3. El Usuario elije el Docentes a Modificar y presiona el botón Modificar.	4. El sistema muestra una pantalla con todos los datos del docente de forma editable.

5. El Usuario modifica los datos necesarios y pincha el botón Aceptar.	6. El sistema verifica que los campos modificados estén correctos.
	7. El sistema modifica todos lo datos del docente seleccionado.
Flujos Alternos de los Eventos	
	6.1 El sistema muestra un mensaje de error indicando el problema.
Sección Mostrar Docentes.	
1. El Usuario selecciona la opción Listar	2. El sistema busca los docentes existentes y los muestra en una nueva pantalla
Prioridad:	Crítico
Prototipo de Interfaz:	

Tabla # 9: Descripción textual del Caso de Uso: Gestionar Docentes

2.7.7. Descripción del Caso de Uso: Gestionar Computadoras

Caso de Uso:	**Gestionar Computadoras**
Caso de Uso Asociado	**Eliminar Computadora**
Actores:	Usuario (inicia)
Resumen:	El caso de uso comienza cuando un Usuario accede a las opciones de Adicionar, Modificar, o Mostrar las Computadoras de la UCI. El caso de uso termina cuando el sistema realiza la

	operación elegida, y el Usuario accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF7
Precondiciones:	El usuario con el rol de Usuario se debe haber autenticado con anterioridad.
Poscondiciones:	1. Información de la computadora adicionada a la Base de Datos. 2. Información de la computadora modificada en la base de Datos. 3. Mostrar las computadoras existentes.

Flujo Normal de Eventos	
Sección Adicionar Computadora.	
Acción del Actor	**Respuesta del Sistema**
1. El Usuario selecciona la opción de Adicionar Computadora.	2. El sistema muestra la interfaz para que el Usuario introduzca los datos de la nueva Computadora.
3. El Usuario ingresa los datos solicitados por el sistema y presiona el botón aceptar.	4. El sistema verifica que los campos del formulario no estén vacios.
	5. El sistema verifica que esta computadora no este registrado ya en la base de datos.
	6. El sistema registra los datos de la nueva computadora y muestra la nueva computadora ya insertada.

Flujos Alternos	
	4.1 Muestra un mensaje de error indicando que debe llenar todos los campos. (“Llene todos los campos”).
	5.1 El sistema muestra un mensaje de error indicando que ya existe esa computadora en la base de datos.
Sección Modificar Computadora.	
1. El Usuario selecciona la opción Modificar Computadora.	2. El sistema busca en la base de datos todas las Computadoras de la UCI y las muestra en una pantalla para que el Usuario elija la Computadora a Modificar.
3. El Usuario elije la Computadora a Modificar y presiona el botón Modificar.	4. El sistema muestra una pantalla con todos los datos de la computadora de forma editable.
5. El Usuario modifica los datos necesarios y pincha el botón Aceptar.	6. El sistema verifica que los campos modificados estén correctos.
	7. El sistema modifica todos lo datos de la computadora seleccionada.
Flujos Alternos de los Eventos	
	6.1 El sistema muestra un mensaje de error indicando el problema.

Sección Mostrar Computadoras	
1. El Usuario selecciona la opción Listar Computadoras	2. El sistema busca las computadoras existentes en la UCI y la muestra en una nueva pantalla..
Prioridad:	Crítico
Prototipo de Interfaz:	

Tabla # 10: Descripción textual del Caso de Uso: Gestionar Computadoras

2.7.8. Descripción del Caso de Uso: Gestionar Componentes

Caso de Uso:	**Gestionar Componentes**
Caso de Uso Asociado	**Eliminar Componentes**
Actores:	Usuario (inicia)
Resumen:	El caso de uso comienza cuando un Usuario accede a las opciones de Adicionar, Modificar, o Mostrar las Componentes de la UCI. El caso de uso termina cuando el sistema realiza la operación elegida, y el Usuario accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF8
Precondiciones:	El usuario con el rol de Usuario se debe haber autenticado con anterioridad.
Poscondiciones:	1. Información del componente adicionado a la Base de Datos. 2. Información del componente modificado en la base de

	Datos. 3. Mostrar los componentes existentes.
Flujo Normal de Eventos	
Sección Adicionar Componente.	
Acción del Actor	**Respuesta del Sistema**
1. El Usuario selecciona la opción de Adicionar Componente.	2. El sistema muestra la interfaz para que el Usuario introduzca los datos del nueva Componente.
3. El Usuario ingresa los datos solicitados por el sistema y presiona el botón aceptar.	4. El sistema verifica que los campos del formulario no estén vacios.
	5. El sistema verifica que este componente no este registrado ya en la base de datos.
	6. El sistema registra los datos del nuevo componente y muestra el nuevo componente ya adicionado.
Flujos Alternos	
	4.1 Muestra un mensaje de error indicando que debe llenar todos los campos. (“Llene todos los campos”).

	5.1 El sistema muestra un mensaje de error indicando que ya existe este componente en la base de datos.
Sección Modificar Componente.	
1. El Usuario selecciona la opción Modificar Componente.	2. El sistema busca en la base de datos todos los datos de los Componentes de la UCI y las muestra en una pantalla para que el Usuario elija el Componente a modificar
3. El Usuario elije Componente a modificar y presiona el botón Modificar.	4. El sistema muestra una pantalla con todos los datos del Componente de forma editable.
5. El Usuario modifica los datos necesarios y pincha el botón Aceptar.	6. El sistema verifica que los campos modificados estén correctos.
	7. El sistema modifica todos lo datos del Componente seleccionado.
Flujos Alternos de los Eventos	
	6.1 El sistema muestra un mensaje de error indicando el problema.
Sección Mostrar Componentes	
1. El Usuario selecciona la opción Listar Componentes	2. El sistema busca todos los componentes existentes en la UCI y la

	muestra en una nueva pantalla.
Prioridad:	Crítico
Prototipo de Interfaz:	

Tabla # 11: Descripción textual del Caso de Uso: Gestionar Componentes

2.7.9. Descripción del Caso de Uso: Gestionar Medidas

Caso de Uso:	**Gestionar Medidas**
Caso de Uso Asociado	**Eliminar Medidas**
Actores:	Usuario (inicia)
Resumen:	El caso de uso comienza cuando un Usuario accede a las opciones de Adicionar, Modificar, o Mostrar las Medidas que se toman en la UCI. El caso de uso termina cuando el sistema realiza la operación elegida, y el Usuario accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF10
Precondiciones:	El usuario con el rol de Usuario se debe haber autenticado con anterioridad.
Poscondiciones:	1. Información de las medidas tomadas adicionada a la Base de Datos. 2. Información de las medidas modificada en la base de Datos. 3. Mostrar las medidas ya existentes.
Flujo Normal de Eventos	

Sección Adicionar Medidas	
Acción del Actor	**Respuesta del Sistema**
1. El Usuario selecciona la opción de Adicionar Medida	2. El sistema muestra la interfaz para que el Usuario introduzca la nueva medida.
3. El Usuario ingresa los datos solicitados por el sistema y presiona el botón aceptar.	4. El sistema verifica que los campos del formulario no estén vacios.
	5. El sistema verifica que medida no exista ya en el sistema
	6. El sistema registra la nueva medida
Flujos Alternos	
	4.1 Muestra un mensaje de error indicando que debe llenar todos los campos. (“Llene todos los campos”).
	5.1 El sistema muestra un mensaje de error indicando que ya existe esa medida en la base de datos.
Sección Modificar Medida	
1. El Usuario selecciona la opción Modificar Medida	2. El sistema busca en la base de datos todas las medidas y las muestra en una

	pantalla para que el Usuario elija la medida a modificar.
3. El Usuario elije la medida a modificar y presiona botón modificar.	4. El sistema muestra una pantalla con todos las medidas en forma editable.
5. El Usuario modifica los datos necesarios y pincha el botón Aceptar.	6. El sistema verifica que los campos modificados estén correctos.
	7. El sistema modifica la medida seleccionada.
Flujos Alternos de los Eventos	
	6.1 El sistema muestra un mensaje de error indicando el problema.
Sección Mostrar Medidas	
1. El Usuario selecciona la opción Listar Medidas	2. El sistema busca las medidas y lo muestra en una nueva pantalla se termina el caso de uso.
Prioridad:	Crítico
Prototipo de Interfaz:	

Tabla # 12: Descripción textual del Caso de Uso: Gestionar Medidas

2.7.10 Descripción del Caso de Uso: Generar Reporte

Caso de Uso:	Generar Reporte

Actores:	Usuario(inicia)
Resumen:	El caso de uso permite que los usuarios con determinadas responsabilidades en el sistema introduzcan los reportes diarios de la situación existente en los laboratorios de la UCI.
Referencia:	RF11
Precondiciones:	-
Poscondiciones:	El sistema permite que el usuario tenga los reportes diarios de todos los laboratorios de la UCI

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Usuario selecciona la opción de Generar Reporte	1. El sistema le muestra una interfaz con un formulario para que especifique el medio que se encuentra roto.
2. El usuario ingresa los datos en el formulario (rotura y tiempo).	3. El sistema verifica que los campos del formulario no estén vacio.
	4. Verifica la existencia del reporte en el sistema
	5. El sistema muestra la lista con los reportes con las roturas especificadas anteriormente por el usuario
Flujos Alternos	

Acción del Actor	Respuesta del Sistema
	3.1. Muestra un mensaje de error indicando que debe llenar todos los campos. (“Campos requeridos”).
Prioridad:	Secundario
Prototipo de Interfaz:	

Tabla # 13: Descripción textual del Caso de Uso: Generar Reportes

2.7.11 Descripción del Caso de Uso: Eliminar Proyecto

Caso de Uso:	**Eliminar Proyecto <<extend>>**
Actores:	Especialista GSD (inicia)
Resumen:	El caso de uso comienza cuando un Especialista GSD accede a la opción de Eliminar un proyecto del menú de operaciones. El caso de uso termina cuando el sistema devuelve la información solicitada, y el Especialista GSD accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF5
Precondiciones:	El usuario con el rol de Especialista GSD se debe haber autenticado con anterioridad.
Poscondiciones:	Información del proyecto eliminada de la base de Datos.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**

1. El Especialista GSD selecciona la opción Eliminar Proyecto Productivo.	2. El sistema busca todos los proyectos productivos existentes y los muestra en una pantalla para que el Especialista GSD elija el proyecto productivo a eliminar.
3. El Especialista GSD elige el proyecto productivo y pincha en el botón Eliminar.	4. El sistema muestra una confirmación para estar seguro que lo desea eliminar.
5. El Especialista GSD confirma que desea eliminar el proyecto.	6. El sistema procede a la eliminación de los datos del proyecto productivo seleccionado.
Flujos Alternos	
5.1. Si cancela la acción se culmina el caso de uso del sistema (CUS) sin ejecutar ninguna acción.	
Prioridad:	Secundario
Prototipo de Interfaz:	

Tabla # 14: Descripción textual del Caso de Uso: Eliminar Proyecto

2.7.12 Descripción del Caso de Uso: Eliminar Docente

Caso de Uso:	**Eliminar Docente <<extend>>**
Actores:	Especialista GSD (inicia)
Resumen:	El caso de uso comienza cuando un Especialista GSD accede a la opción de Eliminar docente del menú de operaciones. El caso de uso termina cuando el sistema devuelve la información solicitada, y el

	Especialista GSD accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF3
Precondiciones:	El usuario con el rol de Especialista GSD se debe haber autenticado con anterioridad.
Poscondiciones:	1. Información del docente eliminado de la base de Datos.

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Especialista GSD selecciona la opción Eliminar Docente.	2. El sistema busca todos los docentes existentes y los muestra en una pantalla para que el Especialista GSD elija el docente a eliminar.
3. El Especialista GSD elige el docente y pincha en el botón Eliminar.	4. El sistema muestra una confirmación para estar seguro que lo desea eliminar.
5. El Especialista GSD confirma que desea eliminar el docente.	6. El sistema procede a la eliminación de los datos del docente seleccionado y termina el CUS.
Flujos Alternos	
5.1. Si cancela la acción se culmina el Caso Uso Sistema sin ejecutar ninguna acción.	

Prioridad:	Secundario

Prototipo de Interfaz:	

Tabla # 15: Descripción textual del Caso de Uso: Eliminar Docente

2.7.13 Descripción del Caso de Uso: Eliminar Facultad

Caso de Uso:	**Eliminar Facultad <<extend>>**
Actores:	Especialista GSD (inicia)
Resumen:	El caso de uso comienza cuando un Especialista GSD accede a la opción de Eliminar una facultad del menú de operaciones. El caso de uso termina cuando el sistema devuelve la información solicitada, y el Especialista GSD accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF4
Precondiciones:	El usuario con el rol de Especialista GSD se debe haber autenticado con anterioridad.
Poscondiciones:	1. Información de la facultad eliminada de la base de Datos.

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Especialista GSD selecciona la opción Eliminar Facultad.	2. El sistema busca todas las facultades existentes y los muestra en una pantalla para que el Especialista GSD elija la facultad a eliminar.
3. El Especialista GSD elige la facultad y pincha en el botón	4. El sistema muestra una confirmación para estar seguro que lo desea eliminar.

Eliminar.	
5. El Especialista GSD confirma que desea eliminar la facultad.	6. El sistema procede a la eliminación de los datos de la facultad seleccionada y termina el CUS.
Flujos Alternos	
5.1. Si cancela la acción se culmina el CUS sin ejecutar ninguna acción.	
Prioridad:	Secundario
Prototipo de Interfaz:	

Tabla # 16: Descripción textual del Caso de Uso: Eliminar Facultad

2.7.14 Descripción del Caso de Uso: Eliminar Laboratorio

Caso de Uso:	**Eliminar Laboratorio <<extend>>**
Actores:	Especialista GSD (inicia)
Resumen:	El caso de uso comienza cuando un Especialista GSD accede a la opción de Eliminar un Laboratorio del menú de operaciones. El caso de uso termina cuando el sistema devuelve la información solicitada, y el Especialista GSD accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF6
Precondiciones:	El usuario con el rol de Especialista GSD se debe haber autenticado con anterioridad.
Poscondiciones:	1. Información del laboratorio eliminada de la base de Datos.

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Especialista GSD selecciona la opción Eliminar Laboratorio.	2. El sistema busca todos los laboratorios existentes y los muestra en una pantalla para que el Especialista GSD elija el laboratorio a eliminar.
3. El Especialista GSD elige el laboratorio y pincha en el botón Eliminar.	4. El sistema muestra una confirmación para estar seguro que lo desea eliminar.
5. El Especialista GSD confirma que desea eliminar el laboratorio.	6. El sistema procede a la eliminación de los datos del laboratorio seleccionado y termina el CUS.
Flujos Alternos	
5.1. Si cancela la acción se culmina el CUS sin ejecutar ninguna acción.	
Prioridad:	Secundario
Prototipo de Interfaz:	

Tabla # 17: Descripción textual del Caso de Uso: Eliminar Laboratorio

2.7.15 Descripción del Caso de Uso: Eliminar Solicitud

Caso de Uso:	**Eliminar Solicitud. <<extend>>**
Actores:	Especialista GSD (inicia)
Resumen:	El caso de uso comienza cuando un Especialista GSD accede a la

	opción de Eliminar una Solicitud del menú de operaciones. El caso de uso termina cuando el sistema devuelve la información solicitada, y el Especialista GSD accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF7
Precondiciones:	El usuario con el rol de Especialista GSD se debe haber autenticado con anterioridad.
Poscondiciones:	1. Información de la solicitud eliminada de la base de Datos.

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Especialista GSD selecciona la opción Eliminar Solicitud.	2. El sistema busca todas las solicitudes existentes y los muestra en una pantalla para que el Especialista GSD elija la solicitud a eliminar.
3. El Especialista GSD elige el solicitud y pincha en el botón Eliminar.	4. El sistema muestra una confirmación para estar seguro que lo desea eliminar.
5. El Especialista GSD confirma que desea eliminar la solicitud.	6. El sistema procede a la eliminación de los datos de la solicitud seleccionada y termina el CUS.
Flujos Alternos	
5.1. Si cancela la acción se culmina el CUS sin ejecutar ninguna acción.	

Prioridad:	Secundario
Prototipo de Interfaz:	

Tabla # 18: Descripción textual del Caso de Uso: Eliminar Solicitud

2.7.16 Descripción del Caso de Uso: Eliminar Componente

Caso de Uso:	**Eliminar Componente. <<extend>>**
Actores:	Especialista GSD (inicia)
Resumen:	El caso de uso comienza cuando un Especialista GSD accede a la opción de Eliminar un Componente del menú de operaciones. El caso de uso termina cuando el sistema devuelve la información solicitada, y el Especialista GSD accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF8
Precondiciones:	El usuario con el rol de Especialista GSD se debe haber autenticado con anterioridad.
Poscondiciones:	3. Información del componente eliminado de la base de Datos.

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Especialista GSD selecciona la opción Eliminar Componente.	2. El sistema busca todos los componentes existentes y los muestra en una pantalla para que el Especialista GSD elija el componente a eliminar.
3. El Especialista GS elige el	4. El sistema muestra una confirmación para

componente y pincha en el botón Eliminar.	estar seguro que lo desea eliminar.
5. El Especialista GSD confirma que desea eliminar el componente.	6. El sistema procede a la eliminación de los datos del componente seleccionado y termina el CUS.
Flujos Alternos	
5.1. Si cancela la acción se culmina el CUS sin ejecutar ninguna acción.	
Prioridad:	Secundario
Prototipo de Interfaz:	

Tabla # 19: Descripción textual del Caso de Uso: Eliminar Componente

2.7.17 Descripción del Caso de Uso: Eliminar Características

Caso de Uso:	**Eliminar Características. <<extend>>**
Actores:	Especialista GSD (inicia)
Resumen:	El caso de uso comienza cuando un Especialista GSD accede a la opción de Eliminar Característica del menú de operaciones. El caso de uso termina cuando el sistema devuelve la información solicitada, y el Especialista GSD accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF9
Precondiciones:	El usuario con el rol de Especialista GSD se debe haber autenticado con anterioridad.

Poscondiciones:	7. Información de la característica eliminada de la base de Datos.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Especialista GSD selecciona la opción Eliminar Característica.	2. El sistema busca todas las características existentes y los muestra en una pantalla para que el Especialista GSD elija la característica a eliminar.
3. El Especialista GSD elige la característica y pincha en el botón Eliminar.	4. El sistema muestra una confirmación para estar seguro que lo desea eliminar.
5. El Especialista GSD confirma que desea eliminar la característica.	6. El sistema procede a la eliminación de los datos de la característica seleccionada y termina el CUS.
Flujos Alternos	
5.1. Si cancela la acción se culmina el CUS sin ejecutar ninguna acción.	
Prioridad:	Secundario
Prototipo de Interfaz:	

Tabla # 20: Descripción textual del Caso de Uso: Eliminar Características

2.7.18 Descripción del Caso de Uso: Eliminar Medidas

Caso de Uso:	**Eliminar Medidas. <<extend>>**
Actores:	Especialista GSD (inicia)

Resumen:	El caso de uso comienza cuando un Especialista GSD accede a la opción de Eliminar Medida del menú de operaciones. El caso de uso termina cuando el sistema devuelve la información solicitada, y el Especialista GSD accede a otra opción o simplemente sale de la aplicación.
Referencia:	RF10
Precondiciones:	El usuario con el rol de Especialista GSD se debe haber autenticado con anterioridad.
Poscondiciones:	3. Información de la medida eliminada de la base de Datos.

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Especialista GSD selecciona la opción Eliminar Medida.	2. El sistema busca todas las medidas existentes y los muestra en una pantalla para que el Especialista GSD elija la medida a eliminar.
3. El Especialista GSD elige la medida y pincha en el botón Eliminar.	4. El sistema muestra una confirmación para estar seguro que lo desea eliminar.
5. El Especialista GSD confirma que desea eliminar la medida.	6. El sistema procede a la eliminación de los datos de la medida seleccionada y termina el CUS.
Flujos Alternos	
5.1. Si cancela la acción se culmina el CUS sin ejecutar ninguna acción.	

Prioridad:	Secundario
Prototipo de Interfaz:	

Tabla # 21: Descripción textual del Caso de Uso: Eliminar Medidas

2.8 Conclusiones

A partir de los Procesos identificados, se definieron los Casos de Uso del Negocio, modelando en un diagrama sus relaciones con los Actores, lo que brinda una visión global de los procesos a analizar.

Se identificaron los Actores y Trabajadores del Negocio, que resultan los involucrados en los procesos, de modo que se sientan las bases para la definición posterior de los Usuarios que interactuarán con el Sistema. Se definieron además las Reglas de Negocio.

CAPÍTULO 3: CONSTRUCCIÓN DE LA SOLUCIÓN PROPUESTA

3.1 Introducción.

El análisis de un sistema se centra en la investigación del problema y no en la manera de definir una solución, mientras que el diseño pone de relieve una solución lógica: cómo el sistema satisface los requerimientos funcionales, requerimientos de calidad y las restricciones, es decir, en esta parte del proceso de desarrollo del software se decide como se va a llevar a cabo el mismo.

En este capítulo se realiza el análisis y diseño de la propuesta de solución, a través de los flujos de Análisis y Diseño e Implementación, se presenta el diagrama de clases del análisis y del diseño de los diferentes casos de usos definidos en el capítulo anterior, a demás de los diagramas de interacción correspondientes a cada uno de ellos, se muestra el modelo de datos. Finalmente se realiza el modelo de despliegue donde se representan los nodos en los que se distribuye la aplicación y el de componentes para una mejor descripción de la solución propuesta.

3.2. Modelo de análisis.

El análisis se interesa por refinar y estructurar los requisitos funcionales, obteniendo de esta forma una visión el sistema. Se estructura por clases y paquetes estereotipados. El diagrama de clases es el diagrama principal de análisis para un sistema y se desarrolla buscando una solución ideal

En la construcción del Modelo de Análisis se tienen que identificar las clases que describen la realización de los casos de uso, los atributos y las relaciones entre ellas, es el resultado de la actividad de analizar los casos de uso. Entre las principales ventajas que aporta al proceso de desarrollo de un software es que suaviza la transición al diseño, sirve para tener una visión general de la propuesta de sistema, y entre otras cosas apoya el cambio a otra plataforma de programación puesto que no esta ligado a un lenguaje en particular.

El objetivo principal de este flujo son los diagramas de clases de análisis, los cuales muestran qué clases participan en las realizaciones de los distintos casos de usos, constituyendo la base sobre la cual debe realizarse el Modelo de Diseño.

3.3. Diagramas de Clases del Análisis.

Estos diagramas se encuentran en el Anexo 1

3.3.1 Diagramas de Interacción del análisis

La secuencia de acciones en un caso de uso comienza cuando un actor invoca el caso de uso mediante el envío de algún tipo de mensaje al sistema. Si consideramos el "interior" del sistema, un objeto de interfaz recibirá este mensaje del actor. El objeto de interfaz enviará a su vez un mensaje a algún otro objeto, y de esa forma los objetos implicados interactuarán para llevar a cabo el caso de uso (RUMBAUHG, 2000).

Los diagramas de interacción están constituidos por dos tipos: los diagramas de Colaboración y los diagramas de Secuencia. Ambos expresan información similar, pero en una forma diferente.

Diagrama de Secuencia

Muestra la interacción entre varios objetos y los enlaces que existen entre ellos. Representa las interacciones entre objetos organizadas alrededor de los objetos y sus vinculaciones.

Diagrama de Colaboración

Muestran las interacciones entre un conjunto de objetos, ordenadas según el Tiempo en que tienen lugar. Representa una forma de indicar el período durante el que un objeto está desarrollando una acción directamente o a través de un Procedimiento.

Estos diagramas de interacción se encuentran en el Anexo 2

3.4. Modelo de diseño

En la fase de diseño se modela el sistema de manera que soporte todos los requisitos, tanto funcionales como no funcionales, creándose así una entrada apropiada para las actividades de implementación.

El modelo de diseño es un modelo de objetos que describe la realización física de los casos de uso, centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar, constituyendo una entrada principal en la actividad de implementación (RUMBAUHG, 2000).

Para un mayor entendimiento del diseño de las clases realizado, es conveniente hacer una breve descripción del funcionamiento del framework que se utilizó, pues al estar el diseño completamente ligado al lenguaje de programación como ya se ha dicho, sin conocer como funciona al menos de manera general el Symfony es muy difícil poder entender el flujo que describen los diagramas.

Descripción general del funcionamiento del Symfony

Symfony organiza el código fuente en una estructura de tipo proyecto y almacena los archivos del proyecto en una estructura estandarizada de tipo árbol. Dentro de un proyecto, las operaciones se agrupan de forma lógica en aplicaciones. Cada aplicación a su vez está formada por uno o más módulos y un módulo normalmente representa a una página web o a un grupo de páginas con un propósito relacionado, los módulos además almacenan las acciones, que representan cada una de las operaciones que se puede realizar dentro de él.

Symfony toma lo mejor de la arquitectura MVC y la implementa de forma que el desarrollo de aplicaciones sea rápido y sencillo. En primer lugar, el controlador frontal y el layout son comunes para todas las acciones de la aplicación. Se pueden tener varios controladores y varios layouts, pero solamente es obligatorio tener uno de cada tipo. El controlador frontal es un componente que sólo tiene código relativo al MVC, por lo que no es necesario crear uno, ya que Symfony lo genera de forma automática, es además el único punto de entrada a la aplicación, carga la configuración y determina la acción a ejecutarse. Cuando el controlador frontal recibe una petición, utiliza el sistema de enrutamiento para asociar el nombre de una acción y el nombre de un módulo con la URL escrita (o pinchada) por el usuario.

Las acciones son el corazón de la aplicación, puesto que contienen toda la lógica de la misma. Utilizan el modelo y definen variables para la vista. Cuando se realiza una petición web en una aplicación Symfony, la URL define una acción y los parámetros de la petición. Son métodos con el nombre executeNombreAccion de una clase llamada nombreModuloActions que hereda de la clase sfActions y se encuentran agrupadas por módulos. La clase que representa las acciones de un módulo se encuentra en el archivo actions. class.php, en el directorio actions/ del módulo.

Symfony maneja automáticamente las sesiones del usuario y es capaz de almacenar datos de forma persistente entre peticiones. Utiliza el mecanismo de manejo de sesiones incluido en PHP y lo mejora para hacerlo más configurable y más fácil de usar. Antes de ser ejecutada, cada acción pasa por un filtro especial que verifica si el usuario actual tiene privilegios de acceder a la acción requerida.

Las clases de la capa del modelo también se generan automáticamente, en función de la estructura de datos de la aplicación. La librería Propel se encarga de esta generación automática, ya que crea el esqueleto o estructura básica de las clases y genera automáticamente el código necesario. Cuando Propel encuentra restricciones de claves foráneas (o externas) o cuando encuentra datos de tipo fecha, crea métodos especiales para acceder y modificar esos datos.

La abstracción de la base de datos es completamente invisible al programador, ya que la realiza otro componente específico llamado Creole. Así, si se cambia el sistema gestor de bases de datos en cualquier momento, no se debe reescribir ni una línea de código, ya que tan sólo es necesario modificar un parámetro en un archivo de configuración.

La vista se encarga de producir las páginas que se muestran como resultado de las acciones. La vista en Symfony está compuesta por diversas partes, estando cada una de ellas especialmente preparada para que pueda ser fácilmente modificable por la persona que normalmente trabaja con cada aspecto del diseño de las aplicaciones.

Normalmente se trabaja con las plantillas (que son la presentación de los datos de la acción que se está ejecutando) y con el layout (que contiene el código HTML común a todas las páginas). Estas partes están formadas por código HTML que contiene pequeños trozos de código PHP, que normalmente son llamadas a los diversos helpers disponibles. Los helpers son funciones de PHP que devuelven código HTML y que se utilizan en las plantillas.

El Layout, que también se denomina plantilla global, almacena el código HTML que es común a todas las páginas de la aplicación, para no tener que repetirlo en cada página. El contenido de la plantilla se integra en él, o si se mira desde el otro punto de vista, decora la plantilla. Este comportamiento es una implementación del patrón de diseño llamado “decorator”. Normalmente se utiliza el layout para mostrar la navegación, el logotipo del sitio, etc. Incluso es posible definir más de un layout y decidir en cada acción cuál se va a utilizar.

3.4.1. Diagramas de Clases de Diseño.

El diseño tiene en cuenta los requisitos no funcionales: cómo el sistema cumple sus objetivos. Se realiza para que el mismo se implemente sin ningún tipo de problemas, es decir, para satisfacer los detalles de la implementación. Estos diagramas se encuentran en el Anexo 3

3.5 Diseño de la Base de Datos.

El diseño de la base de datos modela el tratamiento de la información con carácter persistente dentro del sistema. Para este trabajo fueron construidos dos modelos para la representación de los datos persistentes: el Modelo Lógico de Datos y el Modelo Físico de Datos, ambos proporcionan una flexibilidad óptima para el soporte de la informatización entre el Modelo de Análisis, Modelo de Diseño, y la Base de Datos Física.

3.5.1 Modelo Lógico de datos (Diagrama de Clases Persistentes)

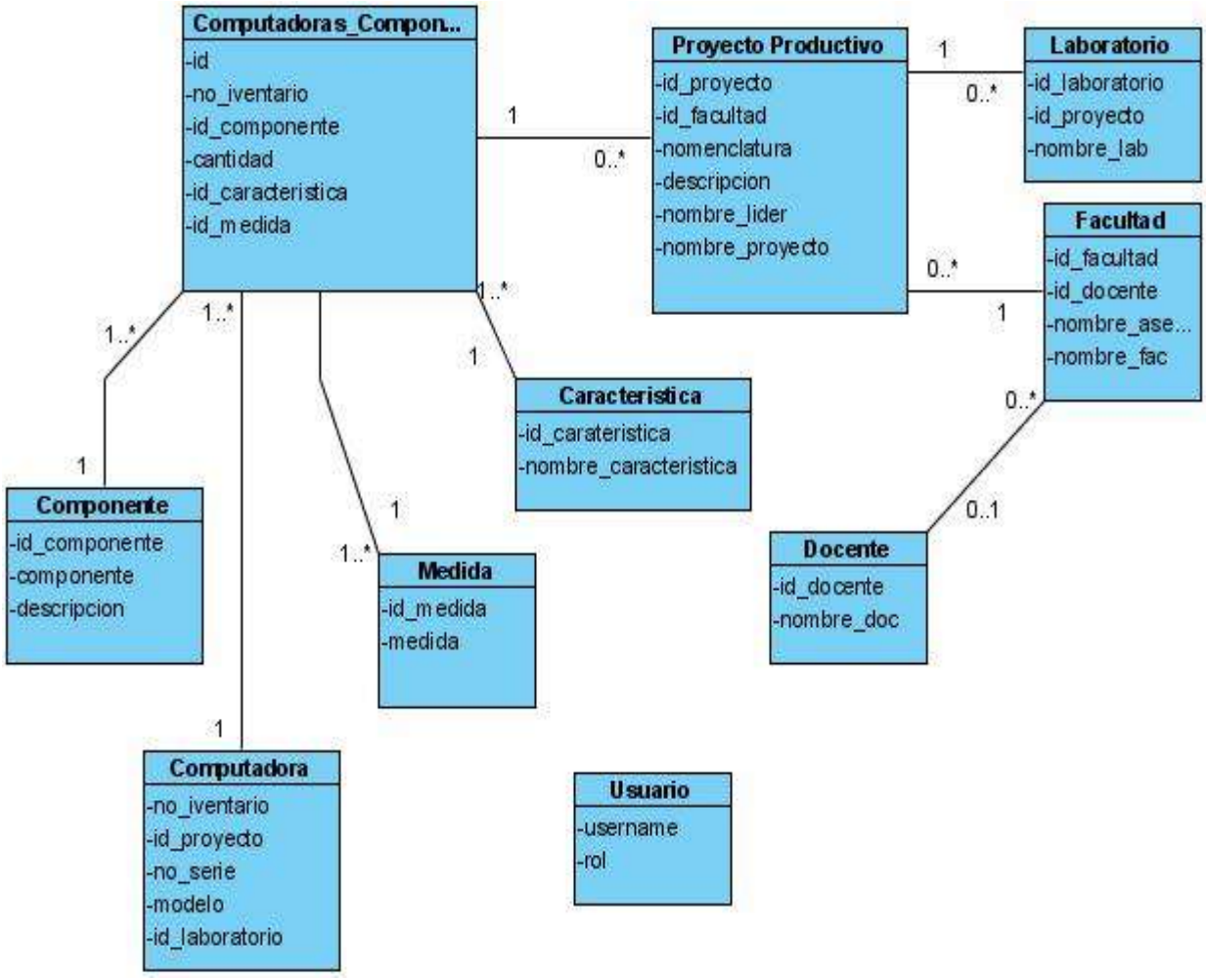


Fig. # 6: Diagrama de clases persistentes

3.5.2 Modelo Físico de datos (Diagrama Entidad- Relación)

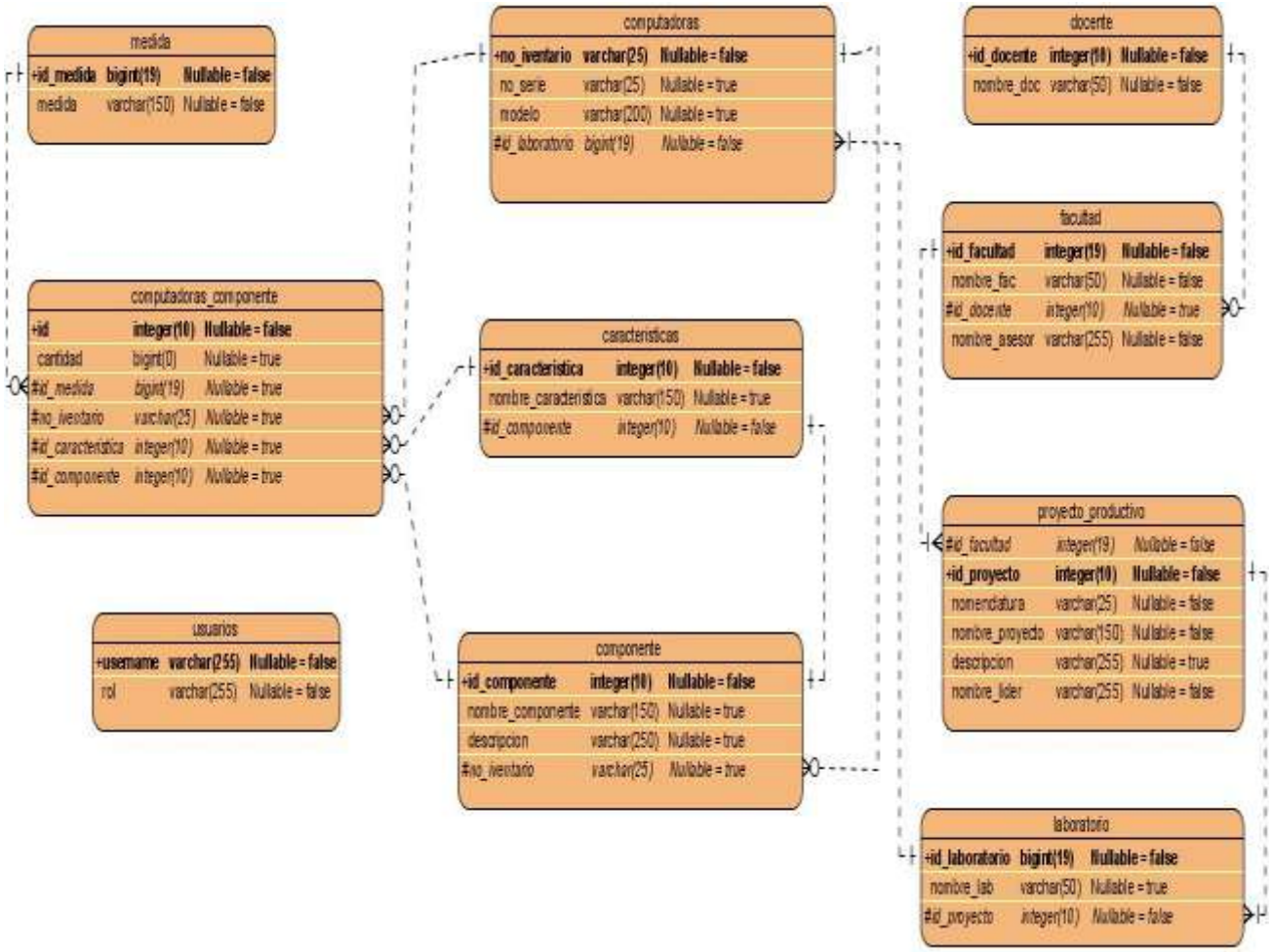


Fig. # 7 Diagrama Entidad- Relación

3.6 Modelo de Despliegue.

Diagrama de despliegue.

El diagrama de despliegue es un artefacto que modela la arquitectura en tiempo de ejecución de un sistema. En él se indica la situación física de los componentes lógicos desarrollados, lo que significa que sitúa el software en el hardware que lo contiene. A continuación se muestra el diagrama de despliegue, que representa la distribución física del sistema en términos de cómo se distribuirán las funcionalidades

entre los nodos, donde cada nodo representa un recurso de cómputo, siendo estos procesadores o dispositivos hardware que se necesitan para el despliegue del sistema:

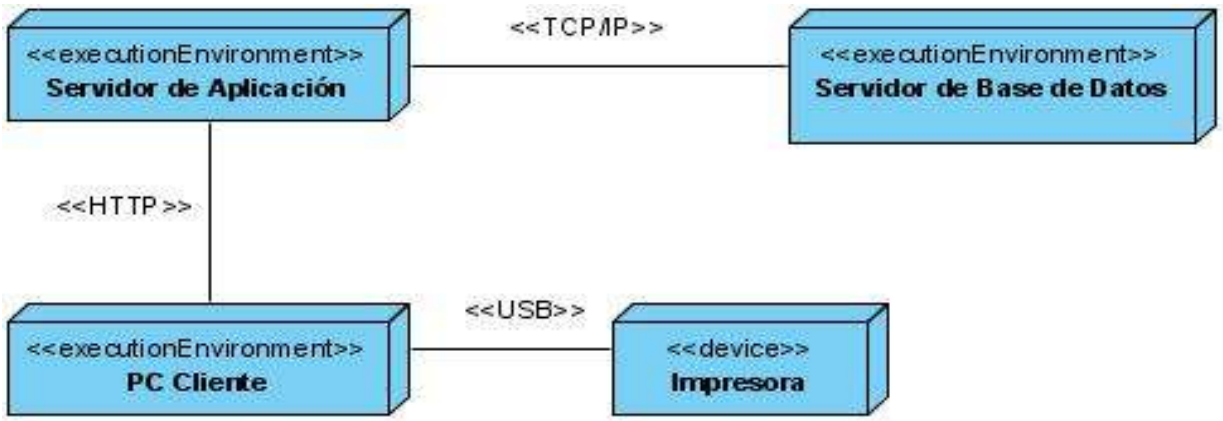


Fig. # 8 Diagrama de despliegue

3.7 Modelo de Implementación.

3.7.1 Diagrama de Componentes**.**

El diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes software, sean éstos componentes de código fuente, binarios o ejecutables. Desde el punto de vista del diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión del software, la reutilización, y las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo. Los elementos de modelado incluidos un diagrama de componentes serán componentes y paquetes.

El Diagrama de Componentes, en resumen: define cómo las clases, artefactos y otros elementos de bajo nivel, se unen para formar componentes de alto nivel y las conexiones entre ellos.

A continuación se muestra el diagrama de componentes correspondiente al sistema que se propone en la presente investigación.

Vista General:

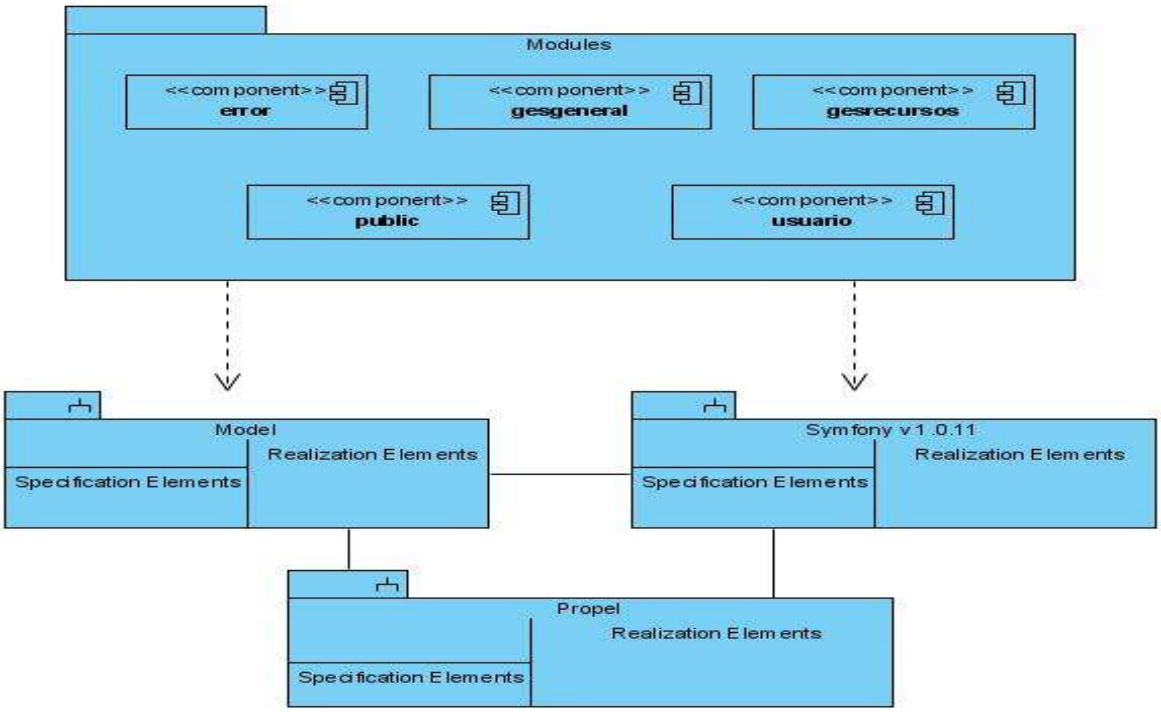


Fig. # 9: Diagrama de Componente (Vista General)

Para una mayor comprensión del Diagrama de Componentes, se expondrá una vista detallada de los paquetes Model y Módulo.

Vista Detallada: Paquete Model.

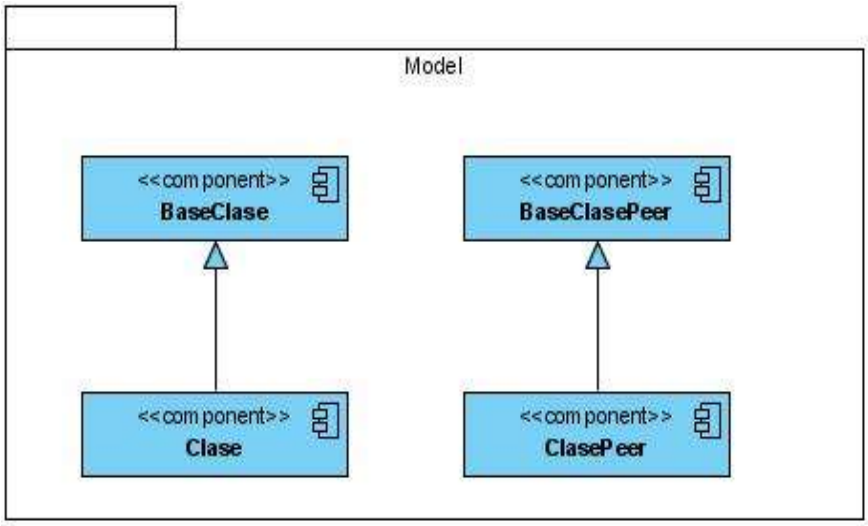


Fig.# 10: Vista Detallada: Paquete Model

Vista Detallada Paquete Modules

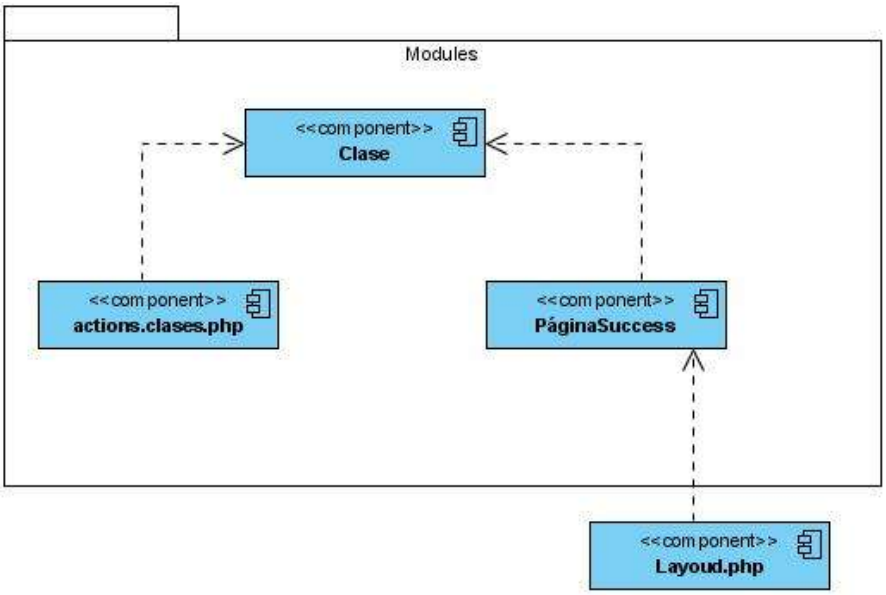


Fig.# 11: Vista Detallada: Paquete Modules

3.8 Conclusiones

En este capítulo fueron modelados los artefactos que tienen lugar en los flujos de trabajo de análisis, diseño e implementación, sustentándose de esta forma la base para la construcción del sistema.

CAPÍTULO 4 ESTUDIO DE FACTIBILIDAD

4.1 Introducción

Desde los primeros momentos del desarrollo de un software, resulta necesario determinar si el mismo resultará factible o no. Para ello se debe realizar un estudio detallado de los beneficios que este aporta y de las inversiones que implicará tanto en la esfera organizativa (entiéndase estructuras, procesos y personas), como en la económica y técnica (teniendo en cuenta habilidades, experiencias, recursos), para llevar a cabo su implementación. Este estudio incluye una planificación del trabajo a realizar referente al tiempo que demorará el desarrollo del software a partir de la cantidad de personas requeridas y del tamaño del mismo.

Este capítulo está dedicado al estudio de factibilidad. Se abordan varios aspectos que permiten determinar la viabilidad o factibilidad del desarrollo del sistema propuesto. Se realiza la estimación de costo del proyecto, se plantean los beneficios tangibles e intangibles que reportaría la aplicación y se realiza el análisis de costo y beneficio.

4.2 Planificación

Uno de los objetivos de la planificación es lograr estimaciones razonables. Actualmente existen varias técnicas para la estimación de costos, recursos y tiempo relacionados al desarrollo del software. La técnica de estimación utilizada en este caso es la denominada **Análisis de Puntos de Casos de Uso**. Este método se basa en la posibilidad de predecir el tamaño de un sistema a partir de las características de sus requisitos, expresados en los casos de uso. Debe aclararse que esta técnica no concibe los gastos logísticos y por conceptos de compras de licencias, etc., que deben ser sumados para que se logre obtener una estimación más exacta.

4.2.1 Estimación basada en casos de uso

Es un método de estimación de esfuerzo de un proyecto de desarrollo de software a partir de los casos de uso.

El método utiliza los actores y casos de uso identificados para calcular el esfuerzo que constará desarrollarlos. A los casos de uso se les asigna una complejidad basada en transacciones, que son pares

de pasos acción-usuario/respuesta-sistema de los escenarios de los casos de uso. A los actores se les asigna una complejidad basada en el tipo de actor, es decir, si son interfaces con usuarios o si son interfaces con otros sistemas. También se utilizan factores de entorno y de complejidad técnica para afinar el resultado.

Una vez asignada complejidad a actores y casos de uso y establecidos los factores técnicos y de entorno, se calculan los puntos de caso de uso no ajustados o UUCP, el TCF (factor de complejidad técnica) y el EF (factor del entorno). Con ellos, se calculan los puntos de caso de uso o UCP, que finalmente se traducen a esfuerzo en horas-hombre con un sencillo cálculo. Con el objetivo de realizar una buena estimación se llevan a cabo una serie de pasos fundamentales como son:

- Cálculo de los Puntos de Caso de Uso sin ajustar.
- Cálculo de los Puntos de Caso de Uso ajustados.
- Estimación del esfuerzo a través de los puntos de caso de uso.
- Cálculo del costo del proyecto.

Paso 1: Calcular los Puntos de Caso de Uso sin ajustar (UUCP)

UUCP = UAW + UUCW

Donde:

- UUCP: Puntos de Casos de Uso sin ajustar
- UAW: Factor de Peso de los Actores sin ajustar
- UUCW: Factor de Peso de los Casos de Uso sin ajustar Para calcular UAW:

Tipo de actor	**Descripción**	**Factor de peso**	**Actores**	**Total**

Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación. (API, Application Interface Programming)	1	0	0
Medio	Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto.	2	0	0
Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica.	3	1	3

$$UAW = \sum cant\ actores * peso$$

UAW=3

Tabla # 22: Factor de peso de los actores sin ajustar

Tipo de CU	Descripción	Peso	Cantidad de CU	Total
Simple	El caso de uso tiene de 1 a 3 transacciones.	5	8	40
Medio	El caso de uso tiene de 4 a 7 transacciones.	10	2	20
Complejo	El caso de uso tiene más de 8 transacciones.	1	8	120

Tabla # 23: Factor de peso de los casos de uso sin ajustar

Finalmente los Puntos de Caso de Uso sin ajustar son:

$$UUCW = \sum cant\ CU * Peso$$

UUCW=180

UUCP = UAW + UUCW

UUCP=3+180

UUCP=183

Paso 2. Cálculo de los Puntos de casos de uso ajustados.

UCP=UUCP*TCF*EF

Donde:

- UCP: Puntos de casos de uso ajustados.
- UUCP: Puntos de casos de uso sin ajustar.
- TCP: Factor de complejidad técnica.
- EF: Factor de ambiente.

El factor de complejidad técnica (TCF) se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada factor se cuantifica en un valor desde 0 (aporte irrelevante) hasta 5 (aporte muy relevante) (Pressman, 2005).

Factor	Descripción	Peso	Valor asignado	Total
T1	Sistema distribuido	2	4	8
T2	Tiempo de respuesta	1	4	4
T3	Eficiencia del usuario final	1	4	4
T4	Funcionamiento Interno complejo	1	5	5
T5	El código debe ser reutilizable	1	4	4
T6	Facilidad de instalación	0.5	5	2.5

T7	Facilidad de uso	0.5	4	2
T8	Portabilidad	2	4	8
T9	Facilidad de cambio	1	4	4
T10	Concurrencia	1	5	5
T11	Incluye objetivos especiales de seguridad	1	4	4
T12	Provee acceso directo a terceras partes	1	0	0
T13	Se requieren facilidades especiales de entrenamiento de usuarios	1	2	2

Tabla # 24: Factor de complejidad técnica.

$$TCF = 0.6 + 0.01 * \sum(peso * valor\ asignado)$$

TCF = 0.6 + 0.01 * 52.5

TCF = 0.6 + 0.525

TCF=1.125

El factor de ambiente (EF) está relacionado con las habilidades y entrenamiento del grupo de desarrollo que realiza el sistema. Cada factor se cuantifica con un valor desde 0 (aporte irrelevante) hasta 5 (aporte muy relevante) (Pressman, 2005).

Factor	**Descripción**	**Peso**	**Valor asignado**	**Total**
E1	Familiaridad con el modelo de proyecto utilizado	1.5	4	6
E2	Experiencia en la aplicación	0.5	5	2.5
E3	Experiencia en la orientación a objetos.	1	4	4
E4	Capacidad del analista líder.	0.5	4	2

E5	Motivación.	1	5	5
E6	Estabilidad de requerimientos	2	5	10
E7	Personal Part–Time	-1	3	-3
E8	Dificultad del lenguaje de programación	-1	3	-3

Tabla # 25: Factor de ambiente

$$EF = 1.4 - 0.03 * \sum (peso * valor\ asignado)$$

EF = 1.4 – 0.03 * 23.5

EF = 1.4 – 0.705

EF=0.695

Finalmente los Puntos de Caso de Uso ajustados son:

UCP = UUCP * TCF * EF

UCP = 183 *1.125 * 0.695

UCP= 143.0831

Paso 3: Estimación del esfuerzo a través de los puntos de caso de uso

E=UCP*CF

Donde:

- E: Esfuerzo estimado en horas hombres.
- UCP: Punto de casos de usos ajustados.
- CF: Factor de conversión.

Para obtener el factor de conversión (CF) se cuentan cuantos valores de los que afectan el factor ambiente (E1...E6) están por debajo de la media (3), y los que están por arriba de la media para los restantes (E7, E8). Si el total es 2 o menos se utiliza el factor de conversión 20 Horas-Hombre / Punto de Casos de uso. Si el total es 3 o 4 se utiliza el factor de conversión 28 Horas-Hombre / Punto de Casos de

uso. Si el total es mayor o igual que 5 se recomienda efectuar cambios en el proyecto ya que se considera que el riesgo de fracaso del mismo es demasiado alto (Pressman, 2005).

En este caso se puede decir que:

CF = 20 Horas-Hombre / Punto de Casos de uso.

E=143.0831*20

E=2861.6625 Horas-Hombre

Paso 4. Calcular esfuerzo de todo el proyecto.

Actividad	**Porcentaje %**	**Horas-Hombres**
Análisis	10	786.9875
Diseño	20	1618.48125
Implementación	40	2861.6625
Pruebas	15	1157.975
Sobrecarga (otras actividades)	15	1157.975
Total	100	7583.08125

Tabla # 26: Cálculo del esfuerzo

Si E_T = 7583.08125 horas-hombre y cada mes los desarrolladores trabajan como promedio 240 horas, eso daría un:

E_T = 31.5961 mes-hombre.

Esto quiere decir que 1 persona puede realizar el problema analizado en 32 meses aproximadamente.

Paso 5. Costo del proyecto.

Se asume como salario promedio mensual $100.00

CHM = 100 * Salario Promedio

CHM = 100.00 $/mes

Costo = Salario Promedio* E_T

Costo = 100.00 * 31.5961

Costo = $3159.61

4.3 Beneficios tangibles e intangibles

El desarrollo e implantación del software propuesto trae a la Universidad de las Ciencias Informáticas una serie de beneficios importantes debido a que se disminuye en gran medida el trabajo humano con relación al método de trabajo que tienen actualmente, esto trae aparejado una disminución de errores cometidos en la entrada de datos, así como repeticiones de los mismos al no tener un sistema informatizado que controle de manera eficiente las actividades.

Además permite mantener el control detallado y organizado de los recursos y de los usuarios que hacen uso de ellas para la producción, proporcionará un punto de partida para brindar seguridad a su información. Brinda también la posibilidad de conocer las actualizaciones que se realizan en los inventarios, sin necesidad de inspeccionar cada uno, dejando un mínimo de esfuerzo para los trabajadores de la entidad. Proporciona una descongestión en la red, debido a que el uso que se hace actualmente del correo electrónico cambiará, no utilizándose ya como medio de envío y recepción de los inventarios, y sí como medio de información y comunicación entre trabajadores de la entidad.

4.4 Análisis de costo y beneficios

El desarrollo de la aplicación no constituye un gasto considerable pues todas las herramientas que se han empleado en su desarrollo son libres y de código abierto. El sistema está orientado al trabajador, es de fácil aprendizaje debido a que uno de los objetivos planteados fue no romper con el esquema de trabajo que tienen en la actualidad, debido a esto no reporta grandes gastos por concepto de entrenamiento de los trabajadores del negocio. No son necesarios los gastos por concepto de tecnología pues la Infraestructura Productiva cuenta con la infraestructura necesaria para la implantación de la aplicación. Por todo lo antes planteado se decide que es factible desarrollar un sistema que proporcione estas ventajas.

4.5 Conclusiones

En este capítulo se describió el estudio de factibilidad realizado correspondiente al sistema propuesto, teniendo en cuenta el costo estimado y los beneficios que reportará al ser implantado. El estudio realizado, ha proporcionado valiosos argumentos que permiten llegar a la conclusión de que la solución de software, es factible pues reportará importantes beneficios sin incurrir en mayores gastos.

CONCLUSIONES

Después de haber culminado esta investigación quedó informatizado el proceso de Gestión de Recursos Informáticos con lo establecido en los documentos rectores y los requerimientos de los usuarios, alcanzando de esta manera el objetivo propuesto y dándole solución al problema planteado. Los resultados de este trabajo y la utilización de este sistema serán de gran utilidad para la Universidad teniendo en cuenta que se eliminará el trabajo manual que hasta el momento se estaba realizando. Proveerá una mayor confiabilidad en la información obtenida y mejorará el funcionamiento y la organización de este proceso.

RECOMENDACIONES

Se recomienda que este trabajo de diploma sea considerado como material de consulta por las personas que deban realizar un sistema similar o que la teoría que se ha expuesto en este trabajo le sirva de apoyo en futuros proyectos.

Mantener una actualización periódica sobre el sistema, logrando así que se mantenga la fiabilidad y funcionamiento óptimo del mismo así como la integridad de la información que se gestiona a través de él.

Continuar trabajando en el sistema con el propósito de hacerlo totalmente funcional, y que se le puedan adicionar otras funcionalidades de importancia estratégica para el grupo empresarial.

GLOSARIO DE TÉRMINOS

Apache: Servidor de páginas Web de código abierto para diferentes plataformas (UNIX, Windows, etc.)

CSS (Cascading Style Sheets): Las hojas de estilo en cascada contienen un conjunto de etiquetas que definen el formato que se aplicará al contenido de las páginas de una Web. Se llaman "en cascada" porque una hoja puede heredar los formatos definidos en otra hoja de forma que no hace falta que vuelva a definirlos. Estas hojas permiten la separación entre el contenido y la presentación en una Web.

Framework: Es una estructura de soporte definida, en la cual otro proyecto de software puede ser organizado y desarrollado. Es un patrón de la arquitectura que proporciona una plantilla extensible para aplicaciones dentro de un dominio específico.

HTML (Hypertext Markup Language): Lenguaje basado en marcas que indican las características del texto, utilizado para definir documentos de hipertexto en Web.

HTTP (Hypertext Transfer Protocolo): Protocolo cliente-servidor utilizado para el intercambio de páginas Web (HTML).

JavaScript: Es un lenguaje interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas Web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C.

Negocio: Cualquier ambiente o entorno en cual esta enmarcado el problema

Open Source: Código abierto o código libre. Software que distribuye de forma libre su código fuente, de forma que los desarrolladores pueden hacer variaciones, mejoras o reutilizarlo en otras aplicaciones. También conocido como free software.

PHP (Hypertext Preprocessor): Lenguaje de programación para el desarrollo de webs dinámicas, con sintaxis parecida a C. Originalmente se conocía como Personal Hombre Page tools, herramientas para páginas personales (en Internet).

Proceso del Negocio: Funciones que se desarrollan en el ambiente o entorno que se define como negocio.

Proyecto: Acciones y recursos que permiten en un tiempo dado el logro de un resultado para el cual fue concebido, un Proyecto definido puede durar más de un año o período fiscal.

Recursos: Son los elementos del ordenador que utilizan los dispositivos para poder funcionar correctamente.

Sistema Informático: Un sistema informático como todo sistema, es el conjunto de partes interrelacionadas, hardware, software y de Recurso Humano.

Software: Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

BIBLIOGRAFÍA CONSULTADA

Castro, Fidel. 2003. Discurso dirigido a los estudiantes de la UCI. *Granma.* 2003.

Development, S. L.JS. SOLUCIONES INFORMATICAS PARA LA EMPRESA. [En línea] http://www.jsd.es/software-erp/3-47-48-47.htm.

E.Kendall, Kenneth. 1997. *Análisis y diseño de sistemas.* México : s.n., 1997. 968-880-694-3.

ERPs Open Source. [En línea] [Citado el: 2009 de Mayo de 10.] http://edays.netau.net/2009/04/09/erp-privativos-erp-opensource/.

FERNÁNDEZ, ADOLFO.R DE SOTO and E. C. 2006. *Nuevas Tendencias en Sistema de información.* 2006. págs. 129-158.

Fernando González Ladrón de Guevara, Alba Soraya Aguilar Jiménez, Jorge Moreno Cantón, Ignacio Miralles Terol, Javier Muñoz Giner. 2008. *Análisis de sistemas ERP software libre.* España : s.n., 2008.

Group, P. G. 1996 – 2008. [En línea] 1996 – 2008. [Citado el: 14 de marzo de 2009.] www.postgresql.org/docs/manuals/.

Hispalinux. (1999-2002). Modelado de Sistemas com UML. [En línea] (1999-2002). http://www.ibiblio.org/pub/linux/docs/LuCaS/Tutoriales/doc-modelado-sistemas-UML/multiple-html/x219.html.

J.B.López. 2005.. *Un Estudio Comparativo de Herramietas para el Modelado con UML.* Colombia : s.n., 2005.

Jacobson, James, y otros. 2000. *El Lenguaje Unificado de Modelado.* Madrid : s.n., 2000. pág. pág. 526. Vol. Manual de Referencia. 84-7829-037-0.

Lidia y Vallecillo Fuentes, Antonio. 2003.. *Una Introducción a los Perfiles UML.* España : s.n., 2003.

MaestrosdelWeb. 2008. MaestrosdelWeb. [En línea] 2008. http://www.maestrosdelweb.com/principiantes/%C2%BFque-son-las-bases-de-datos/.

Paradigm, Visual. 2009. Visual Paradigm. [En línea] 2009. [Citado el: 10 de enero de 2009.] http://www.visual-paradigm.com/product/vpuml/..

Porteiro, Marisel Sosa. 2009. El VERSAT-Sarasola: Sistema cubano de Gestión Contable-Financiero. [En línea] 2009. [Citado el: 16 de enero de 2009.] http://www.betsime.disaic.cu/secciones/eco_enemar_07.htm#2..

Pressman, Roger S. 2005. *Ingeniería del Software. Un enfoque práctico.* 2005.

RUMBAUHG, J. 2000. *El Lenguaje Unificado de Modelado. Manual de referencia.* Madrid : s.n., 2000.

Sanchez, María A. Mendoza. 2004. *Metodologías De Desarrollo De Software.* Perú : s.n., 2004.

Torres, Jose Luis. 2009. Especificación de requisitos en Ingeniería de Software. [En línea] 2009. http://www.uag.mx/ieee/contsep01/requerimientos.htm.

VEGAS, J. 2005, 2006. .Desarrollo de Aplicaciones Web . [En línea] 2005, 2006. http:/www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node17/.

BIBLIOGRAFÍA REFERENCIADA

Castro, Fidel. 2003. Discurso dirigido a los estudiantes de la UCI. *Granma.* 2003.

J.B.López. 2005.. *Un Estudio Comparativo de Herramietas para el Modelado con UML.* Colombia : s.n., 2005.

Jacobson, James, y otros. 2000. *El Lenguaje Unificado de Modelado.* Madrid : s.n., 2000. pág. pág. 526. Vol. Manual de Referencia. 84-7829-037-0.

Lidia y Vallecillo Fuentes, Antonio. 2003.. *Una Introducción a los Perfiles UML.* España : s.n., 2003.

MaestrosdelWeb. 2008. MaestrosdelWeb. [En línea] 2008. http://www.maestrosdelweb.com/principiantes/%C2%BFque-son-las-bases-de-datos/.

Pressman, Roger S. 2005. *Ingeniería del Software. Un enfoque práctico.* 2005.

RUMBAUHG, J. 2000. *El Lenguaje Unificado de Modelado. Manual de referencia.* Madrid : s.n., 2000.

Torres, Jose Luis. 2009. Especificación de requisitos en Ingeniería de Software. [En línea] 2009. http://www.uag.mx/ieee/contsep01/requerimientos.htm.

ANEXOS

Anexo 1. Diagramas de Clases del Análisis

Diagrama de Clases del Análisis del Caso de Uso: Autenticar Usuario.

Diagrama de Clases del Análisis del Caso de Uso: Gestionar Usuario

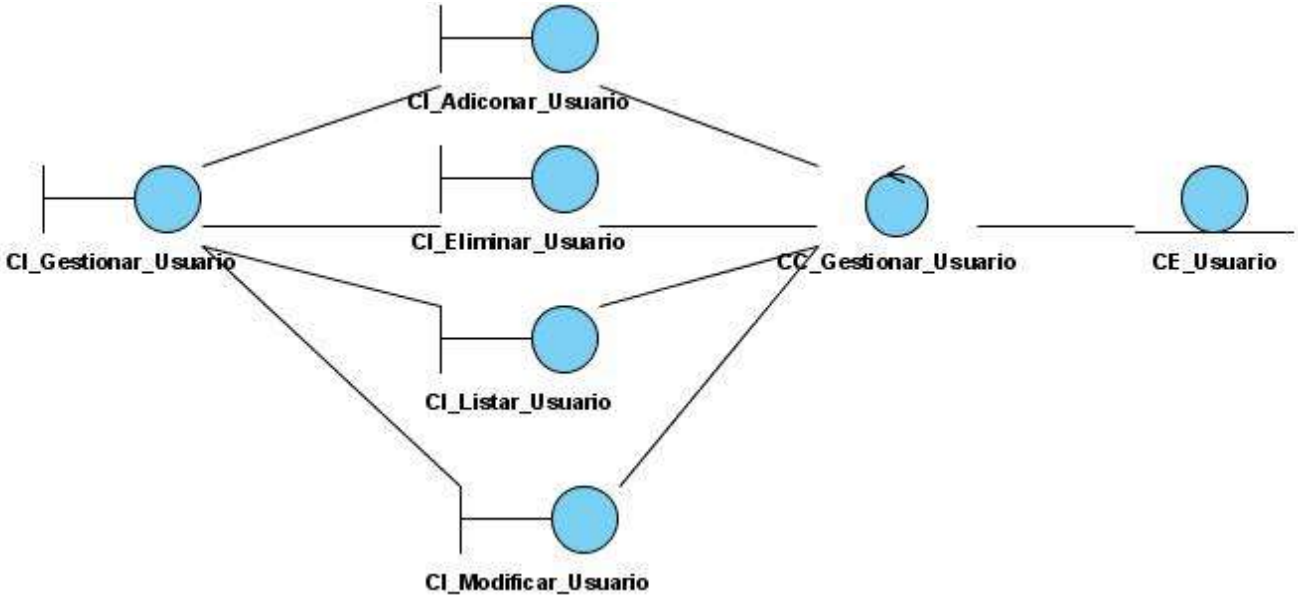


Diagrama de Clases del Análisis del Caso de Uso: Gestionar Docente

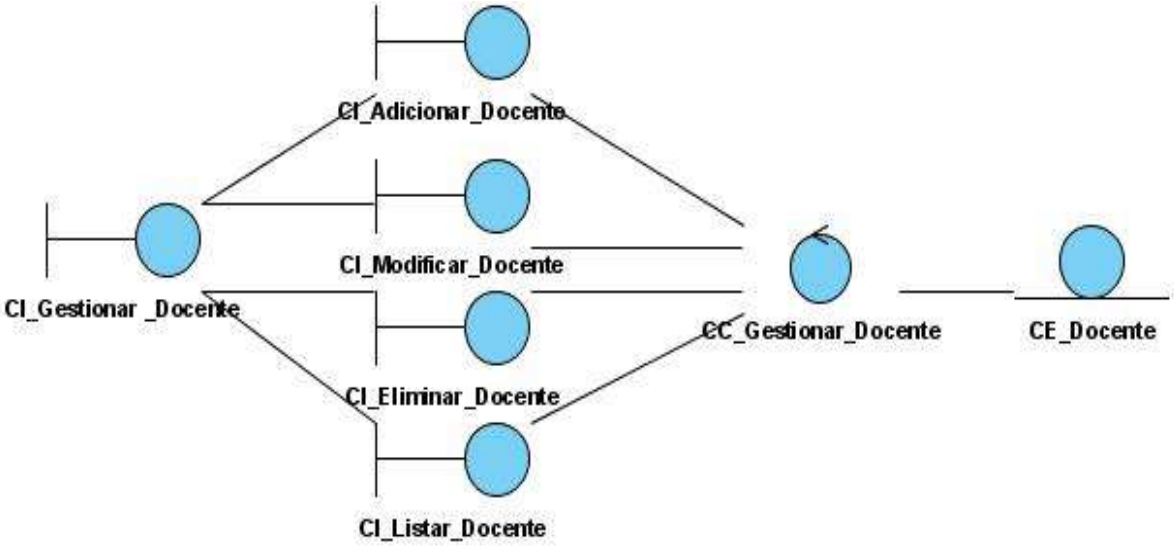


Diagrama de Clases del Análisis del Caso de Uso: Gestionar Facultad

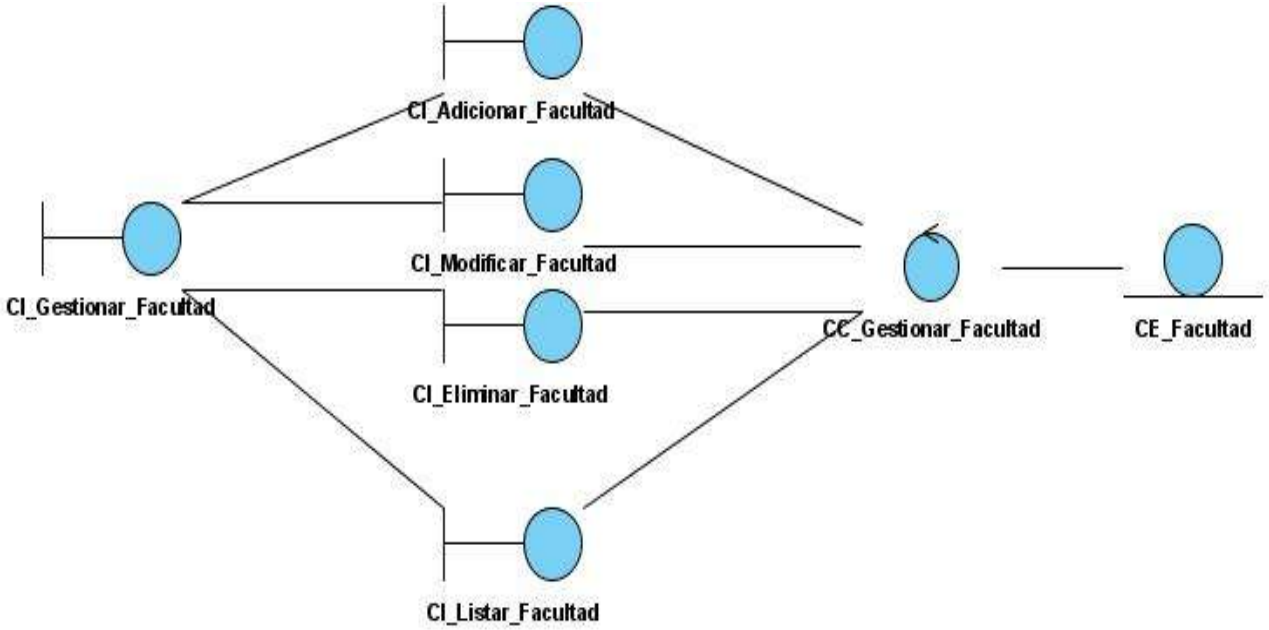


Diagrama de Clases del Análisis del Caso de Uso: Gestionar Productivo.

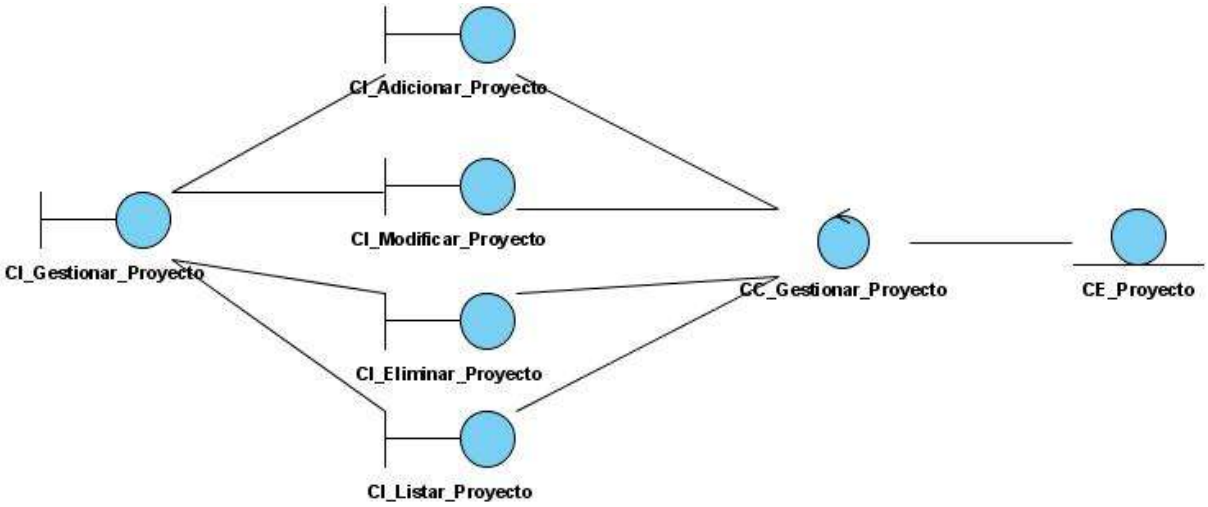


Diagrama de Clases del Análisis del Caso de Uso: Gestionar Laboratorio

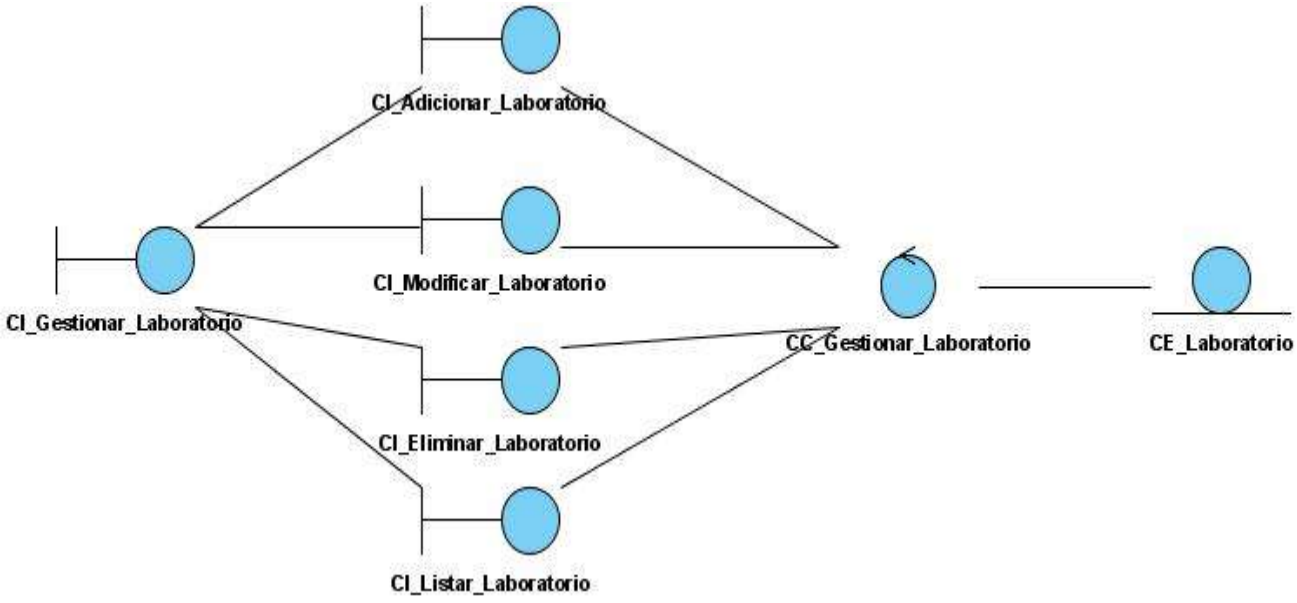


Diagrama de Clases del Análisis del Caso de Uso: Gestionar Características

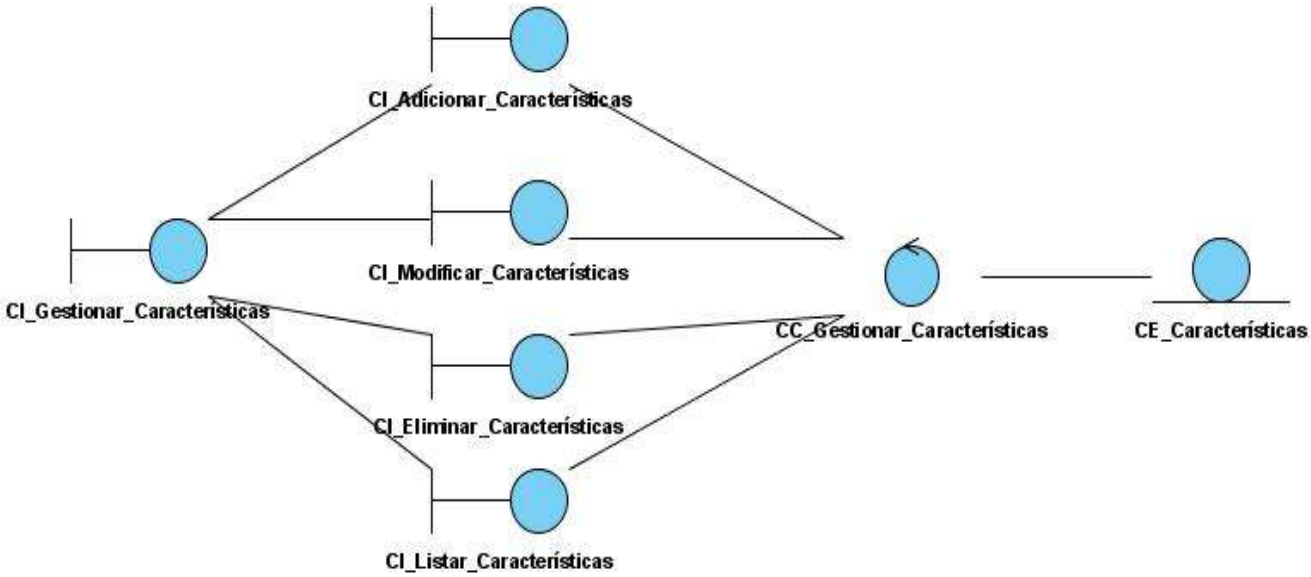


Diagrama de Clases del Análisis del Caso de Uso: Gestionar Componente

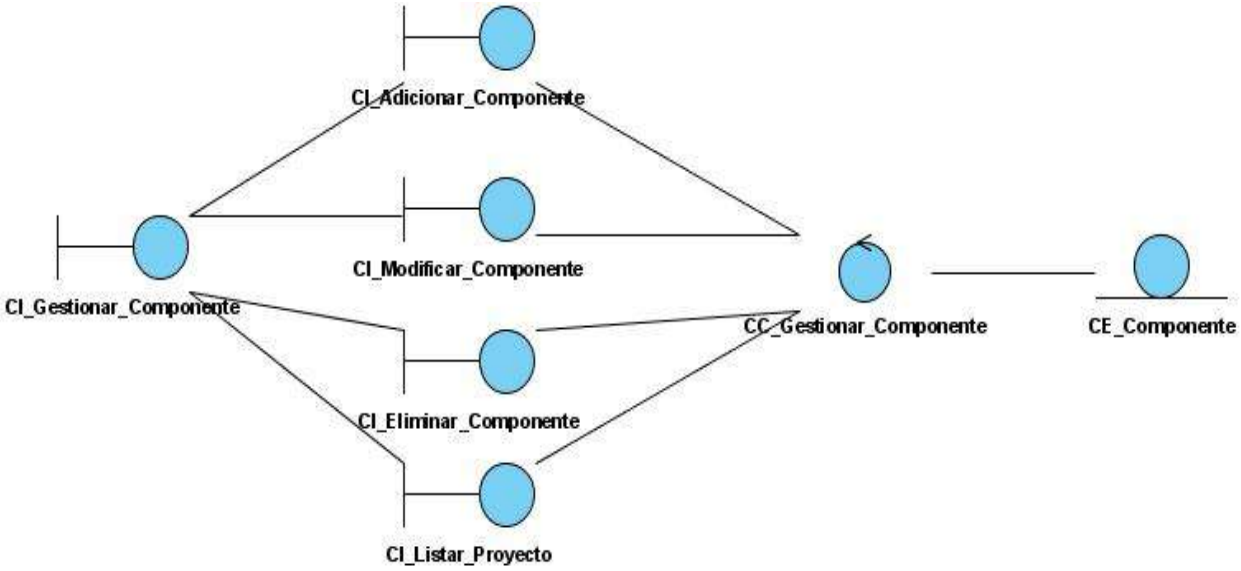


Diagrama de Clases del Análisis del Caso de Uso: Gestionar Medidas

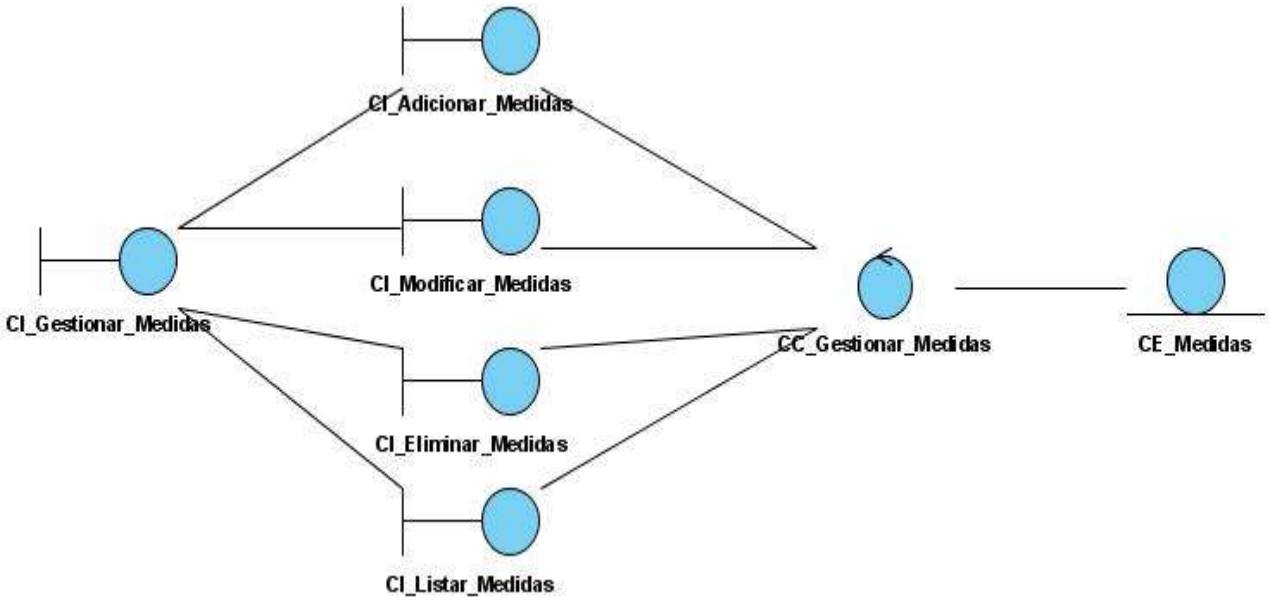


Diagrama de Clases del Análisis del Caso de Uso: Gestionar Computadoras

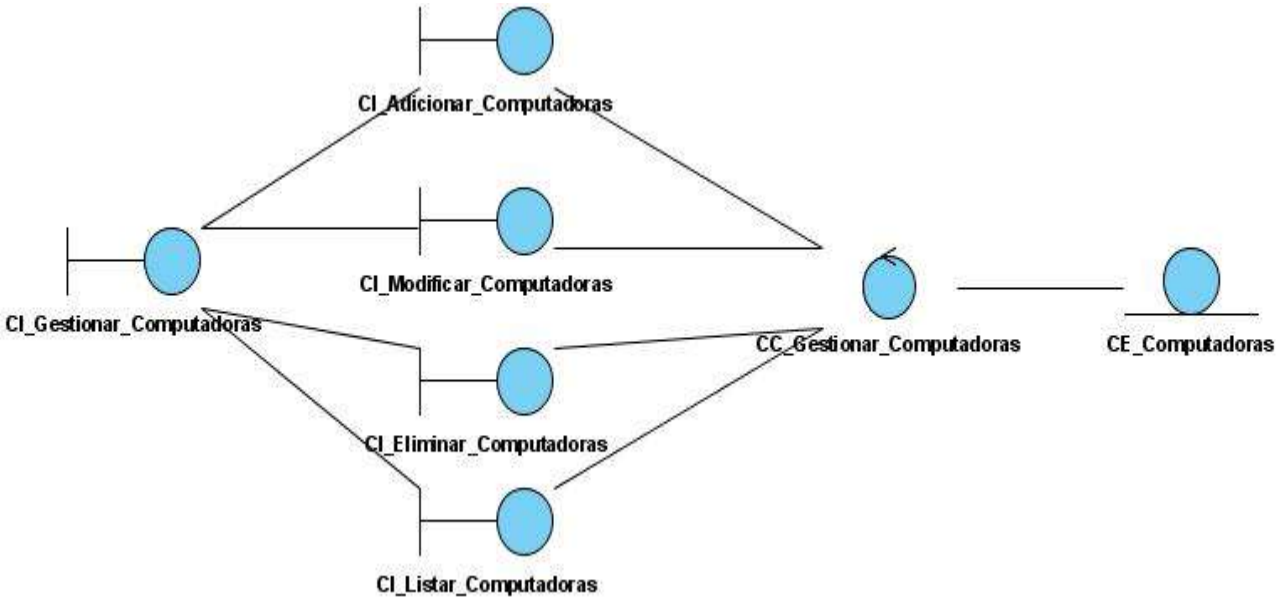
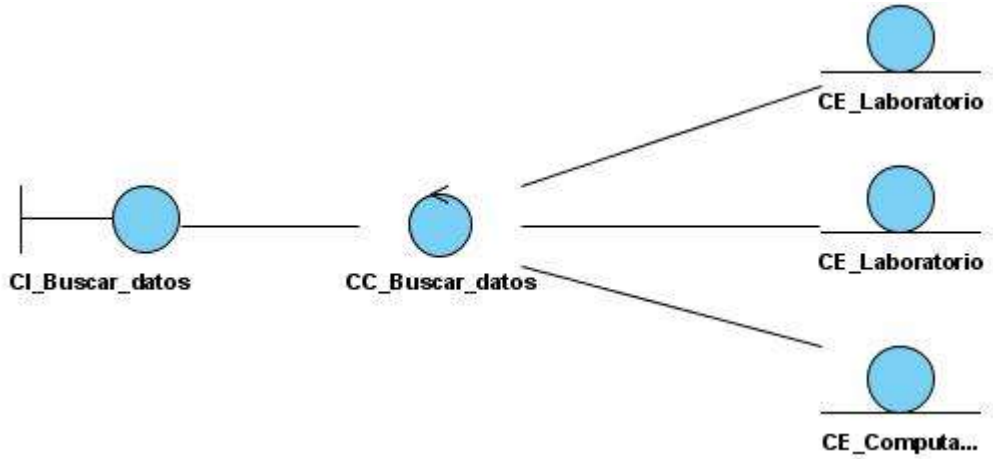


Diagrama de Clases del Análisis del Caso de Uso: Gestionar Reporte

Diagrama de Clases del Análisis del Caso de Uso: Realizar Búsqueda de datos a nivel de universidad.



Anexo 2.Diagramas de Interacción del Análisis.

Diagramas de Colaboración: Autenticar Usuario

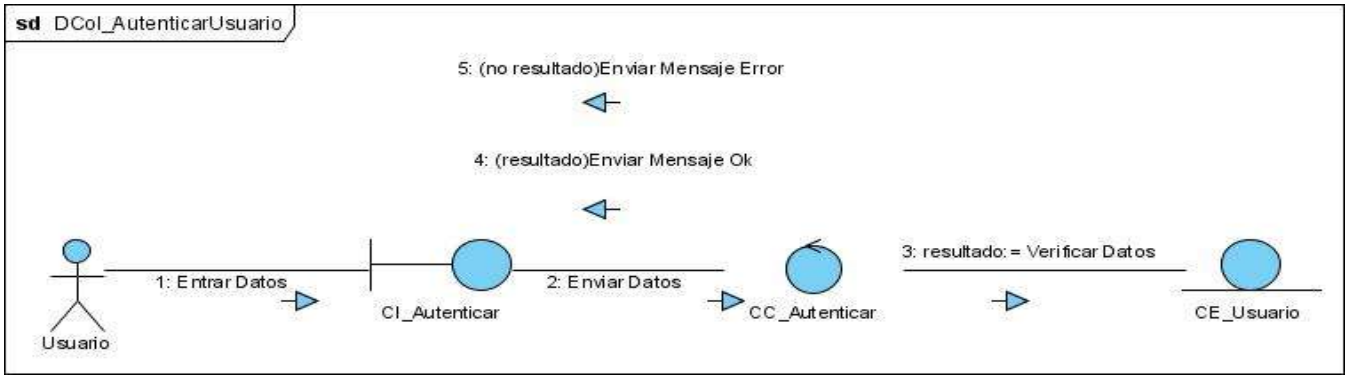


Diagrama de Colaboración: Gestionar Usuario (Sección Adicionar).

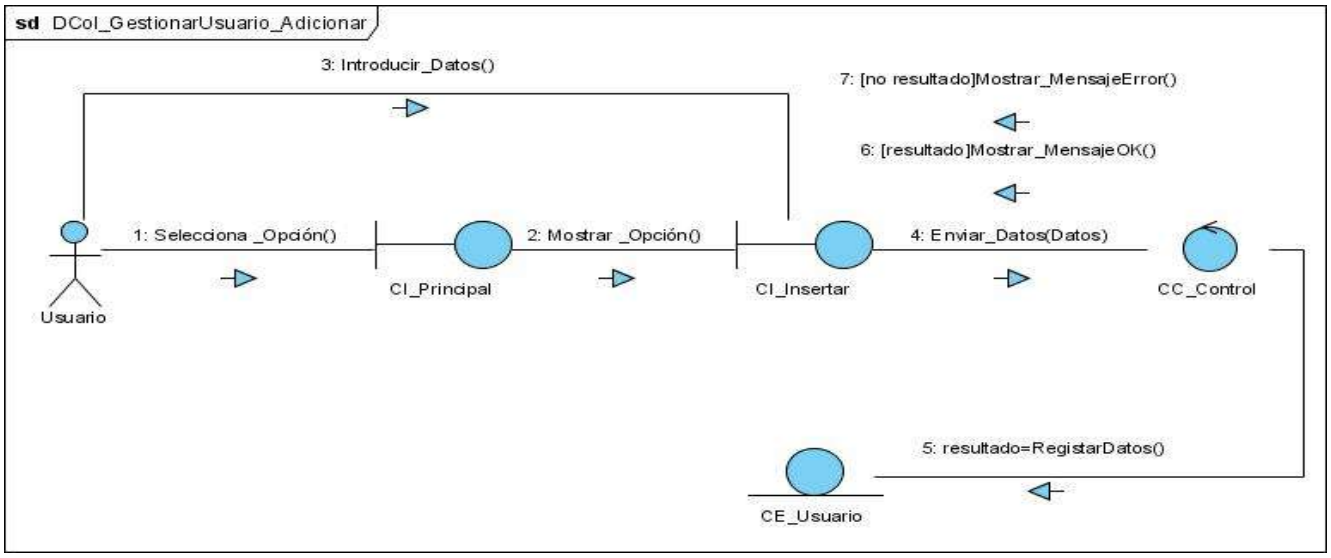


Diagrama de Colaboración: Gestionar Usuario (Sección Listar).

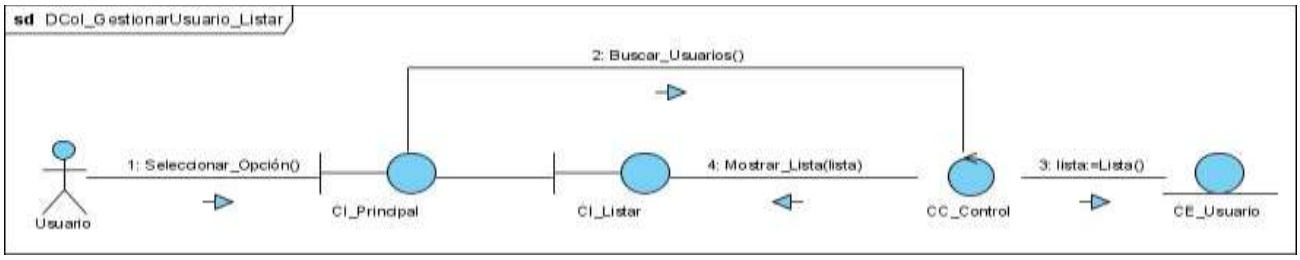


Diagrama de Colaboración: Gestionar Usuario (Sección Modificar).

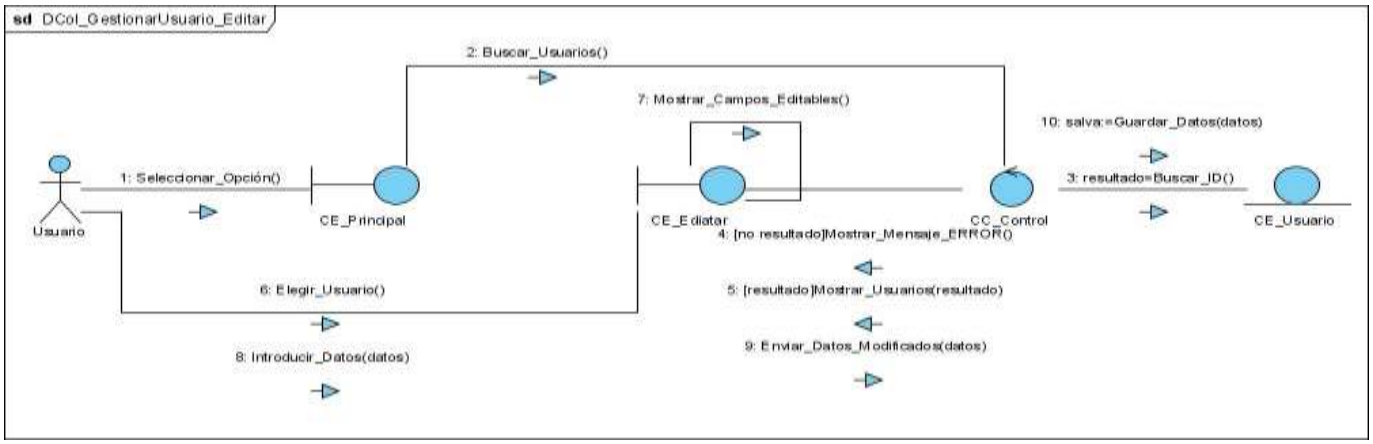


Diagrama de Colaboración: Gestionar Usuario (Sección Eliminar).

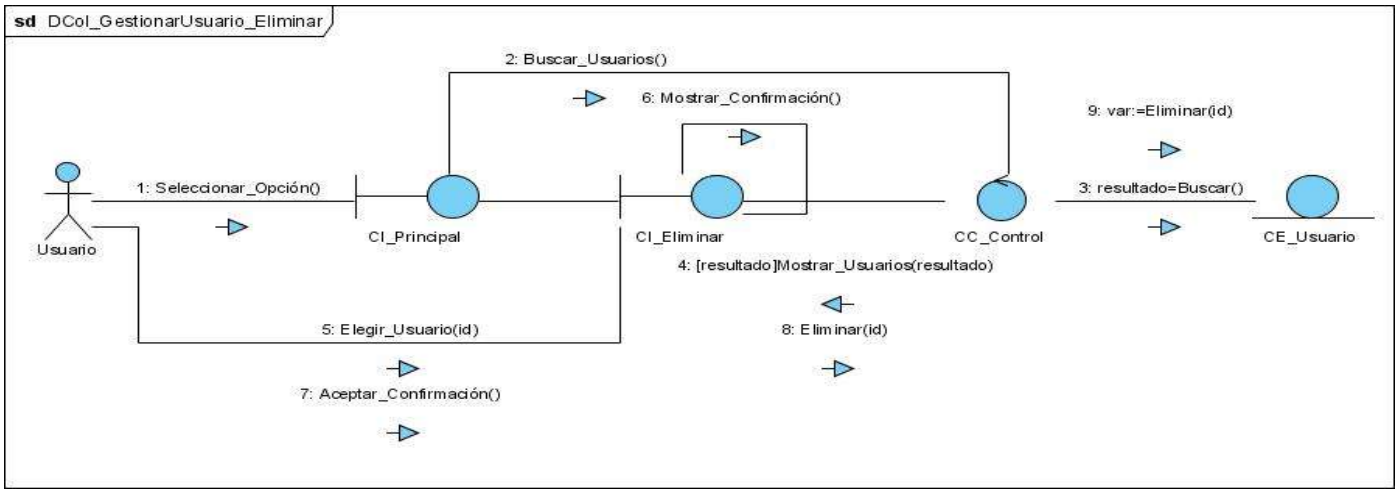


Diagrama de Colaboración: Gestionar Docente (Sección Adicionar).

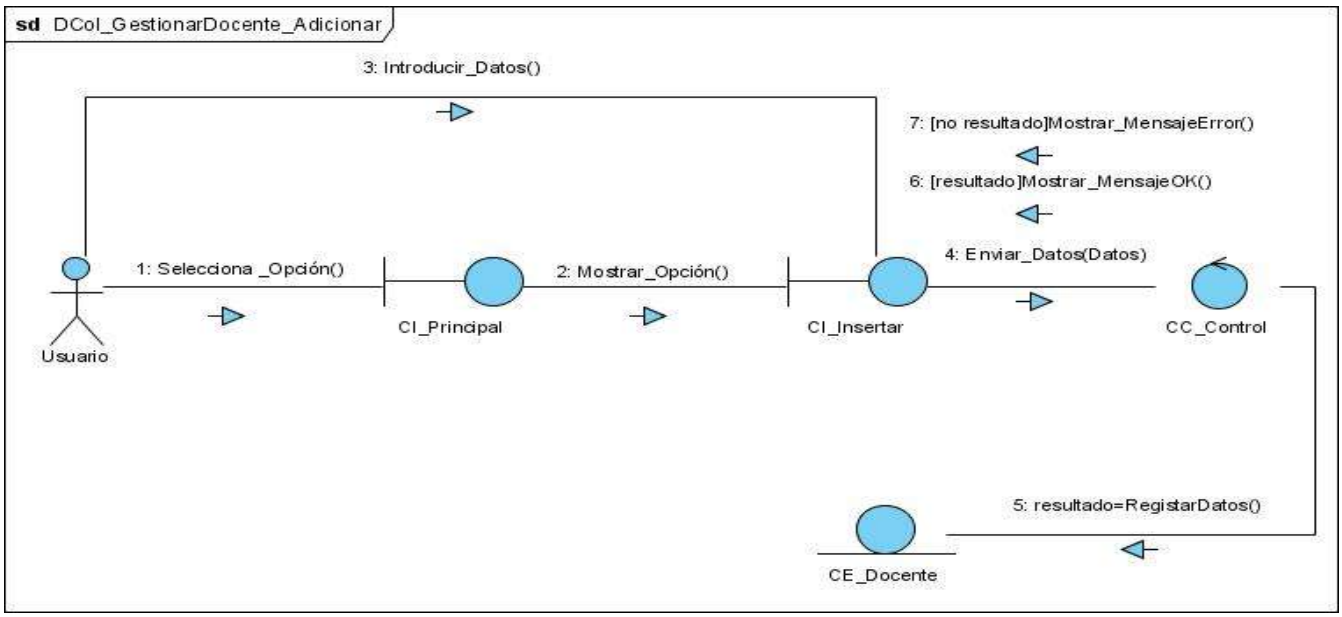


Diagrama de Colaboración: Gestionar Docente (Sección Listar).

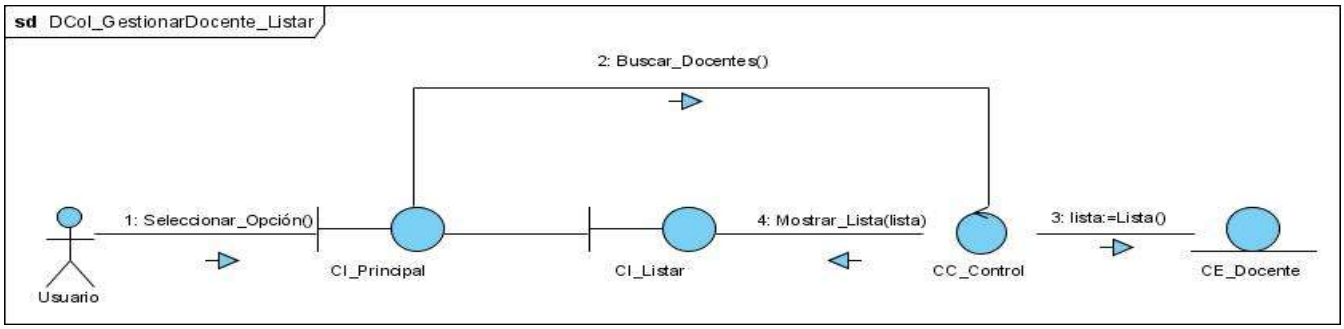


Diagrama de Colaboración: Gestionar Docente (Sección Adicionar).

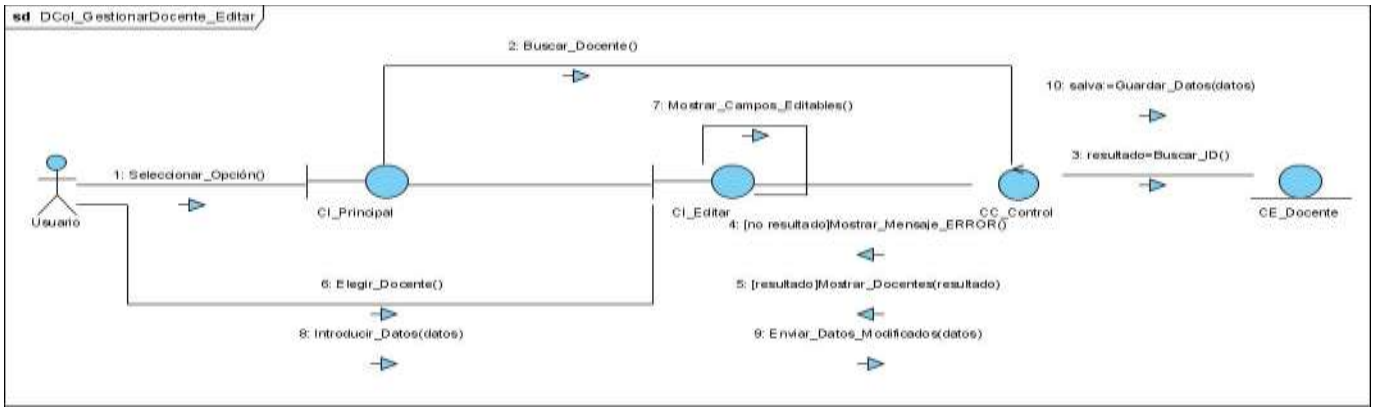


Diagrama de Colaboración: Gestionar Docente (Sección Eliminar).

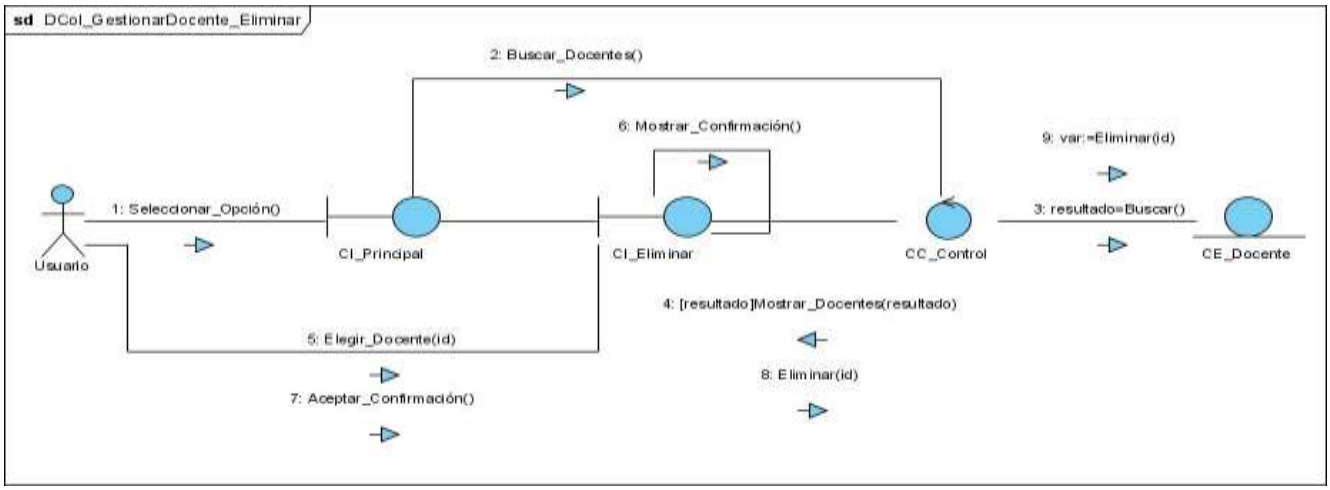


Diagrama de Colaboración: Gestionar Facultad (Sección Adicionar).

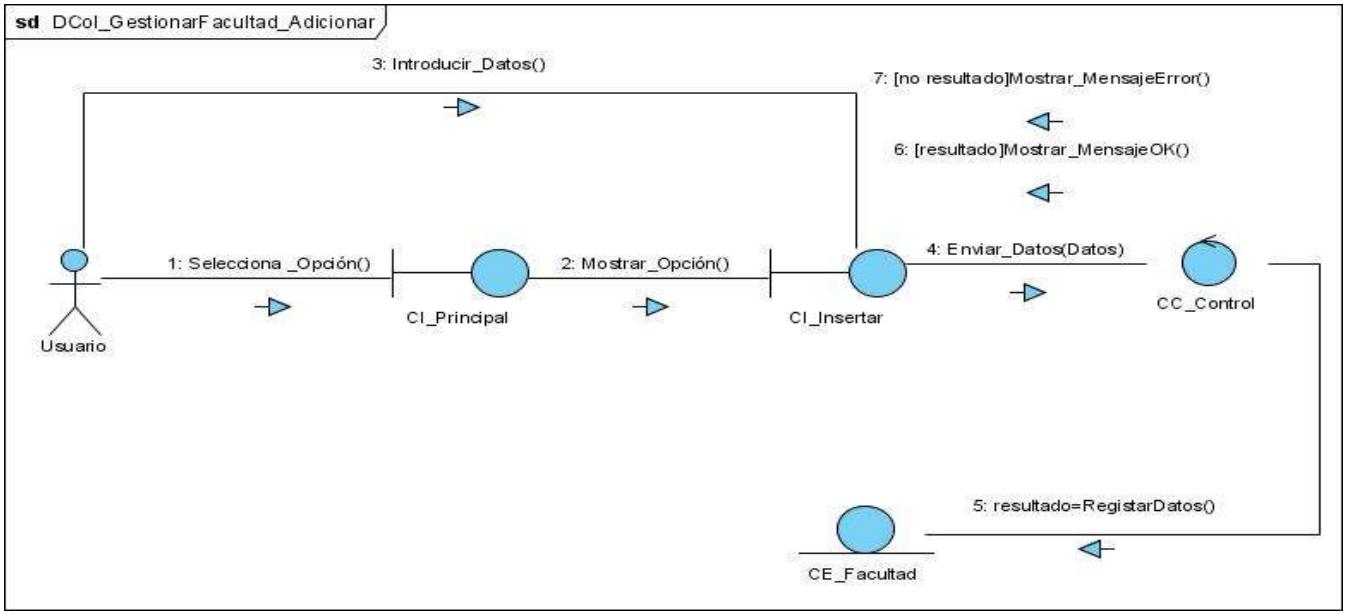


Diagrama de Colaboración: Gestionar Facultad (Sección Listar).

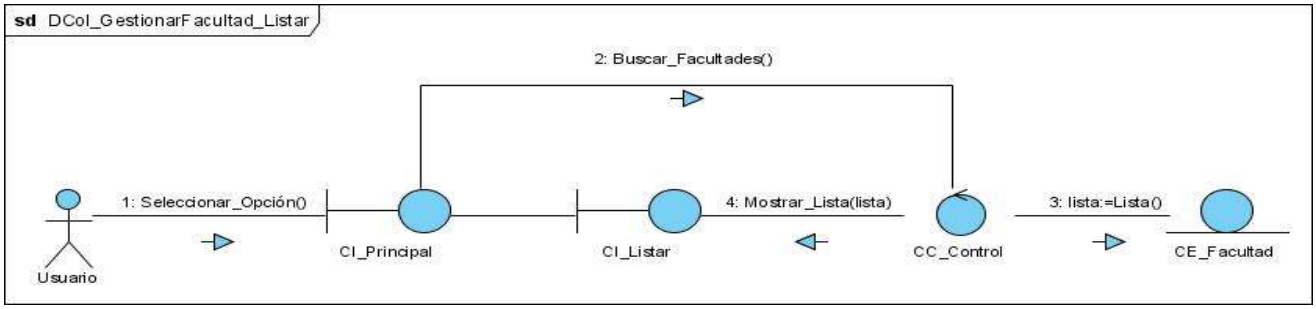


Diagrama de Colaboración: Gestionar Facultad (Sección Modificar).

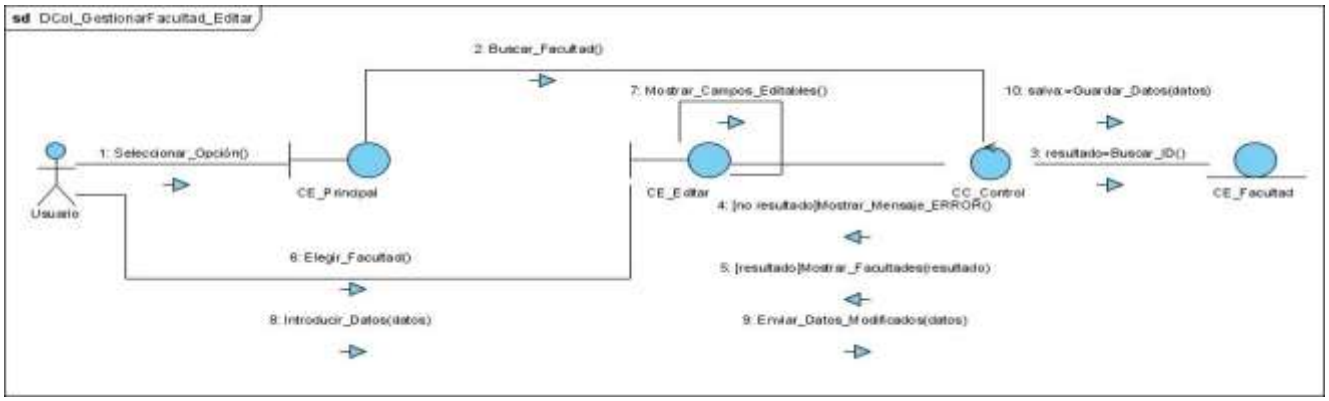


Diagrama de Colaboración: Gestionar Facultad (Sección Eliminar).

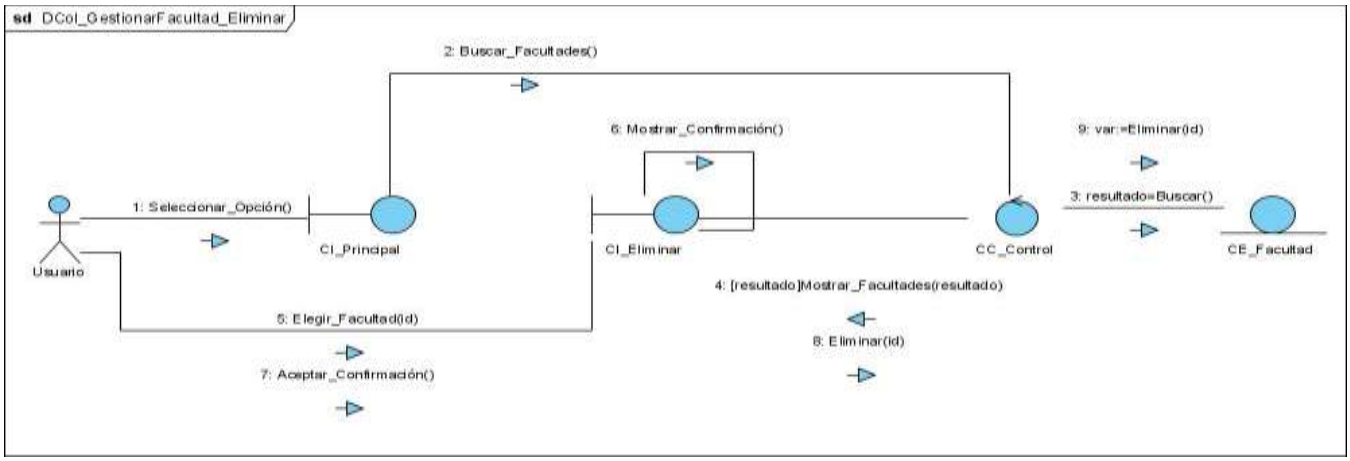


Diagrama de Colaboración: Gestionar Proyecto Productivo (Sección Adicionar).

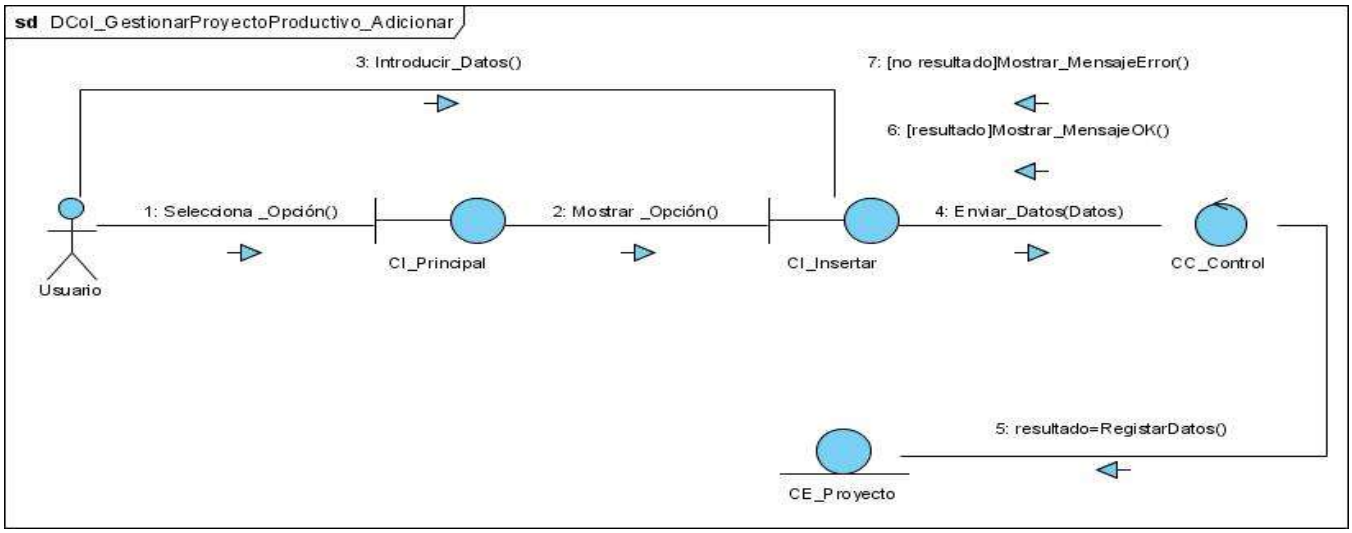


Diagrama de Colaboración: Gestionar Proyecto Productivo (Sección Listar)

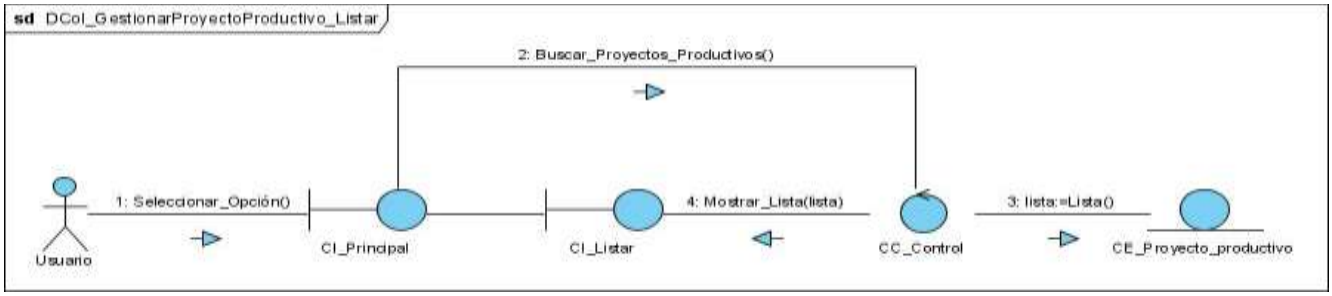


Diagrama de Colaboración: Gestionar Proyecto Productivo (Sección Modificar).

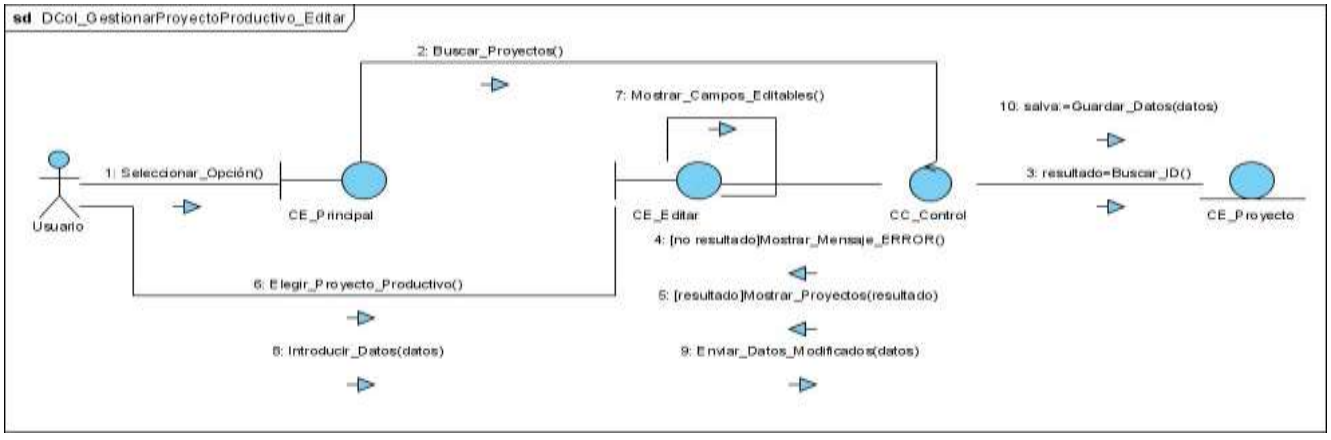


Diagrama de Colaboración: Gestionar Proyecto Productivo (Sección Eliminar).

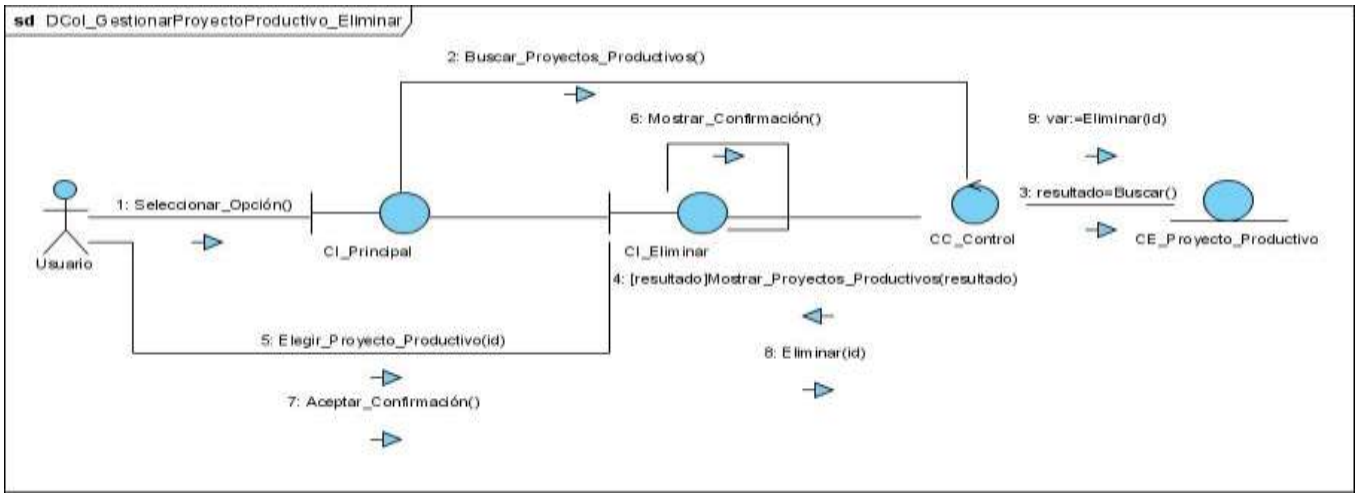

sd DCol_GestionarProyectoProductivo_Eliminar
2: Buscar_Proyectos_Productivos()
6: Mostrar_Confirmación()
9: var:=Eliminar(id)
1: Seleccionar_Opción()
3: resultado=Buscar()
Usuario
CI_Principal
CI_Eliminar
CC_Control
CE_Proyecto_Productivo
4: [resultado]Mostrar_Proyectos_Productivos(resultado)
5: Elegir_Proyecto_Productivo(id)
8: Eliminar(id)
7: Aceptar_Confirmación()

Diagrama de Colaboración: Gestionar Laboratorios (Sección Adicionar).

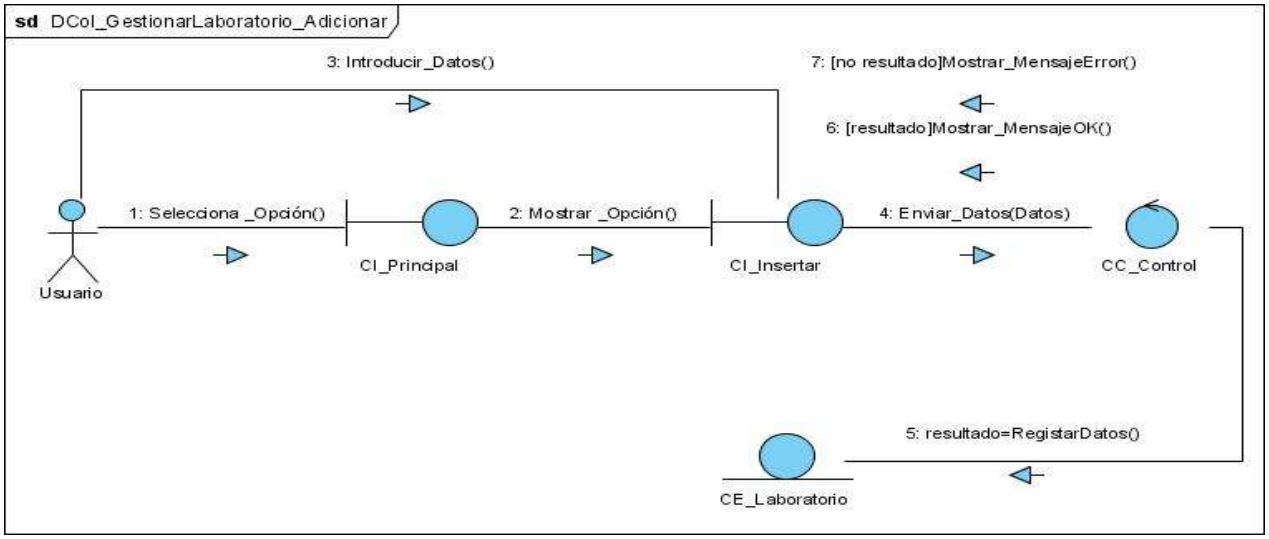


Diagrama de Colaboración: Gestionar Laboratorios (Sección Listar).

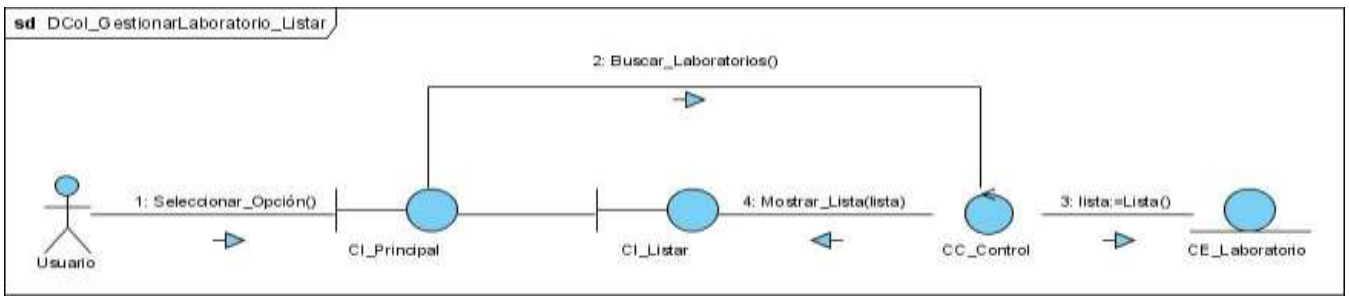


Diagrama de Colaboración: Gestionar Laboratorios (Sección Modificar).

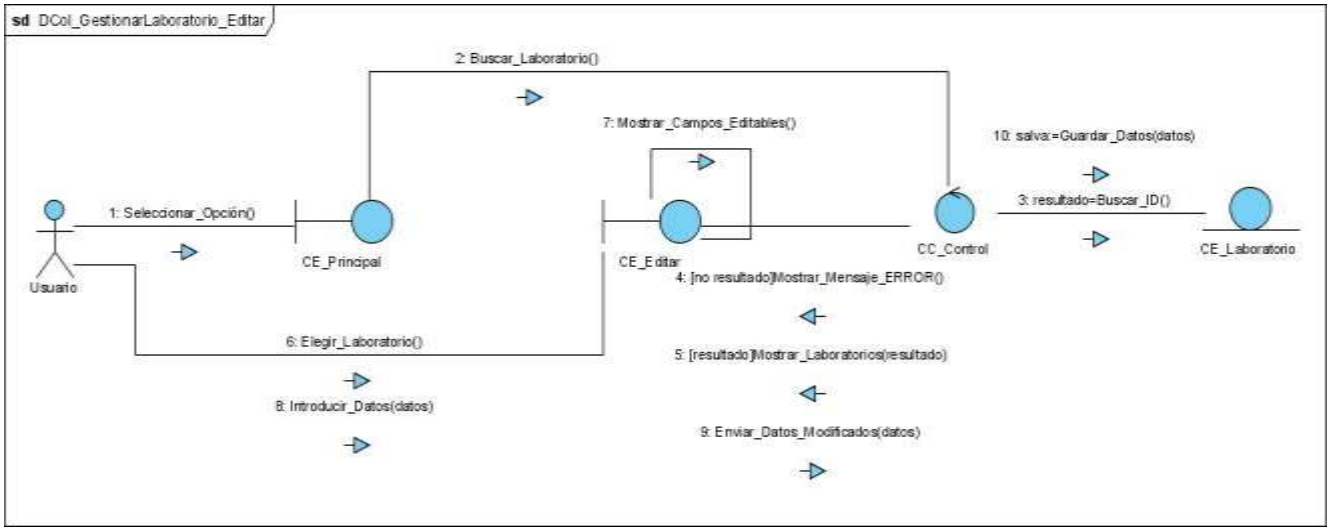


Diagrama de Colaboración: Gestionar Laboratorios (Sección Eliminar).

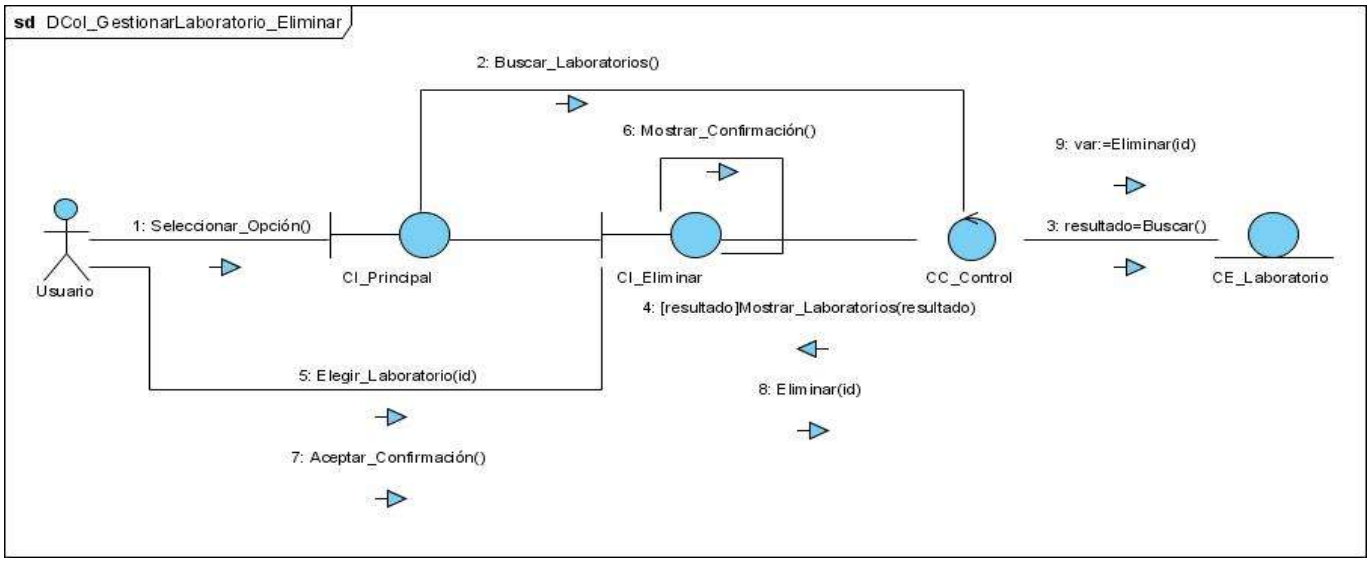


Diagrama de Colaboración: Gestionar Características (Sección Adicionar).

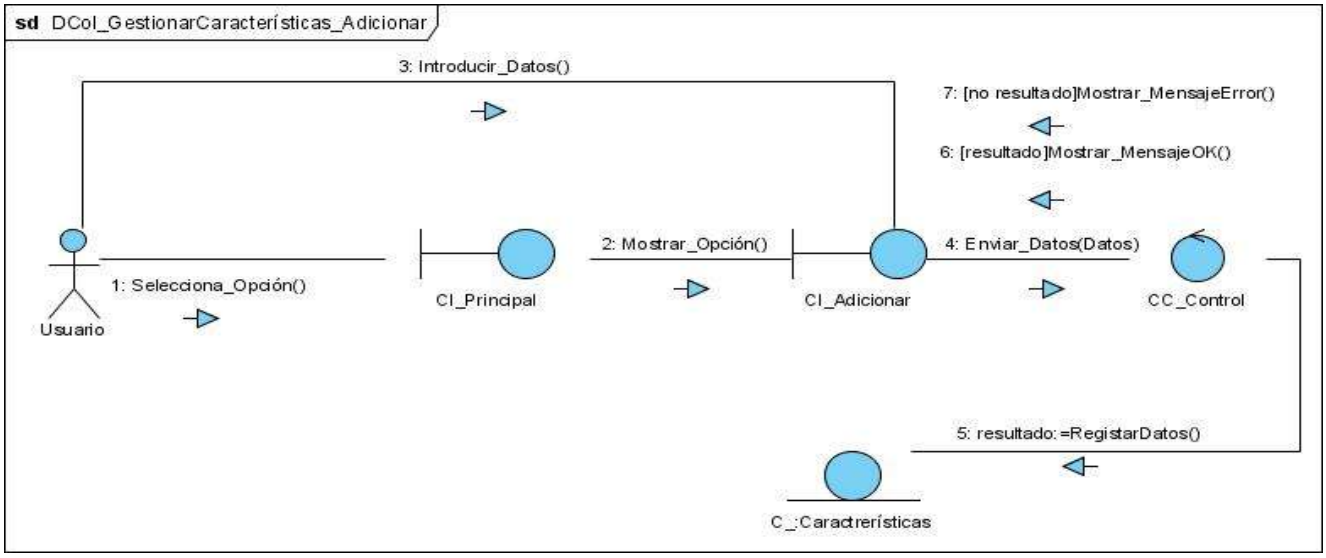


Diagrama de Colaboración: Gestionar Características (Sección Listar).

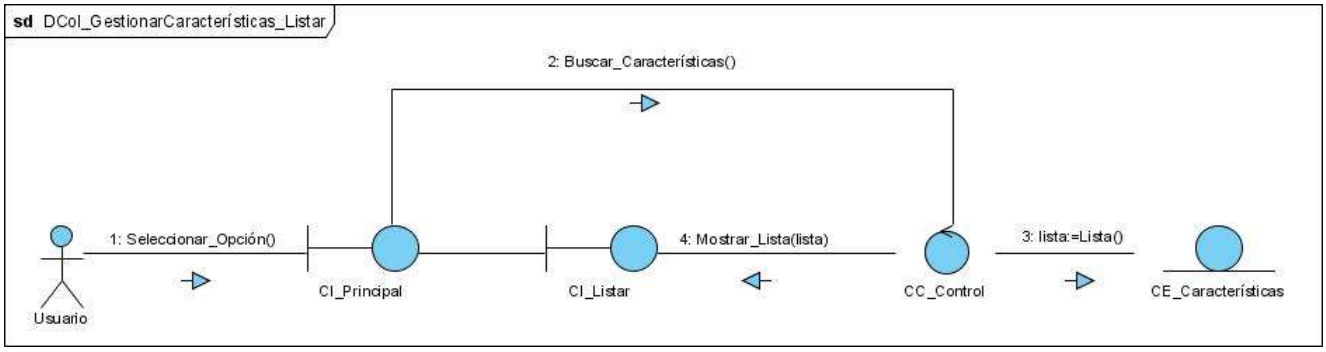


Diagrama de Colaboración Gestionar Características (Sección Modificar).

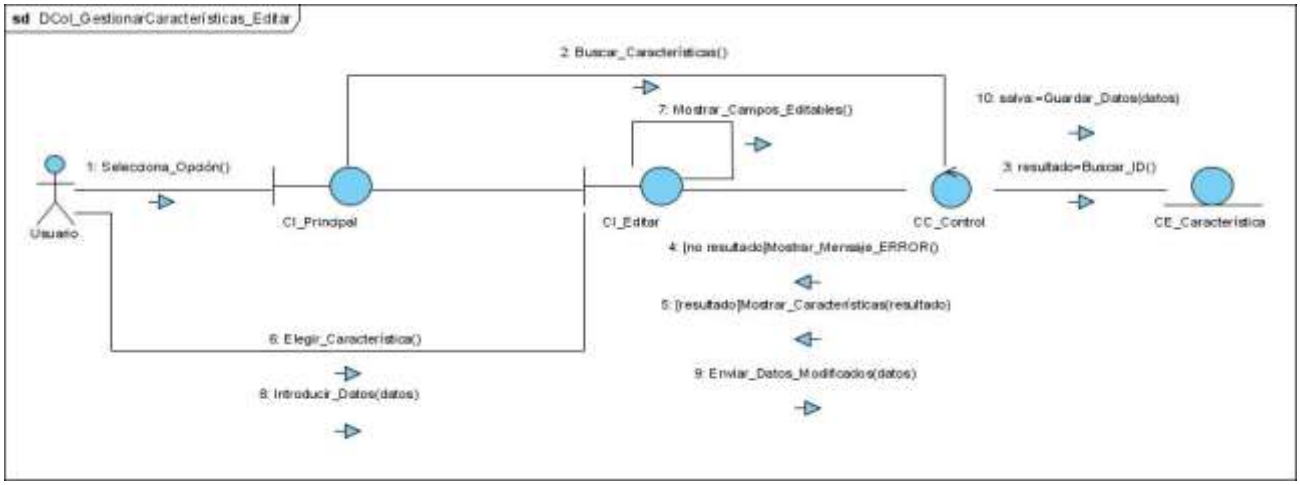


Diagrama de Colaboración: Gestionar Características (Sección Eliminar).

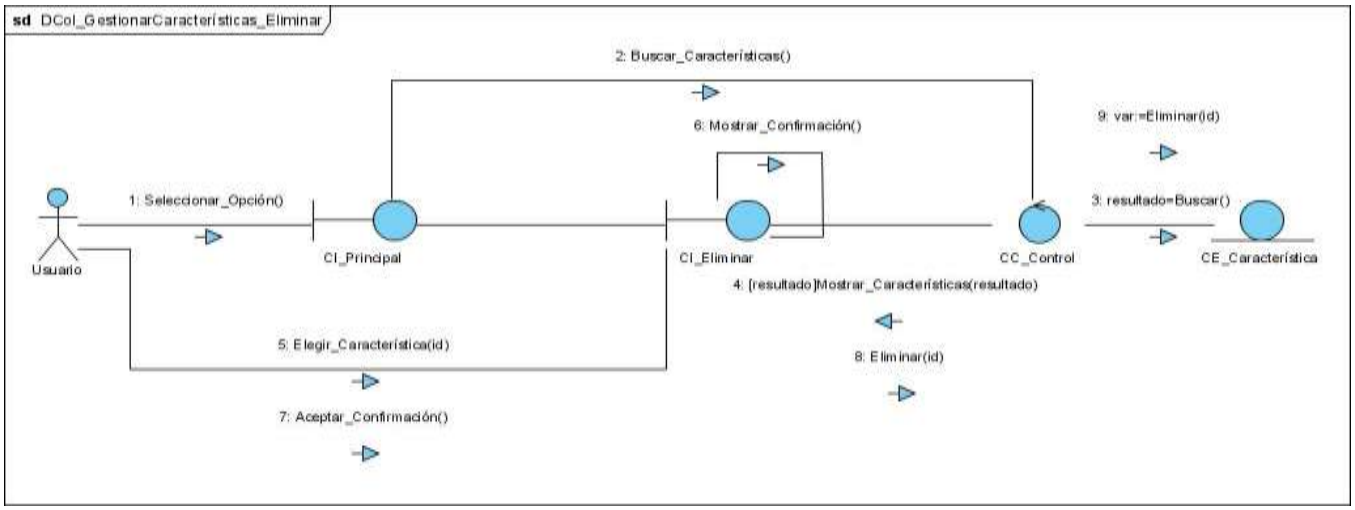


Diagrama de Colaboración: Gestionar Componentes (Sección Adicionar).

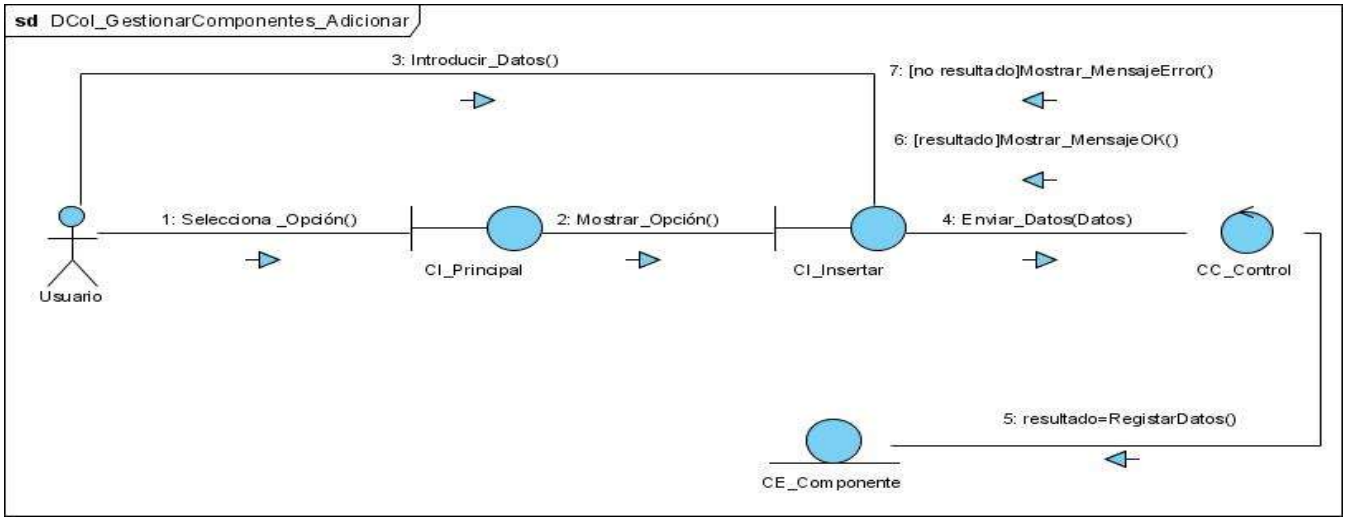


Diagrama de Colaboración: Gestionar Componentes (Sección Listar)

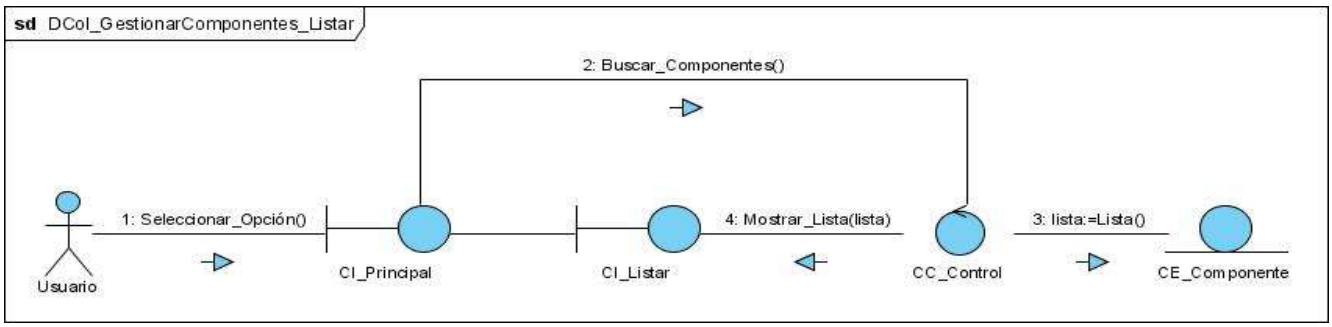


Diagrama de Colaboración: Gestionar Componentes (Sección Modificar).

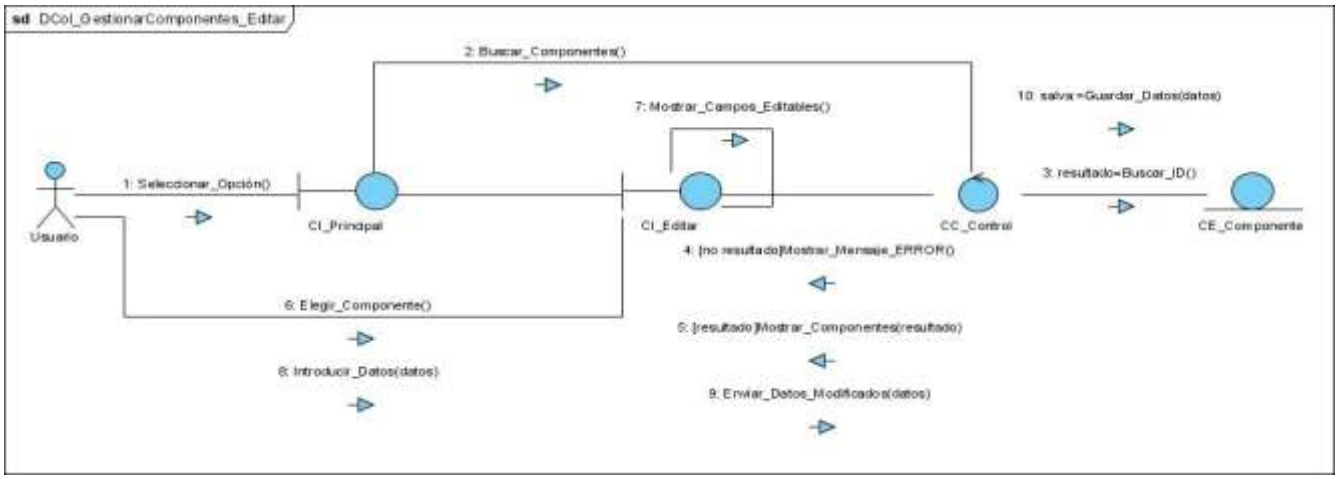


Diagrama de Colaboración: Gestionar Componentes (Sección Eliminar).

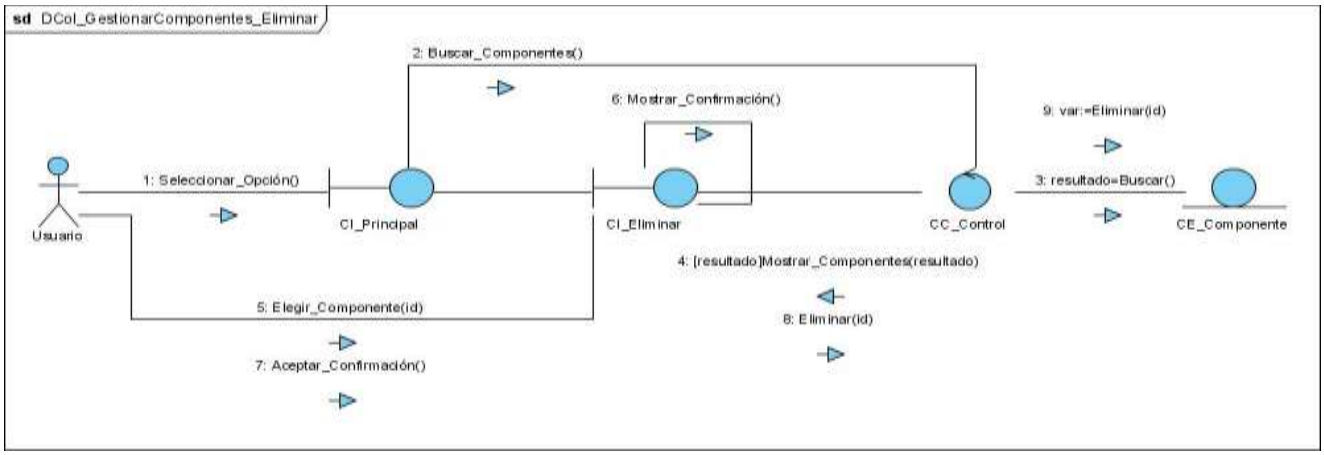


Diagrama de Colaboración: Gestionar Medidas (Sección Adicionar).

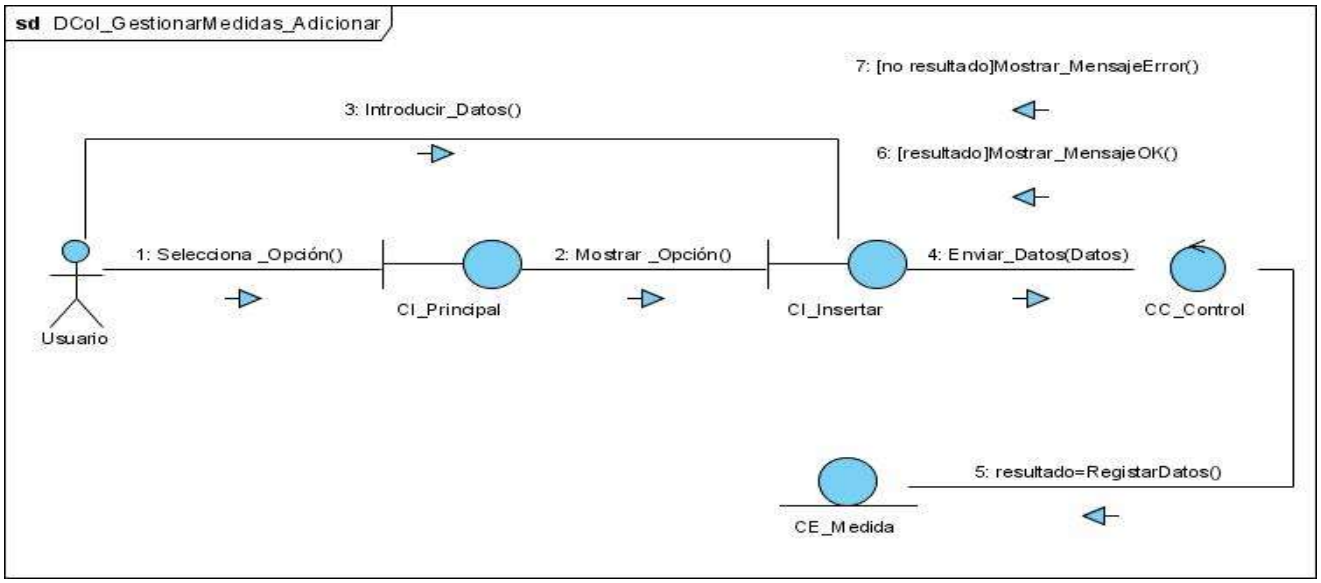


Diagrama de Colaboración: Gestionar Medidas (Sección Listar).

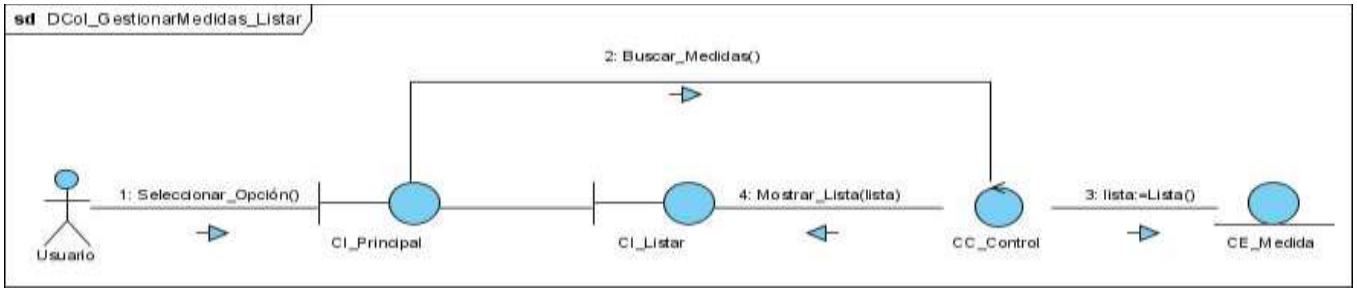


Diagrama de Colaboración: Gestionar Medidas (Sección Modificar).

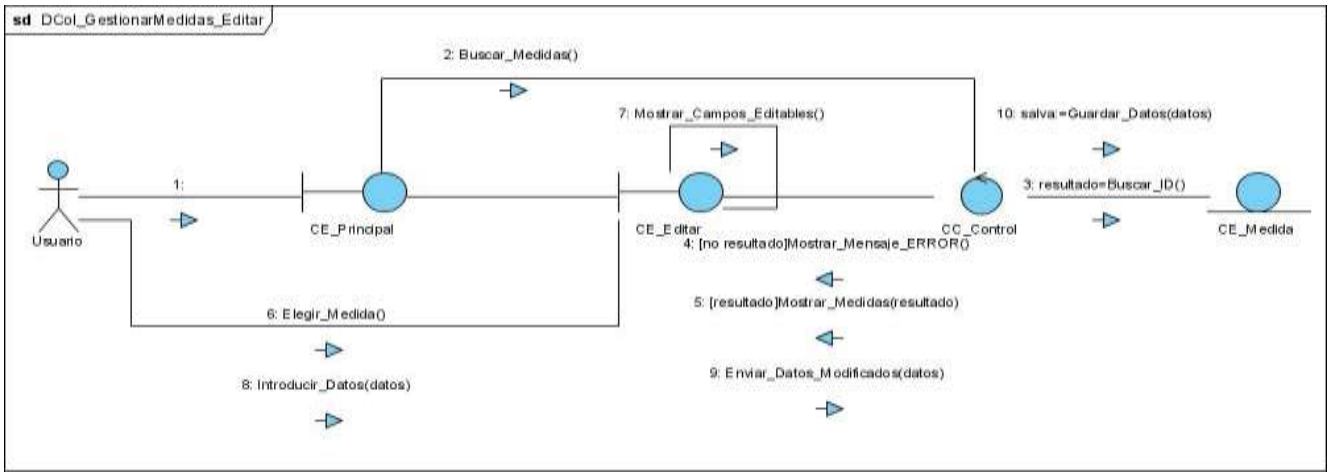
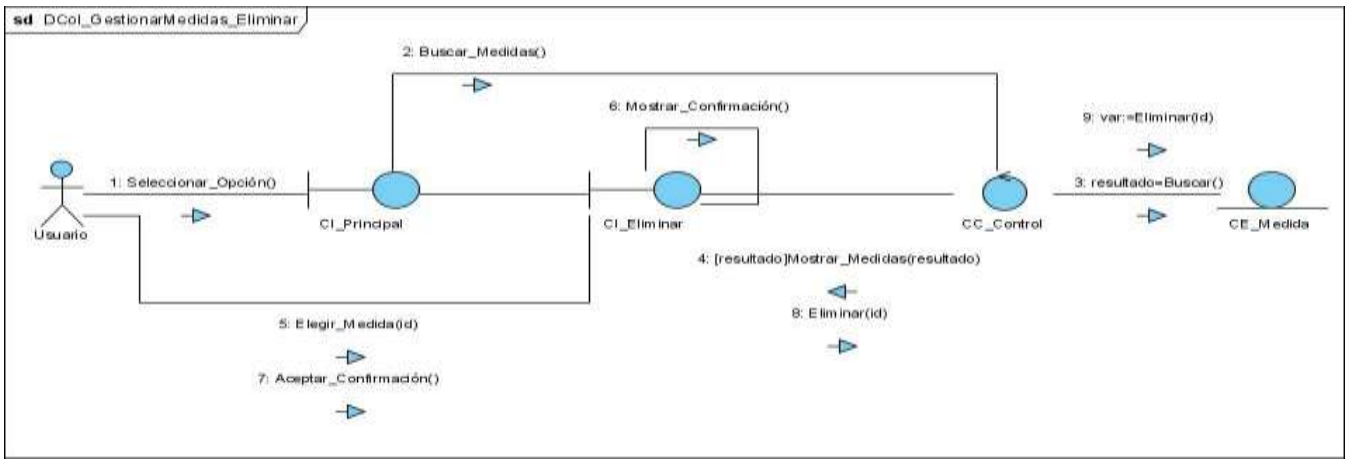
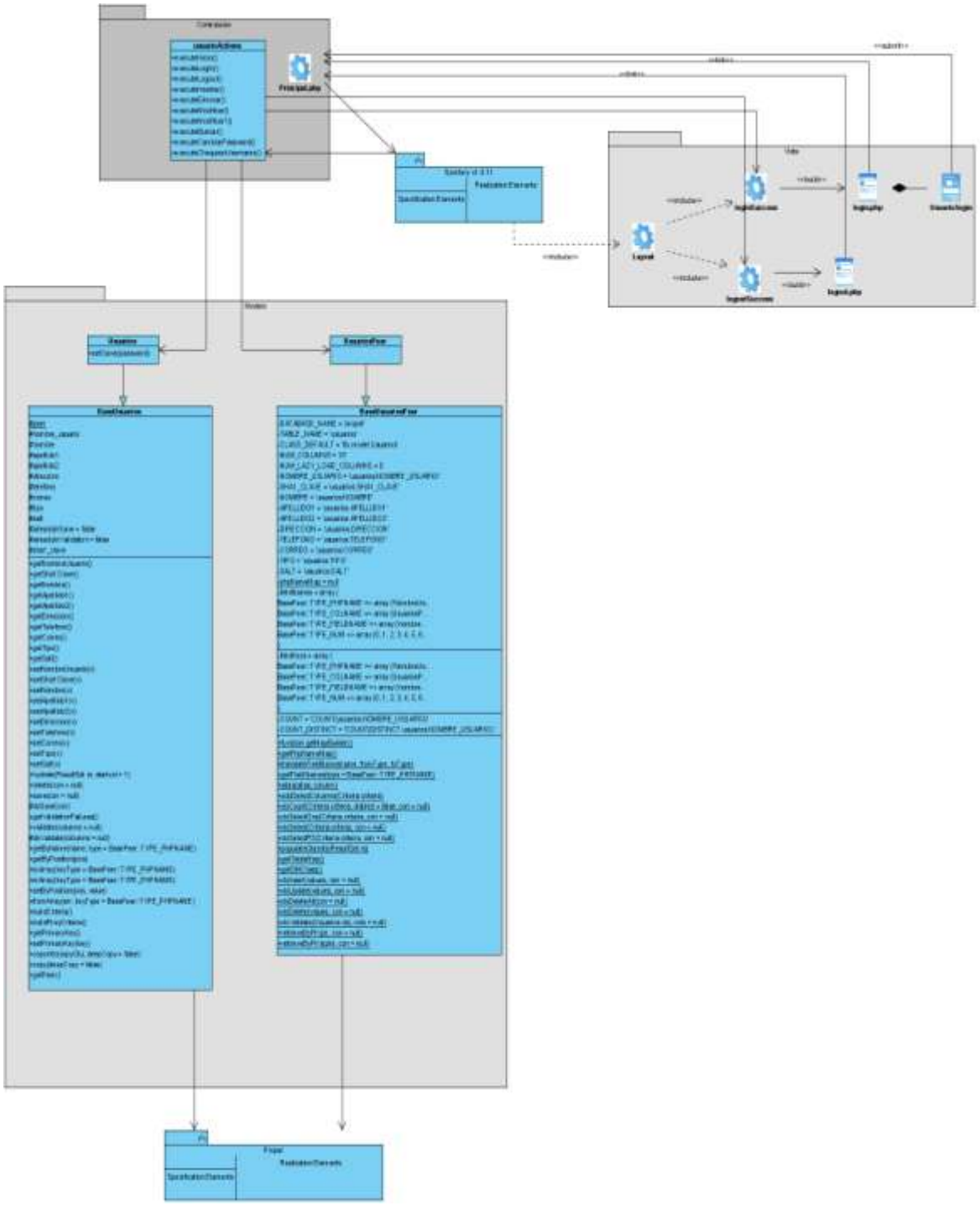


Diagrama de Colaboración: Gestionar Medidas (Sección Eliminar).

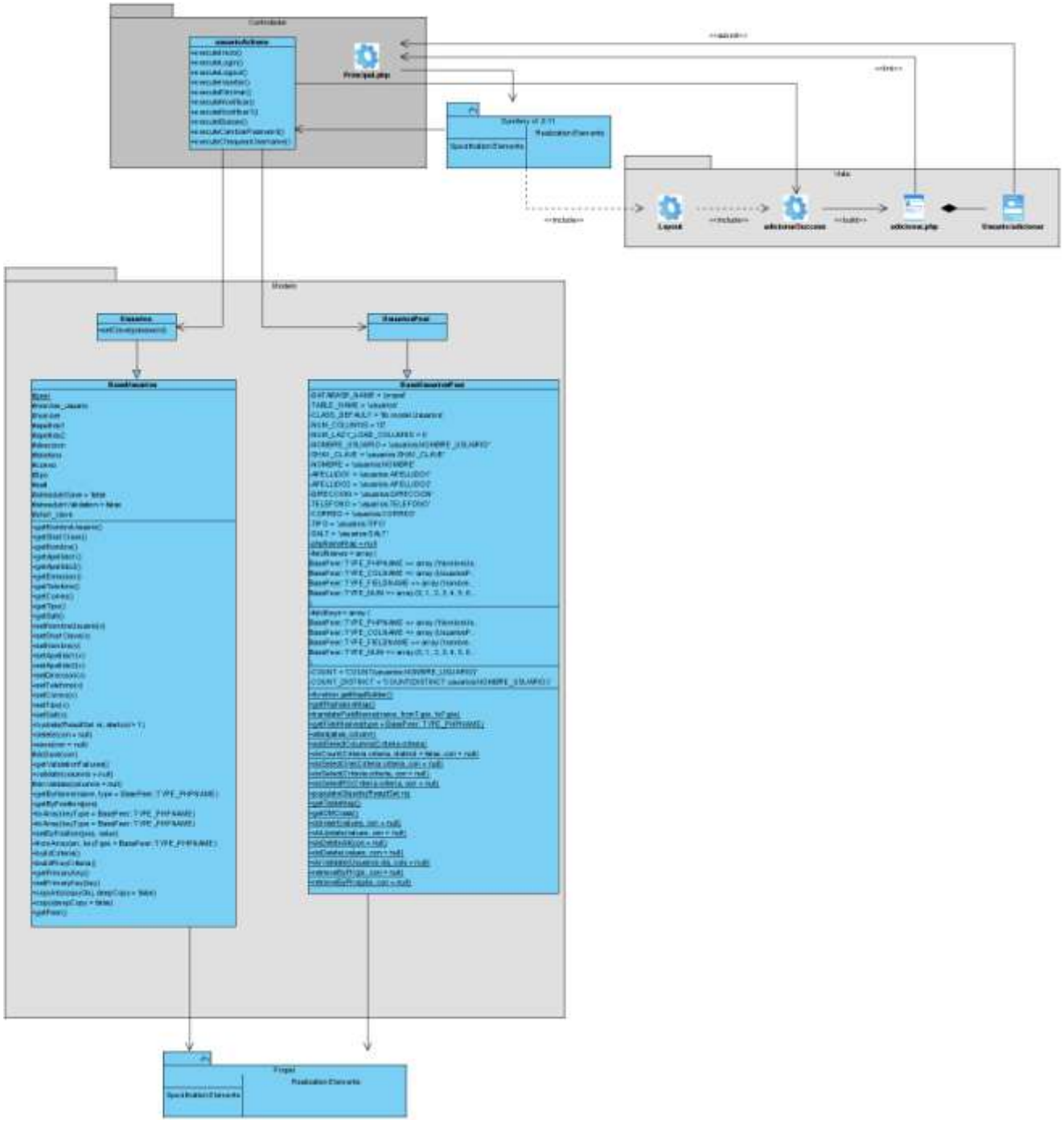


Anexo 3 Diagramas de Clases de Diseño

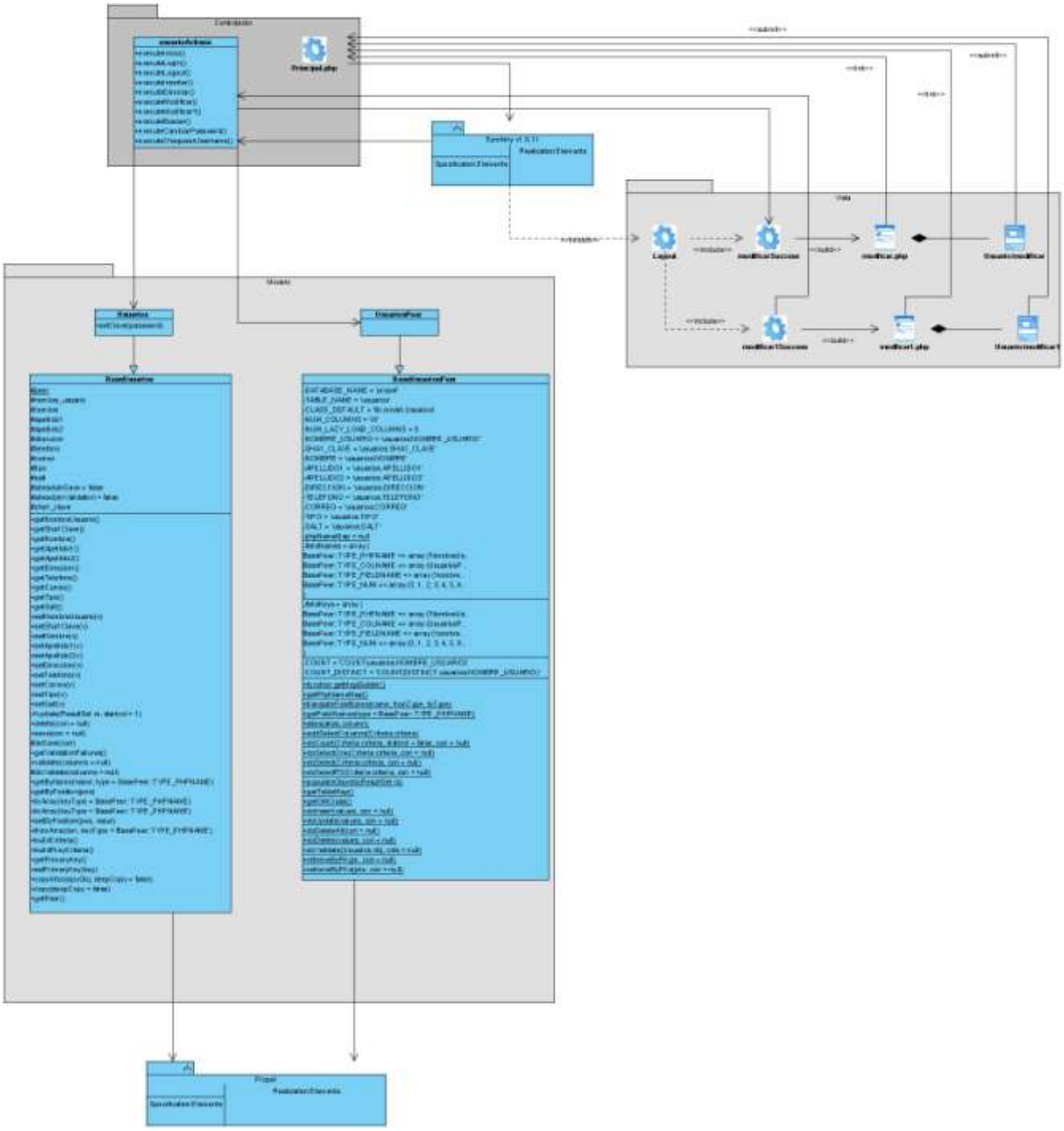
Diagramas Clases de Diseño del Caso de Uso: Autenticar Usuario.



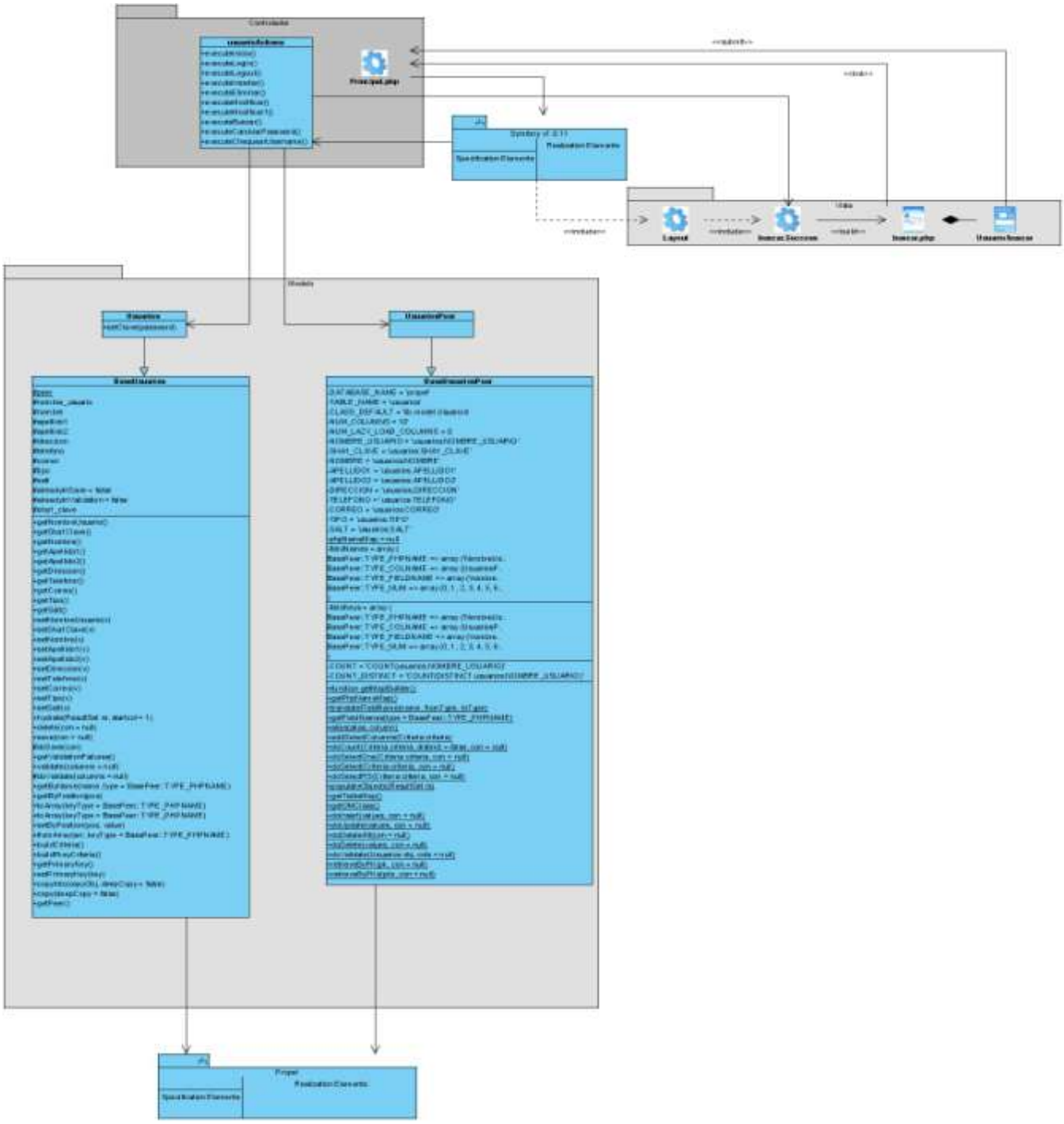
Diagramas Clases de Diseño del Caso de Uso: Gestionar Usuario sección Adicionar.



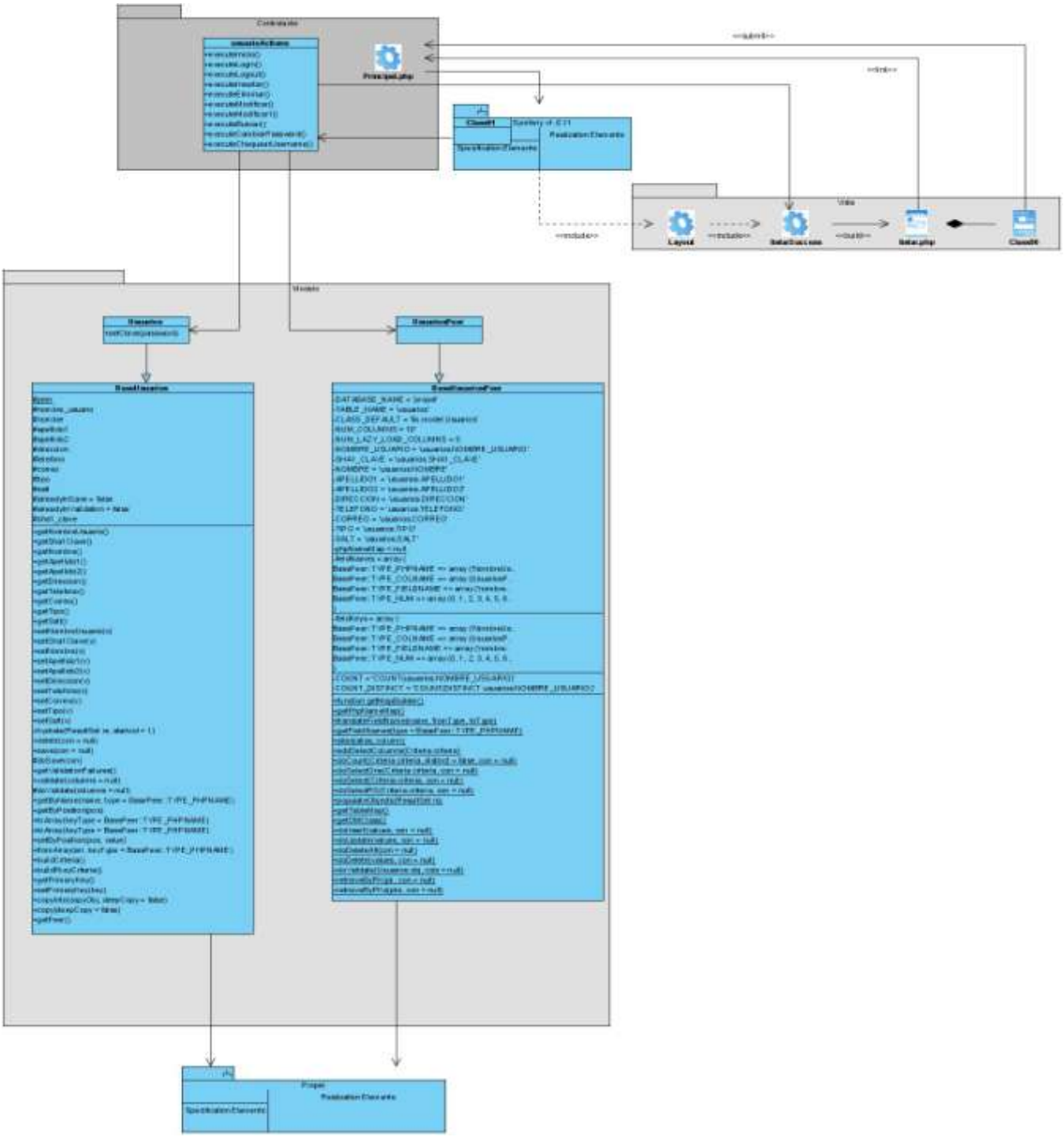
Diagramas Clases de Diseño del Caso de Uso: Gestionar Usuario sección Modificar.



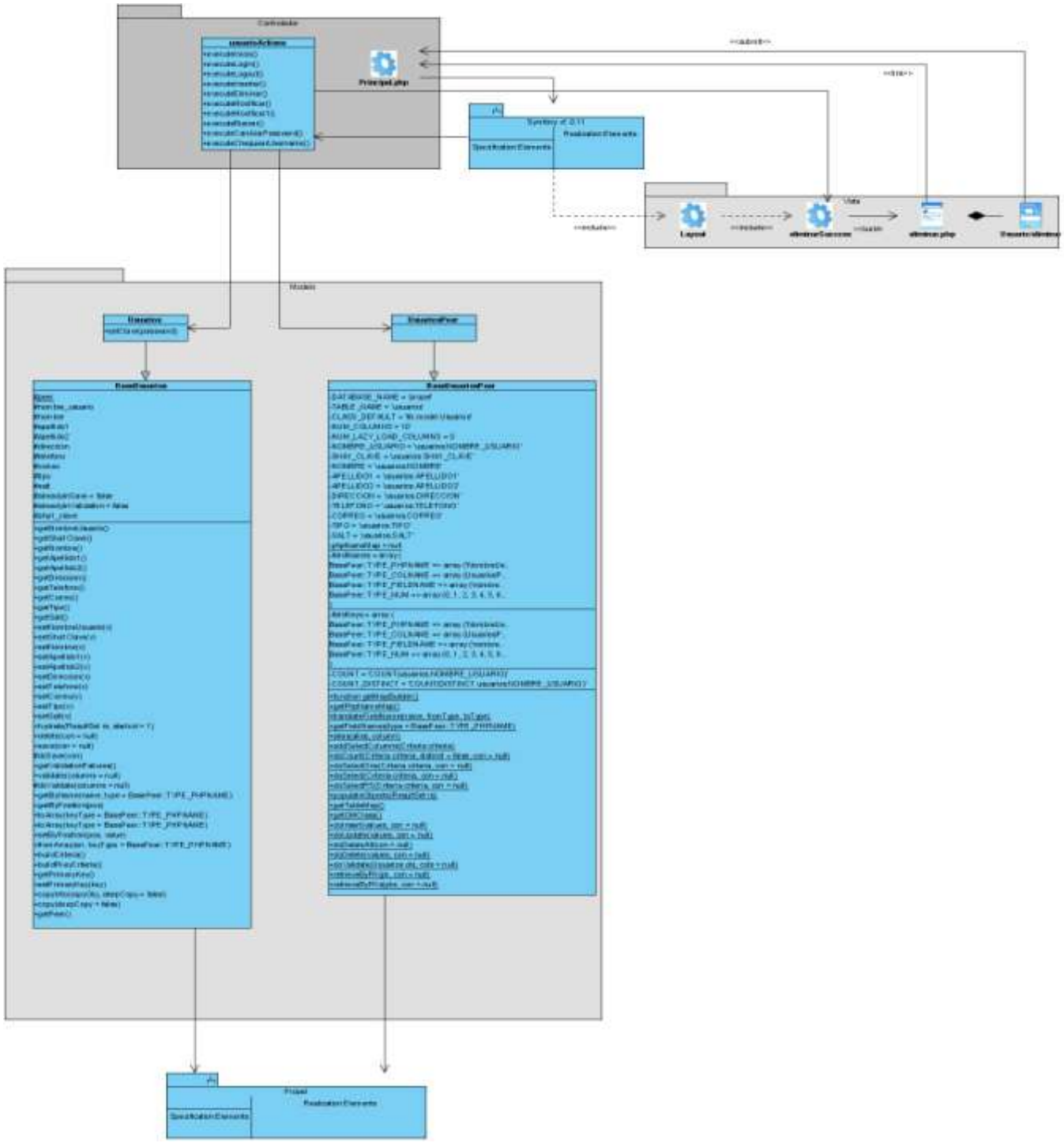
Diagramas Clases de Diseño del Caso de Uso: Gestionar Usuario sección Buscar



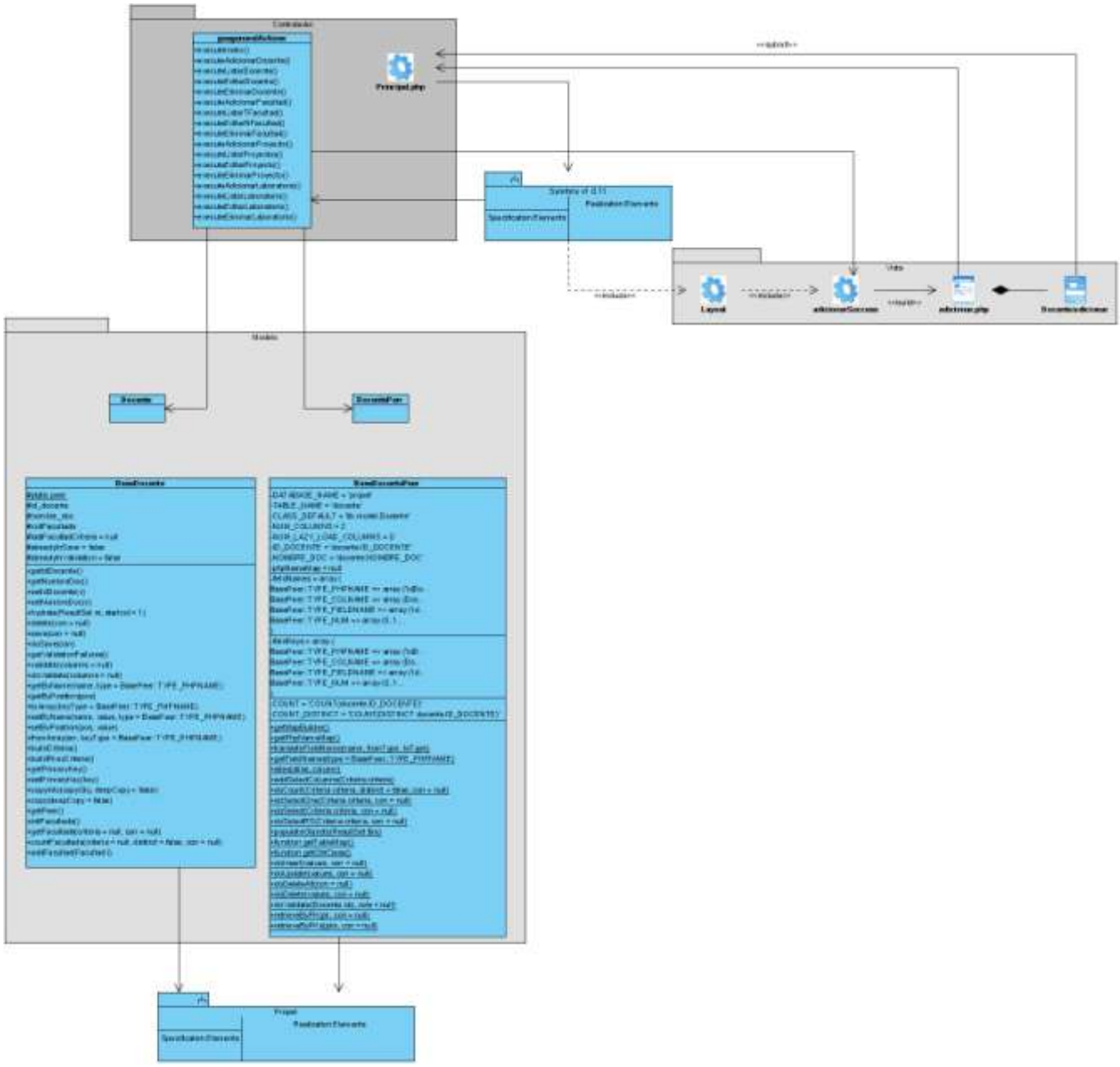
Diagramas Clases de Diseño del Caso de Uso: Gestionar Usuario sección Listar.



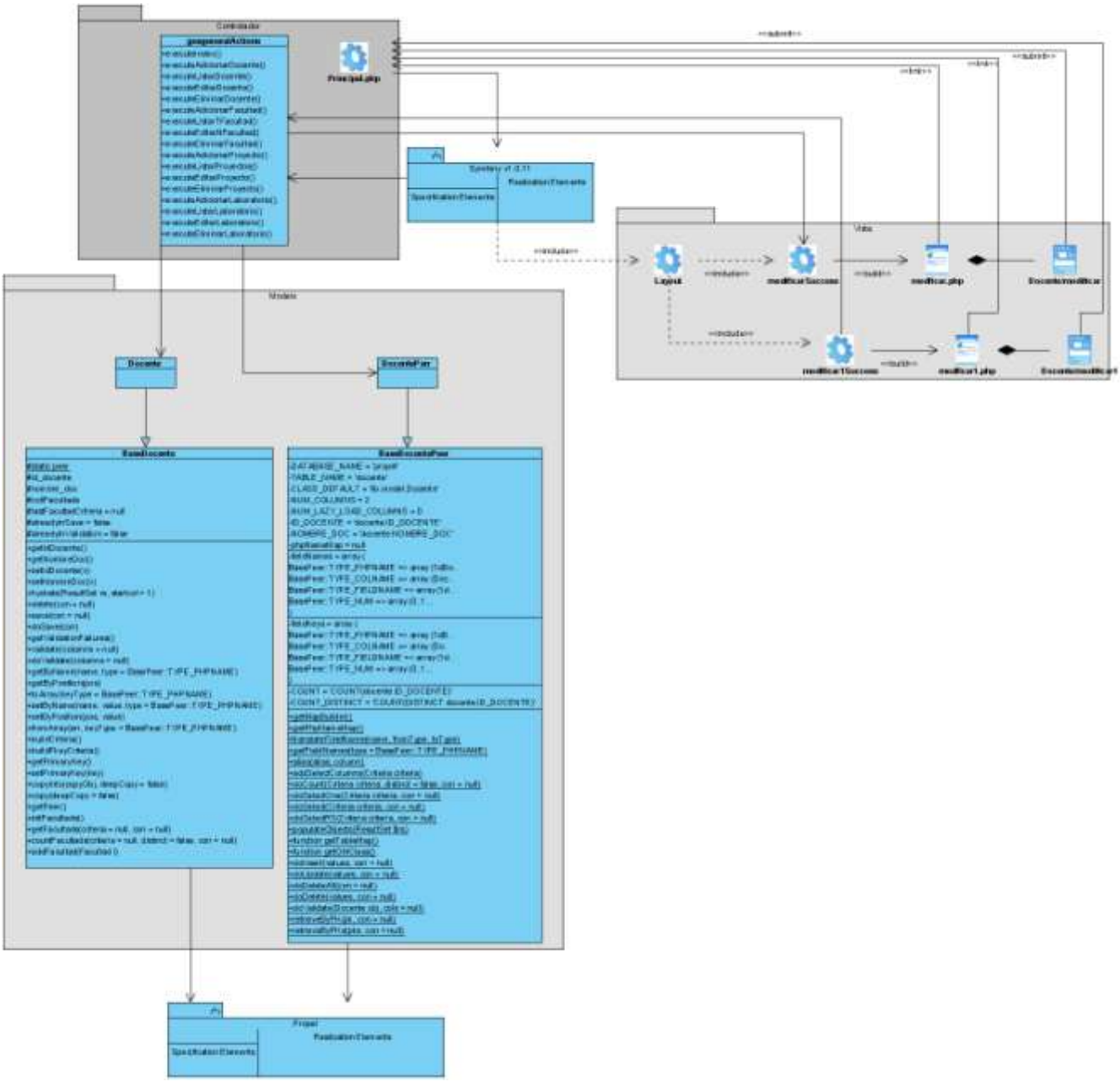
Diagramas Clases de Diseño del Caso de Uso: Gestionar Usuario sección Eliminar



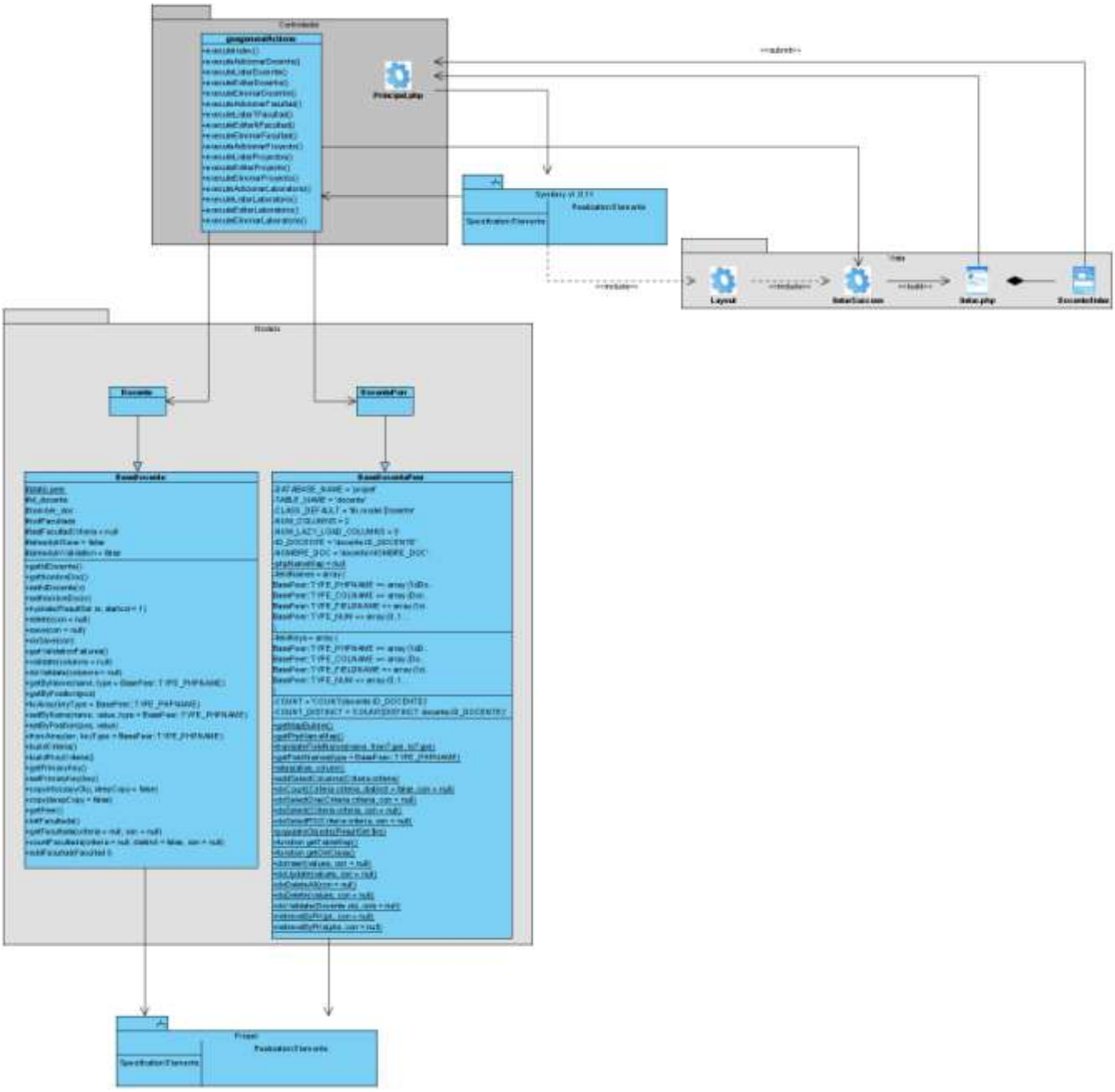
Diagramas Clases de Diseño del Caso de Uso: Gestionar Docente sección Adicionar.



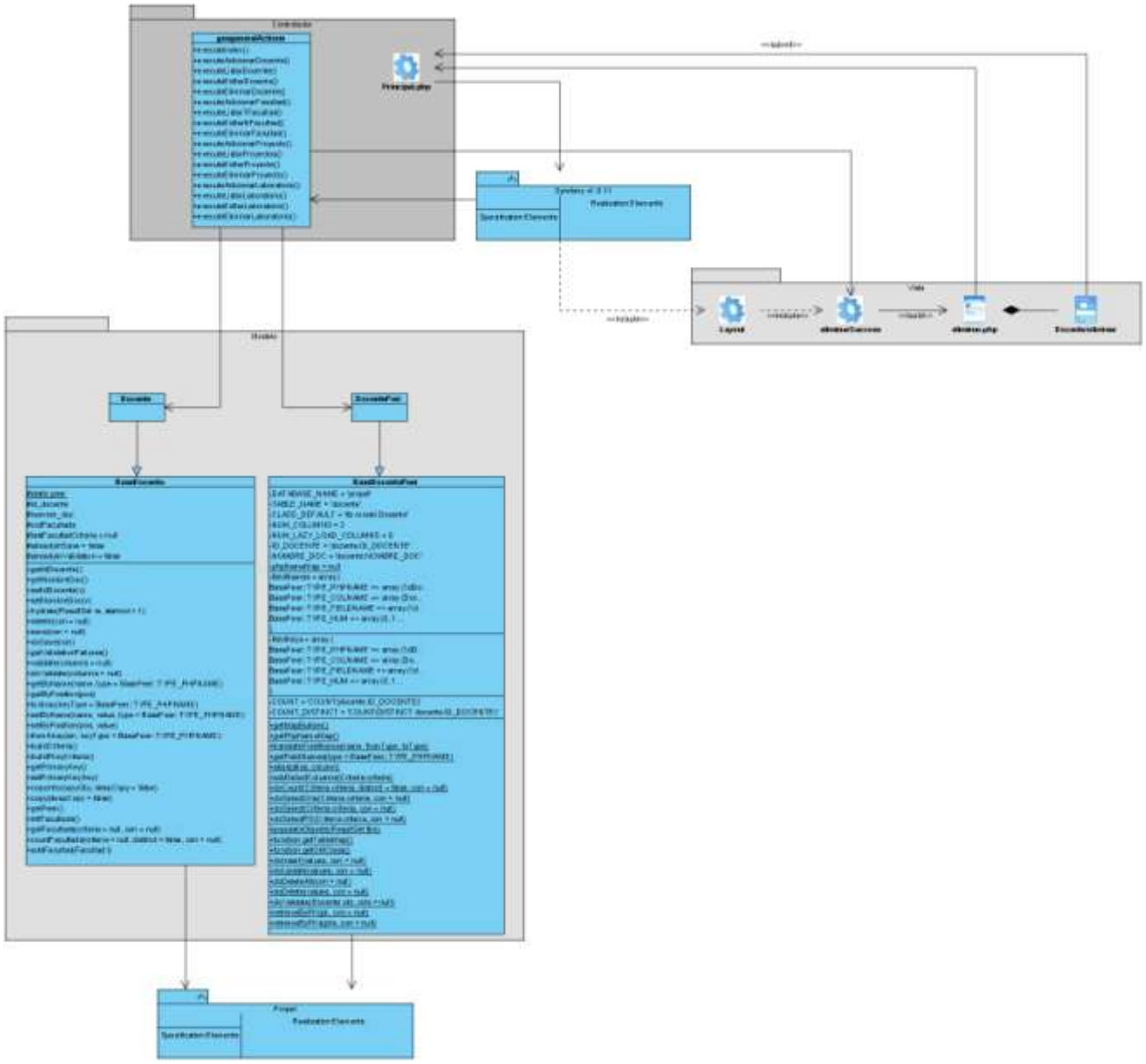
Diagramas Clases de Diseño del Caso de Uso: Gestionar Docente sección Modificar.



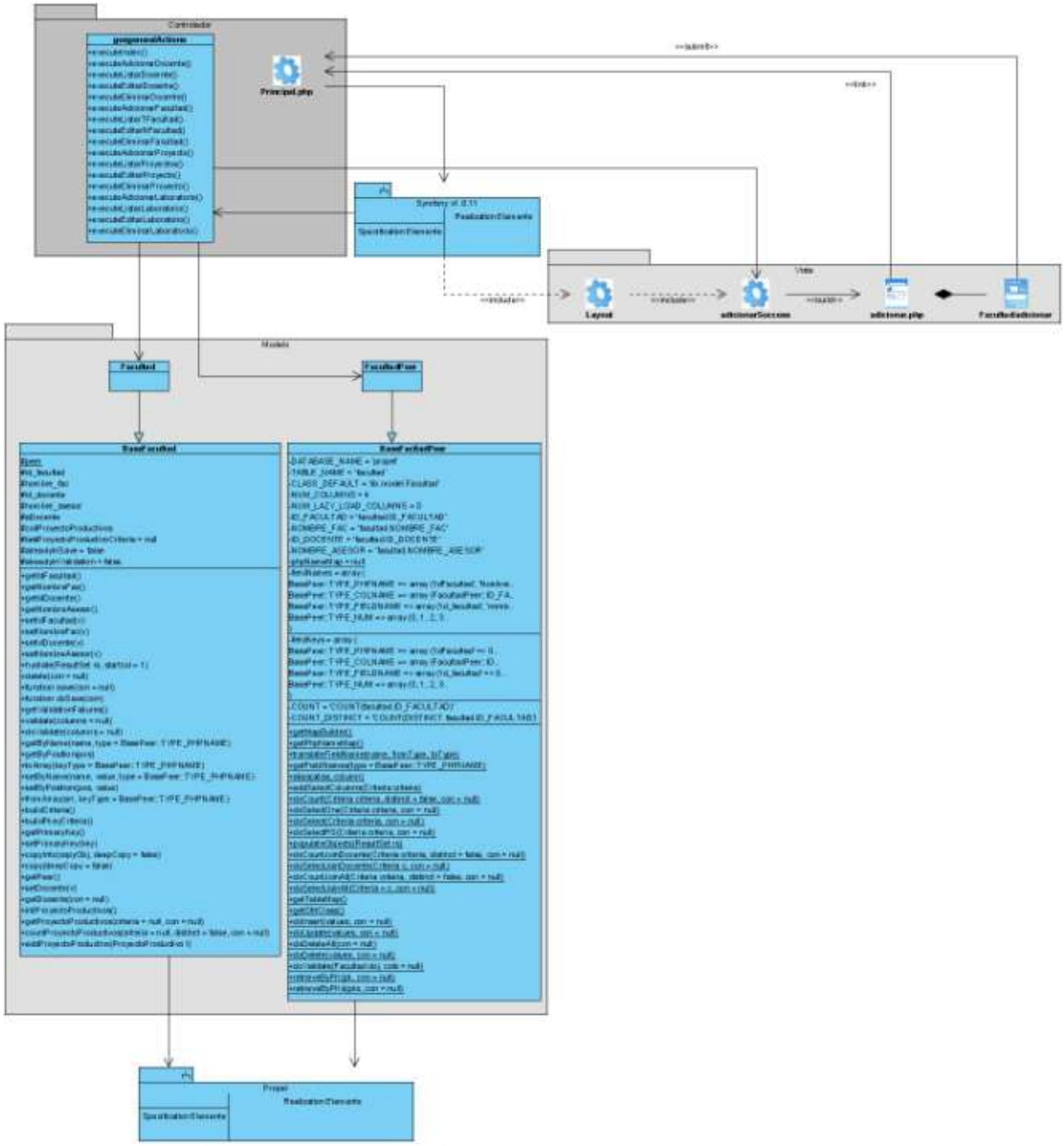
Diagramas Clases de Diseño del Caso de Uso: Gestionar Docente sección Listar.



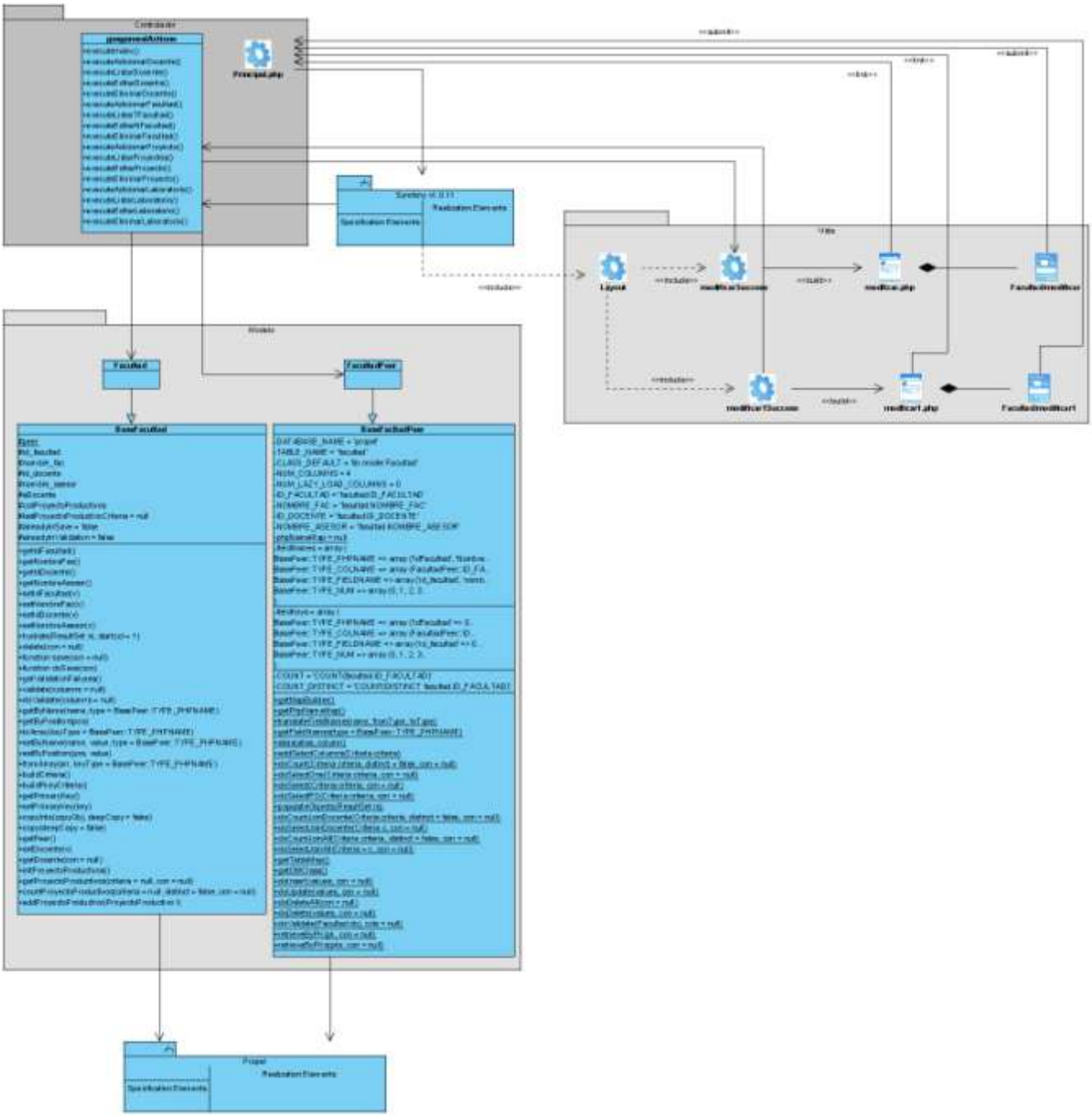
Diagramas Clases de Diseño del Caso de Uso: Gestionar Docente sección Eliminar



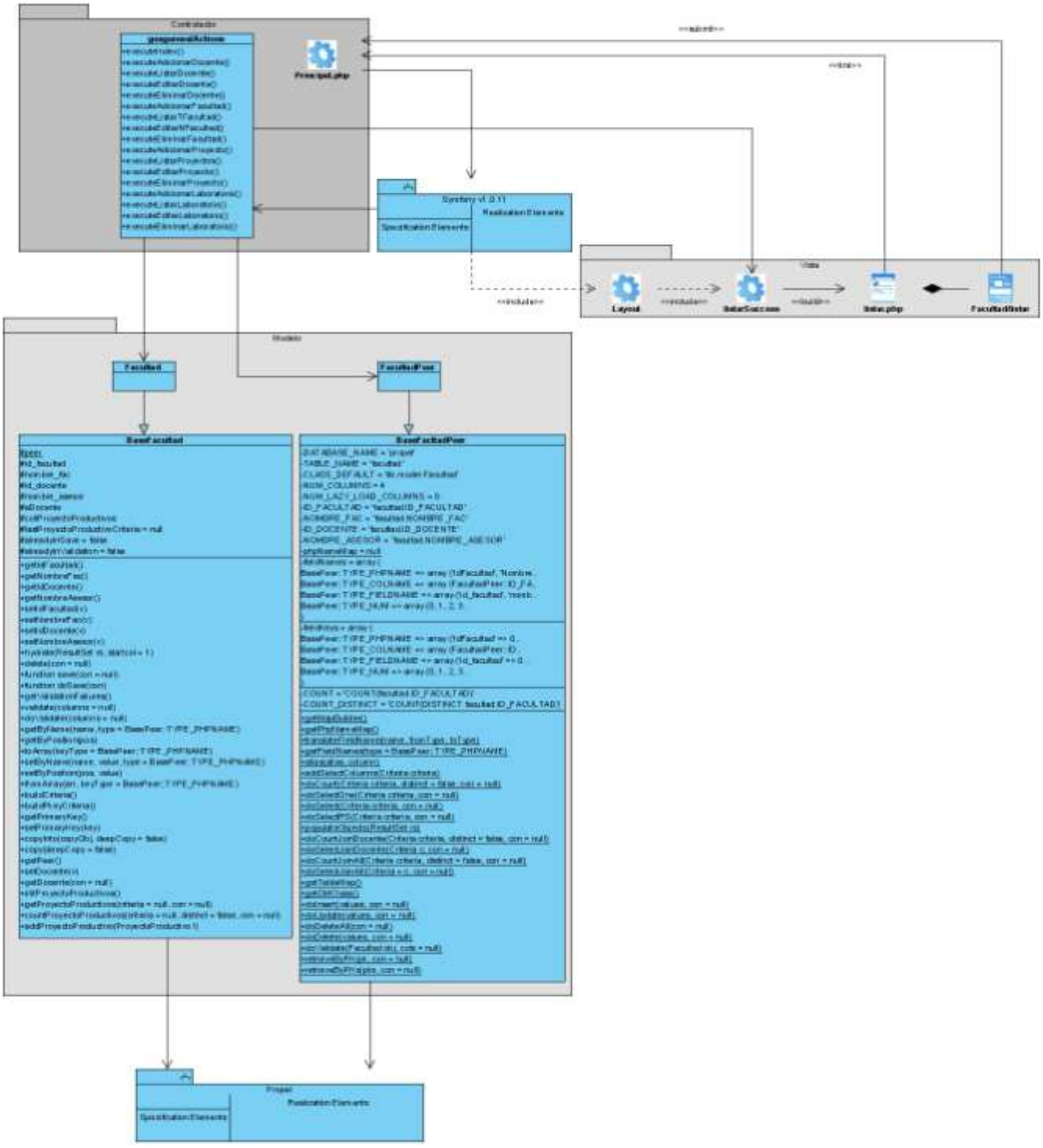
Diagramas Clases de Diseño del Caso de Uso: Gestionar Facultad sección Adicionar



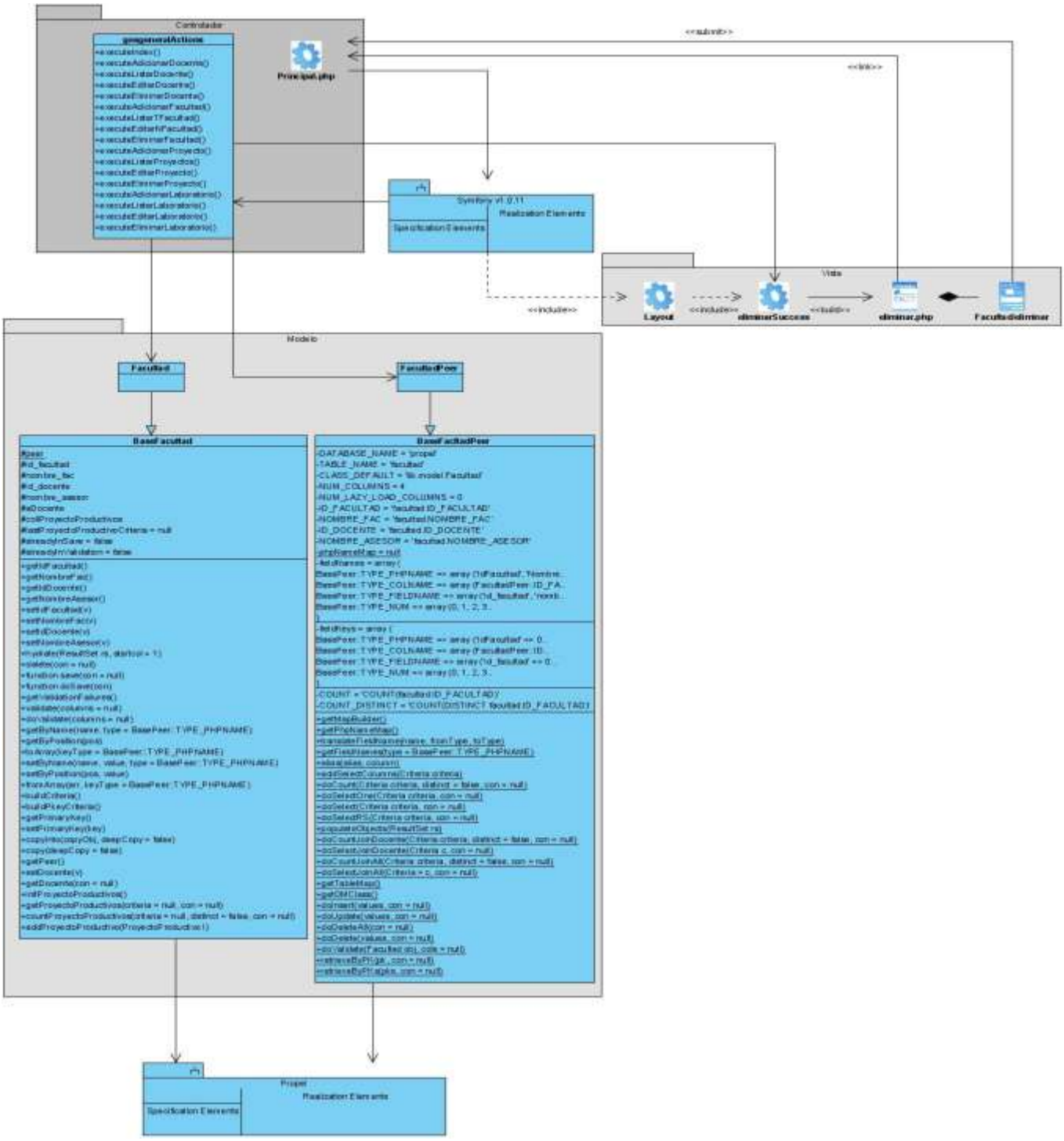
Diagramas Clases de Diseño del Caso de Uso: Gestionar Facultad sección Modificar



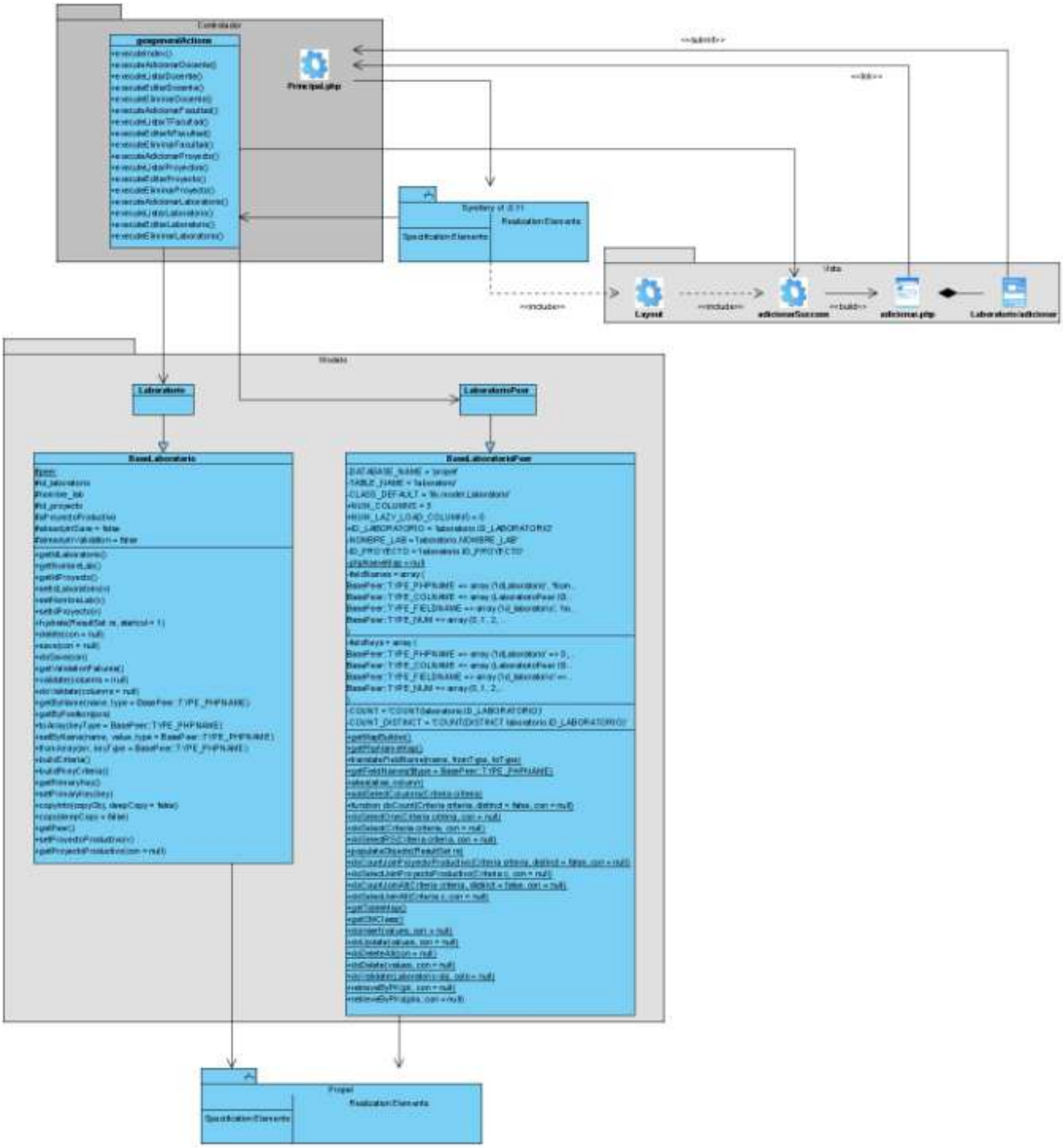
Diagramas Clases de Diseño del Caso de Uso: Gestionar Facultad sección Listar



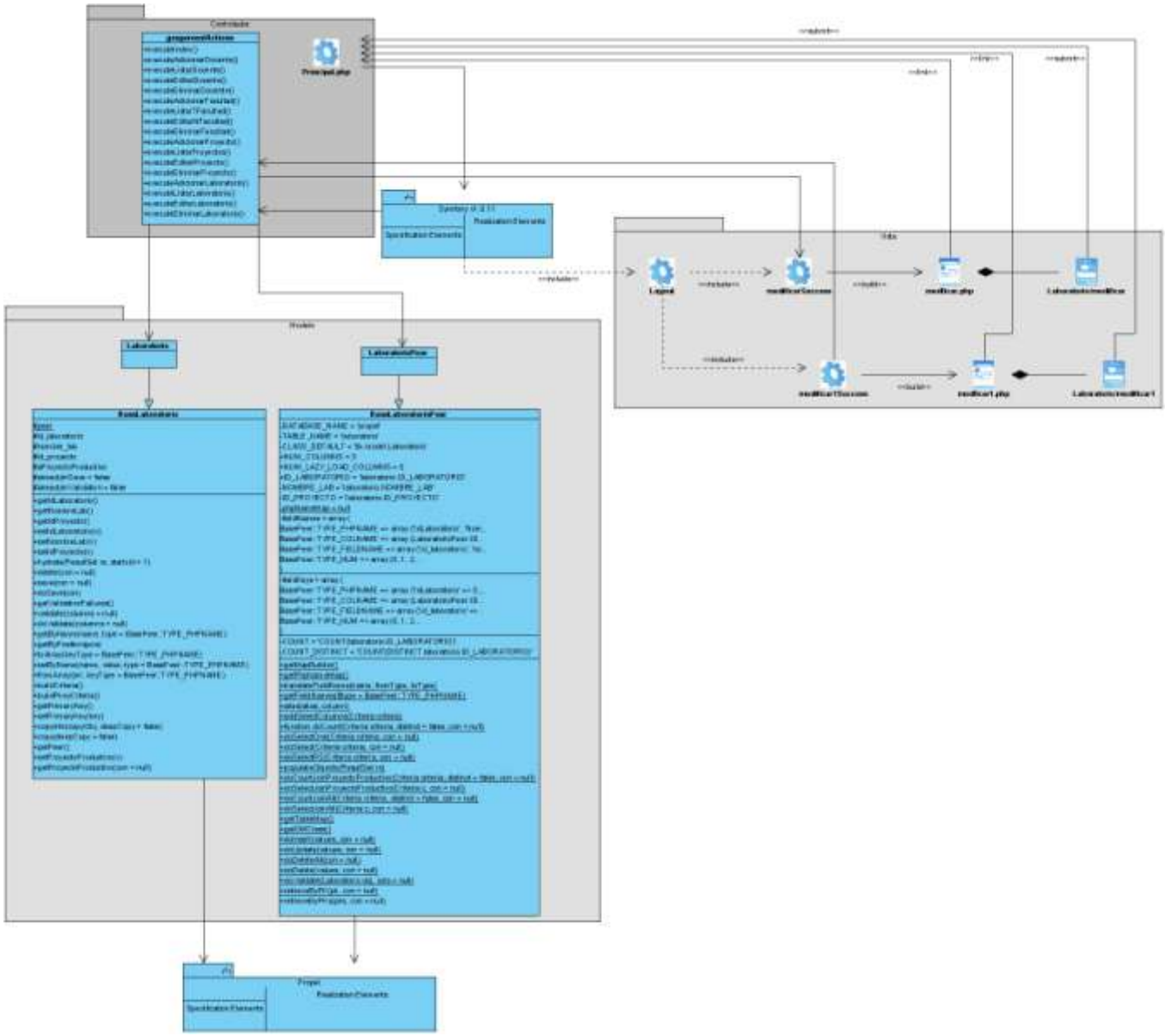
Diagramas Clases de Diseño del Caso de Uso: Gestionar Facultad sección Eliminar.



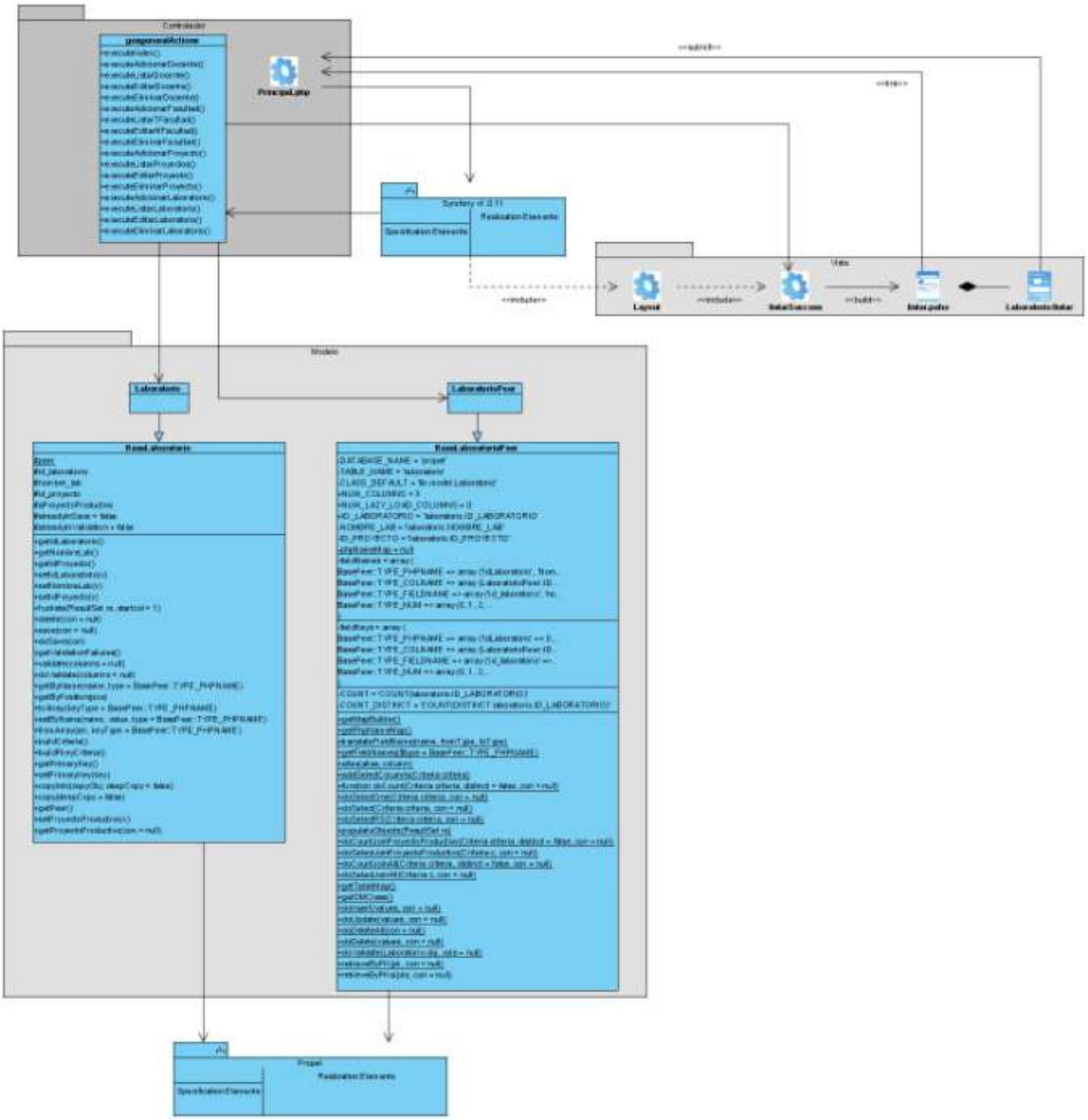
Diagramas Clases de Diseño del Caso de Uso: Gestionar Laboratorios sección Adicionar.



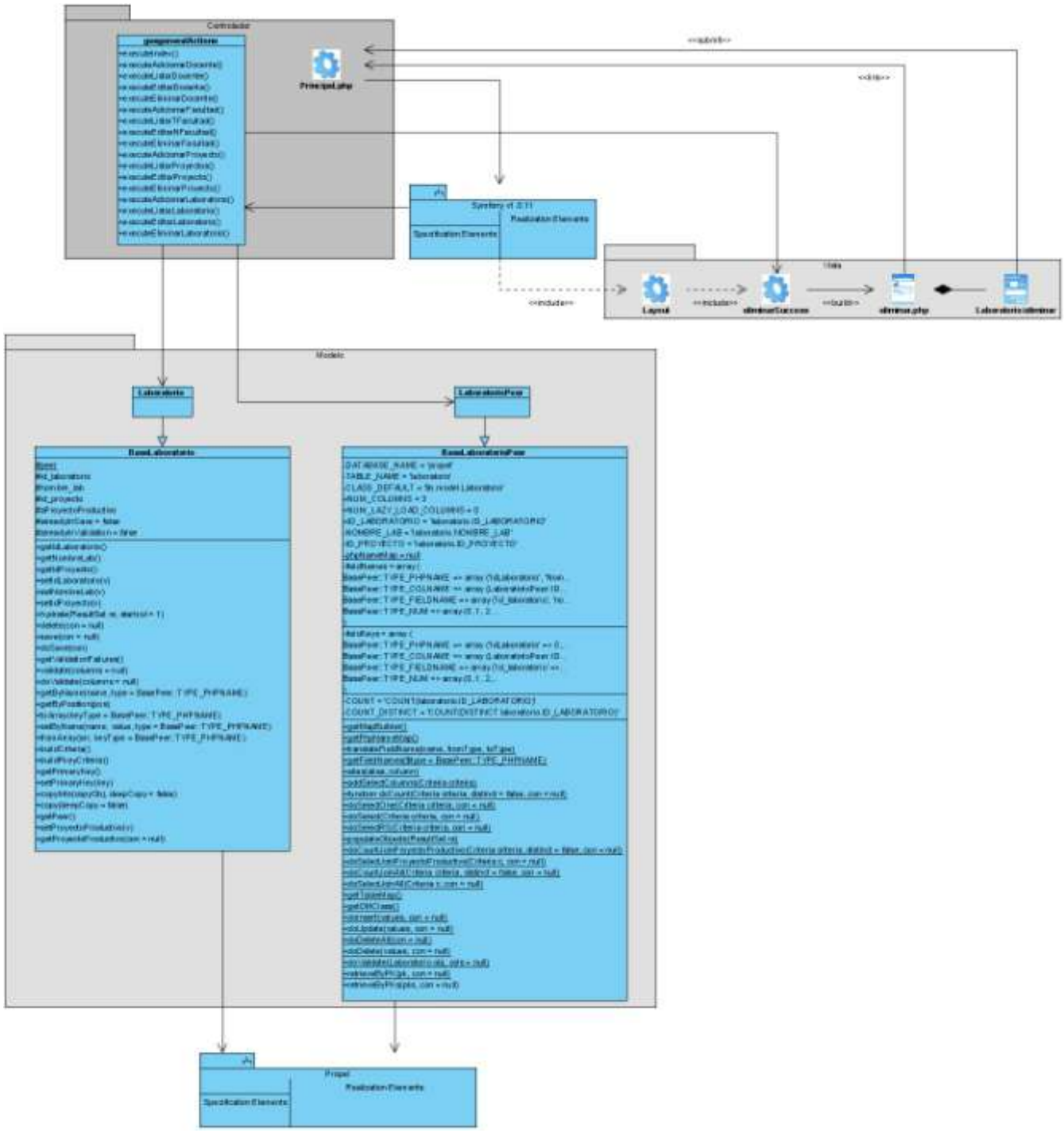
Diagramas Clases de Diseño del Caso de Uso: Gestionar Laboratorios sección Modificar.



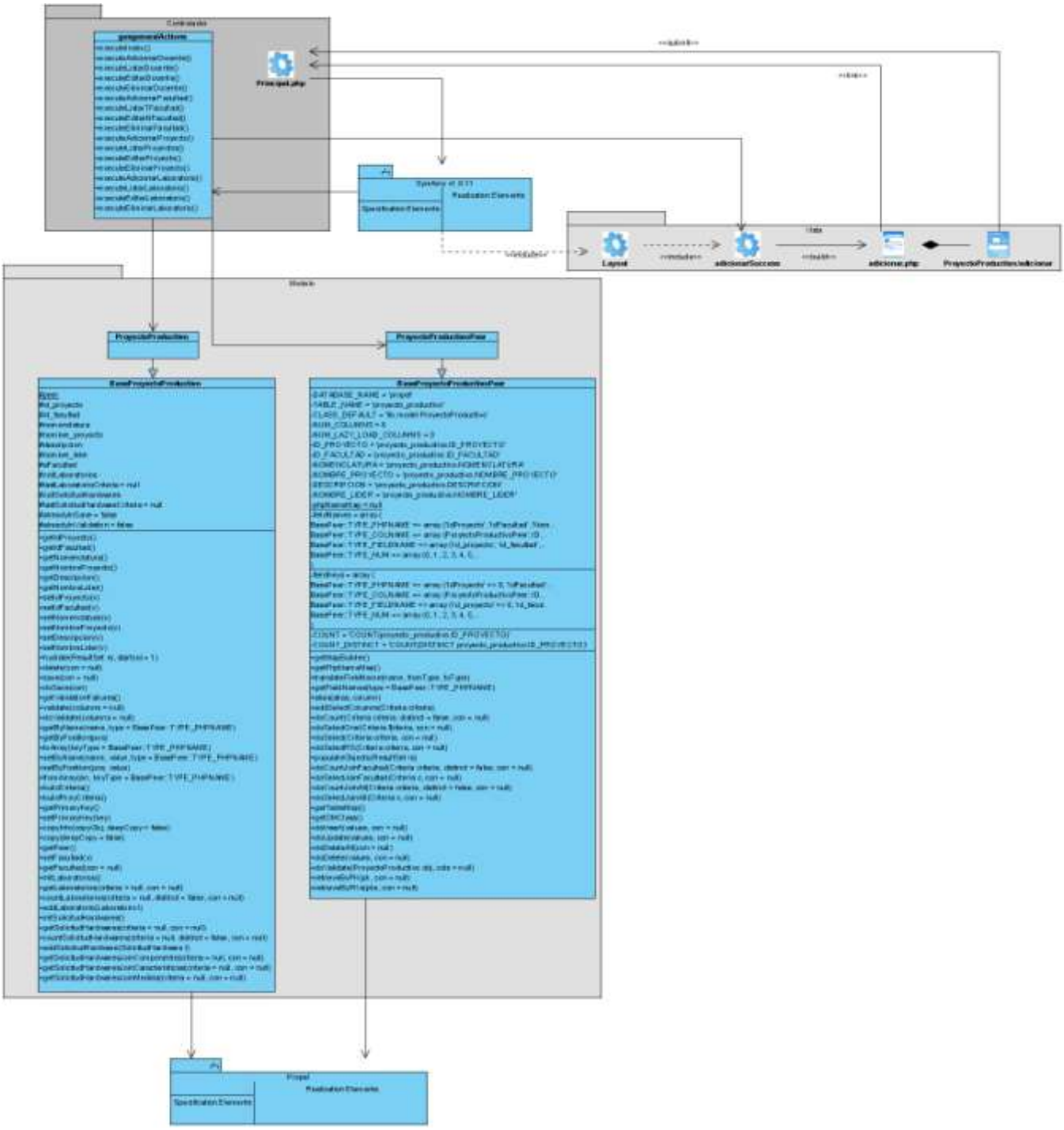
Diagramas Clases de Diseño del Caso de Uso: Gestionar Laboratorios sección Listar



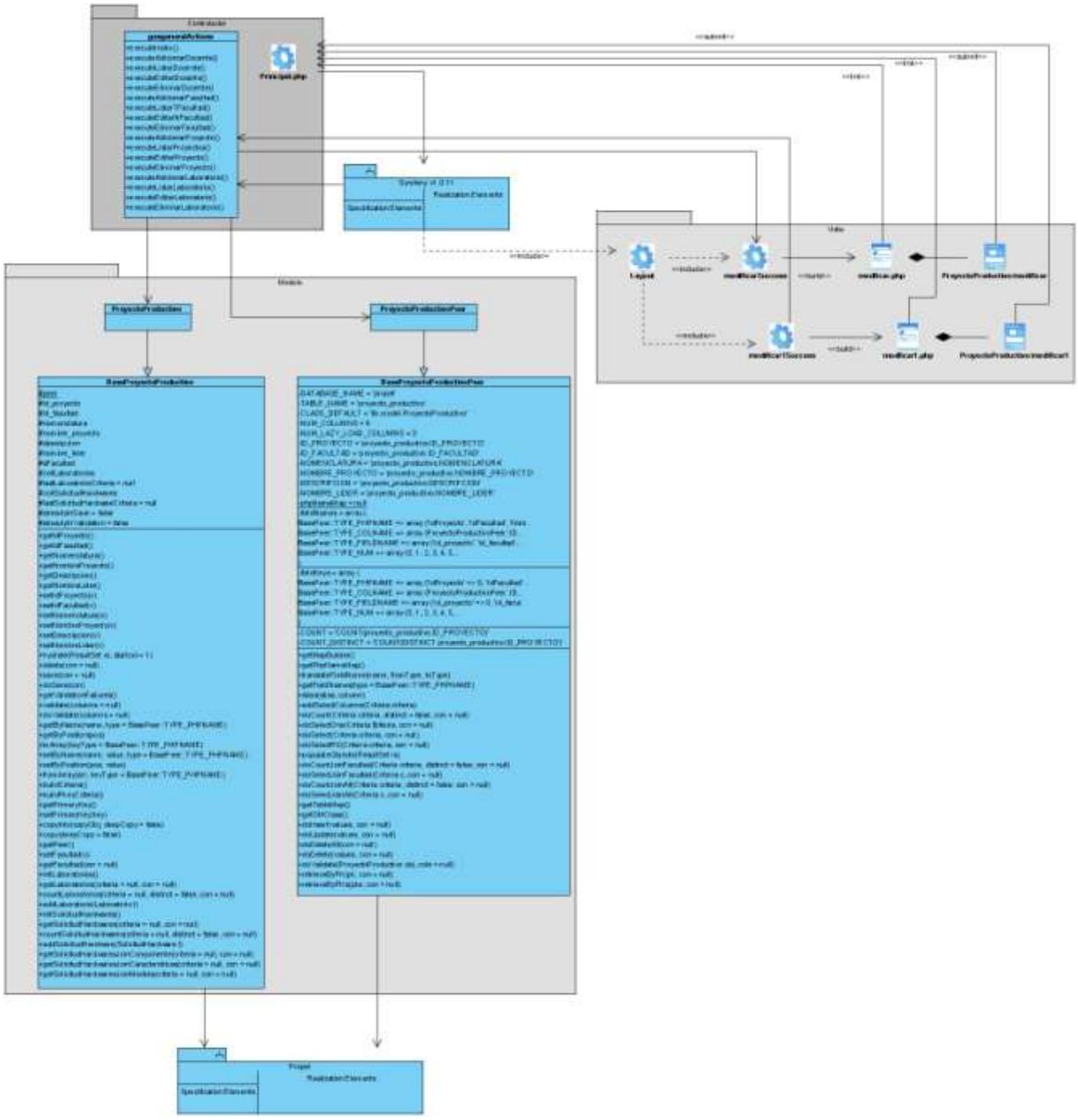
Diagramas Clases de Diseño del Caso de Uso: Gestionar Laboratorios sección Eliminar



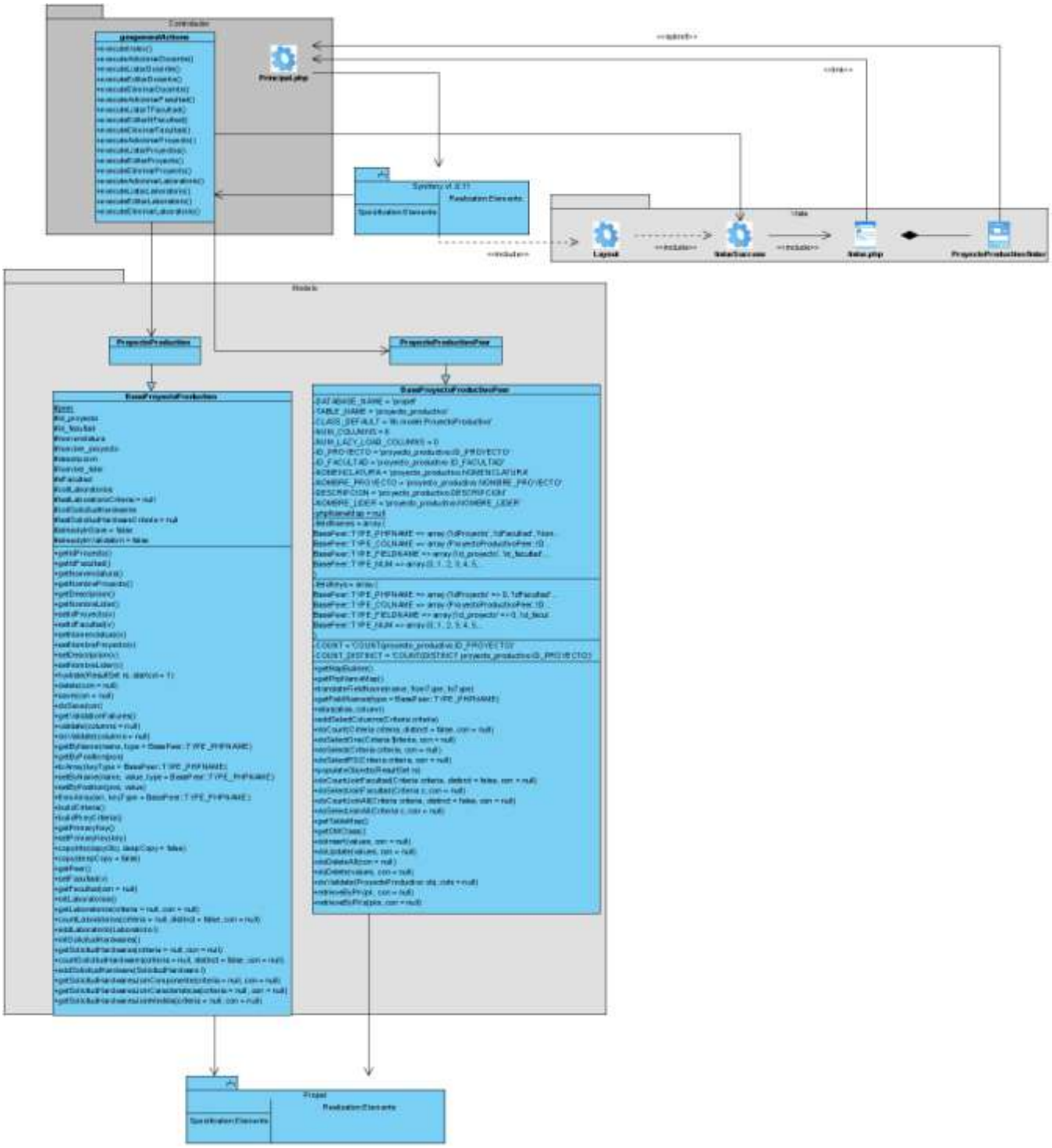
Diagramas Clases de Diseño del Caso de Uso: Gestionar Proyecto Productivo sección Adicionar.



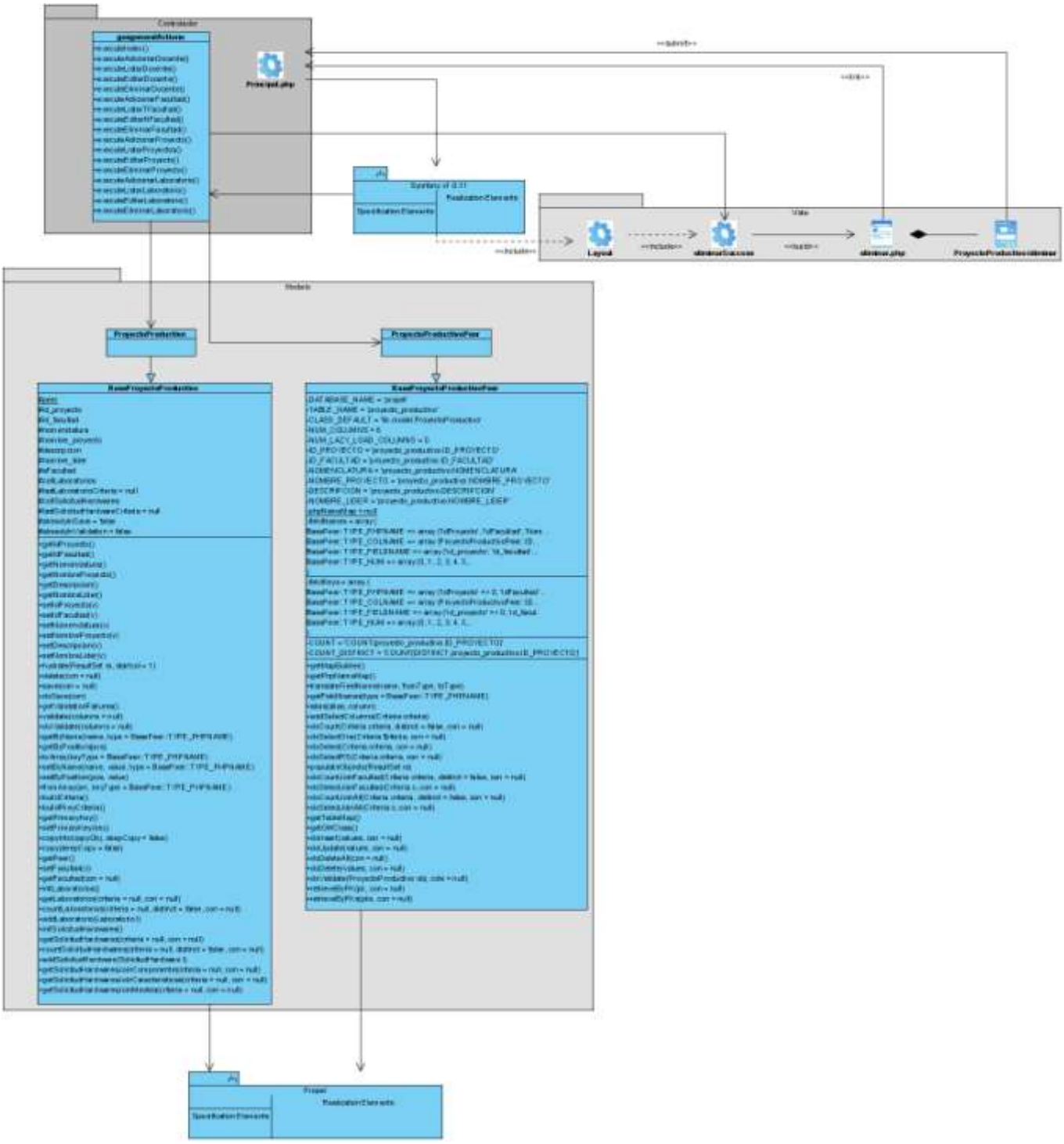
Diagramas Clases de Diseño del Caso de Uso: Gestionar Proyecto Productivo sección Modificar.



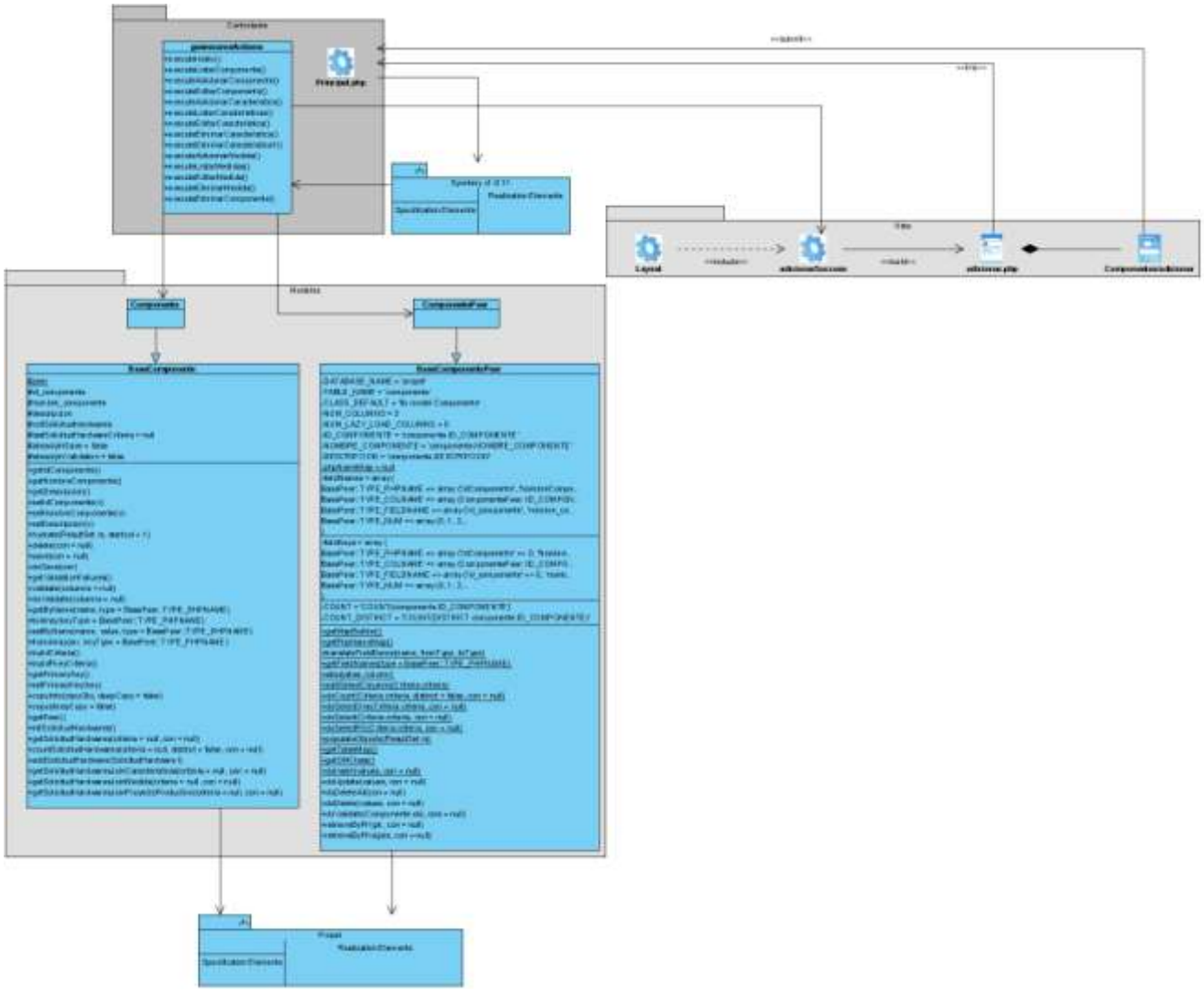
Diagramas Clases de Diseño del Caso de Uso: Gestionar Proyecto Productivo sección Listar.



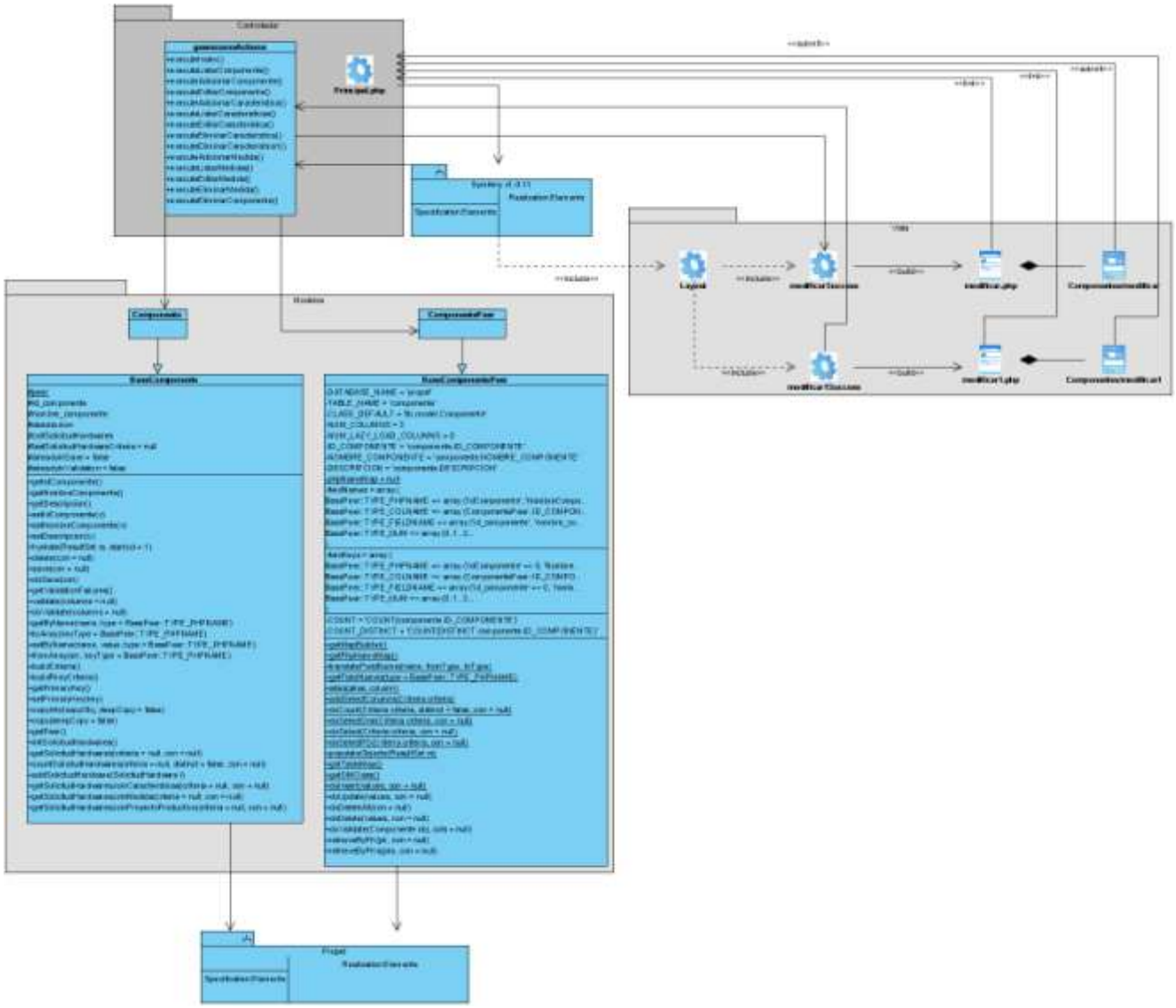
Diagramas Clases de Diseño del Caso de Uso: Gestionar Proyecto Productivo sección Eliminar.



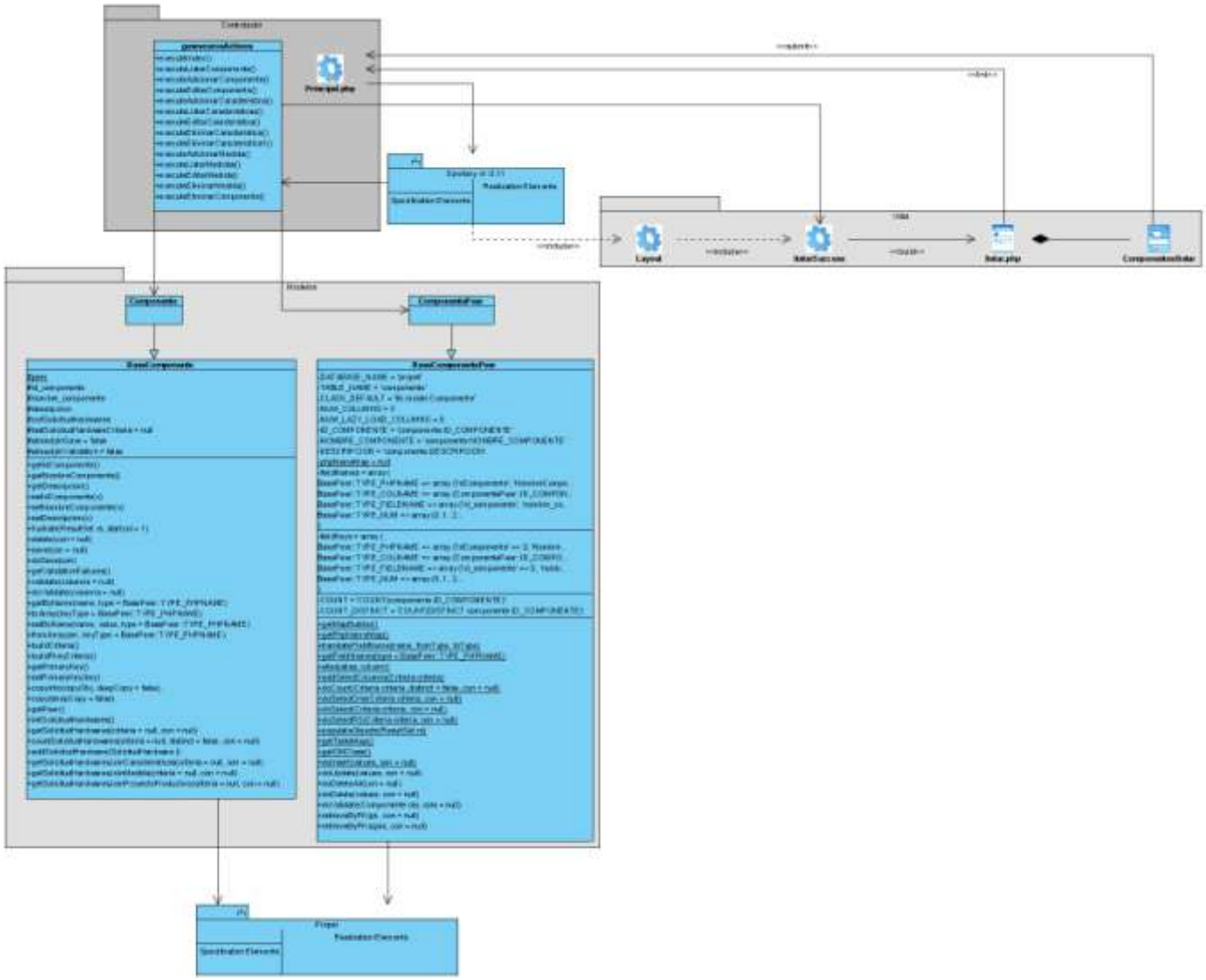
Diagramas Clases de Diseño del Caso de Uso: Gestionar Componentes sección Adicionar.



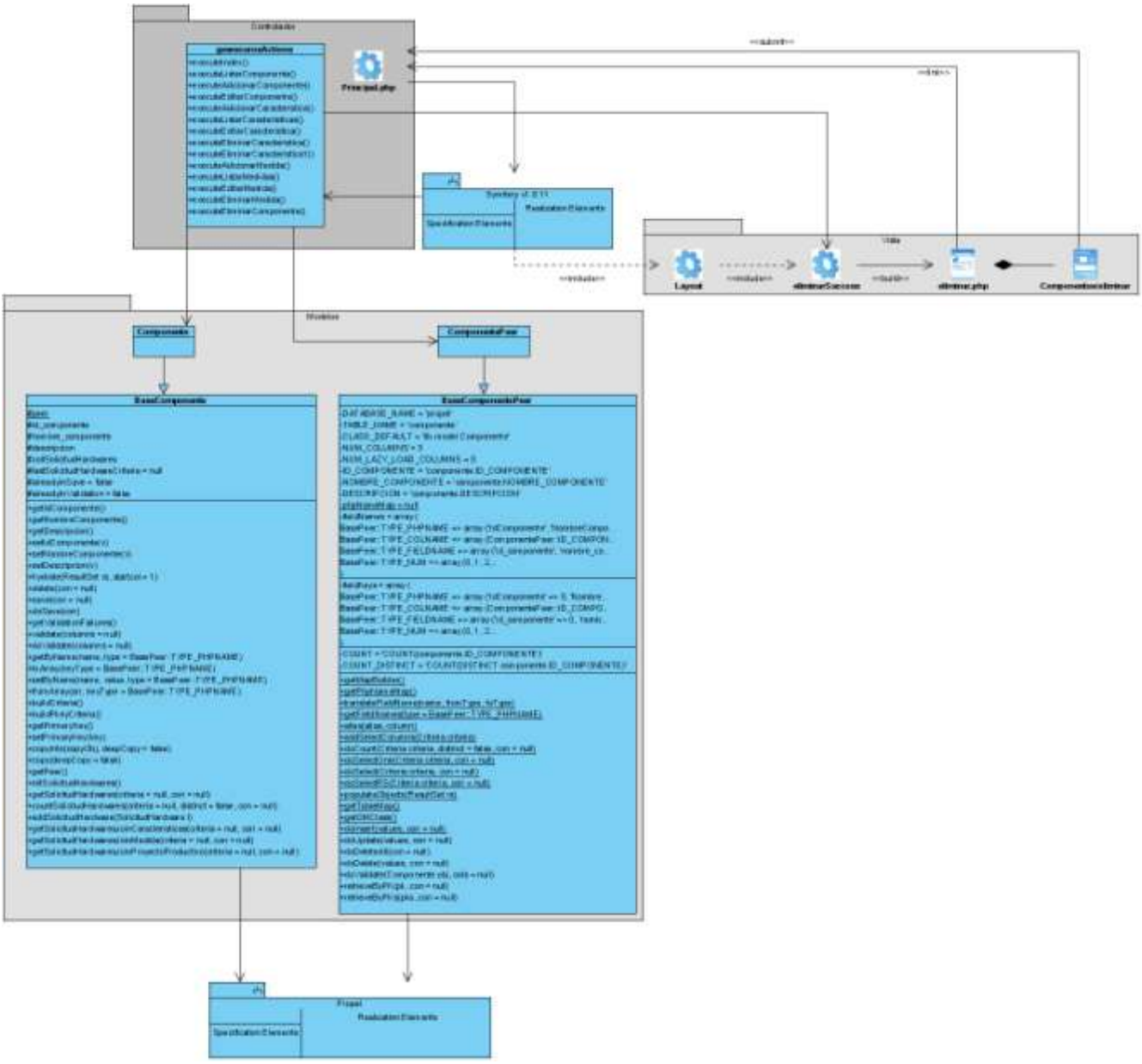
Diagramas Clases de Diseño del Caso de Uso: Gestionar Componentes sección Modificar



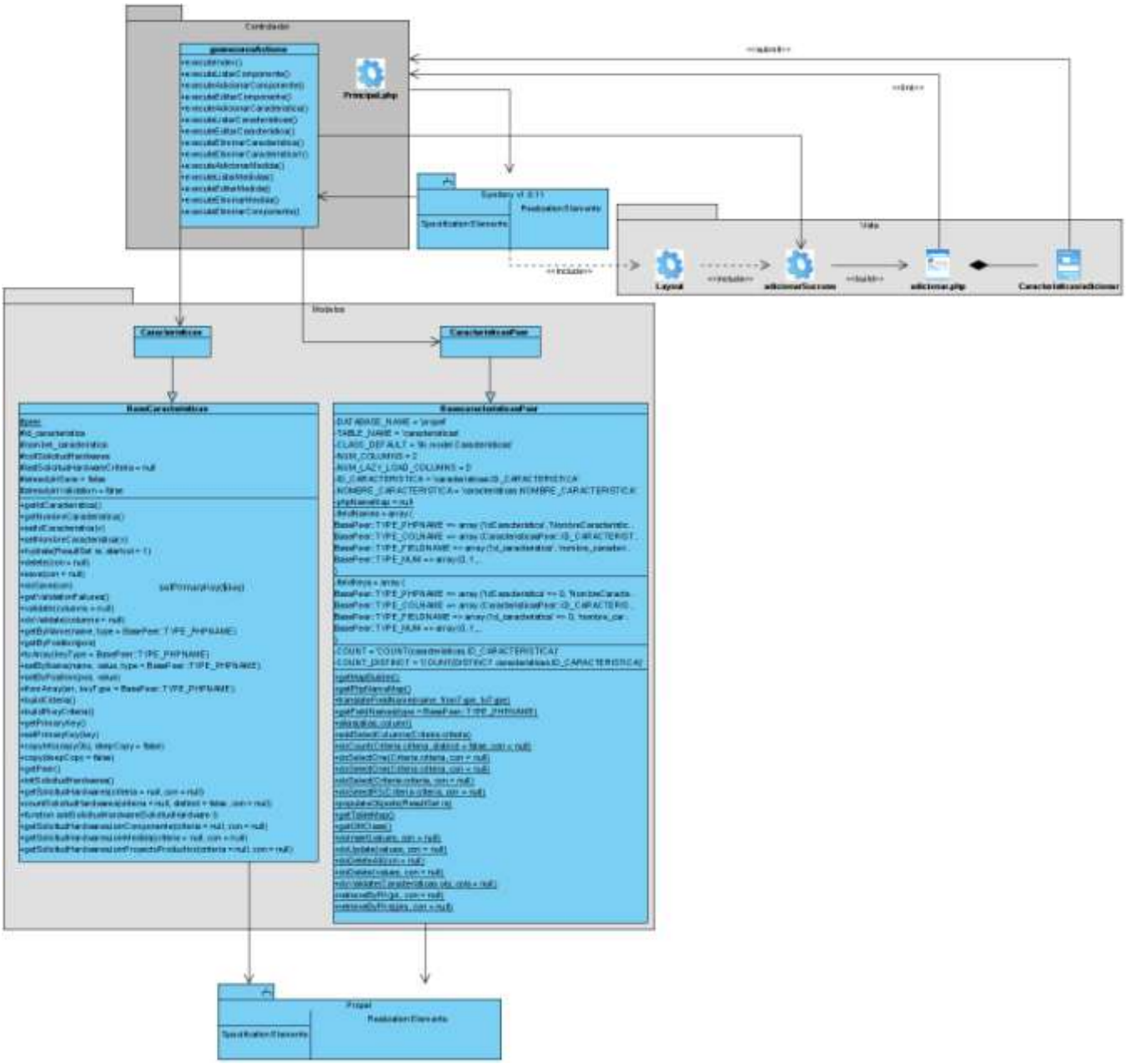
Diagramas Clases de Diseño del Caso de Uso: Gestionar Componentes sección Listar



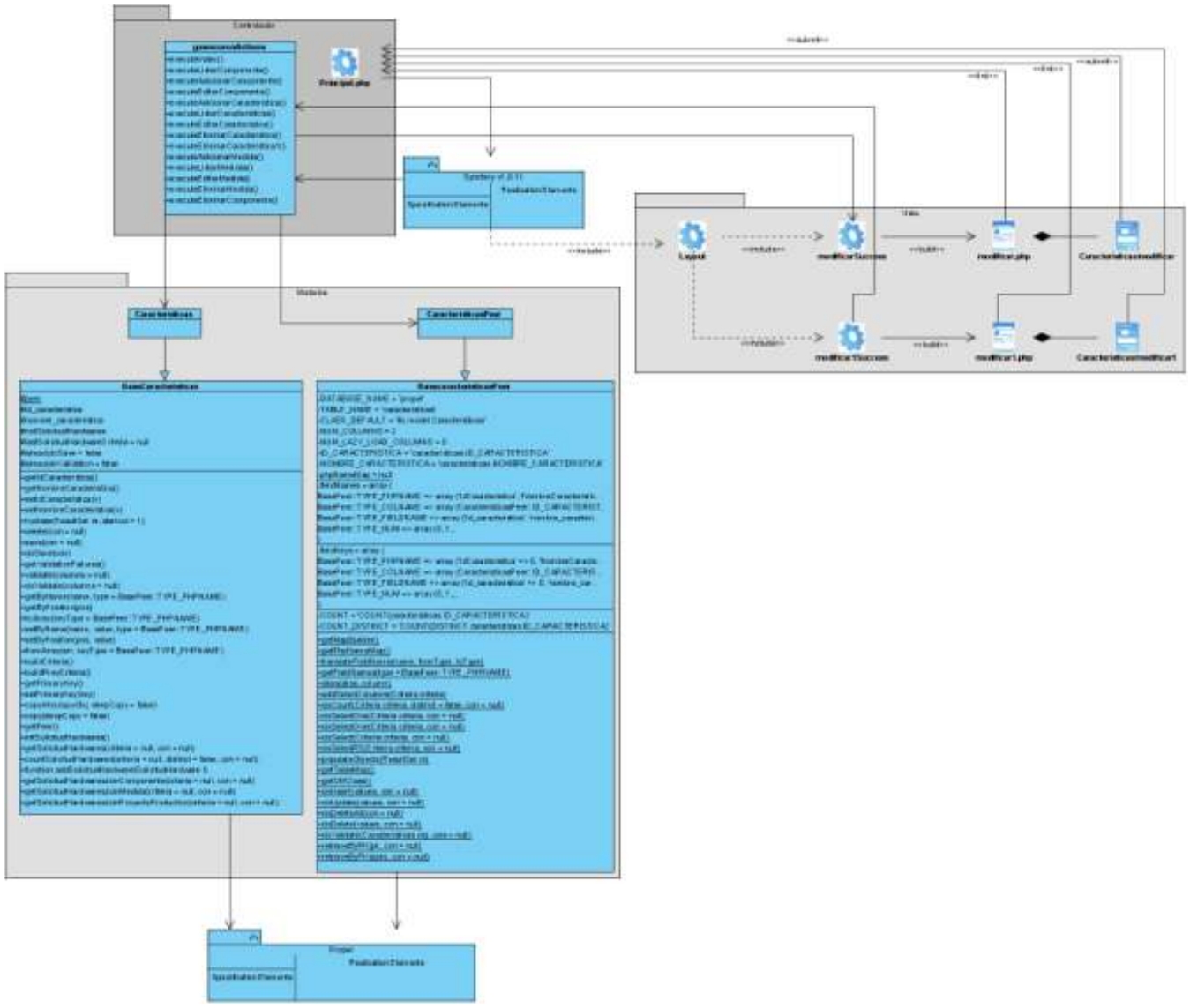
Diagramas Clases de Diseño del Caso de Uso: Gestionar Componentes sección Eliminar.



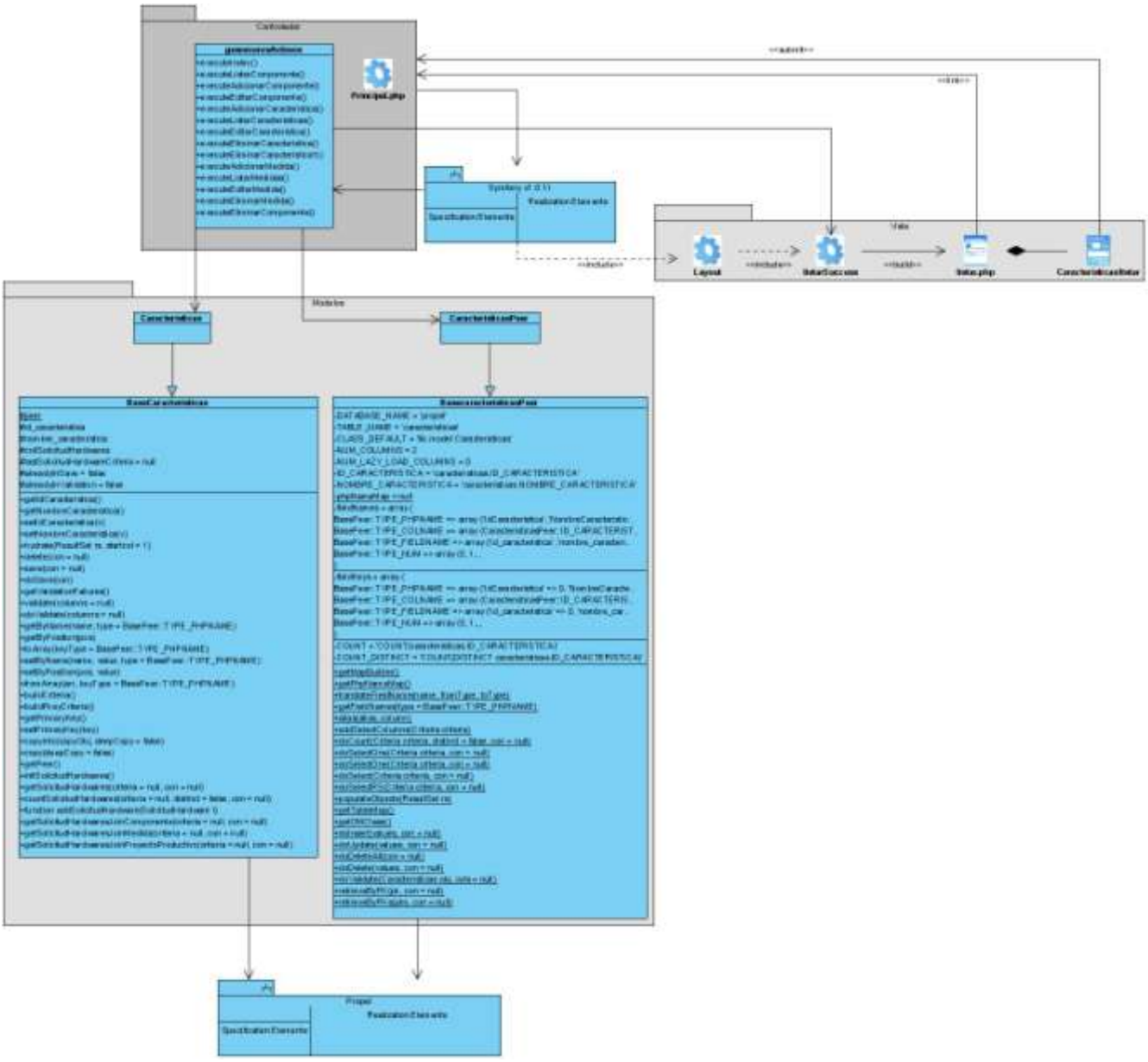
Diagramas Clases de Diseño del Caso de Uso: Gestionar Características sección Adicionar



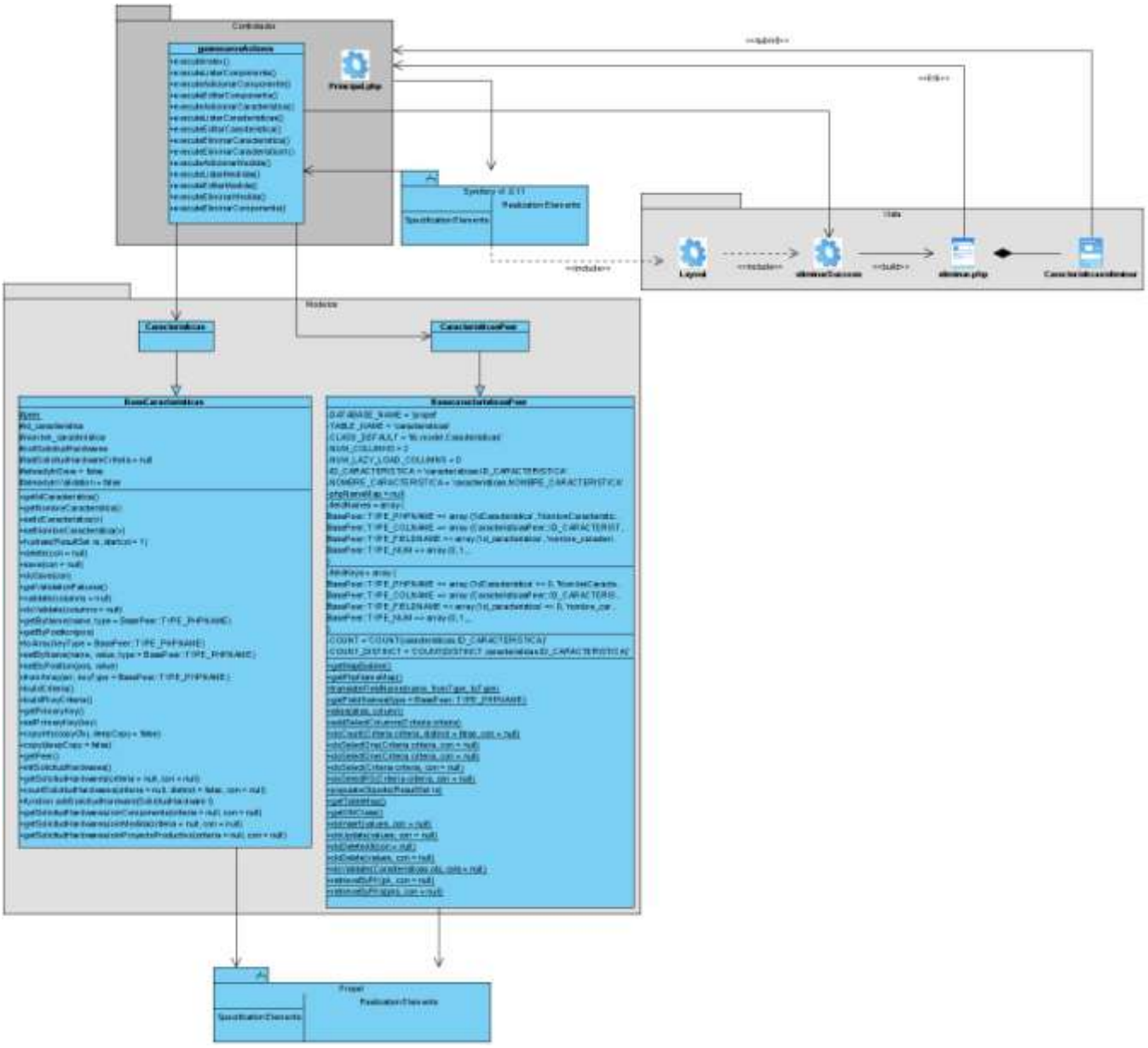
Diagramas Clases de Diseño del Caso de Uso: Gestionar Características sección Modificar



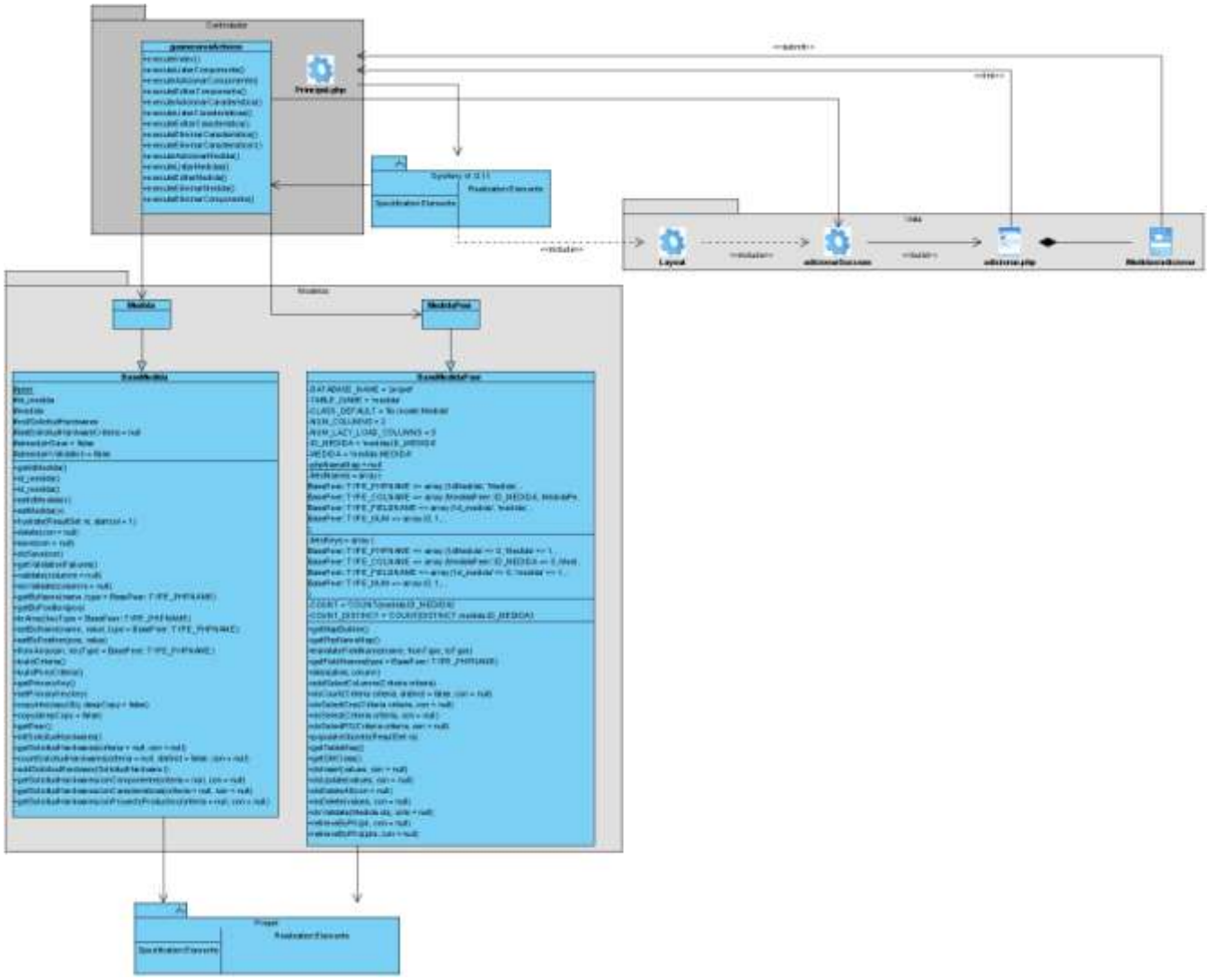
Diagramas Clases de Diseño del Caso de Uso: Gestionar Características sección Listar



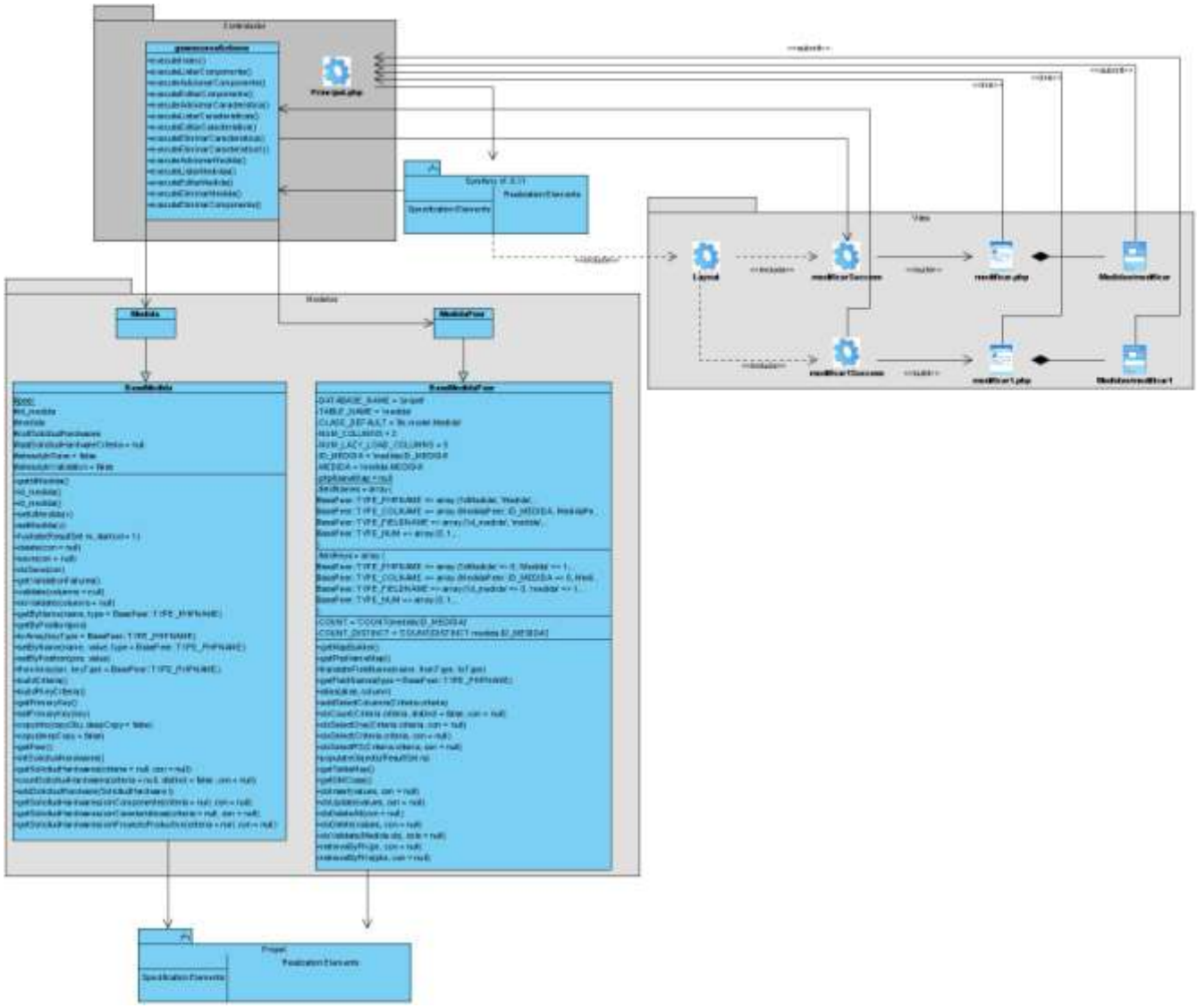
Diagramas Clases de Diseño del Caso de Uso: Gestionar Características sección Eliminar.



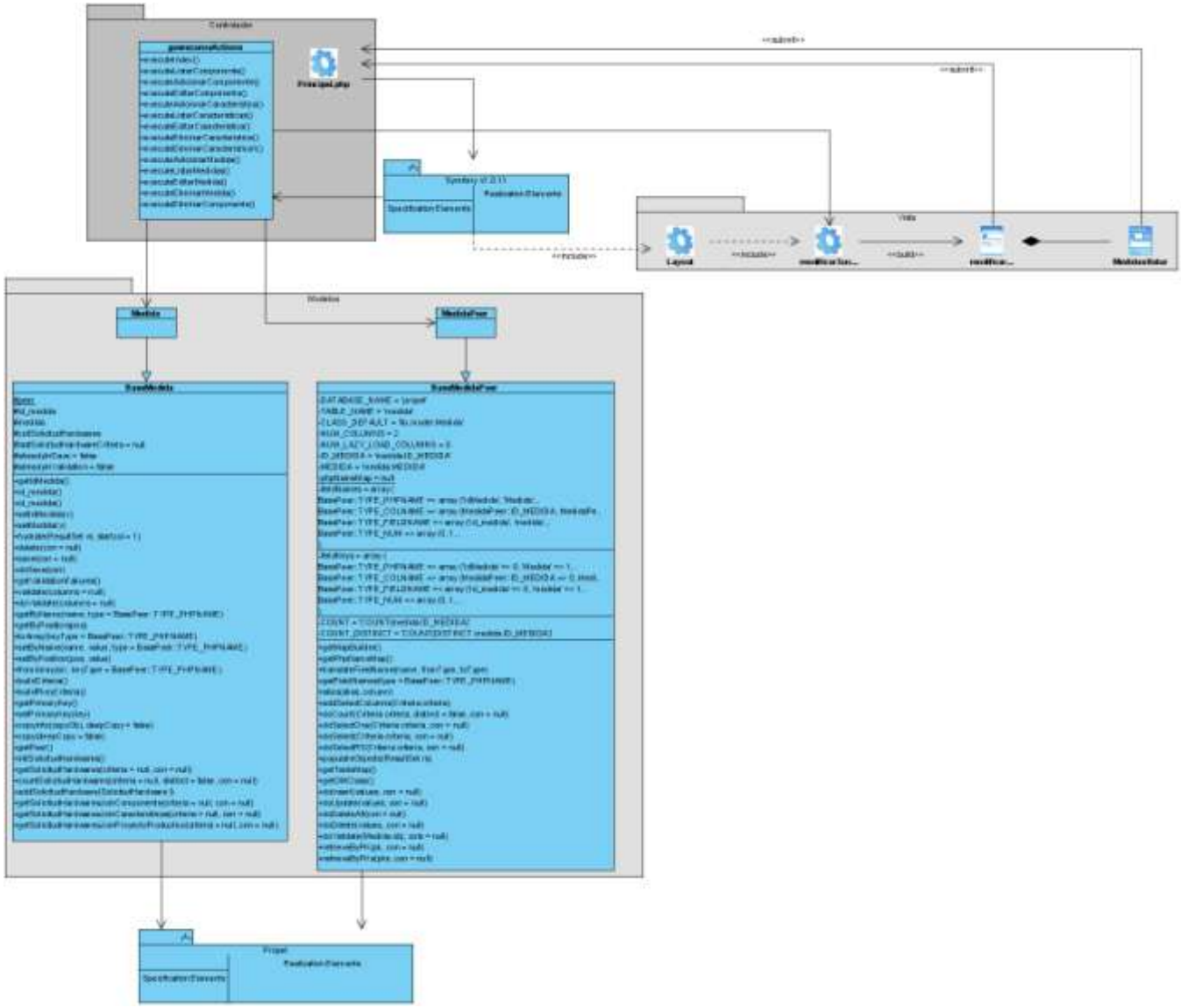
Diagramas Clases de Diseño del Caso de Uso: Gestionar Medidas sección Adicionar



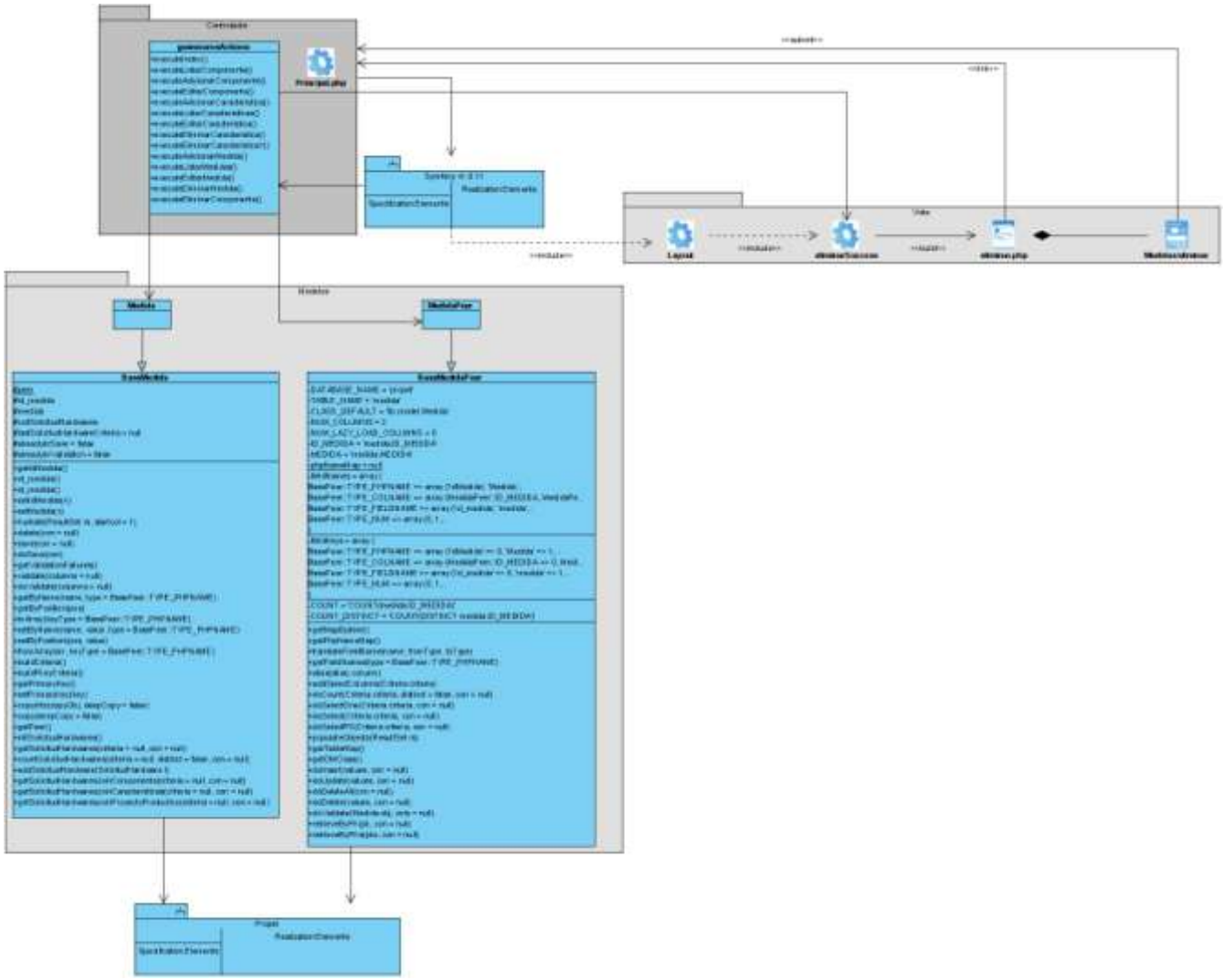
Diagramas Clases de Diseño del Caso de Uso: Gestionar Medidas sección Modificar.



Diagramas Clases de Diseño del Caso de Uso: Gestionar Medidas sección Listar.

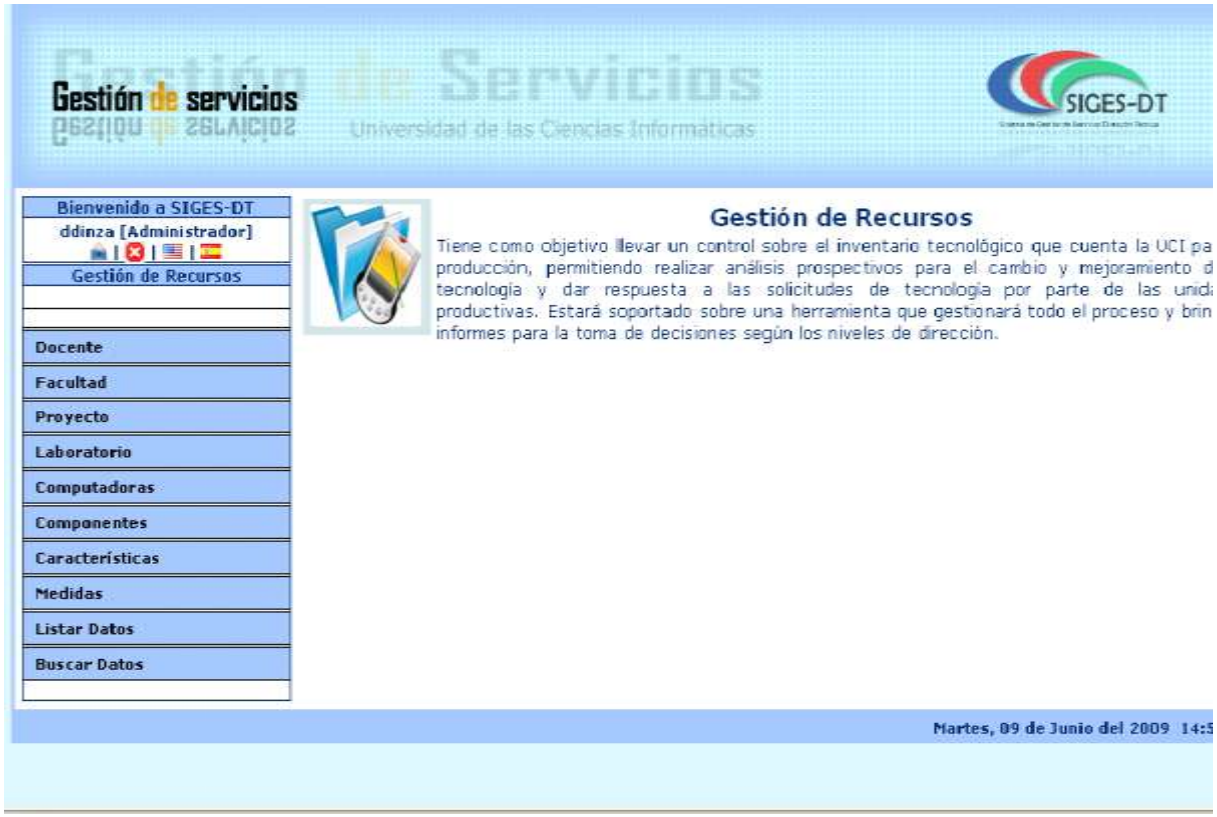


Diagramas Clases de Diseño del Caso de Uso: Gestionar Medidas sección Eliminar

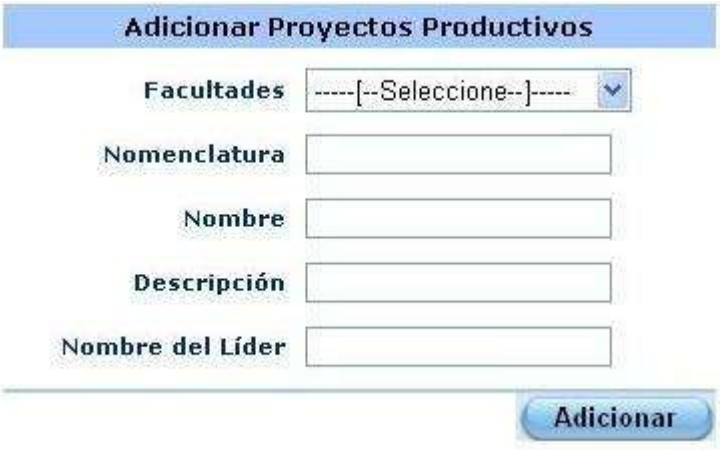


Anexo 4 Prototipos de Interfaz del Sistema de Gestión de Recursos Informáticos.

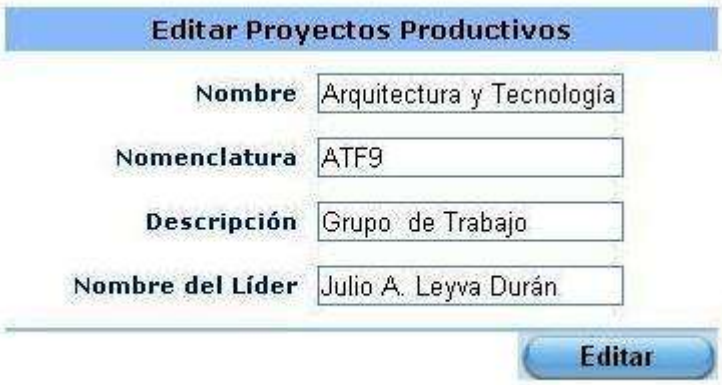
Página Principal del GesRecursos



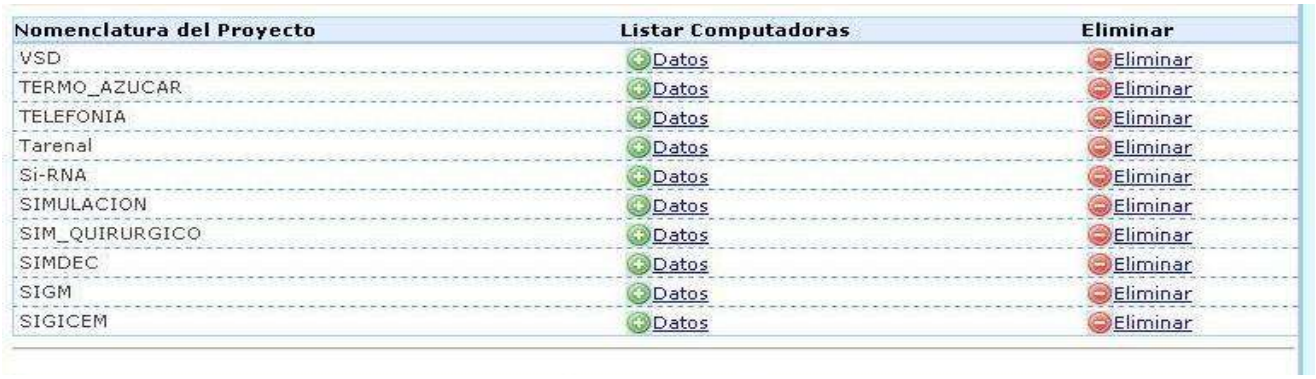
Prototipo de Interfaz de funcionalidad Adicionar Proyecto Productivo.



Prototipo de Interfaz de funcionalidad Editar Proyecto Productivo



Prototipo de Interfaz de funcionalidad Listar Proyecto Productivo



Nomenclatura del Proyecto	Listar Computadoras	Eliminar
VSD	Datos	Eliminar
TERMO_AZUCAR	Datos	Eliminar
TELEFONIA	Datos	Eliminar
Tarenal	Datos	Eliminar
Si-RNA	Datos	Eliminar
SIMULACION	Datos	Eliminar
SIM_QUIRURGICO	Datos	Eliminar
SIMDEC	Datos	Eliminar
SIGM	Datos	Eliminar
SIGICEM	Datos	Eliminar