UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 2 "TELECOMUNICACIONES Y SEGURIDAD INFORMÁTICA"

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE

INGENIERO EN CIENCIAS INFORMÁTICAS

Título: Informatización del procedimiento desarrollado en la UCI para evaluar las premisas técnicas y de mercado que se deben tener en cuenta en la contratación de un proyecto de software.

Autores:

Frank Ernesto Estevez Sanchez

Yovamnis Cordova Perez

Tutores:

Ing. Maylé Díaz Castro

DrC. Rolando Alfredo Hernández León

Ciudad de la Habana Junio del 2009

"Año del 50 Aniversario de la Revolución"

Declaración de Autoría

Declaramos que _____ y _____ somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas (UCI) y a la Facultad 2 para que hagan el uso que estimen pertinente con este trabajo.

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

_____	_____
Firma del Autor	Firma del Autor
Yovamnis Cordova Perez	Frank Ernesto Estevez Sanchez
_____	_____
Firma del Tutor	Firma del Tutor
Ing. Maylé Díaz Castro	DrC. Rolando Alfredo Hernández León

"Lo importante no es hacer cosas extraordinarias sino cosas ordinarias extraordinariamente bien."

Che.

AGRADECIMIENTOS

A nuestros tutores por exigirnos, apoyarnos, guiarnos y ayudarnos a formarnos como profesionales.

A la Universidad de las Ciencias Informáticas por habernos formados y permitirnos ser mejores revolucionarios.

De Frank:

A Dios por guiar mis pasos por el buen camino. A mis padres y familiares por la dedicación, amor y confianza que me han brindado a lo largo de toda mi vida. Al ingeniero Yoel Rivera Suárez por colaborar arduamente en la programación de este trabajo. A mis compañeros que a lo largo de estos cinco años han batallado conmigo día a día. A todos los que de una forma u otra fueron partícipes de mi formación como profesional y como una mejor persona.

De Yovamnis:

A Dios por estar cada día a mi lado y permitirme llegar hasta donde lo he hecho. A mis padres por el amor que siempre me han brindado. A mi querida hermana Yanine por ser tan dulce y comprensiva. A mi tío Diosmedes por el apoyo en todos estos años de mi vida estudiantil. A mis amigos del alma: Sandy Noa Cabrera y Antonio Hernández Domínguez por tolerarme y apoyarme siempre a lo largo de mi estudio en la Universidad.

DEDICATORIA

De Frank:

A mis padres que siempre han estado conmigo e inculcaron en mí el deseo de ser cada día mejor, a mis hermanos, a mi esposa por confiar en mí y darme las fuerzas que en algún momento me faltaron para seguir adelante y a toda mi familia que siempre me ha apoyado incondicionalmente.

De Yovamnis:

A mis padres que desde pequeño siempre me inculcaron la necesidad de superarme, de no conformarme y cada día aprender algo nuevo. A mi querida hermana, pues sin ella todos estos años hubiesen sido imposibles. A mis mejores amigos en Baracoa: Leo y Raulier.

RESUMEN

En la Universidad de las Ciencias Informáticas el proceso de evaluación de las premisas para la contratación de un proyecto informático no garantiza la información necesaria para aceptar o rechazar los proyectos, provocando que a menudo estos comiencen a desarrollarse sin un previo análisis de factibilidad técnica y de mercado.

En el presente trabajo se informatizó un procedimiento desarrollado en la Universidad de las Ciencias Informáticas para evaluar las premisas técnicas y de mercado que se deben tener en cuenta para contratar un proyecto de software, garantizando de esta forma la información imprescindible para tomar decisiones acertadas en torno a los mismos.

Para el análisis del sistema se siguieron los pasos que propone el *Rational Unified Process* (RUP), se utilizó el Lenguaje Unificado de Modelado (UML) como soporte a la metodología y el *Visual Paradigm* para el modelado visual. Con la informatización de este procedimiento se logró que la universidad cuente con un sistema capaz de realizar el proceso de evaluación de las premisas para aceptar o rechazar los proyectos informáticos, ganando en capacidad organizativa, confiabilidad y seguridad.

PALABRAS CLAVES

Evaluación, proyecto, software, premisas, factibilidad.

ÍNDICE

INTRODUCCIÓN

Las decisiones desempeñan un papel fundamental en la vida de las personas, nadie ignora que además del hombre comer, vestir, calzar y otros procesos inherentes en su vida, también depende de las decisiones que sea capaz de tomar o que en la gran mayoría de los casos tomen otras personas. Las mismas pueden tomarse con total seguridad o bajo cierto grado de incertidumbre, pueden ser programadas o no, lo que sí es seguro que de ellas dependa el futuro de cada persona, empresa o país.

Las decisiones pueden ser complejas o sencillas, todo obedece al problema en cuestión; una de las más difíciles son las que repercuten en el bienestar social de todo un pueblo, en el desarrollo económico de una nación o de una empresa; ante estas, se busca la opinión de personal calificado, de expertos en la situación problemática a resolver.

Cuba, como nación, no está ajena a la toma de decisiones acertadas como alternativa para poder fortalecer su economía; es visible a los ojos de su pueblo el esfuerzo que realiza por impulsar entre otros aspectos la producción de software; tan encomiable labor ha sido asignada principalmente a la Universidad de las Ciencias Informáticas (UCI).

La UCI es un centro creado a raíz de la Batalla de ideas, cuya misión es formar de manera continua profesionales integrales comprometidos con la Patria, ser el soporte de la informatización del país y la competitividad internacional de la industria cubana del software. Por ello es necesario lograr una buena organización y control del proceso productivo.

Para desarrollar un proyecto de software que surge con el objetivo de dar respuesta a una situación problemática que está afectando la vida económica y social de una empresa, territorio o país se tienen que conjugar varios factores, sería arriesgado someterse a la realización de una tarea sin antes analizar el alcance de la misma, sin verificar si se cuenta con los recursos humanos necesarios para su acometido. Es imprescindible estudiar también la factibilidad en cualquiera de sus tres dimensiones: técnica, de mercado y económica, valorar los riesgos para así tener una idea de nuestra posibilidad de éxito.

Qué debemos evitar y hacia dónde debemos encaminar nuestros esfuerzos constituyen piezas claves para lograr una buena y competitiva producción. El doctor Hernández en su libro: “**Curso básico de**

gestión de proyectos", expresa: **"**Para elaborar un proyecto de investigación-desarrollo o de innovación tecnológica y estar seguro del éxito es necesario tener una justificación sólida del mismo, lo que facilitará obtener financiamiento para su ejecución e introducción en la práctica social. Esta justificación se apoyará en los estudios de factibilidad técnica, económica y de mercado". (1)

Por los estudios y consultas realizadas se puede decir que en la actualidad no se cuenta con un sistema que evalúe las premisas para aceptar o rechazar un proyecto informático. La Universidad de las Ciencias Informáticas tiene un procedimiento utilizado por el personal de la Alternativa Bolivariana para la Exportación de Tecnologías (ALBET) que se emplea en la evaluación de proyectos que llegan a la misma.

Con este procedimiento se compendian informaciones relacionadas con la empresa: nombre, fecha de fundación, estado legal de la compañía, cantidad de empleados, publicaciones, bases de datos, software, inversionista principal, entre otros. (ver anexo 1) Estas secciones aportan un buen número de datos acerca de la misma pero no brindan la información necesaria para aceptar o rechazar los proyectos de software, dando así origen a que el **problema Científico** del presente trabajo de diploma sea que: el procedimiento existente en la UCI no garantiza con suficiente agilidad y veracidad la información necesaria para aceptar los proyectos y garantizar su contratación con alta probabilidad de éxito.

El **Objeto de estudio** se enmarca en el**:** Proceso de contratación de proyectos de software. Se establece como **objetivo general:** Informatizar el método Multicriterio desarrollado en la UCI que permita una evaluación ágil y veraz de las premisas técnicas y de mercado que se deben tener en cuenta para contratar proyectos de software con alta probabilidad de éxito y como **campo de acción:** las premisas para contratar un proyecto de software.

Analizado y expuesto todo lo anterior se plantea como **hipótesis** que**:** si se informatiza el procedimiento desarrollado en la UCI para evaluar la factibilidad técnica y de mercado de los proyectos de software se podrá obtener una evaluación ágil y veraz, y así lograr la contratación de estos con alta probabilidad de éxito.

Variables

Variable independiente: Procedimiento para evaluar las premisas que se deben tener en cuenta para contratar un proyecto de software.

Variable dependiente: Información necesaria para rechazar o aceptar un proyecto.

Los **objetivos específicos** de este trabajo lo constituyen:

1. Elaborar el marco teórico de la investigación.
2. Definir el procedimiento de evaluación de las premisas técnicas y de mercado.
3. Informatizar el procedimiento de evaluación de las premisas técnicas y de mercado.

Este trabajo está conformado por cinco capítulos:

El **Capítulo 1**: Fundamentación teórica: en este capítulo se hace un estudio del estado del arte. Se analizan un conjunto de herramientas posibles a utilizar para realizar el análisis, diseño e implementación del sistema, justificando el por qué de su uso.

El **Capítulo 2**: Características del sistema: En este capítulo se modela el negocio, que proporciona una base para determinar las actividades fundamentales que serán objeto de automatización y a partir de las funcionalidades identificadas se realiza la propuesta del sistema.

El **Capítulo 3**: Análisis y diseño del sistema: En este capítulo se modelan los diagramas de clases del análisis y del diseño, así como los diagramas de interacción correspondientes.

El **Capítulo 4:** Implementación: En este capítulo se modelan los diagramas de despliegue y de componentes.

El **Capítulo 5:** Estudio de Factibilidad: En este capítulo se evalúa la factibilidad y beneficios del sistema propuesto, utilizando el método de estimación por Puntos de Casos de Uso. Se obtienen valores de importantes indicadores como son: esfuerzo y tiempo de desarrollo del sistema.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

Las nuevas tecnologías han impactado en la vida de las personas, las empresas y su forma de laborar. Las mismas representan uno de los aspectos determinantes en el funcionamiento dinámico de las instituciones. En la medida que el concepto de nuevos productos o servicios informáticos se ponga a disposición de la sociedad, mayor será la eficiencia en algunos sectores. La correcta selección de las tecnologías a utilizar para desarrollar proyectos informáticos, adecuándose a las características del cliente, será de suma importancia para lograr un producto que cumpla con los requerimientos planteados por la principal beneficiada: la sociedad. En el presente capítulo se definirá un conjunto de conceptos relacionados con la gestión de proyectos. Se hará un análisis sobre el procedimiento de evaluación de premisas técnicas y de mercado: Multicriterio, así como de la tendencia del uso de aplicaciones web para los software de gestión. También se realizará un recorrido informativo a través de las tecnologías, donde se analizarán las ventajas y desventajas de las mismas, proponiendo lenguajes, gestor de base de datos, metodología a utilizar y las herramientas case para modelado de sistemas.

1.2 Métodos utilizados en la investigación

Con el propósito de desarrollar las tareas propuestas, se utilizaron los métodos de investigación siguientes:

Métodos Teóricos: Se utilizaron para conocer el estado del arte de los procedimientos de evaluación de premisas.

- **Analítico – sintético:** Este método fue empleado en la revisión bibliográfica la cual está conformada por un conjunto de libros, publicaciones y documentos en soporte electrónico, que se encuentra situada en Páginas Web, Internet, Trabajos de Curso, entre otros, permitiendo sintetizar todos los apuntes y datos tomados al respecto; extrayendo los elementos comunes que se relacionan con el objeto de estudio.

- **Hipotético – deductivo:** es la vía primera de inferencias lógicas deductivas para arribar a conclusiones particulares a partir de la Hipótesis, que después se pueden comprobar experimentalmente. (3) Este método posibilitó que a partir de la hipótesis planteada y siguiendo las reglas lógicas de deducción se llegase a nuevas conclusiones similares a las predicciones inferidas con los hechos científicos, comprobándose la veracidad de la misma.

- **Inductivo - deductivo:** A partir de la información con que se contaba al inicio del trabajo y mediante la inducción se elaboró una hipótesis del problema científico planteado, para a través de la deducción organizar los datos y llegar a formar la teoría sobre la informatización del procedimiento para la evaluación de las premisas técnicas y de mercado que se deben tener en cuenta en la contratación de un proyecto de software.

Métodos Empíricos: Se utilizaron en la recogida de la información para poder entender y modelar la situación que tiene lugar en el proceso de evaluación de premisas, así como obtener los requerimientos del sistema.

- **Las entrevistas:** son un excelente instrumento heurístico para combinar los enfoques prácticos, analíticos e interpretativos implícitos en todo proceso de comunicar. Es a través de ellas que las percepciones, las actitudes y las opiniones de los contestantes, que no pueden inferirse de la observación, se hacen entonces accesibles. (2) A través de las mismas se profundizaron en las características y funciones del procedimiento que se informatizó y los requerimientos del sistema. Se efectuaron cinco entrevistas al DrC. Rolando Alfredo Hernández, once a la Ing. Maylé Díaz Castro y seis al estudiante Sandy Noa Cabrera (Arquitecto del proyecto: Servicios Telemáticos).

1.3 Conceptos fundamentales

La definición de conceptos juega un papel importante en la comprensión de todo un contenido, según se conozca la etimología de cada palabra que se lea mayor será el análisis y asimilación de la información que se usará. Antes de horadar en todo el estudio del arte se hace necesario conocer algunos términos que no por mencionados se sabe plenamente su significado.

1.3.1 Proyecto

Un proyecto es la célula básica para la organización, ejecución, financiamiento y control de actividades vinculadas con la investigación científica, el desarrollo tecnológico, la innovación tecnológica, la prestación de servicios científicos y tecnológicos de alto nivel de especialización, las producciones especializadas, la formación de recursos humanos, la gerencia y otras, que materializan objetivos y resultados propios o de los programas en que están insertados y que tienen a su disposición un grupo de recursos materiales y humanos para lograr en un tiempo determinado los objetivos propuestos. (1)

1.3.2 Gestión de Proyecto

La Gestión de Proyecto es la aplicación de conocimientos, habilidades y técnicas de planificación, organización y control para optimizar las actividades de un proyecto y obtener los objetivos propuestos, cumpliendo con el tiempo establecido, el presupuesto previsto y la calidad requerida. (1)

En resumen, la gestión de proyecto suma áreas tan distintas como la incorporación del proyecto, la gestión de costes, la gestión de calidad, la gestión del tiempo, la gestión de recursos humanos o la gestión de la comunicación (entre los miembros y el exterior). Así, la gestión de proyecto forma un ciclo dinámico que transcurre del planteamiento a la ejecución y control. (4)

1.3.3 Anteproyecto

El anteproyecto es la fase del trabajo en la que se definen las características generales de la obra, en sus aspectos funcionales, formales, constructivos y económicos, proporcionando una primera imagen global del proyecto y un avance del presupuesto.

Incluye la recogida y sistematización de la información precisa, el planteamiento del programa técnico de necesidades y una estimación orientativa del coste económico, que permitan al autor del encargo adoptar una decisión inicial. (5)

1.3.4 Estudio de factibilidad

Un estudio de factibilidad abarca todos los datos e informaciones importantes para un proyecto de inversión; este material se procesa y presenta en forma sistemática, suficientemente detallada y de tal manera que facilite una decisión en cuanto a la implementación técnica y económica del proyecto. (6)

1.4 Estado del arte

En la actualidad, la eficacia y productividad con rapidez se han convertido en premisas indispensables para las empresas de todo el mundo. En la rama de la Informática y el desarrollo de software donde las empresas que con mayor rapidez y eficiencia empleen sus recursos, dígase capital humano, tecnológico y logísticos serán las que marquen la vanguardia en este sector.

El uso de las decisiones acertadas, rápidas y precisas sobre en qué proyectos se deben invertir los recursos anteriormente expuestos y en cuáles no, hacen indispensable un estudio de factibilidad para determinar la mejor inversión.

1.4.1 Evaluación de proyectos

La evaluación de proyectos surge de la necesidad de valerse de un método científico, que permita cuantificar las ventajas y desventajas que implica asignar recursos escasos, y de uso imprescindible a una determinada iniciativa, la cual necesariamente, deberá estar al servicio de la sociedad y del hombre que en ella vive.

La **Evaluación de Proyectos** es "un instrumento o herramienta que genera información, permitiendo emitir un juicio sobre la conveniencia y confiabilidad de la estimación preliminar del beneficio que genera el Proyecto en estudio. (7)

1.4.1.1 Método Multicriterio

Hernández en su folleto: Curso básico de gestión de proyectos, propone un método de evaluación de proyectos denominado: “Método Multicriterio”. Este procedimiento combina los métodos Multicriterio con los cuantitativos. A partir de criterios de expertos, utilizando procedimientos estadísticos determina de manera muy rápida un índice de aceptación para los proyectos. En el mismo se establece una serie de criterios para que sean evaluados por expertos, estas evaluaciones se procesan con el fin de obtener un resultado que será valorado por un decisor quien aprobará o no el proyecto de software. Este método se adecua con mayor facilidad al área de los proyectos informáticos, permite la formación de un criterio con elevado grado de objetividad, el consenso logrado sobre la base de los criterios es muy confiable, garantiza la libertad de opiniones y además es bastante sencillo y explícito a la hora de la evaluación de los criterios que surgen de los estudios de factibilidad (Técnica, Mercado, Económica). (1)

Estos estudios de factibilidad tienen gran importancia, sirven para realizar estimaciones o análisis previos al desarrollo de un proyecto y en base a ello tomar la mejor decisión, su objetivo principal es: auxiliar a una organización a lograr sus metas, estableciendo un instrumento que permita a un determinado personal la toma de decisiones que en este caso se refiere a proyectos de software. Cada uno de ellos tiene sus peculiaridades entre las que se destacan las siguientes:

- **Factibilidad Técnica**

Es el análisis tecnológico de todos aquellos factores que justificarán la mejor combinación de estos para determinar la viabilidad del proyecto. (9)

Algunos aspectos técnicos a tener en cuenta son:

¿Cómo se va a producir o dar el servicio, cuáles son los montos de inversión?

Definición de las características técnicas del producto, localización, selección de tecnología y equipo, maquinaria y equipo, lista de bienes y servicios necesarios para el proyecto, materias primas, mano de obra y programa de inversión. (10)

- **Factibilidad de Mercado**

En la factibilidad de mercado se trata de identificar la reacción del medio externo ante el producto que se obtendrá del proyecto y preparar un plan de comercialización. (1)

Algunos aspectos de mercado a tener en cuenta son:

¿Existe un mercado? ¿Cuáles son los ingresos que proyectarán, en qué se basan?

Definición del producto, magnitud y tendencias del mercado, penetrabilidad del mercado, estrategia comercial y determinación de ventas potenciales del proyecto. (10)

Factibilidad Económica

El estudio de Factibilidad económico-financiero es la herramienta imprescindible para conocer la totalidad de los gastos en que incurrirá la Empresa al incorporar el nuevo sistema, como así también el incremento de los costos por cargas de estructura que demandará su funcionamiento luego de la implementación. (9)

Algunos aspectos económicos a tener en cuenta son:

¿Es viable financieramente el proyecto? ¿Cómo se va a estructurar su financiamiento?
Monto de inversión, estructura de crédito y capital, proyecciones de ingresos, costos y resultados, balances proforma, flujo de efectivo. (10)

Actualmente la Factibilidad económica se estima usando los cálculos del VAN y la TIR que permiten seleccionar aquellos proyectos que tienen rendimiento con la mejor alternativa. Ambos criterios de selección se apoyan en un concepto de rango superior al beneficio, pues tienen en cuenta el flujo de caja, el cual no depende de criterios de valoración contables subjetivos. Además, ambos criterios tienen en cuenta cuando se produce la entrada o salida de fondos en la empresa, lo cual tiene una importancia fundamental en la medida en que existe un valor del dinero en tiempo. Además permite considerar el riesgo asociado al proyecto, pues se encuentra implícito en la tasa de rentabilidad que actúa como tipo de descuento en el VAN y como tasa de referencia en la TIR.

Tanto el VAN como la TIR ofrecen en condiciones normales soluciones consistentes para aceptar o rechazar un proyecto, pues si el valor actual es positivo, su tasa interna de rentabilidad será superior al costo de los flujos de caja y viceversa. Sin embargo, tiene que discrepar cuando se tiene que elegir entre varios proyectos posibles.

Actualmente el cálculo del VAN y la TIR se encuentra automatizado y conociendo el flujo de caja se hace muy fácil su determinación usando software profesionales como Storm. (1)

Por las razones antes expuestas sobre el método Multicriterio, la poca información que genera el procedimiento empleado en la UCI por el personal de ALBET para la aceptación de proyectos de software y al no hallazgo de acuerdo a las bibliografías consultadas de ningún software que estime las factibilidades técnicas y de mercado, se decide informatizar el mismo para aplicarlo en el proceso de

evaluación de las premisas de orden técnico y de mercado de proyectos de software en la Universidad de las Ciencias Informáticas.

1.4.2 Tendencias en el mundo sobre el uso de aplicaciones web para los sistemas de gestión de la información

1.4.2.1 Sistemas de gestión de la información

La gestión de información: es el proceso de analizar y utilizar la información que se ha recabado y registrado para permitir a los administradores (de todos los niveles) tomar decisiones documentadas. La gestión de información, según Ponjuán es el proceso mediante el cual se obtienen, despliegan o utilizan recursos básicos (económicos, físicos, humanos, materiales) para manejar información dentro y para la sociedad a la que sirve. Tiene como elemento básico, la gestión del ciclo de vida de estos recursos y se desarrolla en cualquier organización. En particular, también se desarrolla en unidades especializadas que manejan estos recursos en forma intensiva, llamadas unidades de información. (12)

Por lo tanto, la gestión de la información implica:

- Determinar la información que se precisa.
- Recoger y analizar la información.
- Registrarla y recuperarla cuando sea necesaria.
- Utilizarla y divulgarla.

1.4.2.2 Aplicaciones web

Una aplicación web es aquella aplicación que los usuarios pueden manejar accediendo a un servidor web a través de una red haciendo uso de un navegador. La tendencia en el mundo del uso de las aplicaciones Web para los proyectos de gestión de la información radica en las ventajas de las mismas entre las que se encuentran:

- Desarrollo barato, sencillo y rápido.
- Permite tener una alta disponibilidad, ya que pueden realizar consultas desde cualquier parte del mundo donde haya acceso a Internet y a cualquier hora.
- Permite reunir las diferentes áreas de una empresa.

- Tendrá mayor control de datos y mejor seguridad en las diferentes secciones de la aplicación.
- Otorgan la flexibilidad de determinar niveles de acceso según la confidencialidad de los datos así como la posibilidad de realizar transacciones on-line.
- No se requieren complicadas combinaciones de Hardware/Software para utilizar estas aplicaciones.
- Se facilita el trabajo a distancia. Se puede trabajar desde cualquier PC o computador portátil con conexión a Internet o a una red interna o privada.
- Facilita un mejor soporte al sistema pues una nueva versión solamente se ubica en el servidor y automáticamente todos los usuarios pueden acceder a ella. (13)

Todas estas ventajas dejan claro el potencial de las aplicaciones Web. Puesto que el mantenimiento del sistema se concentra en el servidor, el gasto se reduce. En general, es el proveedor del servicio quien se preocupa de tener la aplicación siempre disponible y actualizada.

Después de analizada la tendencia de aplicaciones web en el mundo, se propone la implementación del método Multicriterio para la evaluación de las premisas técnicas y de mercado en el proceso de contratación de proyectos de software haciendo uso de una aplicación web, la cual permitirá a un grupo de expertos emitir sus criterios, otorgándole un peso a cada una de estas premisas así como reflejando de forma cuantitativa su importancia dentro del proyecto, lo que posteriormente permitirá aceptar o no dicho proyecto de software de acuerdo a sus factibilidades: Técnica y de Mercado.

Resulta importante destacar que este estudio unido a uno de factibilidad económica facilita una mejor solidez y argumentación a la hora de tomar decisiones en cuanto a la factibilidad de un proyecto de software en general.

1.4.3 Metodologías de desarrollo de software

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos software en el que se van indicando paso a paso todas las actividades a realizar para lograr el producto informático deseado, indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben de tener. Además detallan la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla. (14)

1.4.3.1 Extreme Programing (XP)

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo.

XP se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes y donde existe un alto riesgo técnico.

El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.
5. Vuelve al paso 1.

En todas las iteraciones de este ciclo tanto el cliente como el programador aprenden. No se debe presionar al programador a realizar más trabajo que el estimado, ya que se perderá calidad en el software o no se cumplirán los plazos. De la misma forma el cliente tiene la obligación de manejar el ámbito de entrega del producto, para asegurarse que el sistema tenga el mayor valor de negocio posible con cada iteración. El ciclo de vida ideal de XP consiste de seis fases: Exploración, Planificación de la Entrega (Release), Iteraciones, Producción, Mantenimiento y Muerte del Proyecto.

Prácticas XP

La principal suposición que se realiza en XP es la posibilidad de disminuir la mítica curva exponencial del costo del cambio a lo largo del proyecto, lo suficiente para que el diseño evolutivo funcione. Esto se consigue gracias a las tecnologías disponibles para ayudar en el desarrollo de software y a la aplicación disciplinada de las siguientes prácticas: el juego de la planificación, entregas pequeñas, metáfora, diseño

simple, pruebas, refactorización (Refactoring), programación en parejas, propiedad colectiva del código, integración continua, 40 horas por semana, cliente in-situ, estándares de programación. (15)

XP es una metodología menos orientada al documento, exigiendo una cantidad más pequeña de documentación para una tarea dada. Su principal orientación es hacia el código: siguiendo un camino que dice que la parte importante de la documentación es el código fuente. (15)

1.4.3.2 Rational Unified Process (RUP)

El Proceso Unificado de Rational o RUP es un proceso para la Ingeniería de Software. Proporciona un acercamiento disciplinado a asignar tareas y responsabilidades dentro de una organización de desarrollo. Su meta es asegurar la producción de software de calidad superior que satisface las necesidades de sus usuarios finales dentro de un tiempo y presupuesto predecibles. (16)

Características de RUP:

- Creado por Jacobson, Rumbaugh y Booch.
- Unifica los mejores elementos de metodologías anteriores.
- Preparado para desarrollar grandes y complejos proyectos.
- Orientado a Objetos.
- Utiliza el UML como lenguaje de representación visual.
- Guiado por Casos de Uso.
- Iterativo e Incremental.
- Centrado en la Arquitectura.

RUP divide el proceso de desarrollo en ciclos, teniendo un producto al final de cada ciclo, cada ciclo se divide en fases que finalizan con un hito donde se debe tomar una decisión importante:

- **Inicio**: Se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema.

- **Elaboración**: Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso identificados. A pesar de que se desarrolla a profundidad una parte del sistema, las decisiones sobre la arquitectura se hacen sobre la base de la

comprensión del sistema completo y los requerimientos (funcionales y no funcionales) identificados de acuerdo al alcance definido.

- **Construcción**: Se obtiene un producto listo para su utilización que está bien documentado. Se obtiene uno o varios entregables del producto que han pasado las pruebas. Se ponen estos entregables a consideración de un conjunto de usuarios.

- **Transición**: El release ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores.

Los elementos del RUP son:

Actividades: Son los procesos que se llegan a determinar en cada iteración.

Trabajadores: Son las personas involucradas en cada proceso.

Artefactos: Son los elementos de información producidos, modificados o usados por los trabajadores para realizar nuevas actividades y son el resultado de estas. Una particularidad de esta metodología es que, en cada ciclo de iteración, se hace exigente el uso de artefactos, siendo por este motivo, una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo del software y es la más adaptable para proyectos de largo plazo. (16)

Debido a las grandes potencialidades que presenta RUP, a su unificación de los mejores elementos de otras metodologías, la guía que brinda para encontrar, organizar, documentar, y seguir los cambios de los requisitos funcionales y restricciones, así como su gran estructuración que permite poseer un universo de documentación a cada elemento significativo del proyecto (documentos de análisis, de diseño, de implementación) y al conocimiento básico de la misma alcanzado a lo largo del estudio de asignaturas como Ingeniería de Software I y II, se decide el uso de la misma como la más conveniente para guiar el proceso de desarrollo del producto de software a informatizar.

1.4.4 Lenguaje de Modelado UML

Hoy día UML (Unified Modeling Languaje) está consolidado como el lenguaje estándar en el análisis y diseño de sistemas informáticos. Mediante el mismo es posible establecer una serie de requerimientos y estructuras necesarias para plasmar un sistema de software previo al proceso intensivo de escribir código. UML es una especificación de notación orientada a objetos. Divide cada proyecto en un número de diagramas que representan las diferentes vistas del proyecto. (17)

UML es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software. UML entrega una forma de modelar cosas conceptuales como lo son procesos de negocio y funciones de sistema, además de cosas concretas como lo son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables. (17)

Los principales beneficios de UML son:

- Mejores tiempos totales de desarrollo (de 50 % o más).
- Modelar sistemas utilizando conceptos orientados a objetos.
- Establecer conceptos y artefactos ejecutables.
- Encaminar el desarrollo del escalamiento en sistemas complejos de misión crítica.
- Crear un lenguaje de modelado utilizado tanto por humanos como por máquinas.
- Mejor soporte a la planeación y al control de proyectos.
- Alta reutilización y minimización de costos. (17)

1.4.5 Herramientas CASE

Las Herramientas CASE (Computer Aided Software Engineering, “siglas del inglés” Ingeniería de Software Asistida por Ordenador) pueden definirse como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un Software. (18)

La mejor razón para la creación de estas herramientas fue el incremento en la velocidad de desarrollo de los sistemas. Por esto, las compañías pudieron desarrollar sistemas sin encarar el problema de tener cambios en las necesidades del negocio, antes de finalizar el proceso de desarrollo. También permite a

las compañías competir más efectivamente usando estos sistemas desarrollados nuevamente para compararlos con sus necesidades de negocio actuales. En un mercado altamente competitivo, esto puede hacer la diferencia entre el éxito y el fracaso.

Las herramientas CASE también permiten a los analistas tener más tiempo para el análisis y diseño y minimizar el tiempo para codificar y probar. Estas herramientas pueden proveer muchos beneficios en todas las etapas del proceso de desarrollo de software, algunas de ellas son:

- Verificar el uso de todos los elementos en el sistema diseñado.
- Automatizar el dibujo de diagramas.
- Ayudar en la documentación del sistema.
- Ayudar en la creación de relaciones en la Base de Datos.
- Generar estructuras de código.

La principal ventaja de la utilización de una herramienta CASE, es la mejora de la calidad de los desarrollos realizados y, en segundo término, el aumento de la productividad. Para conseguir estos dos objetivos es conveniente contar con una organización y una metodología de trabajo, además de la propia herramienta. (18)

1.4.5.1 Rational Rose Enterprise

Rational Rose es la herramienta CASE que soporta de forma completa la especificación del UML 1.1. La misma propone la utilización de cuatro tipos de modelo para realizar un diseño del sistema, utilizando una vista estática y otra dinámica de los modelos del sistema, uno lógico y otro físico. Permite crear y refinar estas vistas creando de esta forma un modelo completo que representa el dominio del problema y el sistema de software. (19)

Rational Rose proporciona un lenguaje común de modelado para el equipo que facilita la creación de software de calidad más rápidamente. Algunas de sus características son las siguientes:

- Soporte para análisis de patrones ANSI C++, Rose J y Visual C++ basado en "Patrones de diseño: elementos reutilizables de software orientados a objetos".

- Característica de control por separado de componentes, modelo que permite una administración más granular y el uso de modelos.
- Capacidad de análisis de calidad de código.
- Modelado UML para trabajar en diseños de base de datos, con capacidad de representar la integración de los datos y los requerimientos de aplicación a través de diseños lógicos y físicos.
- Capacidad de crear definiciones de tipo de documento XML (DTD) para su uso en la aplicación.
- Integración con otras herramientas de desarrollo de Rational.
- Capacidad para integrarse con cualquier sistema de control de versiones SCC-compliant, incluyendo a Rational ClearCase.
- Publicación web y generación de informes para optimizar la comunicación dentro del equipo. (20)

1.4.5.2 Visual Paradigm

Es una herramienta CASE que utiliza UML como lenguaje de modelado. Esta herramienta está desarrollada por Visual Paradigm Internacional una de las principales compañías de herramientas CASE donde su mayor éxito consiste en el uso libre del producto mencionado.

Características

Visual Paradigm utiliza UML como lenguaje de modelado para la construcción de los sistemas software ofreciendo soluciones que permiten a las organizaciones desarrollar las aplicaciones con calidad más rápido, y más barato.

- Tiene la capacidad de ejecutarse sobre diferentes sistemas operativos lo que le confiere la característica de ser multiplataforma.

- Integra diferentes funcionalidades para el desarrollo de aplicaciones como el modelado de UML, el modelado de base de batos, el modelado de requerimientos, el modelado del proceso de negocio, la interoperabilidad, la generación de documentación, entre otros.

- Permite realizar reingeniería inversa de los datos. (21)

La herramienta Visual Paradigm se convierte en la más conveniente para utilizarla en el desarrollo del sistema por ser multiplataforma, tener superior entorno de modelado visual, permitir exportar código de

alrededor de 10 lenguajes incluyendo el PHP y porque una de sus versiones, la *Community Edition* es gratis.

1.4.6 Gestor de Base de Datos

Un Sistema Gestor de Base de Datos (SGBD) es un conjunto de programas que permiten crear y mantener una Base de datos, asegurando su integridad, confidencialidad y seguridad. Por tanto debe permitir:

- Definir una base de datos: especificar tipos, estructuras y restricciones de datos.
- Construir la base de datos: guardar los datos en algún medio controlado por el mismo SGBD.
- Manipular la base de datos: realizar consultas, actualizarla, generar informes. (22)

A continuación se realiza un estudio comparativo entre dos potenciales Gestores de Base de Datos, donde se exponen sus principales características.

1.4.6.1 MySQL

MySQL es un sistema de gestión de bases de datos relacional, licenciado bajo la GPL de la GNU. Su diseño multihilo le permite soportar una gran carga de forma muy eficiente. MySQL fue creada por la empresa sueca MySQL AB, que mantiene el copyright del código fuente del servidor SQL, así como también de la marca. (23) Aunque MySQL es software libre, MySQL AB distribuye una versión comercial de MySQL, que no se diferencia de la versión libre más que en el soporte técnico que se ofrece, y la posibilidad de integrar este gestor en un software propietario, ya que de no ser así, se vulneraría la licencia GPL. (23) Este gestor de bases de datos es, probablemente, el gestor más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración.

Características de MySQL

Las principales características de este gestor de bases de datos son las siguientes:

1. Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multihilo.
2. Soporta gran cantidad de tipos de datos para las columnas.
3. Dispone de API's en gran cantidad de lenguajes (C, C++, Java, PHP, etc.).
4. Gran portabilidad entre sistemas.
5. Soporta hasta 32 índices por tabla.
6. Gestión de usuarios y passwords, manteniendo un muy buen nivel de seguridad en los datos. (23)

1.4.6.2 PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional (ORDBMS) basado en el proyecto POSTGRES, de la universidad de Berkeley. El director de este proyecto es el profesor Michael Stonebraker, y fue patrocinado por Defense Advanced Research Projects Agency (DARPA), el Army Research Office (ARO), el National Science Foundation (NSF), y ESL, Inc. (Pecos)

PostgreSQL es una derivación libre (OpenSource) de este proyecto, y utiliza el lenguaje SQL92/SQL99. Fue el pionero en muchos de los conceptos existentes en el sistema objeto-relacional actual, incluido, más tarde en otros sistemas de gestión comerciales. PostgreSQL es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. (23)

Características de PostgreSQL

A continuación se enumeran las principales características de este gestor de bases de datos:

1. Implementación del estándar SQL92/SQL99.
2. Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP...), cadenas de bits, etc.
3. Incorpora una estructura de datos array.
4. Incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes, etc.
5. Permite la declaración de funciones propias, así como la definición de disparadores.
6. Soporta el uso de índices, reglas y vistas.

7. Incluye herencia entre tablas (aunque no entre objetos, ya que no existen), por lo que a este gestor de bases de datos se le incluye entre los gestores objeto-relacionales.
8. Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos. (23)

Luego del estudio de estos dos sistemas gestores de bases de datos se ha seleccionado para la realización del sistema PostgreSQL debido a las características anteriormente expuestas y además que:

1. Posee una gran escalabilidad. Es capaz de ajustarse al número de CPUs y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta (en algunos benchmarks se dice que ha llegado a soportar el triple de carga de lo que soporta MySQL).

2. Implementa el uso de subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz, y ofreciendo soluciones en campos en las que MySQL no podría.

3. Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos, equiparándolo con los gestores de bases de datos de alto nivel, como puede ser Oracle. (23)

1.4.7 Lenguaje de Programación seleccionado (PHP)

Un lenguaje de programación es aquel que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Consiste en un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Un lenguaje de programación permite a uno o más programadores especificar de manera precisa: sobre qué datos una computadora debe operar, cómo deben ser estos almacenados y transmitidos y qué acciones debe tomar bajo una variada gama de circunstancias. (24)

1.4.7.1 PHP

PHP es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de páginas Web dinámicas, similar al ASP de Microsoft o el JSP de Sun, embebido en páginas HTML y ejecutado en el servidor.

La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas de sí mismo. La meta del lenguaje es permitir rápidamente a los desarrolladores la generación dinámica de páginas. No es un lenguaje de marcas como podría ser HTML, XML o WML. Está más cercano a JavaScript o a C. (25)

¿Cómo Funciona PHP?

A diferencia de Java o JavaScript que se ejecutan en el navegador PHP se ejecuta en el servidor por eso nos permite acceder a los recursos que tenga el servidor como por ejemplo podría ser una base de datos. El programa PHP es ejecutado en el servidor y el resultado es enviado al navegador. El resultado es normalmente una página HTML pero también podría ser una página WML.

Al ser PHP un lenguaje que se ejecuta en el servidor no es necesario que su navegador lo soporte, es independiente del navegador, sin embargo, para que sus páginas PHP funcionen el servidor donde están alojadas debe soportar PHP.

Características

Al ser un lenguaje libre dispone de una gran cantidad de características que lo convierten en la herramienta ideal para la creación de páginas web dinámicas:

- Soporte para una gran cantidad de bases de datos: MySQL, PostgreSQL, Oracle, MS SQL Server, Sybase mSQL, Informix, entre otras.
- Integración con varias bibliotecas externas, permite generar documentos en PDF (documentos de Acrobat Reader) hasta analizar código XML.

- Ofrece una solución simple y universal para las paginaciones dinámicas de la Web de fácil programación.
- Perceptiblemente más fácil de mantener y poner al día que el código desarrollado en otros lenguajes.
- Soportado por una gran comunidad de desarrolladores, como producto de código abierto, PHP goza de la ayuda de un gran grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y reparen rápidamente.
- El código se pone al día continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP.
- Con PHP se puede hacer cualquier cosa que podemos realizar con un script CGI, como el procesamiento de información en formularios, foros de discusión, manipulación de cookies y páginas dinámicas. (25)

Seguridad

PHP es un potente lenguaje y el intérprete, tanto incluido en el servidor Web como módulo o ejecutado como un binario CGI, puede acceder a ficheros, ejecutar comandos y abrir comunicaciones de red en el servidor. Todas estas características hacen que lo que se ejecute en el servidor Web sea seguro por defecto.

PHP ha sido diseñado específicamente para ser un lenguaje más seguro para escribir programas CGI, Perl o C y con la correcta selección de las opciones de configuración de tiempo de compilación y ejecución se consigue la exacta combinación de libertad y seguridad que se necesita. Ya que existen diferentes modos de utilizar PHP, existe también una multitud de opciones de configuración que permiten controlar su funcionamiento. Una gran selección de opciones garantiza que se pueda usar PHP para diferentes aplicaciones, pero también significa que existen combinaciones de estas opciones y configuraciones del servidor que producen instalaciones inseguras. (25)

Debido a las características de este lenguaje se decide su selección pues entre otras cosas analizadas nos ofrece una gran diversidad de funciones que nos permiten desarrollar un conjunto de funcionalidades que van desde enviar un correo electrónico subir un archivo (upload), crear una imagen en tiempo de

ejecución, interactuar con diversos protocolos de comunicación, interactuar con documentos XML, autenticación, creación dinámica de documentos PDF.

De forma general sus principales características son: su rapidez; su facilidad de aprendizaje; su soporte multiplataforma tanto de diversos Sistemas Operativos, como servidores HTTP y de bases de datos; y el hecho de que se distribuye de forma gratuita bajo una licencia abierta.

1.4.8 Servidor Web seleccionado (Apache Web Server)

Un servidor web es un programa que sirve para atender y responder a las diferentes peticiones de los navegadores, proporcionando los recursos que soliciten usando el protocolo HTTP o el protocolo HTTPS (la versión cifrada y autenticada). Un servidor web básico cuenta con un esquema de funcionamiento muy simple, basado en ejecutar infinitamente el siguiente bucle:

1. Espera peticiones en el puerto TCP indicado (el estándar por defecto para HTTP es el 80).
2. Recibe una petición.
3. Busca el recurso.
4. Envía el recurso utilizando la misma conexión por la que recibió petición.
5. Vuelve al segundo punto. (26)

De los servidores web existentes se decide usar el Apache Web Server por las siguientes razones:

- Presenta entre otras características, mensajes de error altamente configurables, bases de datos de autenticación y negociado de contenido.
- Corre en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal.
- Apache 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 qué es lo que se está 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 las capacidades del servidor Web Apache. Actualmente existen muchos módulos para Apache que son adaptables a este, y están ahí para que se instalen cuando hagan falta. Otra cosa importante es que cualquiera que posea una experiencia adecuada en la programación de C o Perl puede escribir un módulo para realizar una función determinada.

- Apache te 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 de log a medida del administrador, de este modo se puede tener un mayor control sobre lo que sucede en el servidor. (27)

Además de todas estas características Apache es muy usado en toda la Universidad amén de existir una amplia disposición de documentos e información para el trabajo con él.

1.4.9 Herramienta de Desarrollo seleccionada (IDE): Zend Studio for Eclipse

Zend Studio for Eclipse se ha convertido en el IDE para PHP más potente del mercado, ofreciendo así al desarrollador profesional de PHP la potencia de Zend Studio y el soporte multilenguaje de Eclipse y su enorme conjunto de extensiones (plugins).

En concreto, Zend Studio para Eclipse aporta:

Las ventajas de Eclipse/PDT:

- Una comunidad de millones de usuarios y miles de desarrolladores.
- Cientos de plugins.
- Soporte multi-lenguaje en una única herramienta (Eclipse).
- Coloreado de sintaxis PHP, autocompletado de código e inspección de métodos y atributos.
- Soporte básico de depuración de scripts PHP.

Valor añadido de Zend Studio:

- Soporte de depuración más avanzada (permitiendo debugging local como remoto).
- Soporte a la refactorización de código.
- Análisis de rendimiento.
- Soporte de baterías de test PHPUnit.
- Integración mejorada con Zend Framework.
- Posibilidad de edición HTML WYSIWYG.
- Integración con Zend Platform.
- Herramientas de migración para los usuarios de Zend Studio. (28)

Por todas las características antes expuestas, entre las que se destacan: el excelente completamiento de código, administración avanzada de proyectos, la agilización del trabajo, así como las infinitas opciones que permiten un desarrollo profesional de las aplicaciones se decide utilizar este IDE para el desarrollo del sistema.

1.4.10 Framework seleccionado (Symfony)

Un Framework es una estructura de software compuesta de componentes personalizables e intercambiables que componen un diseño reutilizable que facilita y agiliza el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas más comunes, estos aceleran el proceso de desarrollo y facilitan la programación, ya que contienen operaciones complejas encapsuladas en instrucciones sencillas, además facilita que el código sea más legible y fácil de mantener, y promueve buenas prácticas de desarrollo como el uso de patrones. (29)

Se usará el framework Symfony como librería base para la programación del proyecto debido a que: es uno de los frameworks PHP más usados 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.

Sigue la estela de Rails para simplificar al máximo el desarrollo de aplicaciones web profesionales con PHP, utilizando las mejores prácticas y los patrones de diseño más importantes. Symfony incorpora muchas de las ideas del RAD (”desarrollo rápido de aplicaciones”) para conseguir que la programación de

las aplicaciones sea lo más productiva, correcta y divertida posible. 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.

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. (29)

1.4.11 Arquitectura

1.4.11.1 Patrón Modelo-Vista-Controlador (MVC)

El Modelo-Vista-Controlador (MVC) se creó para Smalltalk a finales de los setenta; desde entonces su uso se ha ido extendiendo cada día más para la construcción de sistemas con interfaz gráfica. MVC es un patrón de arquitectura utilizado para separar los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos, permitiendo flexibilidad y facilidad a la hora de hacer futuros cambios.

Elementos del patrón:

- Modelo: datos y reglas de negocio
- Vista: muestra la información del modelo al usuario
- Controlador: gestiona las entradas del usuario

Patrón Arquitectónico: Modelo-Vista-Controlador

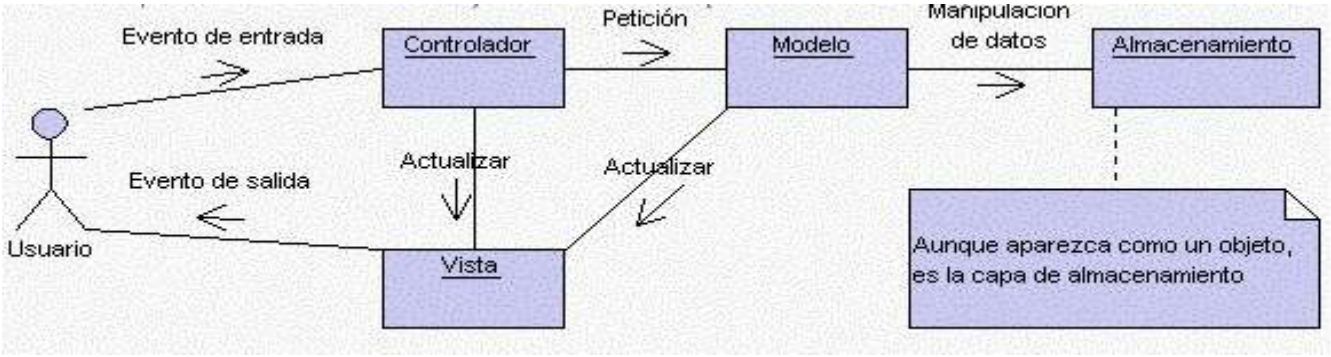


Fig.1 1 Modelo-Vista-Controlador

Un modelo puede tener diversas vistas, cada una con su correspondiente controlador. Un ejemplo clásico es el de la información de una base de datos, que se puede presentar de diversas formas: diagrama de tarta, de barras, tabular, etc. Veamos cada componente:

1. El **modelo** es el responsable de:

   - Acceder a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento.
   - Definir las reglas de negocio (la funcionalidad del sistema). Un ejemplo de regla puede ser: "Si la mercancía pedida no está en el almacén, consultar el tiempo de entrega estándar del proveedor".
   - Llevar un registro de las vistas y controladores del sistema.
   - Notificar a las vistas si estamos ante un modelo activo los cambios que en los datos pueda producir un agente externo (por ejemplo, un fichero bath que actualiza los datos, un temporizador que desencadena una inserción, etc.).

2. El **controlador** es responsable de:

   - Recibir los eventos de entrada (un clic, un cambio en un campo de texto, etc.).

- Contener reglas de gestión de eventos, del tipo "Si Evento Z, entonces Acción W". Estas acciones pueden suponer peticiones al modelo o a las vistas. Una de estas peticiones a las vistas puede ser una llamada al método "Actualizar ()". Una petición al modelo puede ser "Obtener_tiempo_de_entrega (nueva_orden_de_venta)".

3. Las **vistas** son responsables de:

- Recibir datos del modelo y los muestra al usuario.
- Tener un registro de su controlador asociado (normalmente porque además lo instancia).
- Dar el servicio de "Actualización ()", para que sea invocado por el controlador o por el modelo (cuando es un modelo activo que informa de los cambios en los datos producidos por otros agentes). (30)

1.4.11.2 Arquitectura Cliente/Servidor

La arquitectura cliente-servidor se divide en dos partes claramente diferenciadas, la primera es la parte del servidor y la segunda la de un conjunto de clientes. Normalmente el servidor es una máquina potente que actúa de depósito de datos y funciona como un sistema gestor de base de datos (SGBD).

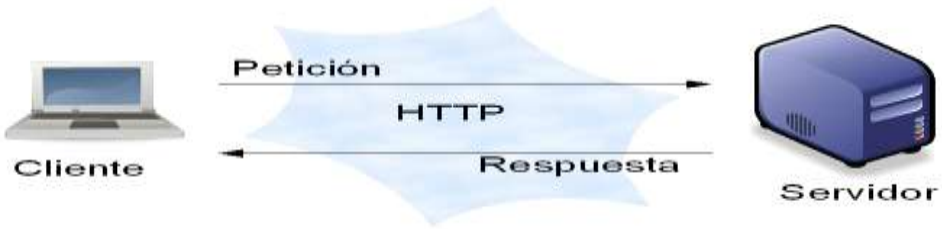


Fig.1 2 Arquitectura Cliente-Servidor

Entre sus principales características se encuentran:

- El servidor presenta una interfaz única y bien definida a todos sus clientes.
- El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa.

- El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.
- Los cambios en el servidor no afectan al cliente. (31)

1.5 Conclusiones

El estado del arte expuesto en este capítulo sobre las tecnologías, así como la tendencia al uso de aplicaciones Web en la realización de software de gestión permitirá una mayor comprensión de lo que existe y definirá hacia donde se deben encaminar los esfuerzos con el propósito de lograr un software a la altura de los objetivos trazados. Para la realización del mismo se establece informatizar el Método Multicriterio, haciendo uso de una aplicación web. Se utilizará RUP como metodología de desarrollo de software, UML como lenguaje de modelado, la herramienta case será Visual Paradigm, se usará como lenguaje de programación PHP, como entorno de desarrollo integrado Zend Studio for Eclipse; el framework Symfony será el que se empleará y como Gestor de base de datos se utilizará PostgreSQL, además de emplearse el Apache como servidor web.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

2.1 Introducción

En este capítulo se realiza una descripción del negocio, se identifican los actores y trabajadores del mismo, así como el diagrama de casos de uso y su formato expandido. Se esbozan los requerimientos funcionales y no funcionales con que debe contar el sistema. Además se identifican los actores y casos de uso del mismo empleando para su modelación el lenguaje UML, que posibilita representar el diagrama de casos de uso del sistema y la especificación de los mismos.

2.2 Descripción del negocio

“Para contratar la ejecución de un proyecto de cualquier tipo y obtener el financiamiento necesario para su ejecución se debe estar seguro de tener altas probabilidades de éxito, por lo que se hace imprescindible realizar estudios de viabilidad técnica, económica y de mercado.” (1)

En el procedimiento **Multicriterio,** que posibilita la evaluación de premisas de orden técnico y de mercado para la contratación de proyectos de software, según Hernández, el analista dirige todo el proceso de evaluación encargándose de seleccionar a los evaluadores y a la vez define los criterios a evaluar, creando así la Plantilla de Criterios Técnicos (ver anexo 2) y de Mercado (ver anexo 3). Todo esto ocurre después de que un cliente ya sea una persona o entidad se presente ante él con el objetivo de verificar cuan factible puede ser la contratación de un proyecto de software determinado. El analista tendrá que realizar un registro de los principales datos del proyecto en la Plantilla de Proyectos (ver anexo 4) antes de gestionar la factibilidad del mismo. Del estudio bibliográfico realizado se proponen un grupo de criterios tanto del orden técnico como de mercado.

En cuanto al estudio técnico los criterios propuestos a evaluar por un grupo de expertos que permita tomar decisiones para aceptar o no el proyecto son los siguientes:

1. Calidad del personal directivo
2. Calidad del personal directo
3. Posibilidades de la tecnología disponible

4. Disponibilidad de las herramientas de software necesarias
5. Acceso al mercado de tecnologías y licencias necesarias
6. Organización del proceso de producción
7. Calidad del mantenimiento de la tecnología.
8. Localización el proyecto
9. Gestión de tiempo
10. Disponibilidad de insumos
11. Nivel de información del cliente
12. Garantía de servicios necesarios
13. Dependencia tecnológica
14. Cumplimientos de la calidad del producto final
15. Relación demanda/capacidad de producción
16. Necesidad de nuevas inversiones
17. Despliegue del producto
18. Costo del proyecto
19. Éxitos pasados
20. Conocimiento de las legislaciones vigentes.

En cuanto al orden de mercado propone los siguientes criterios:

1. Necesidad del producto
2. Productos sustitutivos
3. Productos similares
4. Productos complementarios
5. Universo de probables consumidores
6. Capacidad de pago potencial de los clientes
7. Restricciones legales de investigaciones
8. Dificultades de acceso al mercado
9. Demanda actual
10. Demanda futura
11. Cantidad de competidores

12. Oferta actual
13. Oferta futura
14. Calidad del producto
15. Servicio al cliente
16. Protección al mercado
17. Régimen de mercado
18. Grado de receptibilidad del producto en el mercado
19. Éxitos anteriores
20. Novedad del producto

Cada uno de estos grupos de criterios puede variar de acuerdo a los intereses y necesidades que surjan a la hora de la evaluación, eliminando algunos o añadiendo otros que el analista estime conveniente. Después que se han definido los dos grupos de criterios tanto del orden técnico como de mercado, se selecciona un grupo de expertos. Cada experto determina el peso que tiene cada criterio seleccionado, para lo que se le solicita que valoren cada criterio de acuerdo con su importancia en relación a los demás para el proyecto que se va a evaluar y le asigne un por ciento del total según su opinión. Después que se le ha asignado un peso a cada uno de los criterios, el analista tendrá que verificar la concordancia en el trabajo de expertos, para lo que se utiliza el coeficiente de concordancia de Kendall y el estadígrafo Chi cuadrado (χ^2). (32)

Si no existe concordancia en el trabajo de expertos se rechazan los resultados y se hace necesario repetirlo, pero si existe concordancia el peso de cada criterio es el encontrado anteriormente y se solicita a cada experto que califique cada criterio en una escala de uno a cinco de acuerdo con su comportamiento dentro del estudio de viabilidad técnica o de mercado según lo que se esté analizando en ese momento.

Conociendo el peso promedio de cada criterio dado por los expertos y la calificación promedio dada por los expertos a cada criterio se procede al cálculo de probabilidad de éxito técnico o de mercado. Este resultado se ubica dentro de una matriz de decisión (ver anexo 5) que le permitirá al decisor de acuerdo al cuadrante donde caiga aceptar o no el proyecto de software al cual se le ha gestionado la factibilidad.

2.3 Modelo del negocio

El Modelo del Negocio es la primera disciplina ofrecida por RUP en el ciclo de vida de un software, se desarrolla colosalmente en la fase de inicio, donde identifica las principales actividades de cualquier institución. RUP propone que se realice el modelado del negocio siempre y cuando los procesos sean fáciles de delimitar, si los procesos no son fáciles de identificarlos entonces se efectúa modelo del dominio. En el modelado del negocio desarrollado en el procedimiento **Multicriterio**, se trata de describir los procesos principales, especificando los actores vinculados con los casos de uso, los trabajadores del negocio y las entidades principales a utilizar en el mismo. (33)

2.3.1 Actores del negocio

Los actores del negocio son aquellas personas o sistemas que reciben un beneficio de los procesos asociados al software. Es cualquier individuo, grupo, entidad, organización, máquina o sistema de información externos; con los que el negocio interactúa. (34)

Actores del negocio	Descripción
Cliente	Es la persona, grupo, entidad, empresa u organización que recibe todos los beneficios de los procesos del negocio.

Tabla. 2. 1 Actores del Negocio

2.3.2 Trabajadores del negocio

Un Trabajador del Negocio representa a un ser humano, software o hardware que desempeña un rol dentro de las Realizaciones del Caso de Uso del Negocio. Este trabajador interactúa con entidades y otros trabajadores para que el negocio funcione. Los trabajadores de negocio son roles y no posiciones

organizacionales, ya que una persona puede desempeñar varios roles pero sólo tiene una posición en la organización. (35)

Trabajadores del negocio	Descripción
Analista	Es el encargado de registrar el proyecto, definir los criterios a evaluar, realizar la selección de los expertos, verificar la consistencia de los expertos y calcular el índice de aceptación.
Experto	Expresa la calidad del proyecto de acuerdo con sus conocimientos sobre la temática que trata y califica los criterios empleados según la escala pre-establecida.
Decisor	Aprueba el proyecto de acuerdo con la importancia de los criterios utilizados

Tabla. 2. 2 Trabajadores del Negocio

2.3.3 Diagrama de Casos de Uso del Negocio

Diagrama de casos de uso del negocio

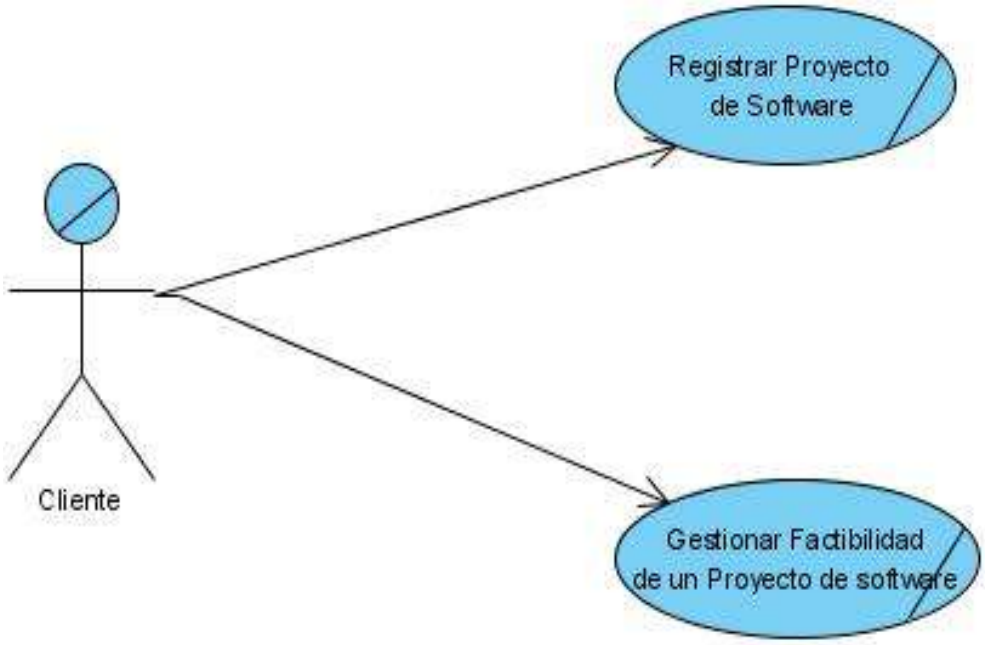


Fig. 1.1 Diagrama de casos de uso del negocio

2.3.4 Descripción textual de los Casos de Uso del Negocio en formato expandido

Nombre del Caso de Uso	Registrar Proyecto de Software
Actores	Cliente (Inicia)
Propósito	Tener control de todos los datos relacionados con un determinado proyecto de software.
Resumen	El cliente se presenta ante un analista con el objetivo de registrar un proyecto de software para así tener de forma más organizada y controlada los datos de proyectos que se le han gestionado la factibilidad y a los que están por gestionárselas.
Flujo Normal de los eventos	
Acciones del Actor	**Respuesta del proceso de negocio**
1- El caso de uso se inicia cuando el cliente	1.1 – El Analista le pide los datos del proyecto.

solicita registrar los datos de un proyecto de software.	
2 – Informa los datos del proyecto	2.1 – El Analista recibe los datos del proyecto. 2.2 – El Analista registra los datos en la Plantilla de Proyectos.
Flujo Alternativo de los eventos	
Acción del actor	**Respuestas del negocio**
Mejoras	

Nombre del Caso de Uso	Gestionar Factibilidad de un Proyecto de Software	
Actores	Cliente (Inicia)	
Propósito	Determinar de acuerdo al orden técnico y de mercado si un proyecto de software determinado es factible o no.	
Resumen	El cliente se presenta ante un analista con el objetivo de saber si un determinado proyecto de software es factible, éste se hará responsable de todas las tareas para gestionar dicha factibilidad como la selección de los criterios y expertos que posteriormente harán las evaluaciones de estos criterios, las cuales van a ser asesoradas por el propio analista y luego un revisor dará las conclusiones en correspondencia con los resultados.	
Flujo Normal de los eventos		
Acciones del Actor		**Respuesta del proceso de negocio**
1- El caso de uso se inicia cuando el cliente solicita gestionar la factibilidad de un proyecto de software.		1 – El Analista define los criterios a evaluar por los expertos creando así la Plantilla de Criterios. 1.2–Selecciona a los expertos que se encargarán de evaluar cada criterio seleccionado. 1.3- Entrega los criterios a los expertos.

	2 – Los Expertos evalúan cada criterio asignándole un peso.
	3 – El analista verifica la consistencia de los expertos y si existe, entrega nuevamente los criterios a los expertos para su calificación.
	4 – Los expertos califican cada criterio asignándole un valor entre 1-5.
	5 – El analista calcula el índice de aceptación del proyecto de software, aplicando el procedimiento correspondiente.
	6 – El decisor analiza los resultados emitidos y decide si el proyecto de software es factible o no.
Flujo Alternativo de los eventos	
Acción del actor	**Respuestas del negocio**
	Actividad 3: El analista verifica la consistencia de los expertos y si no existe **(Ir a 2 Flujo normal de los eventos)**
Mejoras	

2.3.5 Diagrama de Actividades

Diagrama de Actividades: CU <Registrar Proyecto de Software>

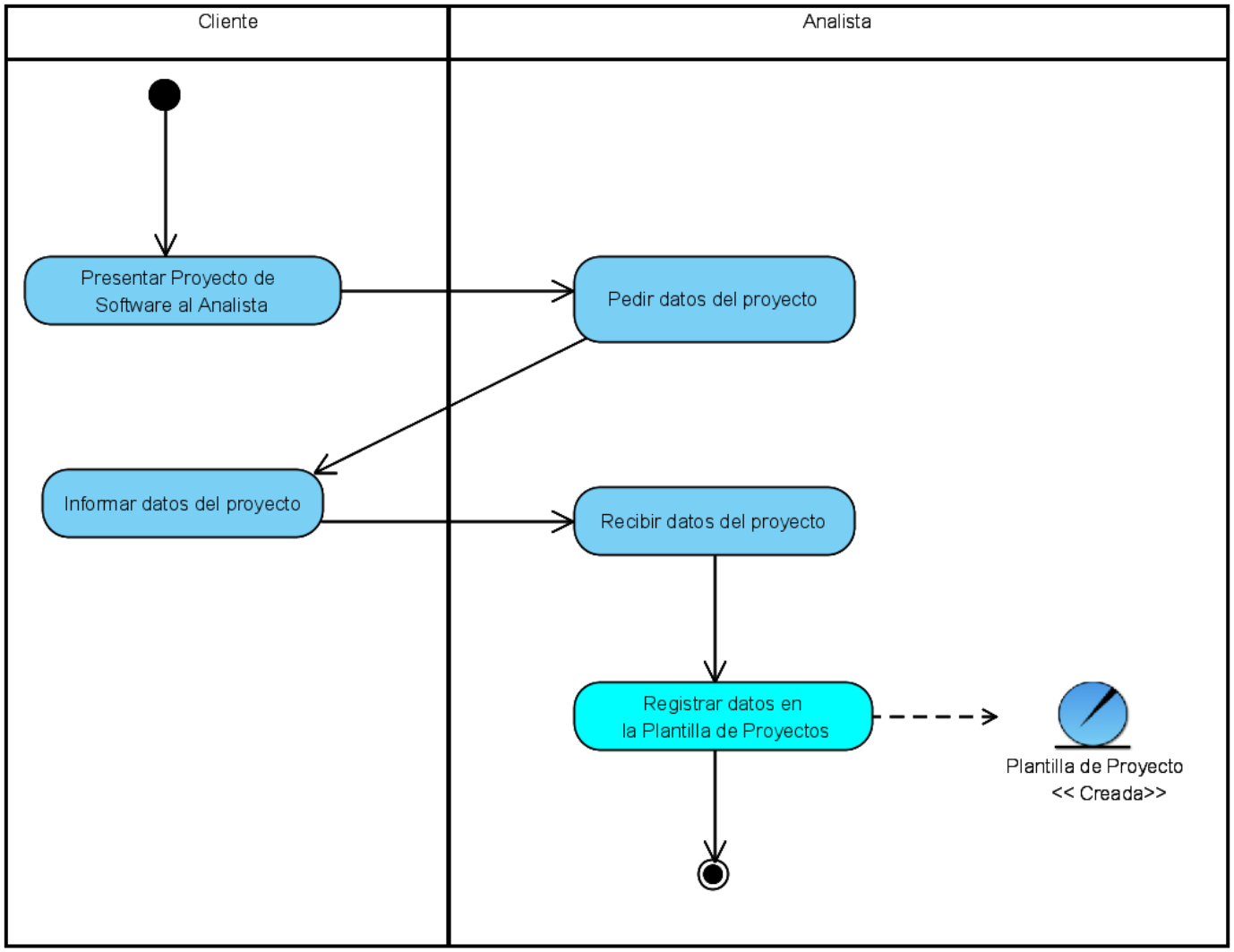


Fig. 2.2 Diagrama de actividades <Registrar Proyecto de Software>

Diagrama de Actividades: CU <Gestionar Factibilidad de un Proyecto de Software>

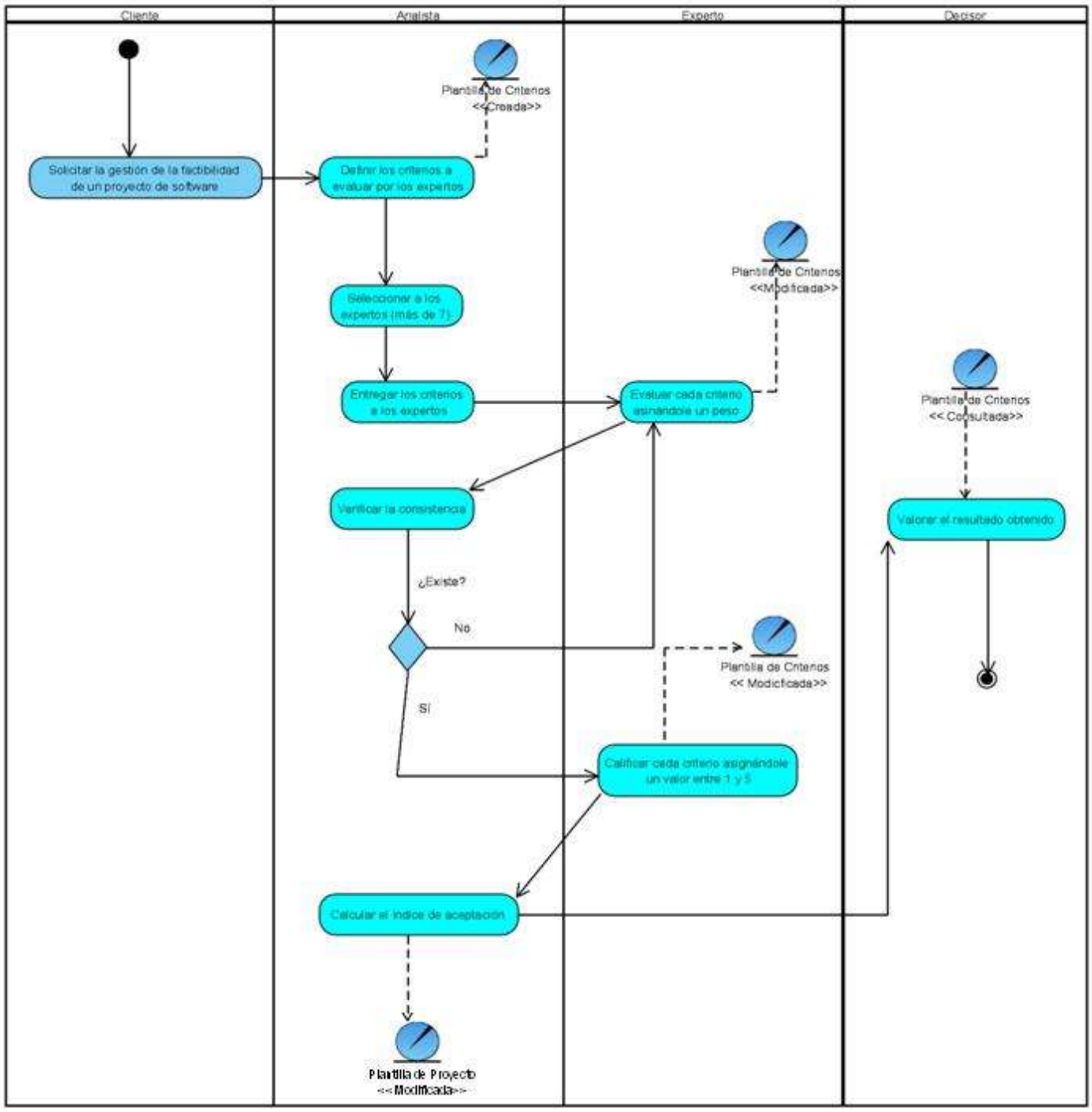


Fig. 3.3 Diagrama de actividades <Gestionar Factibilidad de un Proyecto de Software>

2.3.6 Modelo de Objetos

Modelo de Objetos

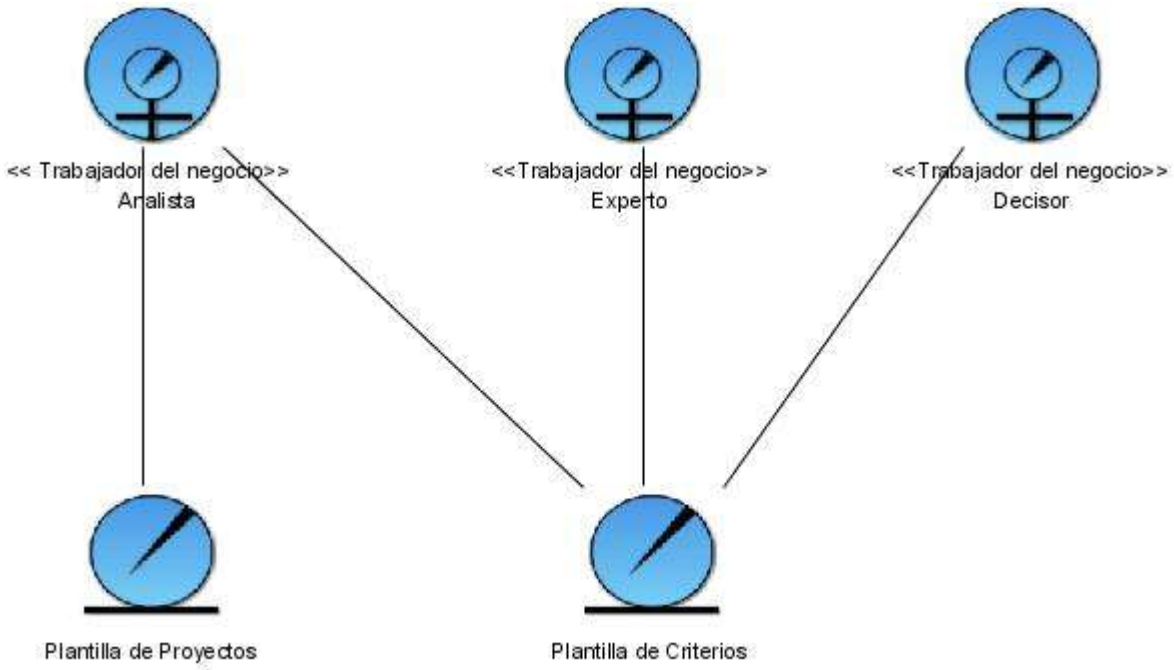


Fig. 4.4 Modelo de Objetos

2.4 Levantamiento de requisitos

A continuación se identifican los principales requisitos funcionales que son las prestaciones que debe cumplir un sistema y los requisitos no funcionales que son las características o cualidades que debe tener el producto.

2.4.1 Requisitos funcionales

R1- Gestionar Usuario

1.1 Adicionar usuario.

1.2 Modificar usuario.

1.3 Eliminar usuario.

R2- Autenticar Usuario.

2.1 Comparar Usuario y contraseña con los usuarios del servidor de aplicaciones LDAP.

2.2 Comparar Usuario con los usuarios del Sistema.

2.3 Asignar privilegios.

R3 Gestionar Proyecto

3.1 Adicionar Proyecto

3.2 Modificar Proyecto

3.3 Eliminar Proyecto

R4 Gestionar Evaluación

4.1 Adicionar Expertos

4.2 Adicionar Criterios

4.3 Eliminar Expertos

4.4 Eliminar Criterios

4.5 Listar Proyectos

R5 Gestionar Criterios

5.1 Adicionar Criterios

5.2 Modificar Criterios

5.3 Eliminar Criterios

R6 Calcular Concordancia

6.1 Calcular la concordancia de los expertos.

R7 Evaluar Criterio

7.1 Asignar evaluación a un criterio

R8 Aprobar Proyecto

8.1 Listar Proyectos

8.2 Mostrar Matriz de Decisión

R9 Calcular Factibilidad

9.1 Calcular índice de aceptación del Proyecto.

R10 Gestionar País

10.1 Adicionar País

10.2 Modificar País

10.3 Eliminar País

R11 Gestionar Entidad

11.1 Adicionar Entidad

11.2 Modificar Entidad

11.3 Eliminar Entidad

R12 Gestionar Criterios Propuestos

12.1 Adicionar Criterio Propuesto

12.2 Modificar Criterio Propuesto

12.3 Eliminar Criterio Propuesto

2.4.2 Requisitos no funcionales

1. Apariencia o interfaz externa

La aplicación propuesta será usada por personas que tengan o no conocimientos básicos de informática, la interfaz debe ser amigable y fácil de usar, de manera que no sea una dificultad para los usuarios el trabajo con la misma.

2. Usabilidad

A los administradores del sistema se les dará un adiestramiento básico en el uso de la aplicación. Estas personas tendrán un nivel de acceso rápido y amplio en la aplicación para poder darle respuesta a cada incidente que ocurra.

3. Rendimiento

Para un funcionamiento óptimo de la aplicación se seguirán las diferentes técnicas de elaboración de sitios Web, que faciliten el acceso rápido a sus páginas. La eficiencia del producto estará determinada en gran medida por el aprovechamiento de los recursos que se disponen en el modelo cliente/servidor, y la velocidad de la consultas de la base de datos. La aplicación propuesta debe ser rápida y el tiempo de respuesta debe ser el mínimo posible.

4. Seguridad

Confiabilidad: La información manejada por el sistema está protegida de acceso no autorizado, y se especifican los diferentes roles existentes para establecer el acceso de los usuarios a determinadas acciones.

Integridad: La información manejada por el sistema permanecerá inalterada a menos que sea modificada por personal autorizado, esta modificación será registrada, asegurando su precisión y confiabilidad.

Disponibilidad: La aplicación deberá estar disponible en todo momento para aquellas personas con acceso a la información y los mecanismos utilizados para lograr la seguridad no deben ser un obstáculo a los usuarios para obtener los datos deseados en un momento dado.

5. Soporte

Se le debe dar un mantenimiento periódico a la aplicación y a la Base de Datos.

6. Portabilidad

El sistema podrá ser usado en cualquier sistema operativo.

7. Requerimientos de software

En las computadoras de los usuarios solo se requiere un navegador Web, bajo cualquier sistema operativo Windows NT en adelante o cualquier distribución de Linux. En el servidor de base de datos se requiere Windows NT en adelante o cualquier distribución de Linux.

8. Requerimientos de hardware

En el cliente se requiere una máquina con 256 MB de RAM y un microprocesador de 1 GHz como mínimo, el servidor Web junto con el servidor de base de datos debe tener 512 MB de RAM, microprocesador de 2.00 GHz y 40 GB de disco duro mínimo.

Todas las máquinas implicadas en la funcionalidad de la aplicación incluyendo las PC clientes deben estar conectadas a una red de 100 Mbps de velocidad con el objetivo de lograr un trabajo más cómodo.

2.5 Modelo de Casos de Uso del Sistema

2.5.1 Actores del Sistema

Actores del sistema	**Descripción**
Usuario	Usuario global que permite la autenticación en el sistema y que valida al mismo dándole un rol ya sea de Administrador, Analista, Experto o de Decisor.

Administrador	Es el encargado de gestionar usuarios que interactúen con el sistema así como realizar la gestión de entidades, países y criterios propuestos al Analista para la evaluación de proyectos de software.
Analista	Es el que inicializa las acciones fundamentales en el sistema al registrar datos del proyecto, seleccionar los criterios, expertos y comprobar la consistencia entre los mismos.
Experto	Su tarea fundamental está dada por la evaluación de los criterios seleccionados por el analista.
Decisor	Se encarga de analizar el resultado de factibilidad emitido por el sistema y tomar la decisión de aceptación o rechazo del proyecto de software.

Tabla. 2.3 Actores del Sistema

2.5.2 Diagrama de Caso de Uso del Sistema.

Diagrama de casos de uso del sistema

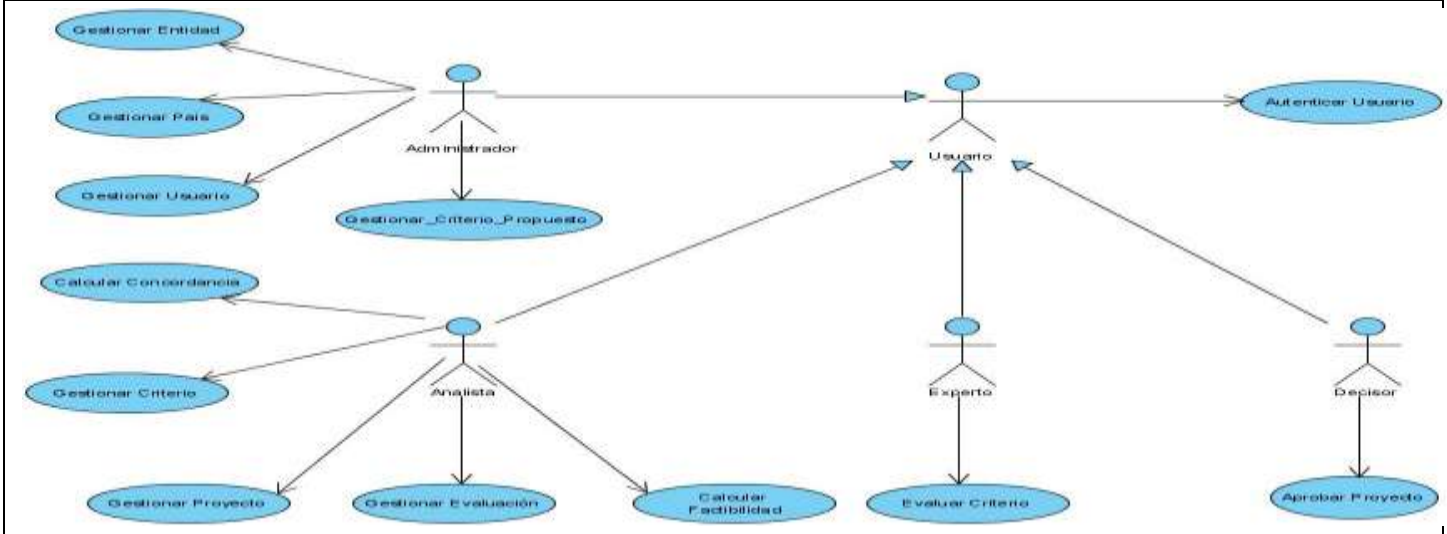


Fig. 5. 5 Diagrama de casos de uso del sistema

2.5.3 Descripción textual de los Casos de Uso del Sistema.

La descripción de los casos de uso del sistema constituye una parte muy valiosa, pues se explica de forma detallada la interacción entre el sistema y el usuario. El anexo 6 muestra la descripción de cada uno de los casos de uso del sistema: Entorno Virtual de Evaluación de Proyectos Informáticos (EVEPI).

2.6 Conclusiones.

En el presente capítulo se realizó un estudio preciso de los procesos fundamentales del negocio que sirvieron como guía para la identificación de las funcionalidades y requerimientos que debe cumplir la aplicación. La descripción del negocio así como la comprensión de su funcionamiento permitieron una visión general del problema y de lo que será el producto, creando la base fundamental para la realización del análisis y el diseño del sistema.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

3.1 Introducción.

El objetivo principal del análisis y diseño del sistema es transformar los requerimientos a una especificación que describa cómo implementar el sistema. El análisis fundamentalmente consiste en obtener una visión que se preocupa de ver qué hace el sistema de software a desarrollar, por tal motivo este se interesa en los requerimientos funcionales. Por otro lado, el diseño es un refinamiento que toma en cuenta los requerimientos no funcionales, por lo cual se centra en como el sistema cumple sus objetivos. (35)

En este capítulo se definirán algunas características del modelo de análisis y diseño así como la representación de sus diagramas correspondientes; además de la realización de los diagramas de interacción, modelo lógico de datos y el modelo físico de datos.

3.2 Modelo de Análisis.

En el flujo de trabajo de análisis se trabaja sobre los aspectos internos del sistema a desarrollar, se pueden estructurar los requisitos de manera que faciliten su comprensión, su preparación, su modificación y en general su mantenimiento. En el modelo de análisis se obtiene un mayor poder expresivo y una mayor formalización y se puede considerar como una primera aproximación al modelo de diseño. Para la construcción del modelo de análisis se identificaron las clases que describen la realización de los casos de uso, que pueden ser de tres tipos fundamentales: clases interfaz, clases entidad y clases controladoras. Las clases Interfaz se utilizan para modelar la interacción entre el sistema y sus actores, lo que clarifican y reúnen los requisitos en los límites del sistema. Representan abstracciones de ventanas, formularios, paneles, interfaces de comunicaciones, interfaces de impresoras, sensores y terminales.

Las clases entidad se utilizan para modelar la información que posee una vida larga y que es a menudo persistente. Modelan la información y el comportamiento asociado de algún fenómeno o concepto y reflejan la información de modo que beneficia a los desarrolladores al diseñar e implementar el sistema, incluyendo su soporte de persistencia.

Las clases controladoras representan la coordinación, secuencia, transacciones, y control de otros objetos, y se usan con frecuencia para encapsular el control de un caso de uso en concreto. Se utilizan para representar derivaciones y cálculos complejos. (35)

3.2.1 Diagramas de Clases del Análisis

DIAGRAMA DE CLASES DEL ANÁLISIS
<CU_ Gestionar Usuario>

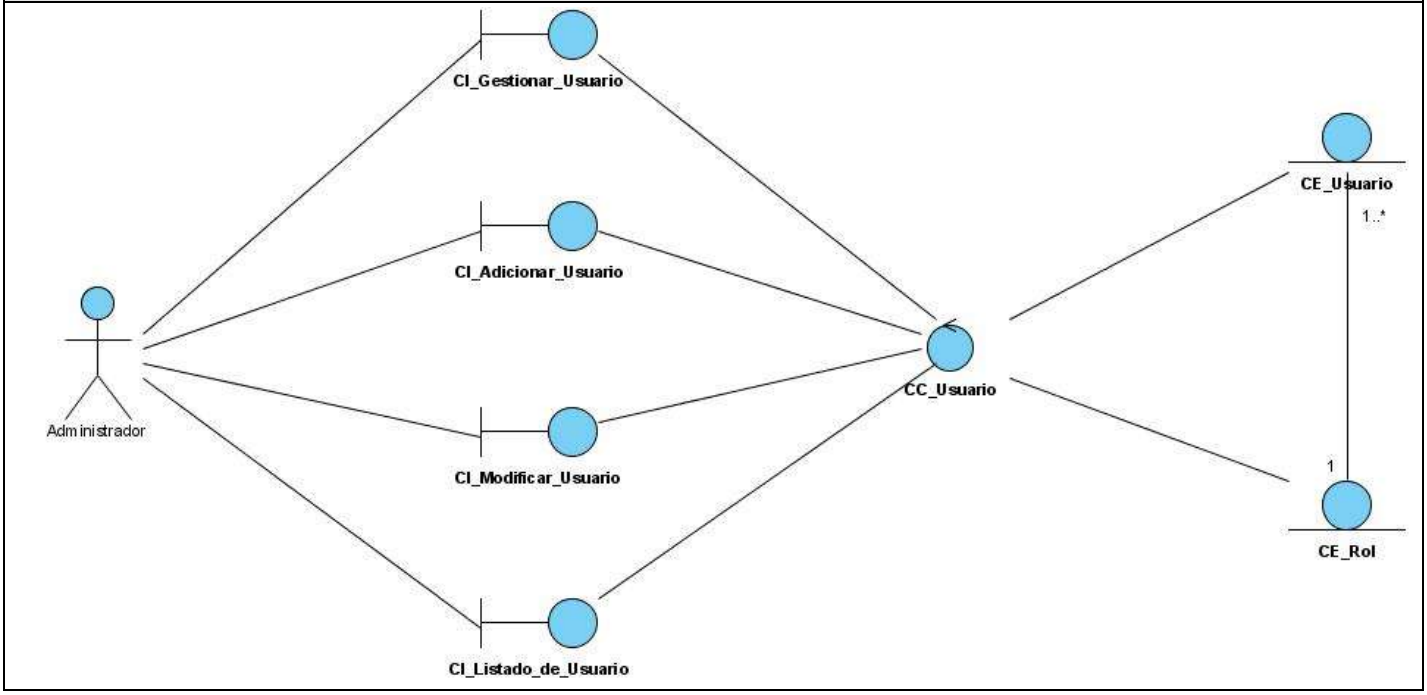


Fig. 3.1 Diagrama de Clases de análisis: Gestionar Usuario

DIAGRAMA DE CLASES DEL ANÁLISIS
<CU_ Gestionar Criterio>

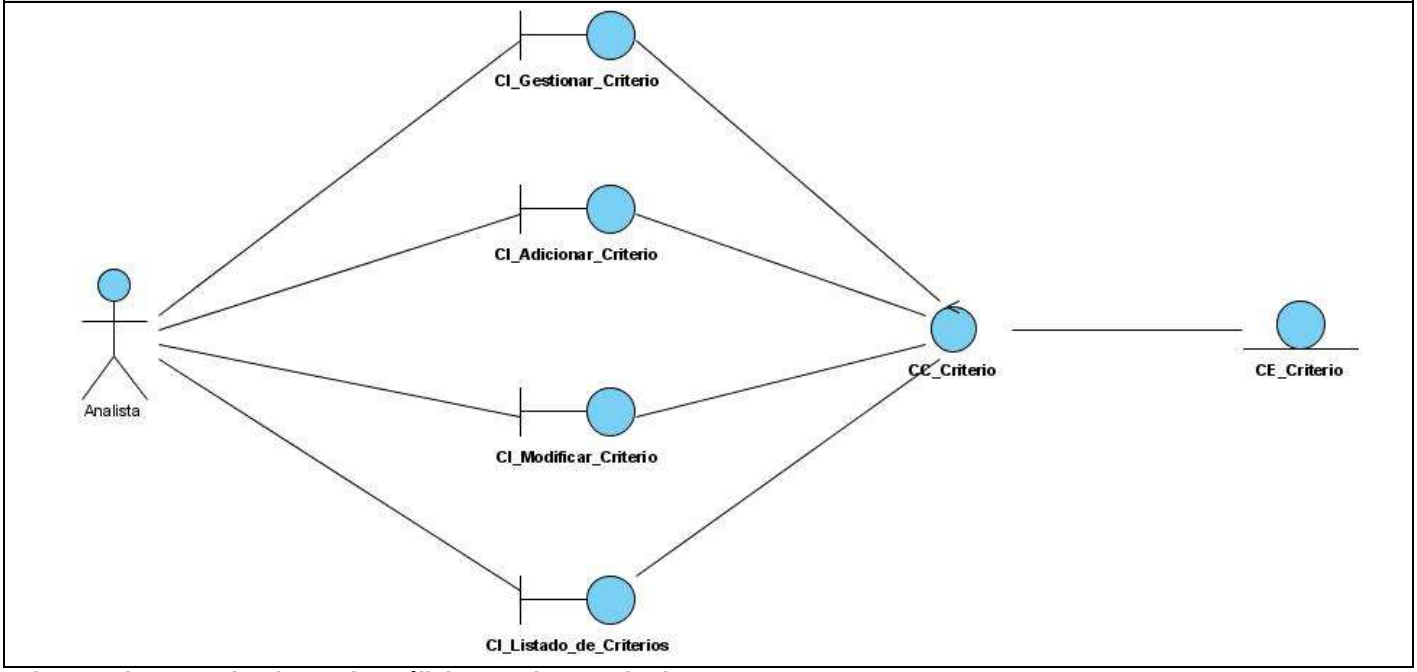


Fig. 3.2 Diagrama de Clases de análisis: Gestionar Criterio

DIAGRAMA DE CLASES DEL ANÁLISIS
<CU_ Calcular Factibilidad>

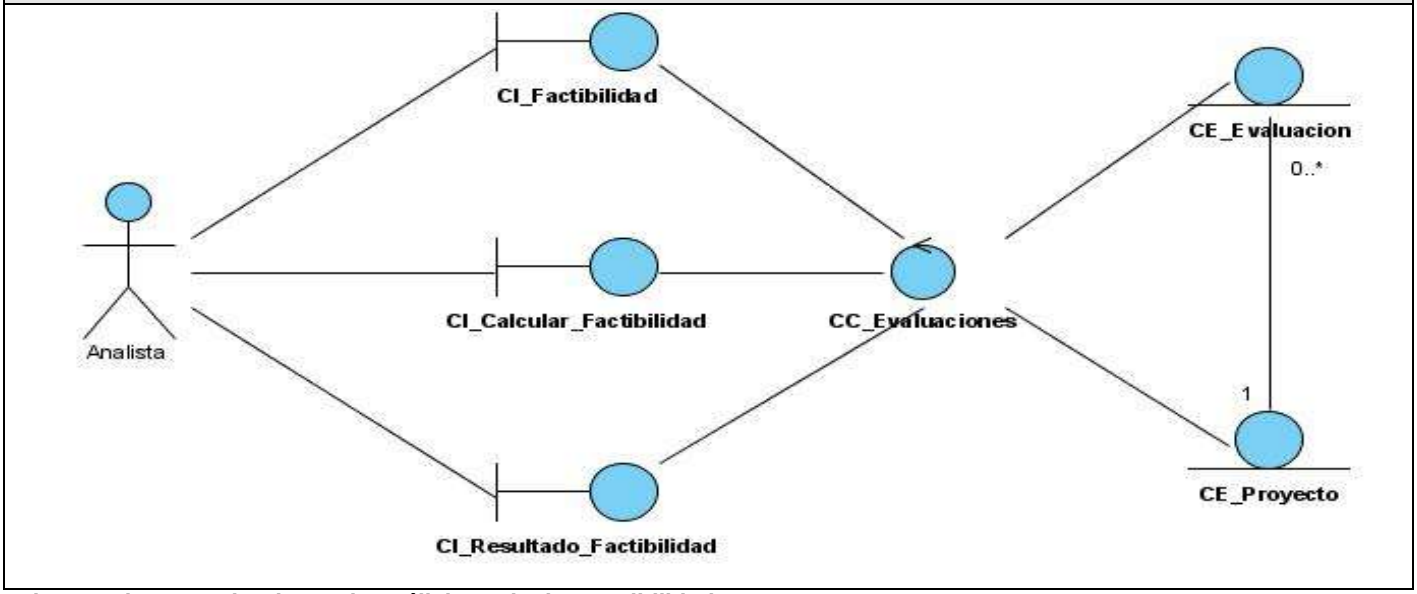


Fig. 3.3 Diagrama de Clases de análisis: Calcular Factibilidad

DIAGRAMA DE CLASES DEL ANÁLISIS
<CU_ Gestionar Proyecto>

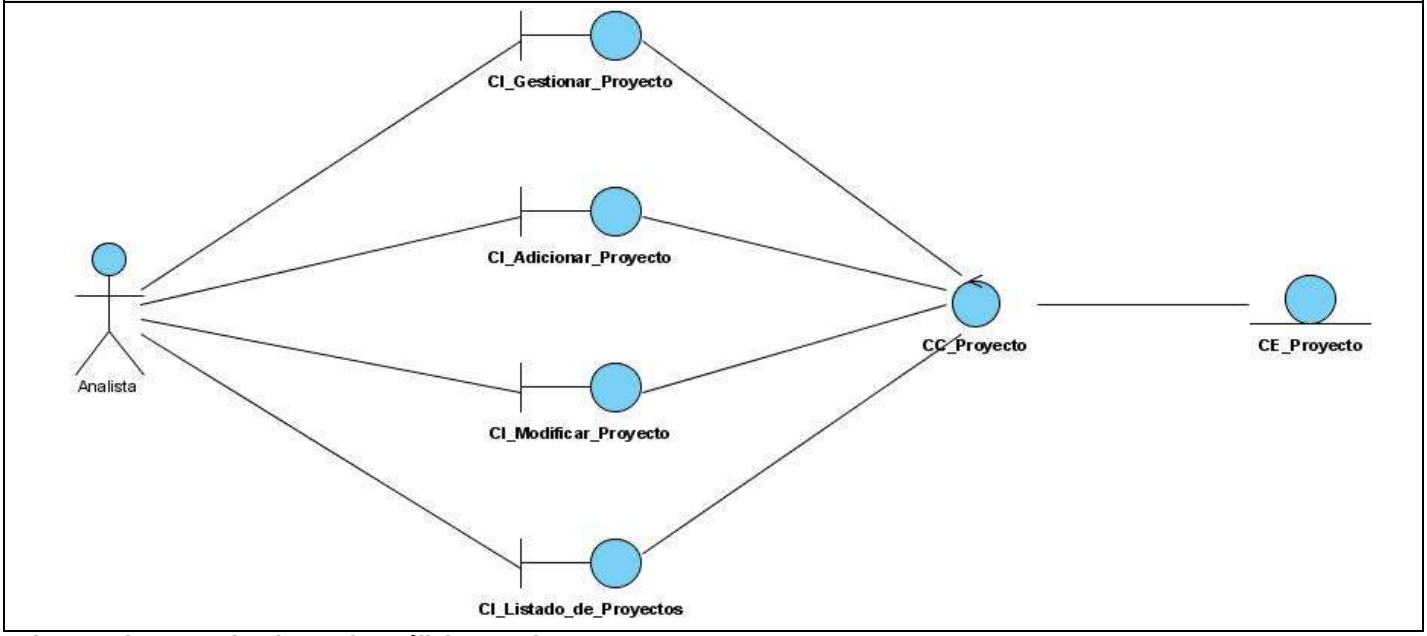


Fig. 3.4 Diagrama de Clases de análisis: Gestionar Proyecto

DIAGRAMA DE CLASES DEL ANÁLISIS
<CU_ Gestionar Evaluación>

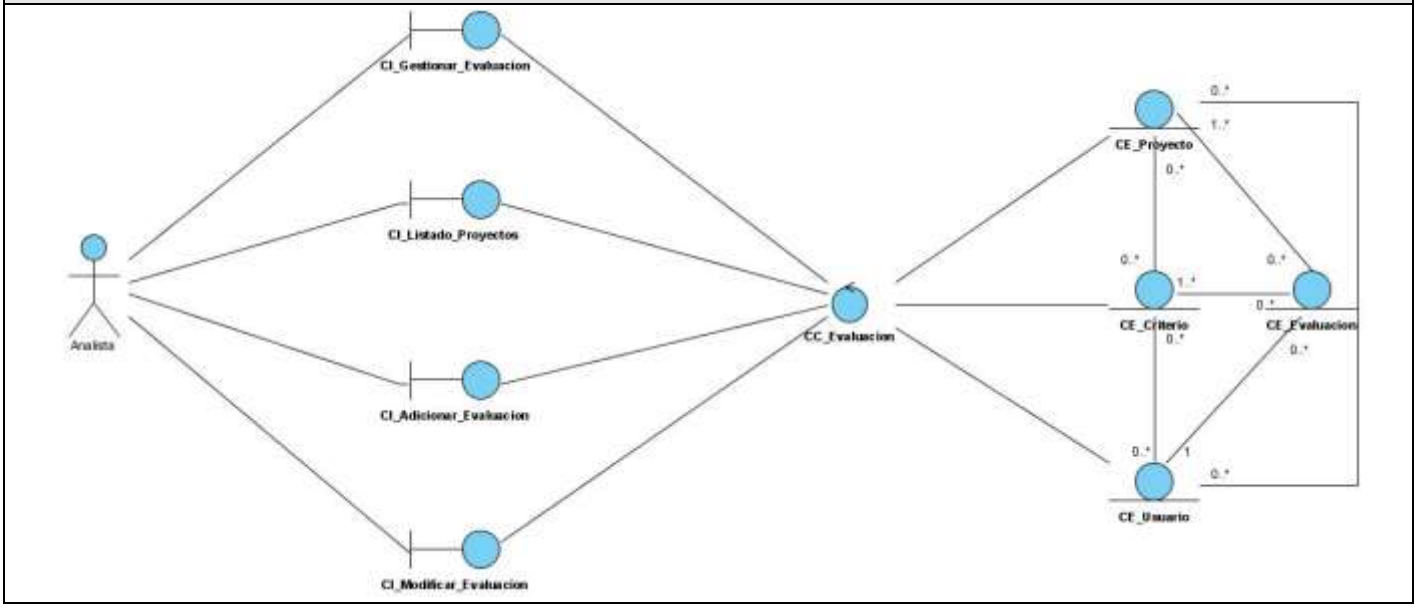


Fig. 3.5 Diagrama de Clases de análisis: Gestionar Evaluación

DIAGRAMA DE CLASES DEL ANÁLISIS
<CU_ Autenticar Usuario>

Fig. 3.6 Diagrama de Clases de análisis: Autenticar Usuario

DIAGRAMA DE CLASES DEL ANÁLISIS
<CU_ Aprobar Proyecto>

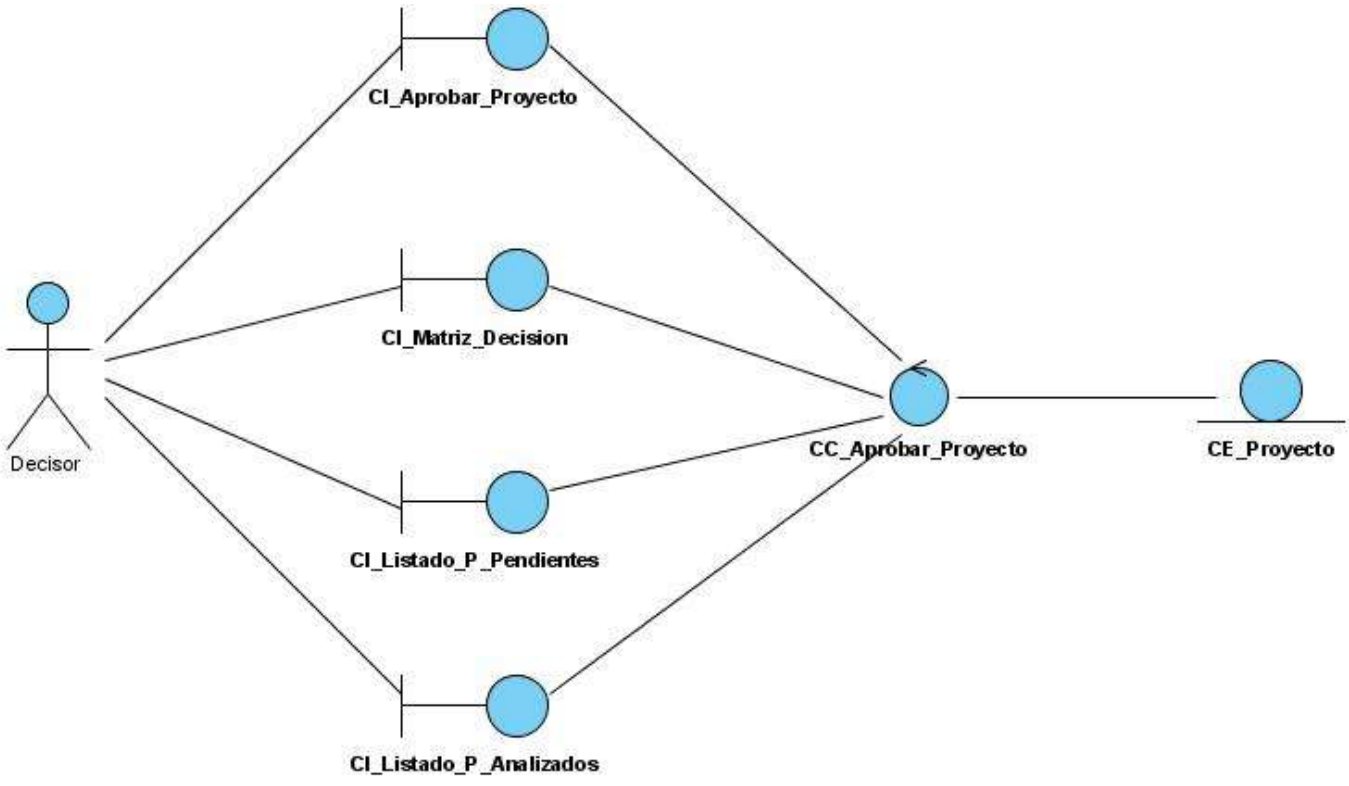


Fig. 3.7 Diagrama de Clases de análisis: Aprobar Proyecto

DIAGRAMA DE CLASES DEL ANÁLISIS
<CU_ Calcular Concordancia>

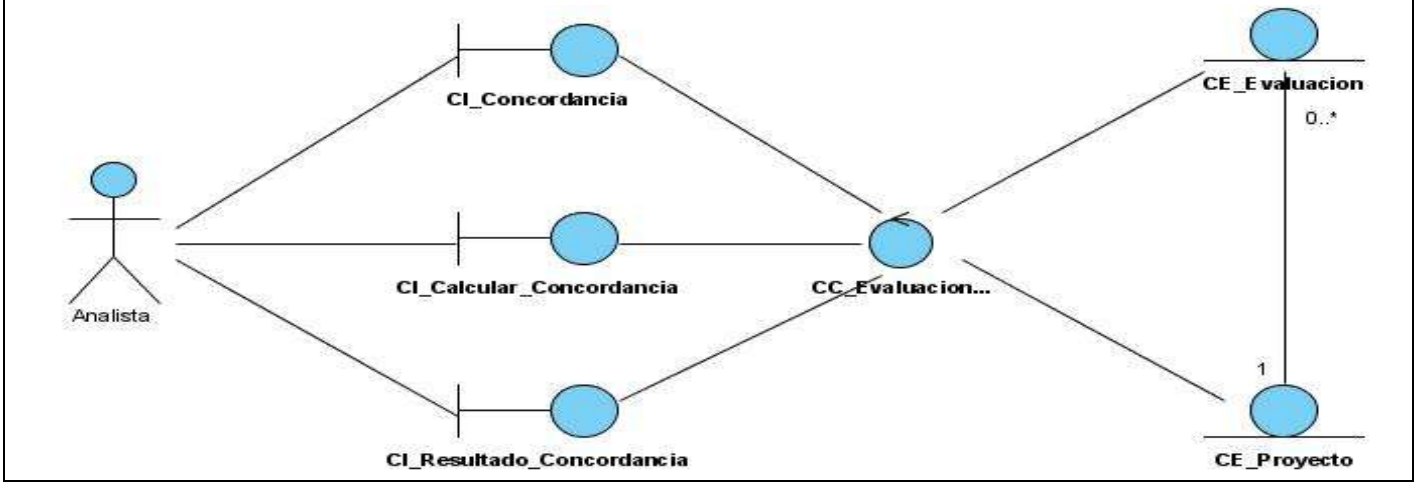


Fig. 3.8 Diagrama de Clases de análisis: Calcular Concordancia

DIAGRAMA DE CLASES DEL ANÁLISIS
<CU_ Evaluar Criterio>

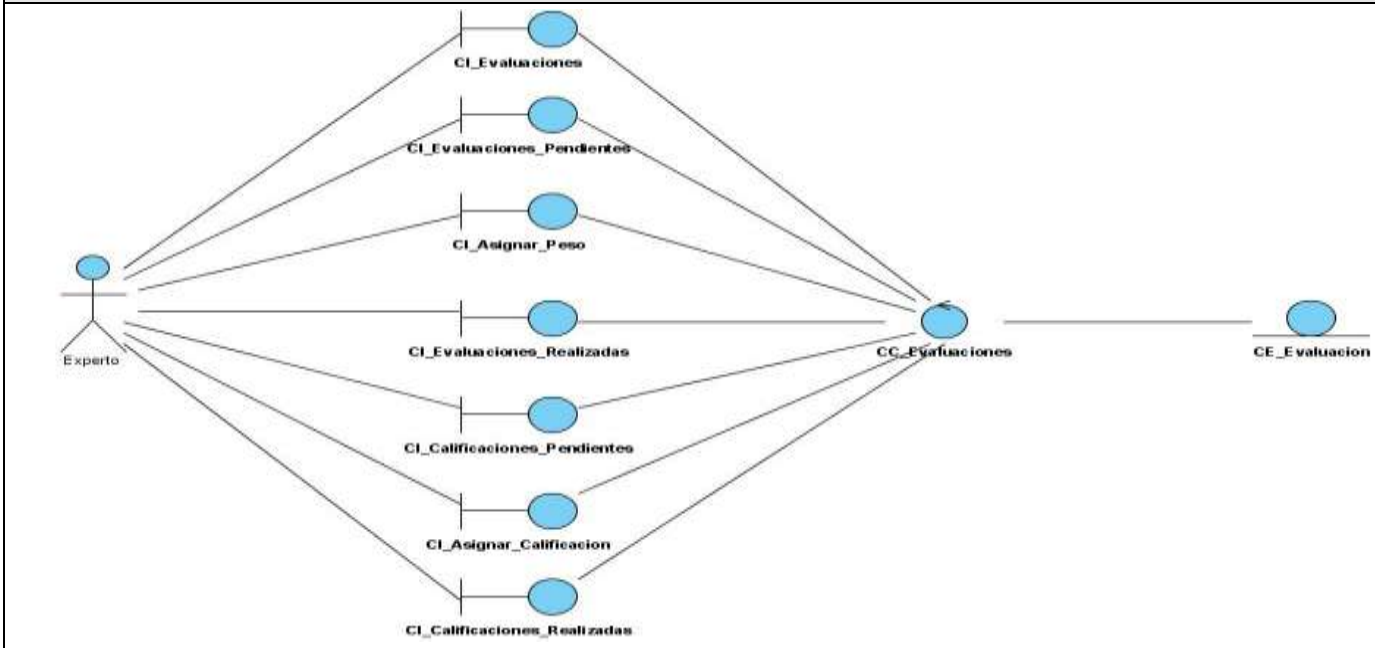


Fig. 3 9 Diagrama de Clases del Análisis: Evaluar Criterio

DIAGRAMA DE CLASES DEL ANÁLISIS
<CU_ Gestionar País>

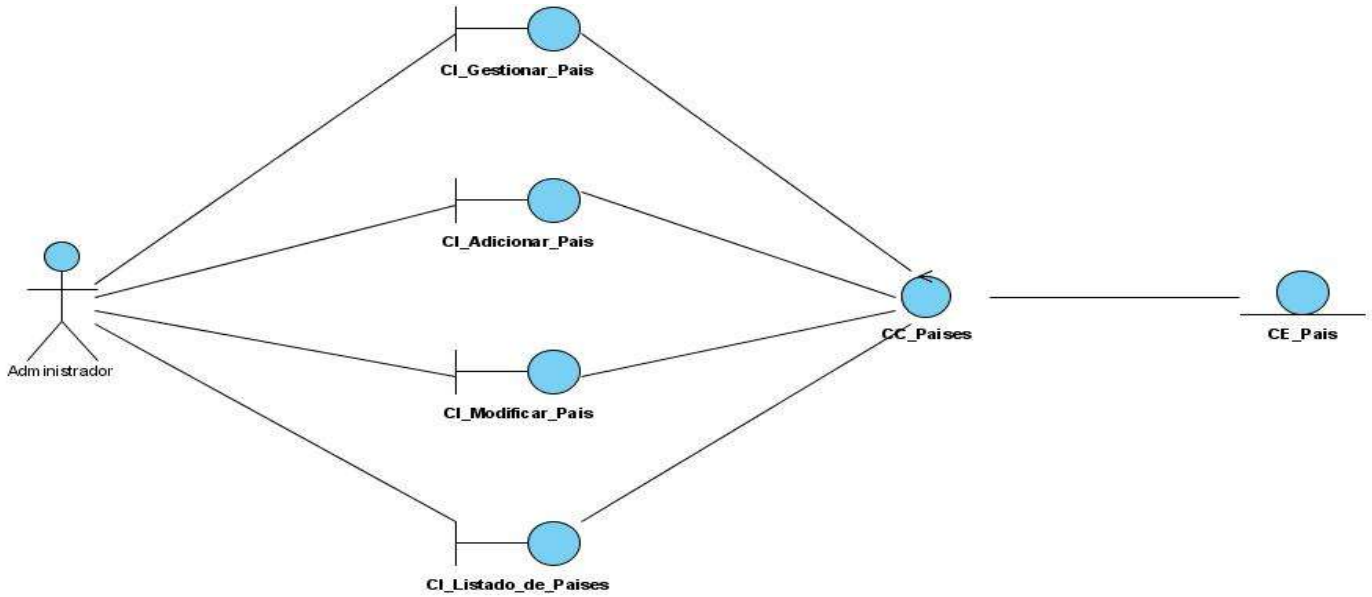


Fig. 3 10 Diagrama de Clases del Análisis: Gestionar País

DIAGRAMA DE CLASES DEL ANÁLISIS
<CU_ Gestionar Entidad>

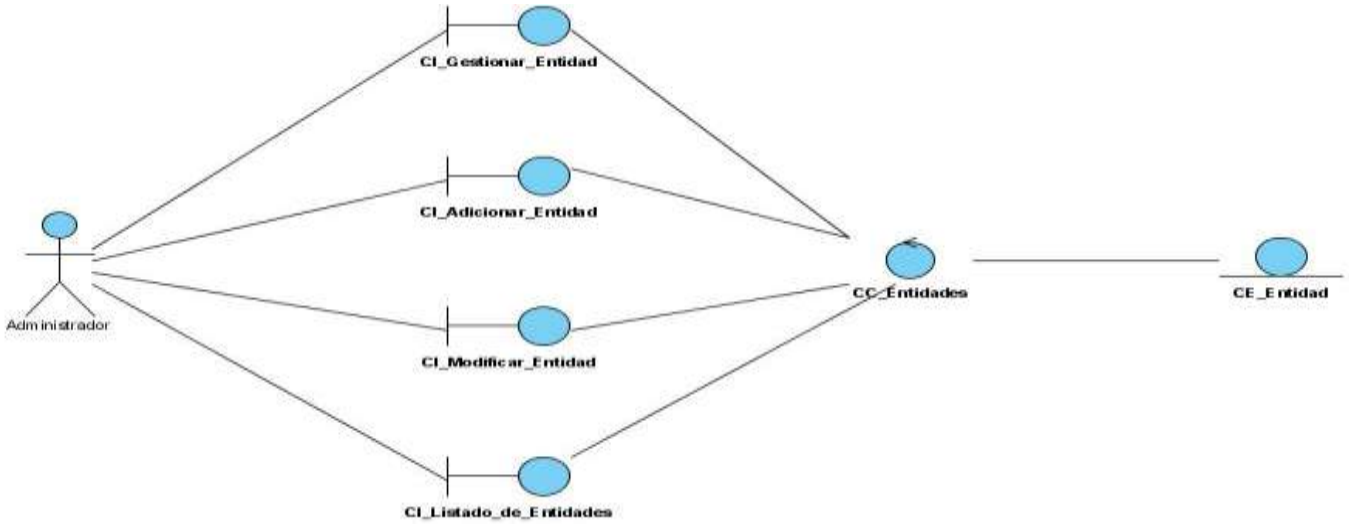


Fig. 3 11 Diagrama de Clases del Análisis: Gestionar Entidad

DIAGRAMA DE CLASES DEL ANÁLISIS
<CU_ Gestionar Criterio Propuesto>

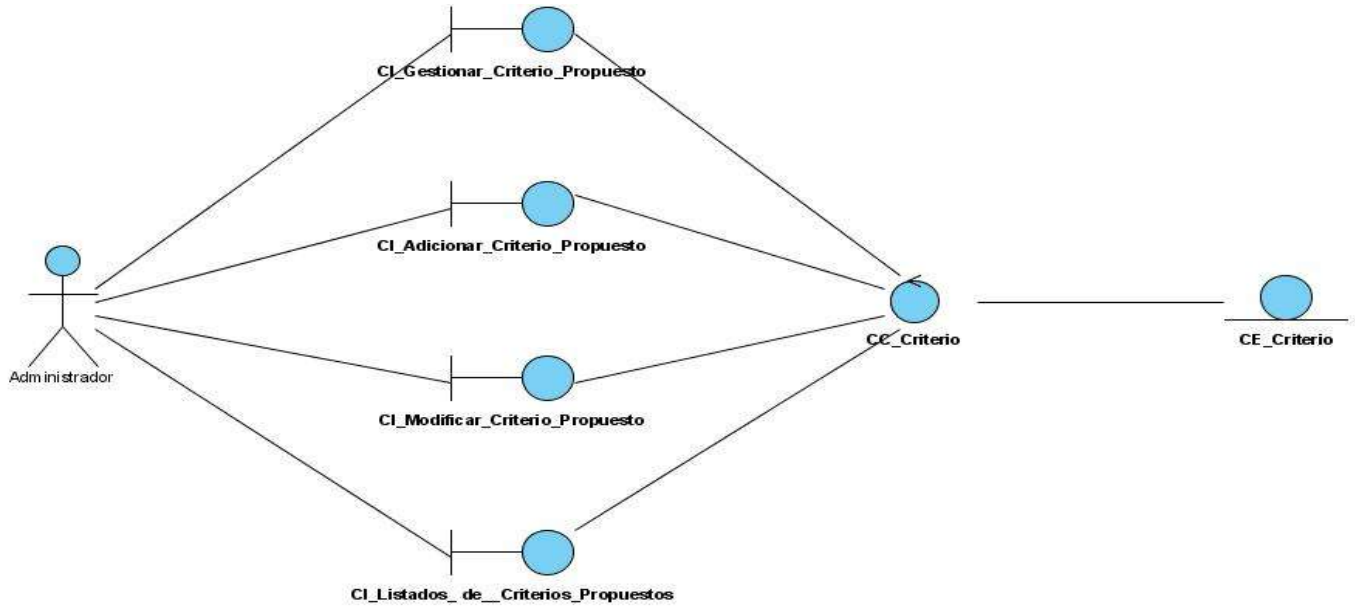


Fig. 3 12 Diagrama de Clases del Análisis: Gestionar Criterio Propuesto

3.3 Diseño

El Modelo de Diseño es una abstracción del Modelo de Implementación y su código fuente, el cual fundamentalmente se emplea para representar y documentar su diseño. Representa a los casos de uso en el dominio de la solución.

El Modelo de Diseño puede contener: los diagramas, clases, paquetes, subsistemas, cápsulas, protocolos, interfaces, relaciones, colaboraciones, atributos, las realizaciones de los casos de uso, entre otros que se puedan considerar para el sistema en desarrollo. (35)

3.3.1 Realización de Casos de Uso del Diseño

Las realizaciones de casos de uso del diseño según el Proceso Unificado de Desarrollo es una colaboración en el modelo de diseño que describe cómo se realiza un caso de uso específico, y cómo se

ejecuta, en términos de clases de diseño y sus objetos. Una realización de casos de uso-diseño proporciona una traza directa a una realización de casos de uso-análisis en el modelo de análisis.

El artefacto subsistema de diseño es una forma de organizar los artefactos del modelo de diseño en piezas más manejables, puede constar de clases de diseño, realizaciones de casos de uso, interfaces y otros subsistemas. (36)

Los estereotipos que usa esta extensión son:

Fig. 3.13. <<Form>>

Grupo de elementos de entrada que son parte de una página cliente. Se relaciona directamente con la etiqueta de igual nombre del HTML.

Fig. 3.14. <<Client Page>>

Una instancia de Página Cliente es una página Web, con formato HTML; mezcla de datos, presentación y lógica. Son interpretadas por el browser. Cada página cliente solo puede ser construida por una página servidor.

Fig. 3.15. <<Server Page>>

Representa la página Web que tiene código que se ejecuta en el servidor. Este código interactúa con recursos en el servidor. Las operaciones representan las funciones del código y los atributos las variables visibles dentro del alcance de la página.

3.3.2 Patrones de Diseño

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño, de optimización de código, reutilización de componentes en un contexto particular. Identifica: clases, instancias, roles, colaboraciones y la distribución de responsabilidades. (36)

3.3.2.1 Patrones GRASP

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. GRASP es un acrónimo que significa General Responsibility Asignment Software Patterns (patrones generales de software para asignar responsabilidades). (37)

Estos fueron los patrones que se aplicaron para realizar el diseño de la aplicación que propone la investigación.

- **Experto**: Es un patrón que se usa más que cualquier otro al asignar responsabilidades; es un principio básico que suele utilizarse en el diseño orientado a objetos. Con él no se pretende designar una idea oscura ni extraña; expresa simplemente la "intuición" de que los objetos hacen cosas relacionadas con la información que poseen.

  Este es uno de los más utilizados en el desarrollo de la aplicación, por ejemplo Propel es la librería externa que utiliza Symfony como ORM, encapsula toda la lógica de los datos y son generadas las clases con todas las funcionalidades comunes de las entidades.

- **Creador**: El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento.

  Un ejemplo de esto lo veríamos en las clases Peer, las cuales crean los diferentes objetos de las clases entidades relacionadas a cada una de ellas.

- **Alta Cohesión**: Asignar una responsabilidad de modo que la cohesión siga siendo alta. Una clase con baja cohesión hace muchas cosas no afines o un trabajo excesivo. Una clase con mucha cohesión es útil porque es bastante fácil darle mantenimiento, entenderla y reutilizarla.

  Un ejemplo de esto es que Symfony permite asignar responsabilidades con una alta cohesión, por ejemplo la clase Actions tiene la responsabilidad de definir las acciones para las plantillas y colabora con otras para realizar diferentes operaciones, instanciar objetos y acceder a las propiedades, es decir, está formada por diferentes funcionalidades que se encuentran estrechamente relacionadas proporcionando que el software sea flexible frente a grandes cambios.

- **Bajo Acoplamiento:** Asignar una responsabilidad para mantener bajo acoplamiento. Las clases deben comunicarse con un número pequeño de clases tanto como sea posible.

  Ejemplo de esto se evidencia en la clase Actions, la cual hereda solamente de sfActions para lograr un bajo acoplamiento de clases.

- **Controlador:** Funciona como intermediario entre una interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado.

  Este patrón **r**ecomienda dividir los eventos del sistema en el mayor número de controladores para poder aumentar la cohesión y disminuir el acoplamiento.

  Ejemplo de esto se evidencia en que todas las peticiones Web son manejadas por un solo controlador frontal (sfController), que es el punto de entrada único de toda la aplicación en un entorno determinado. 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 entrada por el usuario.

3.3.2.2 Patrones GOF

Symfony utiliza una serie de patrones GOF (Gang of Four) como son:

En la categoría Creacionales:

- **Singleton (Instancia única):** Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. En el controlador frontal hay una llamada a sfContext :: createInstance (). En una acción, el método getContext () devuelve la misma instancia, un objeto muy útil que guarda una referencia a todos los objetos del núcleo de Symfony.

- **Abstract Factory (Fábrica abstracta):** Permite trabajar con objetos de distintas familias de manera que las familias no se mezclen entre sí y haciendo transparente el tipo de familia concreta que se esté usando. Cuando el framework necesita por ejemplo crear un nuevo objeto para una petición, busca en la definición de la factoría el nombre de la clase que se debe utilizar para esta tarea.

En la categoría Estructurales:

- **Decorator (Envoltorio):** Añade funcionalidad a una clase, dinámicamente. El archivo layout.php, 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 el Layout, o si se mira desde el otro punto de vista, el Layout decora la plantilla.

3.3.3 Diagramas de Interacción

Los diagramas de interacción se utilizan para modelar los aspectos dinámicos de un sistema, (ver anexo 7) lo que conlleva modelar instancias concretas o prototípicas de clases interfaces, componentes y nodos, junto con los mensajes enviados entre ellos, todo en el contexto de un escenario que ilustra un comportamiento. En el contexto de las clases describen la forma en que grupos de objetos colaboran para proveer un comportamiento. (38)

3.3.4 Diagrama de clases del diseño

La Fig. 3.16 muestra la estructura general de los Diagramas de Clases del Diseño para todos los Casos de Uso del Sistema el cual está dividido en tres paquetes: Vista, Controlador y Modelo.

El paquete Vista está formado por el Layout, denominado también Plantilla Global que guarda el código HTML que es común a todas las páginas del sistema para no tener que repetirlo y su relación con las Páginas Clientes con sus respectivos formularios representados en este paquete con nombres genéricos CP_plantillaSuccess y Form_plantilla respectivamente.

El paquete Controlador contiene la(s) página(s) servidora(s) y la Clase Acción asociadas al caso de uso que se desee modelar y al Controlador Frontal que crea Symfony por defecto encargado de la configuración y de determinar la acción a ejecutarse, siendo el único punto de entrada a la aplicación.

El paquete Modelo está constituido por las cuatro clases que Symfony crea por cada tabla en la Base de Datos. Para este diagrama son:

- **Tabla1:** Clase Personalizada de objetos que hereda todos los métodos de la tabla BaseTabla1.
- **Tabla1Peer:** Clase Personalizada que hereda todos los métodos de la tabla BaseTabla1Peer, tiene métodos estáticos para trabajar con las tablas de la base de datos. Sus métodos devuelven normalmente un objeto o una colección de objetos de la clase objeto relacionada.
- **BaseTabla1:** Clase Base del Modelo de la cual hereda Tabla1.
- **BaseTabla1Peer:** Clase Base del Modelo de la cual hereda Tabla1Peer.

Estructura general de los Diagramas de Clases del Diseño

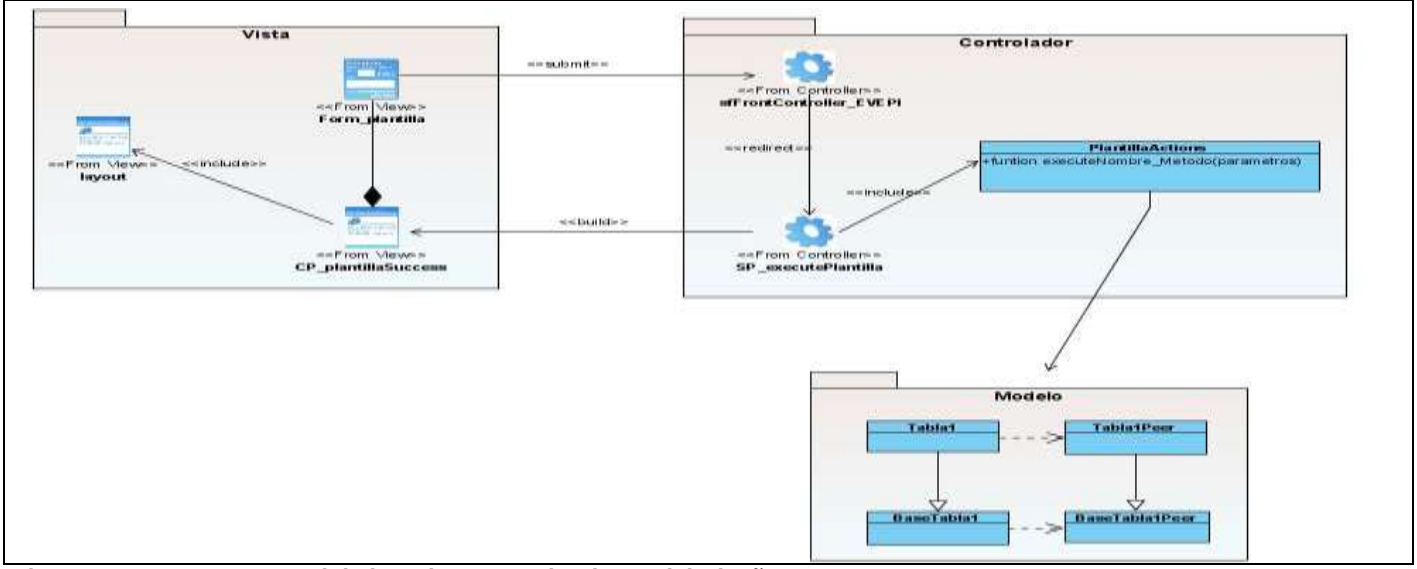


Fig. 3.16 Estructura general de los Diagramas de Clases del Diseño

DIAGRAMA DE CLASES DEL DISEÑO
< CU_ Calcular Factibilidad>

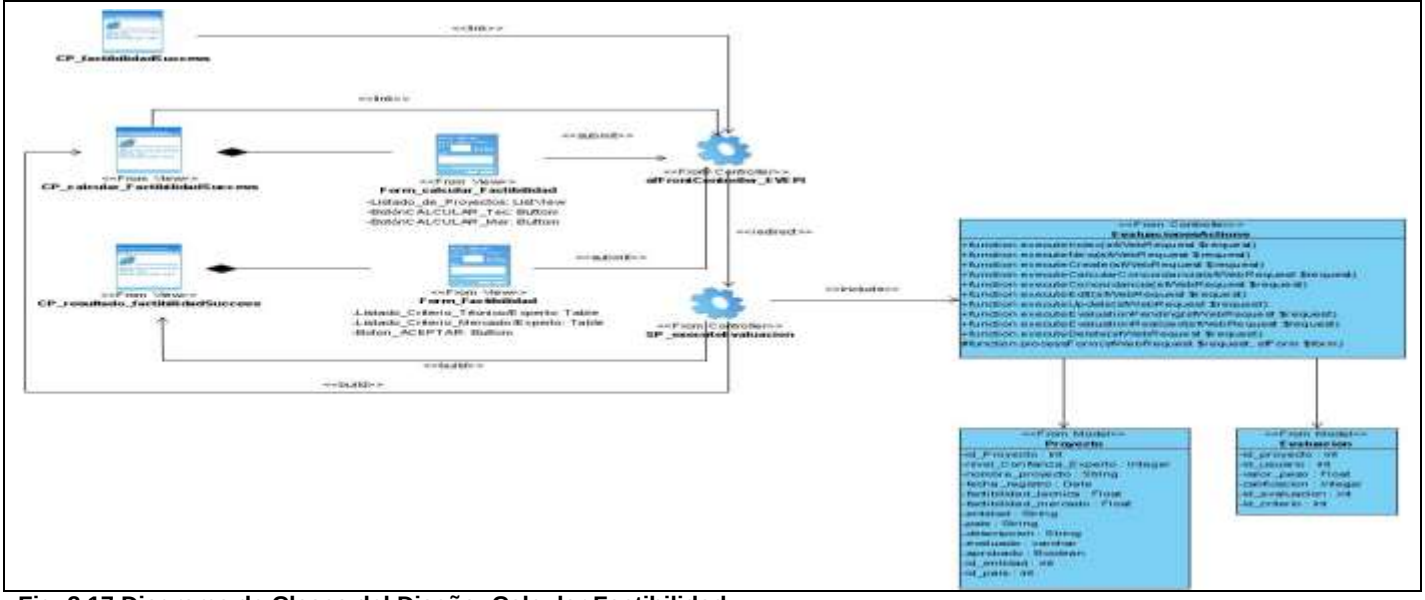


Fig. 3.17 Diagrama de Clases del Diseño: Calcular Factibilidad

DIAGRAMA DE CLASES DEL DISEÑO
< CU_ Aprobar Proyecto>

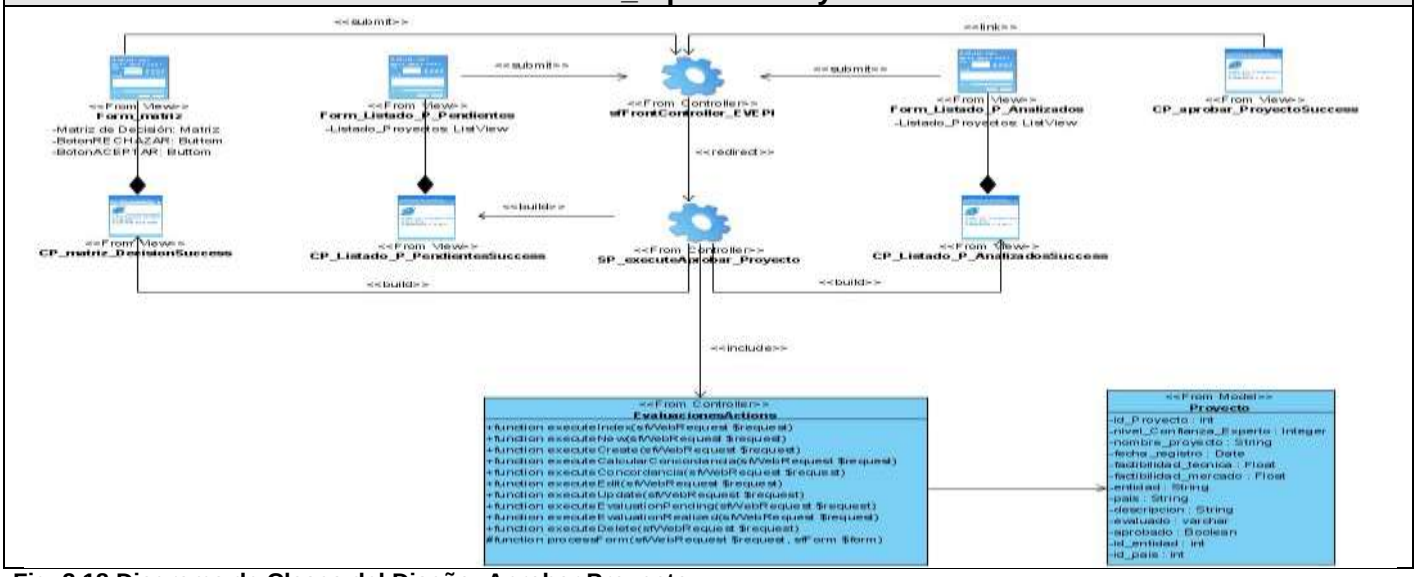


Fig. 3.18 Diagrama de Clases del Diseño: Aprobar Proyecto

DIAGRAMA DE CLASES DEL DISEÑO
< CU_ Gestionar Criterio>

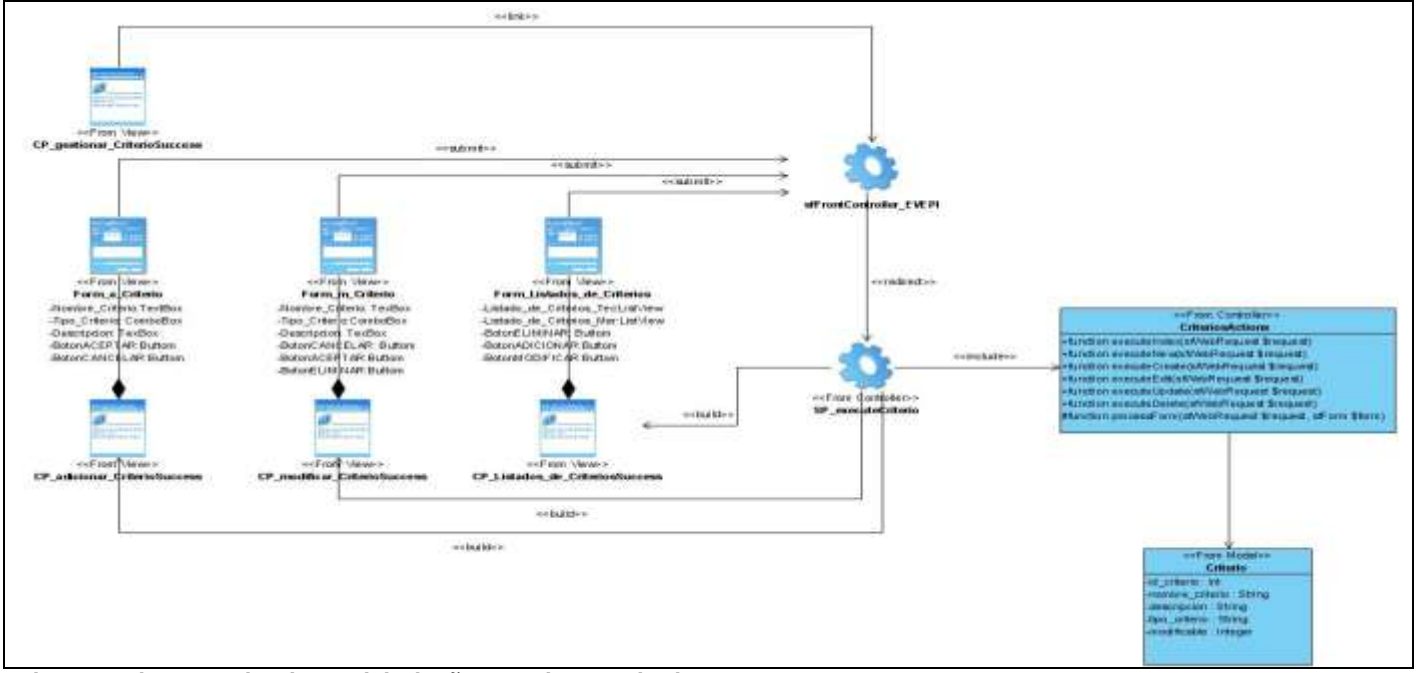


Fig. 3.19 Diagrama de Clases del Diseño: Gestionar Criterio

DIAGRAMA DE CLASES DEL DISEÑO
< CU_ Gestionar Proyecto >

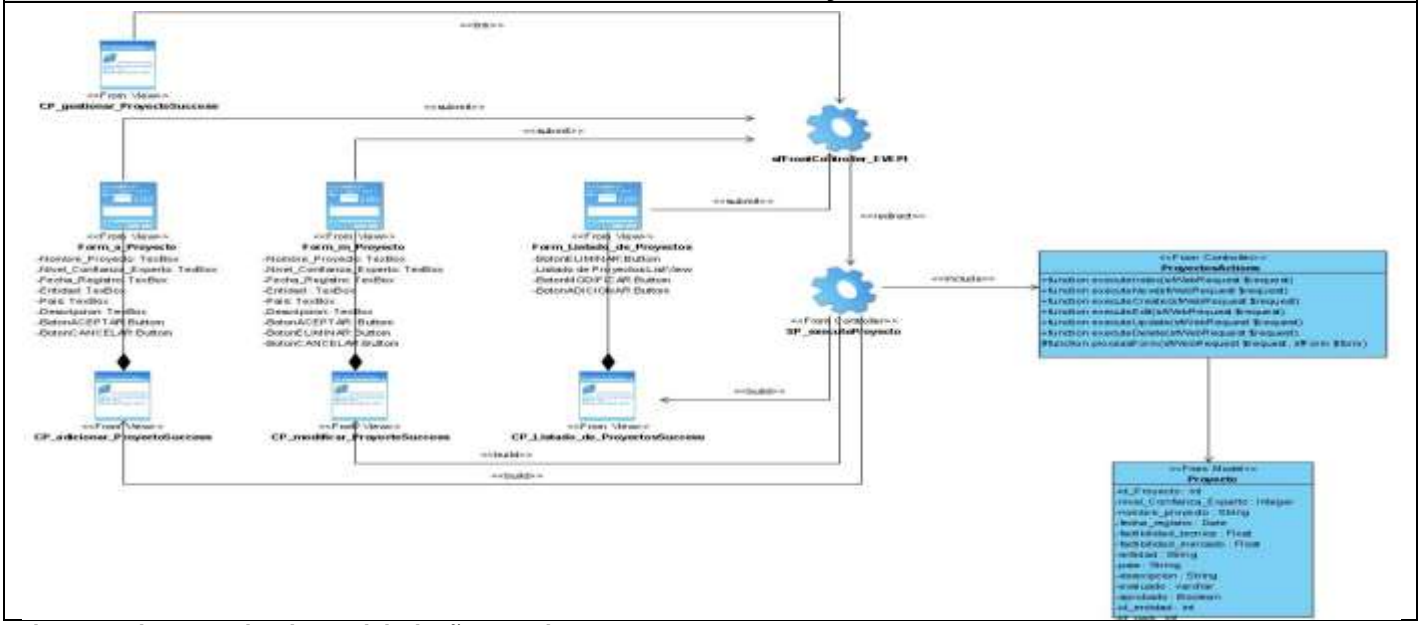


Fig. 3.20 Diagrama de Clases del Diseño: Gestionar Proyecto

DIAGRAMA DE CLASES DEL DISEÑO
< CU_ Gestionar Evaluación>

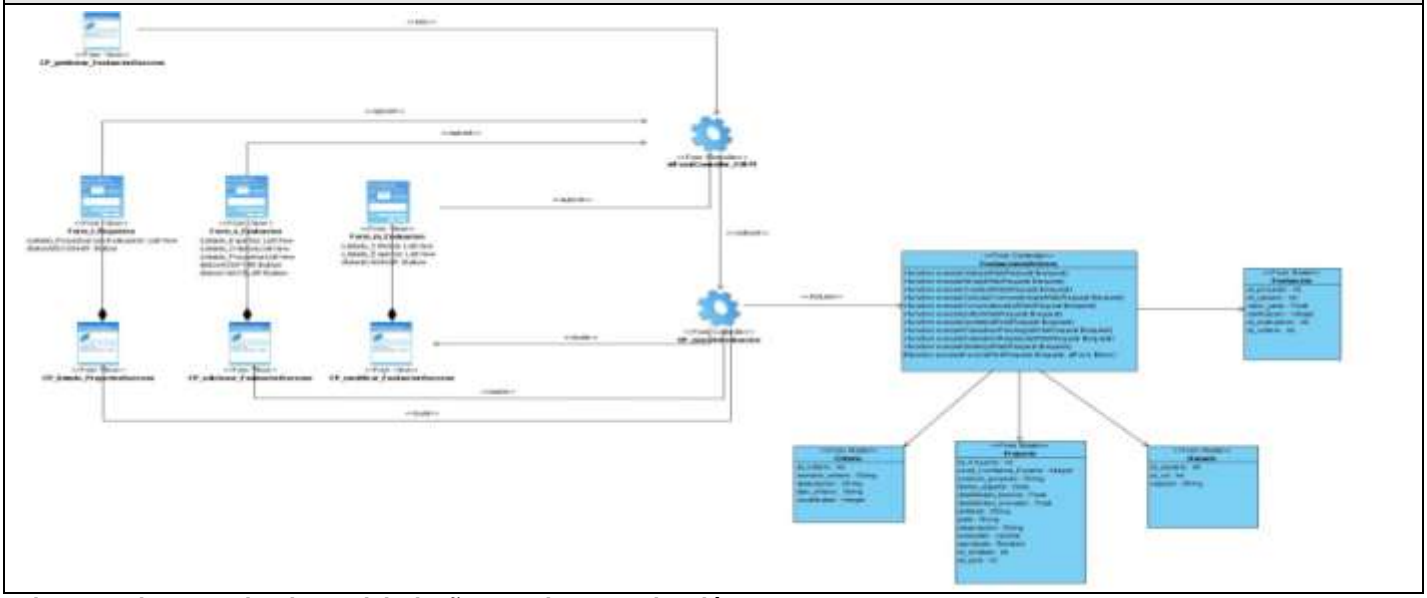


Fig. 3.21 Diagrama de Clases del Diseño: Gestionar Evaluación

DIAGRAMA DE CLASES DEL DISEÑO
< CU_ Evaluar Criterio>

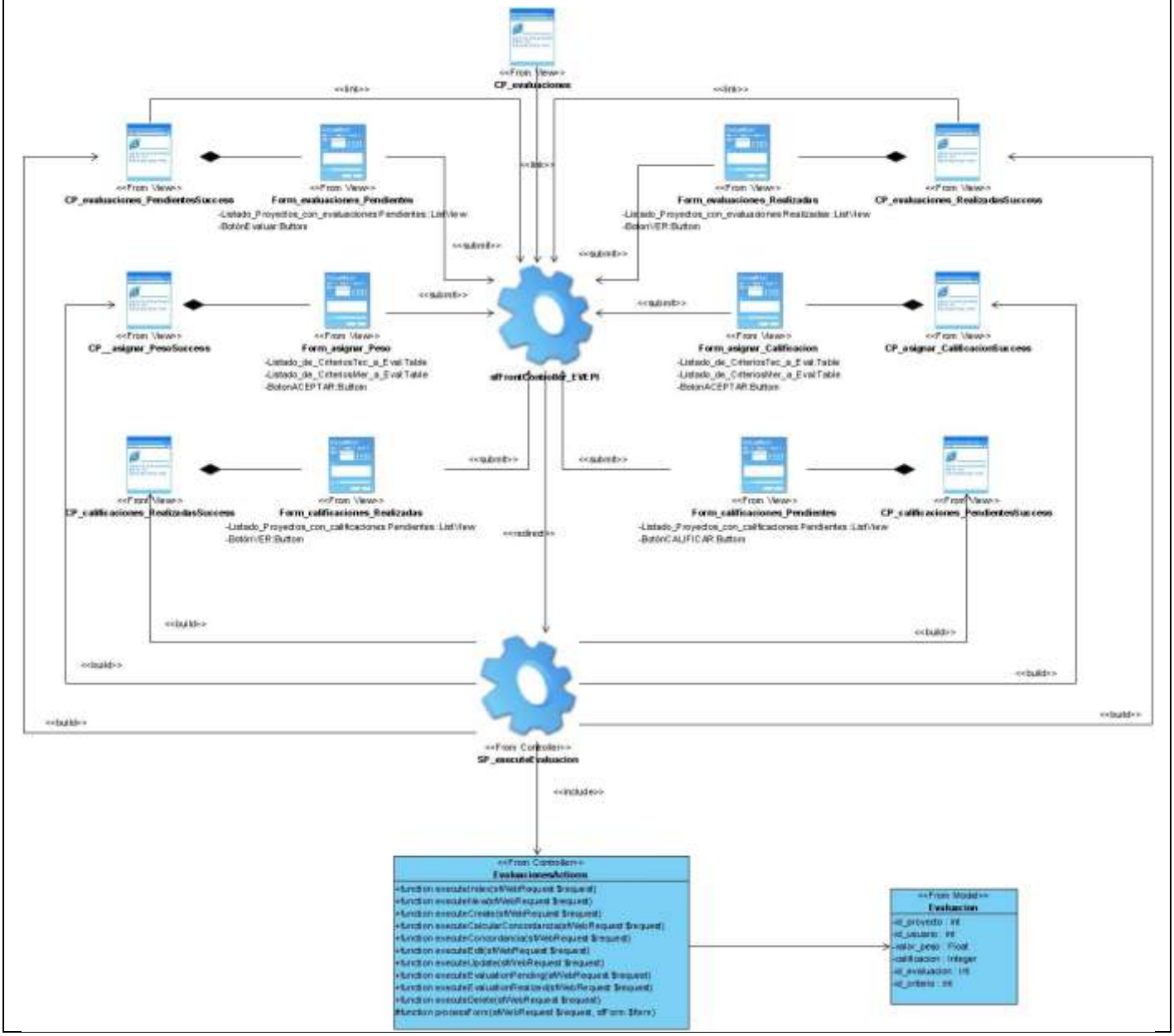


Fig. 3.22 Diagrama de Clases del Diseño: Evaluar Criterio

DIAGRAMA DE CLASES DEL DISEÑO
< CU_ Gestionar Usuario>

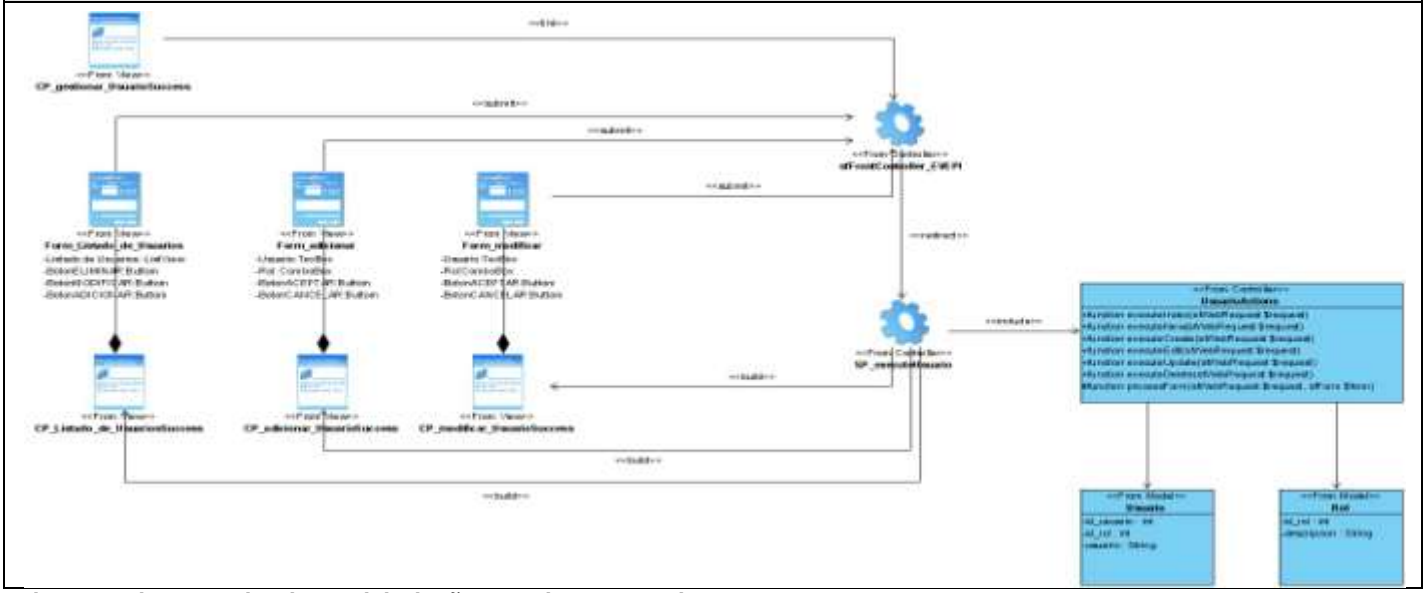


Fig. 3.23 Diagrama de Clases del Diseño: Gestionar Usuario

DIAGRAMA DE CLASES DEL DISEÑO
< CU_ Gestionar País>

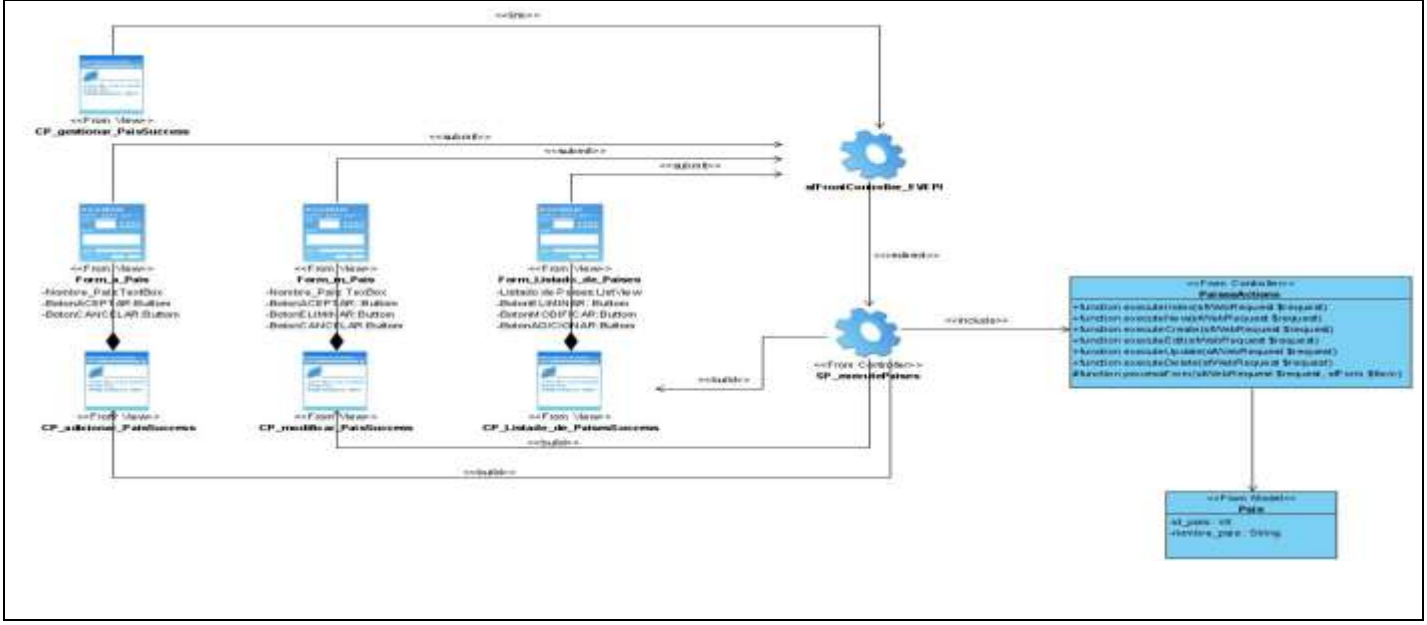


Fig. 3.24 Diagrama de Clases del Diseño: Gestionar País

DIAGRAMA DE CLASES DEL DISEÑO
< CU_ Gestionar Entidad>

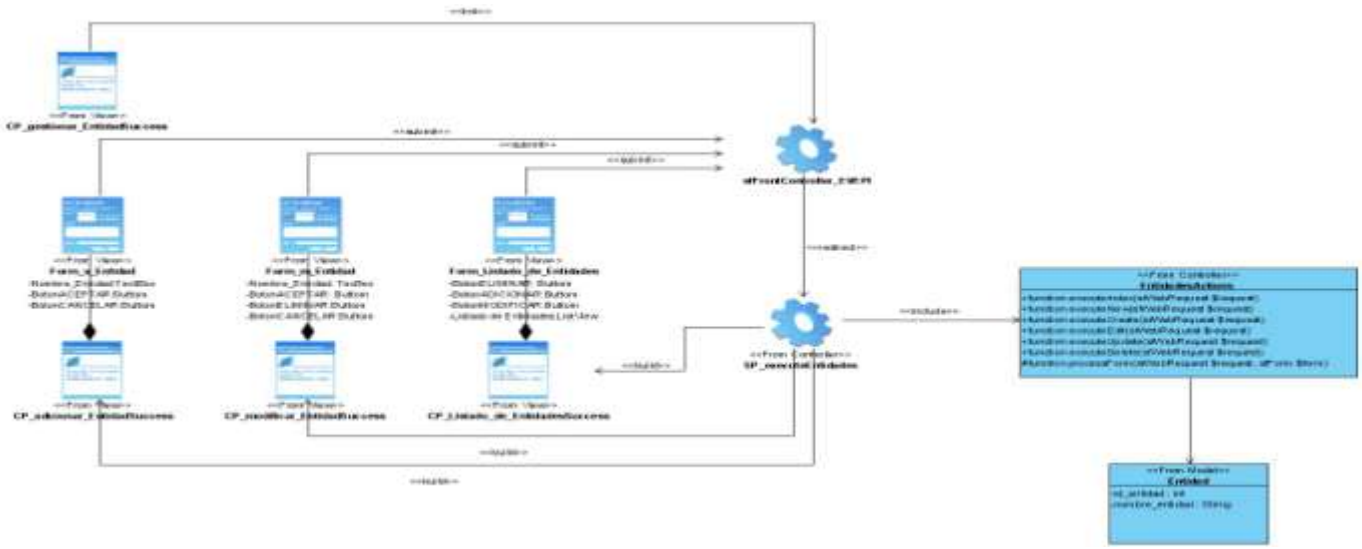


Fig. 3.25 Diagrama de Clases del Diseño: Gestionar Entidad

DIAGRAMA DE CLASES DEL DISEÑO
< CU_ Autenticar Usuario>

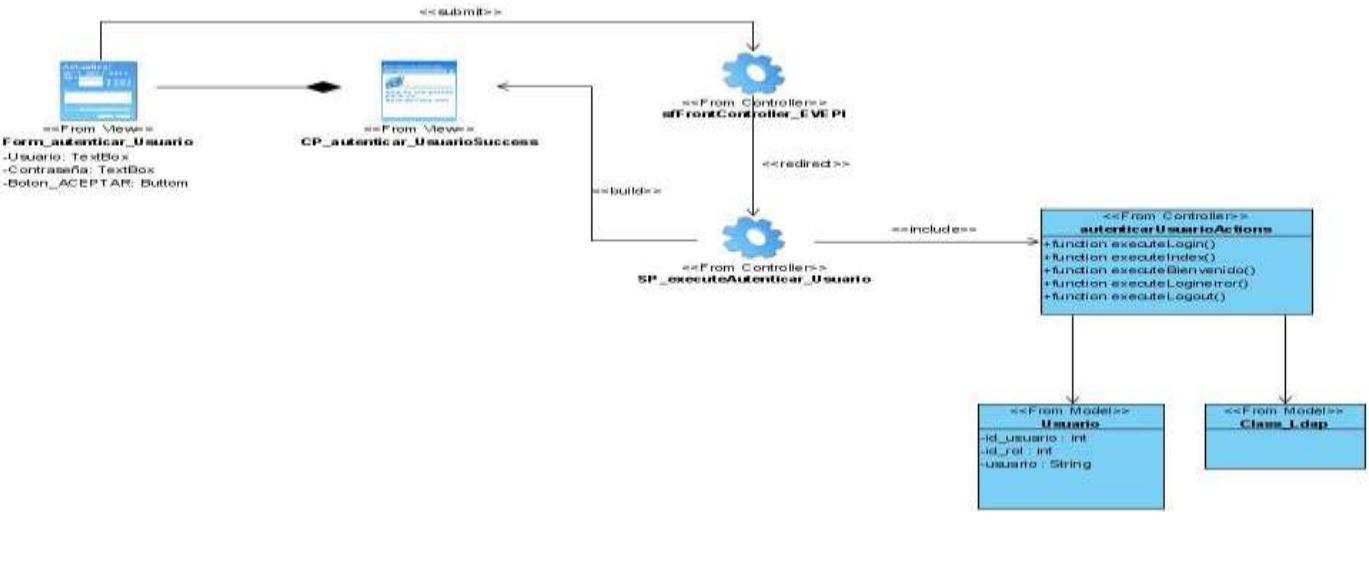


Fig. 3.26 Diagrama de Clases del Diseño: Autenticar Usuario

DIAGRAMA DE CLASES DEL DISEÑO
< CU_ Gestionar Criterio Propuesto>

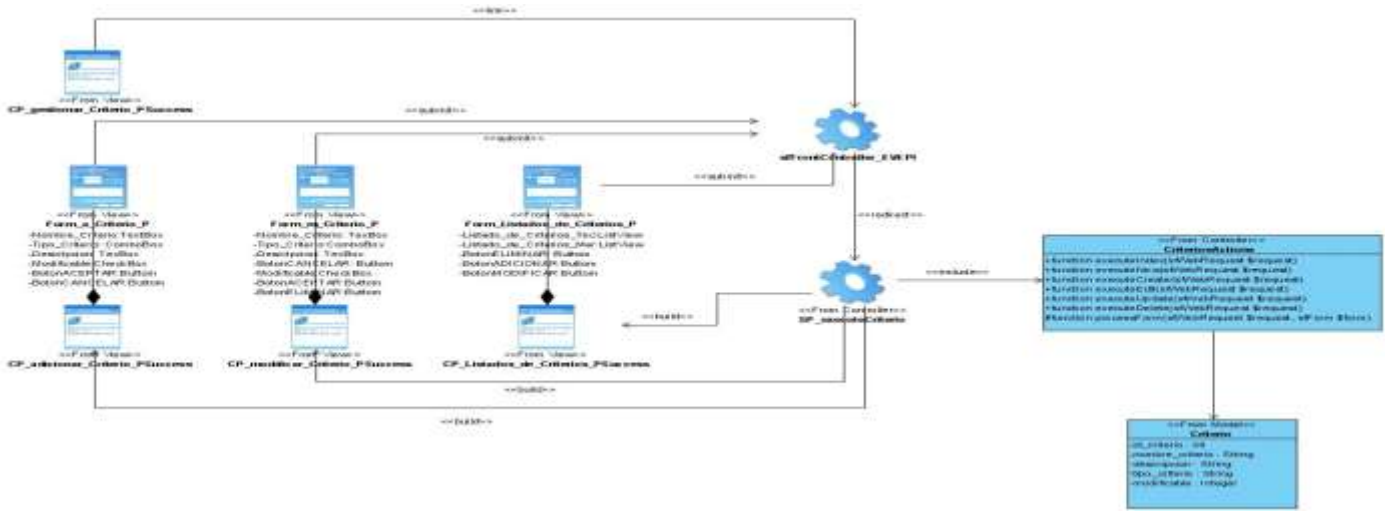


Fig. 3 27 Diagrama de Clases del Diseño: Gestionar Criterio Propuesto

DIAGRAMA DE CLASES DEL DISEÑO
< CU_ Calcular Concordancia>

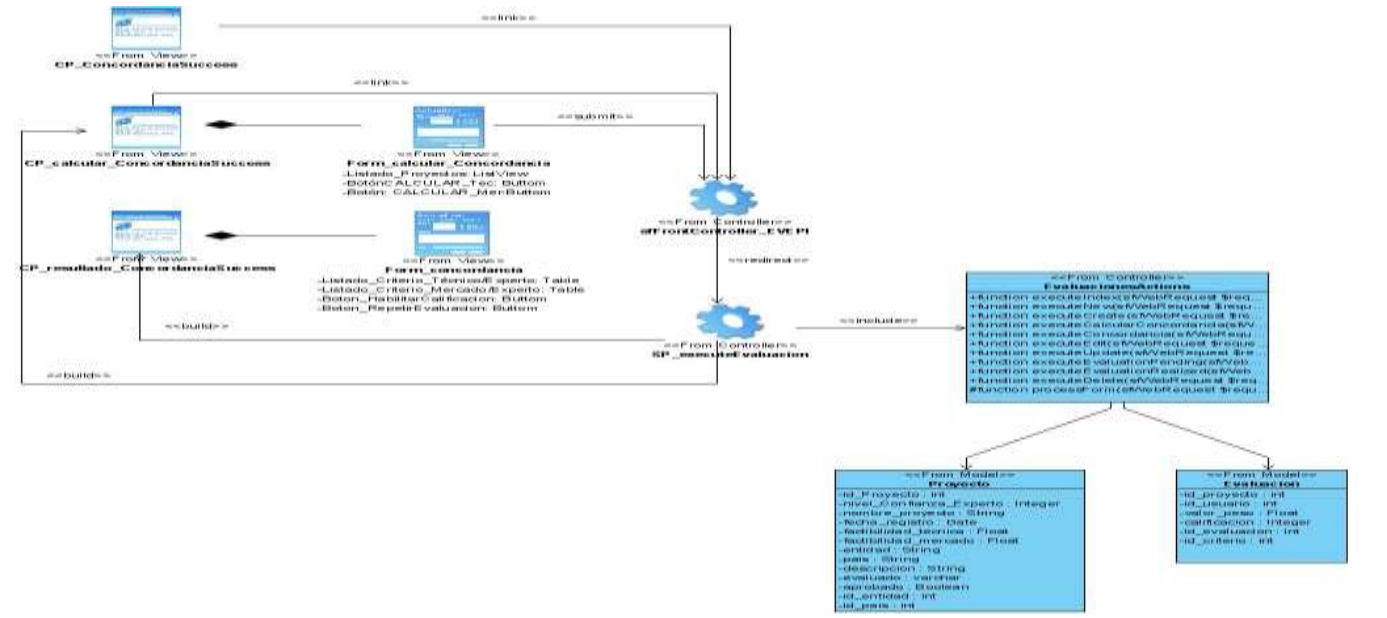


Fig. 3 28 Diagrama de Clases del Diseño: Calcular Concordancia

3.3.5 Modelo Lógico de Datos (Diagrama de Clases Persistentes)

DIAGRAMA DE CLASES PERSISTENTES

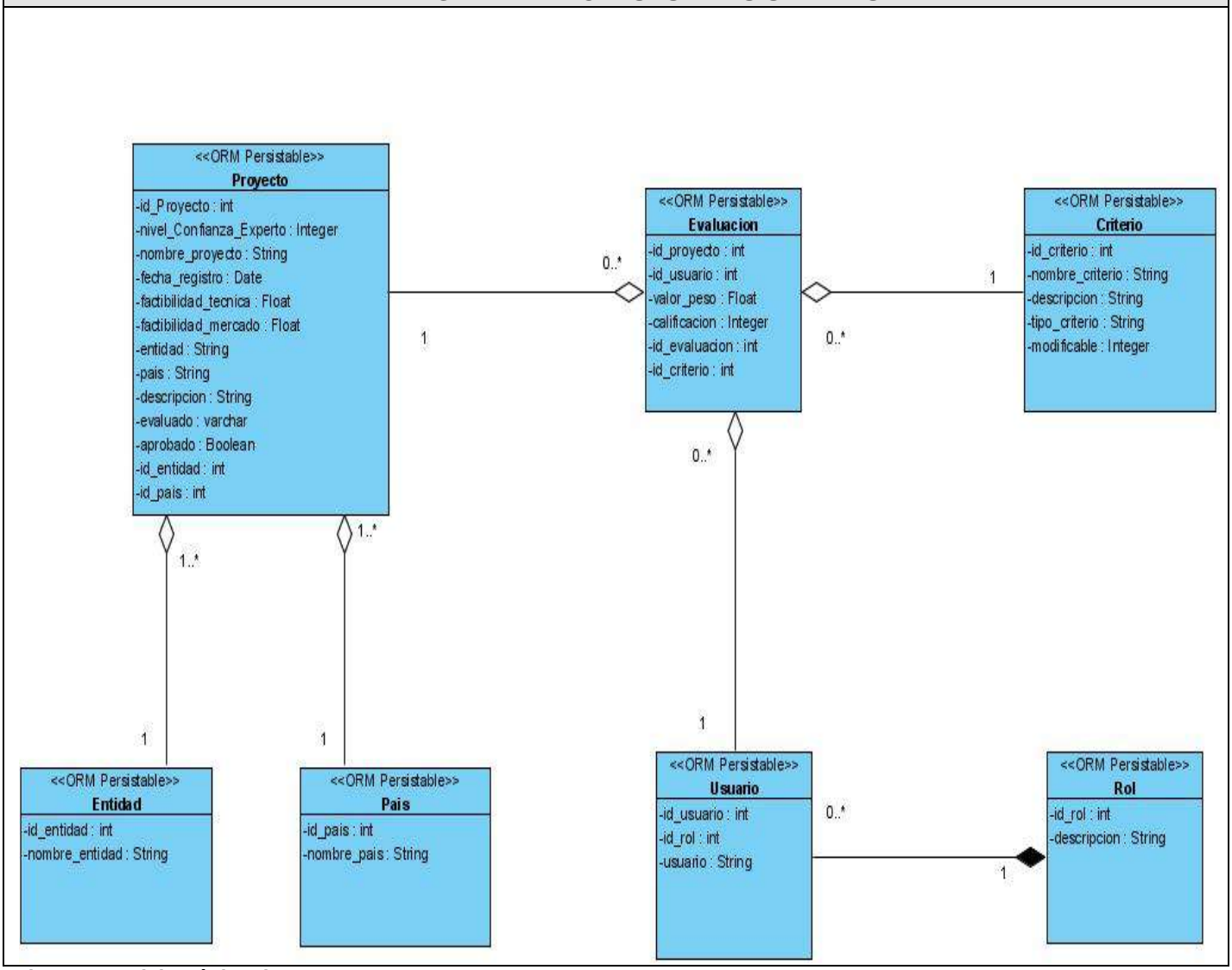


Fig. 3.25 Modelo Lógico de Datos

3.3.6 Modelo Físico de Datos

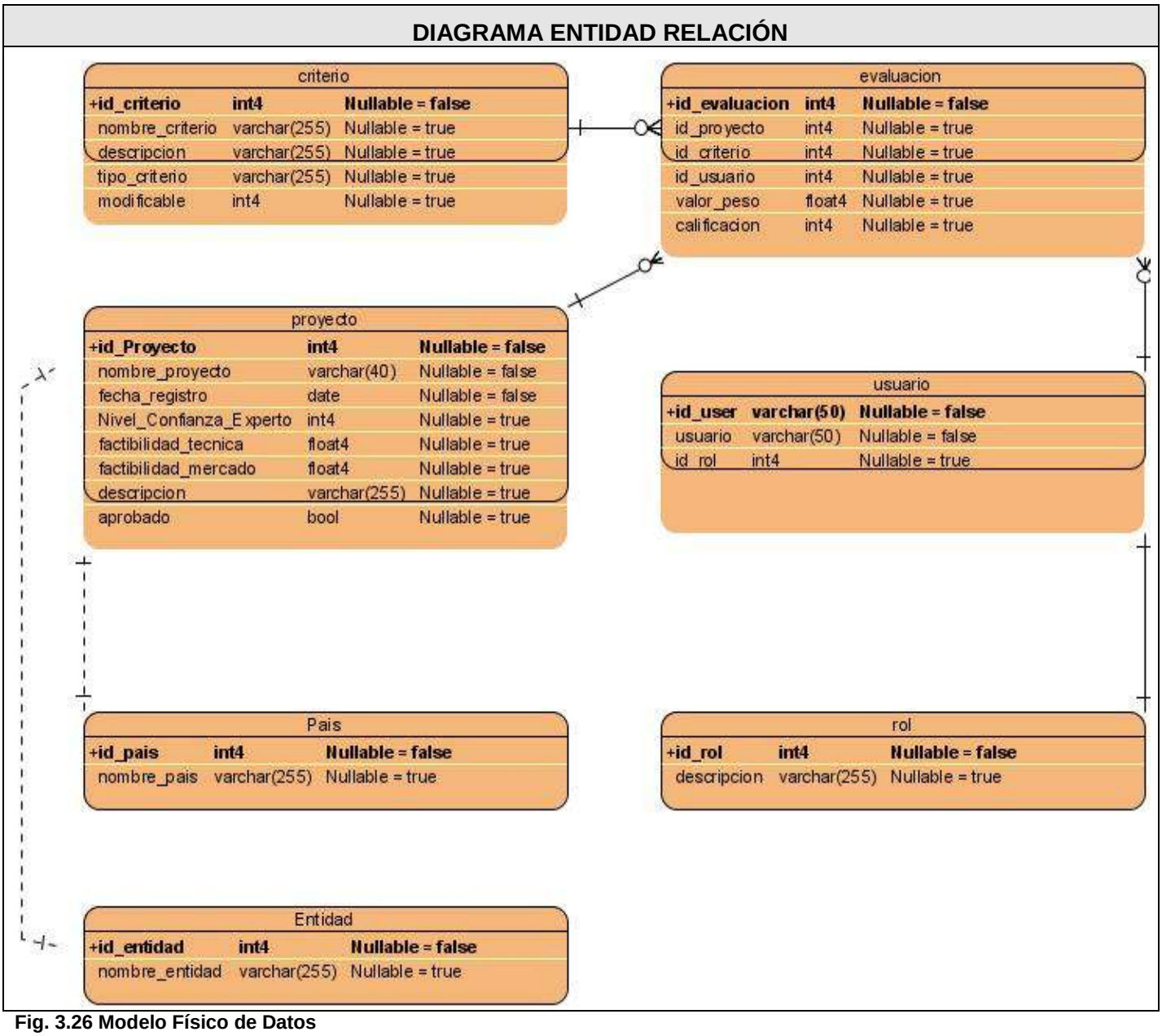


Fig. 3.26 Modelo Físico de Datos

3.4 Conclusiones

En este capítulo se mostraron los artefactos principales generados en el flujo de trabajo de Análisis y Diseño, que son: el modelo de análisis, el modelo de diseño, así como el modelo lógico y físico de datos que permiten una mejor vista para la realización de la base de datos. También se hizo mención a los patrones del diseño que se utilizaron. Con el desarrollo del Análisis se obtuvo una visión del sistema y se utilizó como base para la elaboración del resultado final más importante de este flujo de trabajo que es el modelo de diseño, ya que constituye la entrada principal para la implementación.

CAPÍTULO 4: IMPLEMENTACIÓN

4.1 Introducción.

El siguiente capítulo tiene como objetivo desarrollar los artefactos correspondientes a la implementación del sistema. Se realizarán los diagramas de despliegue y componentes que conforman lo que se conoce como: Modelo de implementación; de esta forma se describen cómo los elementos del modelo del diseño se implementan en términos de componentes y se organizan de acuerdo a los nodos específicos en el modelo de despliegue.

4.2 Diagrama de despliegue

Un diagrama de despliegue muestra las relaciones físicas entre los componentes *hardware* y *software* en el sistema final, es decir, la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes *software* (procesos y objetos que se ejecutan en ellos). Estarán formados por instancias de los componentes *software* que representan manifestaciones del código en tiempo de ejecución (los componentes que sólo sean utilizados en tiempo de compilación deben mostrarse en el diagrama de componentes).

Un diagrama de despliegue es un grafo de nodos unidos por conexiones de comunicación. Un nodo puede contener instancias de componentes *software*, objetos, procesos (caso particular de un objeto). En general un nodo será una unidad de computación de algún tipo, desde un sensor a un *mainframe*. Las instancias de componentes *software* pueden estar unidas por relaciones de dependencia, posiblemente a interfaces (ya que un componente puede tener más de una interfaz). (39)

En la Figura 4.1 se representan los aspectos más importantes a destacar que son:

- La PC Cliente constituye el puesto donde el usuario estará interactuando con la aplicación, este nodo tendrá una conexión <<HTTP>> con el Servidor Web (Apache) encargado de responder a todas las peticiones hechas por el cliente.

- El Servidor Web interactúa con un Servidor de Base de datos (PostgreSQL) mediante <<PDO>> para obtener y brindar los datos solicitados por el usuario, también interactúa con el Servidor de Directorio para la autenticación de los usuarios.

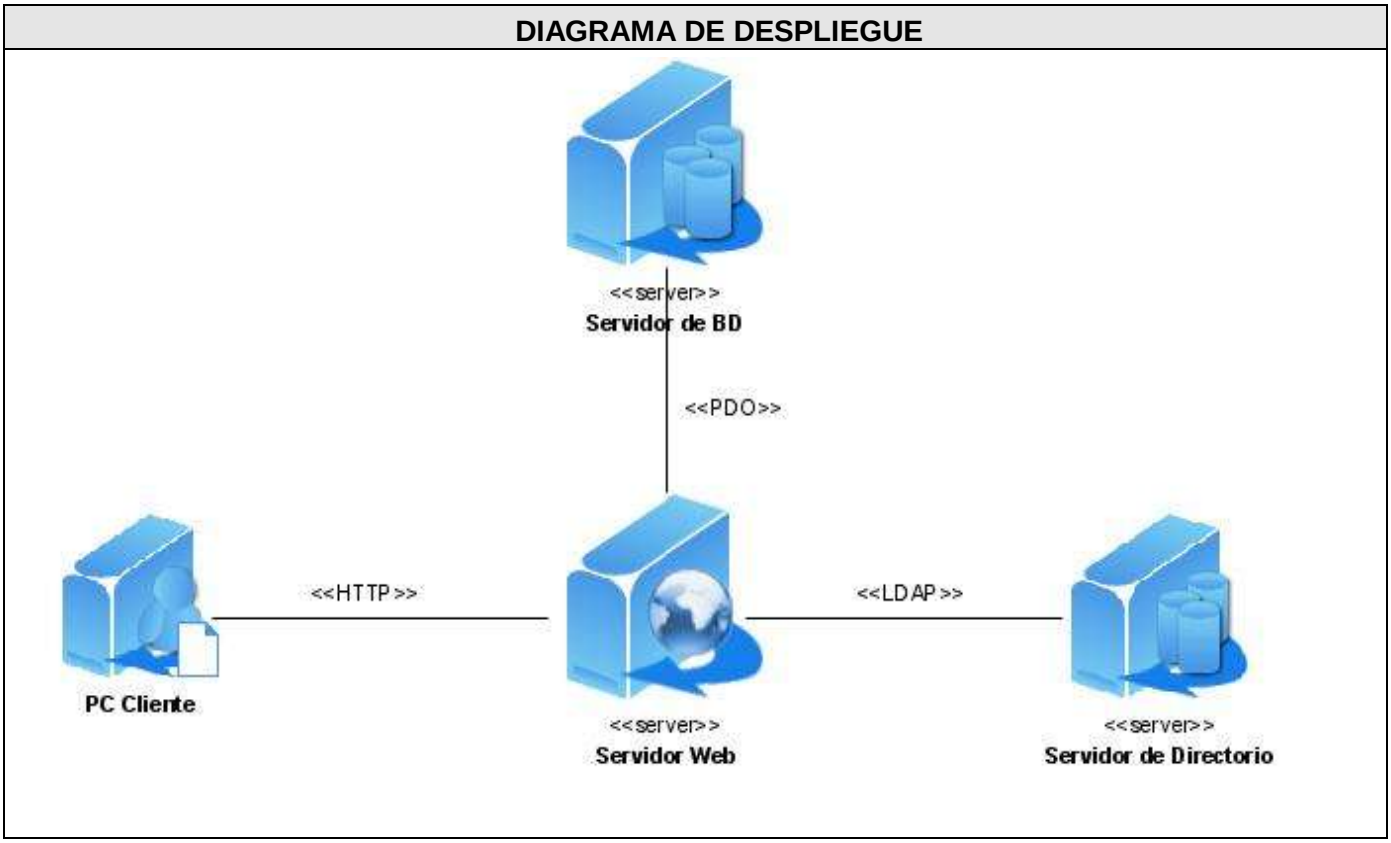


Fig. 4. 1 Diagrama de Despliegue

4.3 Diagrama de componentes

Los diagramas de componentes habitualmente contienen componentes, interfaces y relaciones entre ellos. Y como todos los diagramas, también puede contener paquetes utilizados para agrupar elementos del modelo.

Un 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 dentro de un diagrama de componentes serán componentes y paquetes. En cuanto a los componentes, sólo aparecen tipos de componentes, ya que las instancias específicas de cada tipo se encuentran en el diagrama de despliegue.

Dado que los diagramas de componentes muestran los componentes *software* que constituyen una parte reusable, sus interfaces, y sus interrelaciones, en muchos aspectos se puede considerar que un diagrama de componentes es un diagrama de clases a gran escala. (36)

La Fig. 4.2 muestra la estructura general de los Diagramas de Componentes, la misma está dividida en cuatro paquetes: Vista, Controlador, Acceso a Datos y Base de Datos.

El paquete Vista está formado por el componente Layout.php y su relación con los demás componentes (Páginas Clientes), que en este paquete se representa con nombre genérico: Plantillas.

El paquete Controlador contiene los componentes Actions asociados al caso de uso que se desee representar y al componente Controlador Frontal.

El paquete Acceso a Datos está constituido por los componentes que acceden para cada caso de uso a la(s) tabla(s) de la Base de Datos.

El paquete Base de Datos está integrado por las dos Bases de Datos de las cuales hace uso el sistema EVEPI.

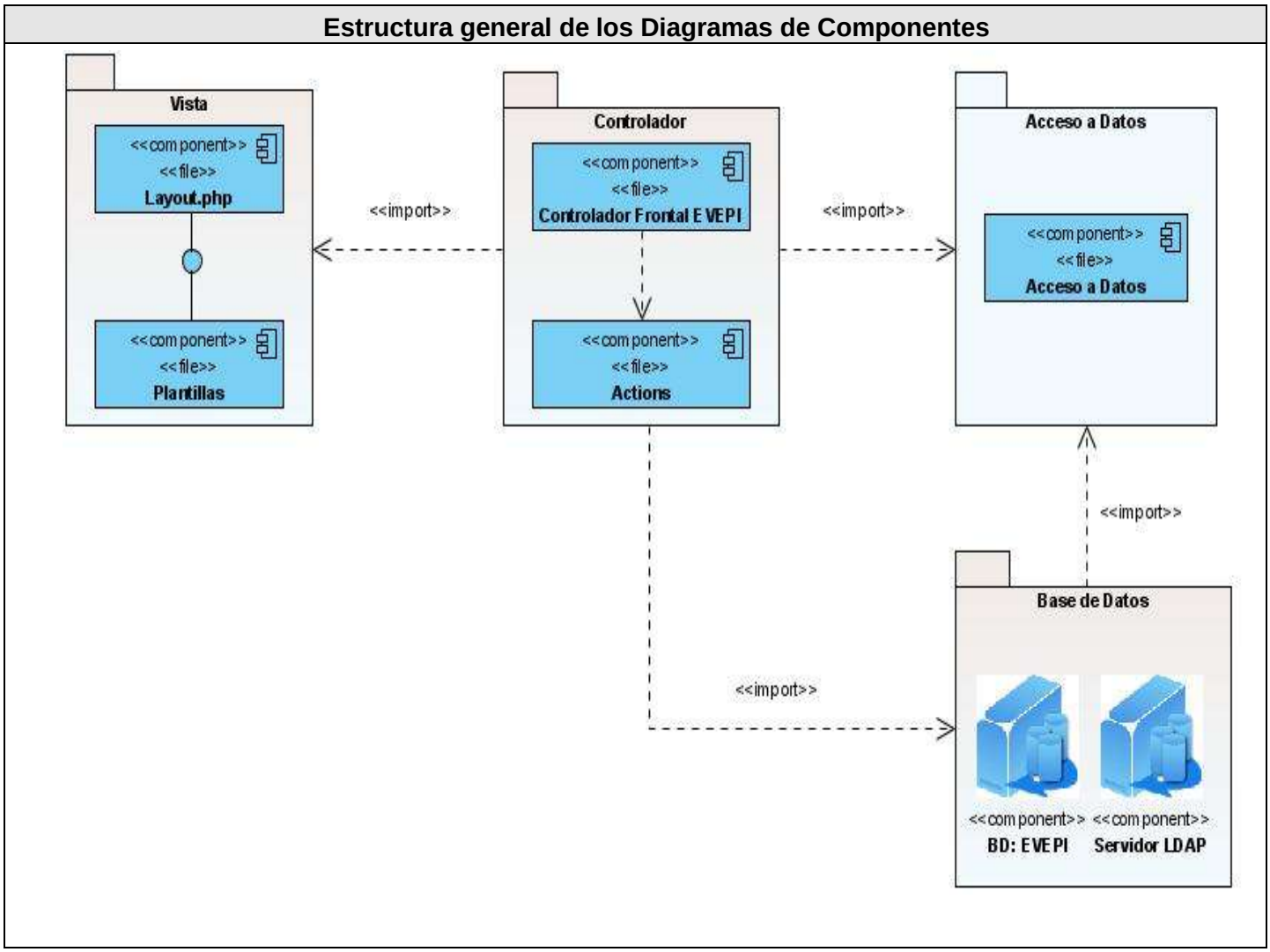


Fig. 4. 1 Estructura general de los Diagramas de Componentes

DIAGRAMA DE COMPONENTES
<CU_ Autenticar Usuario>

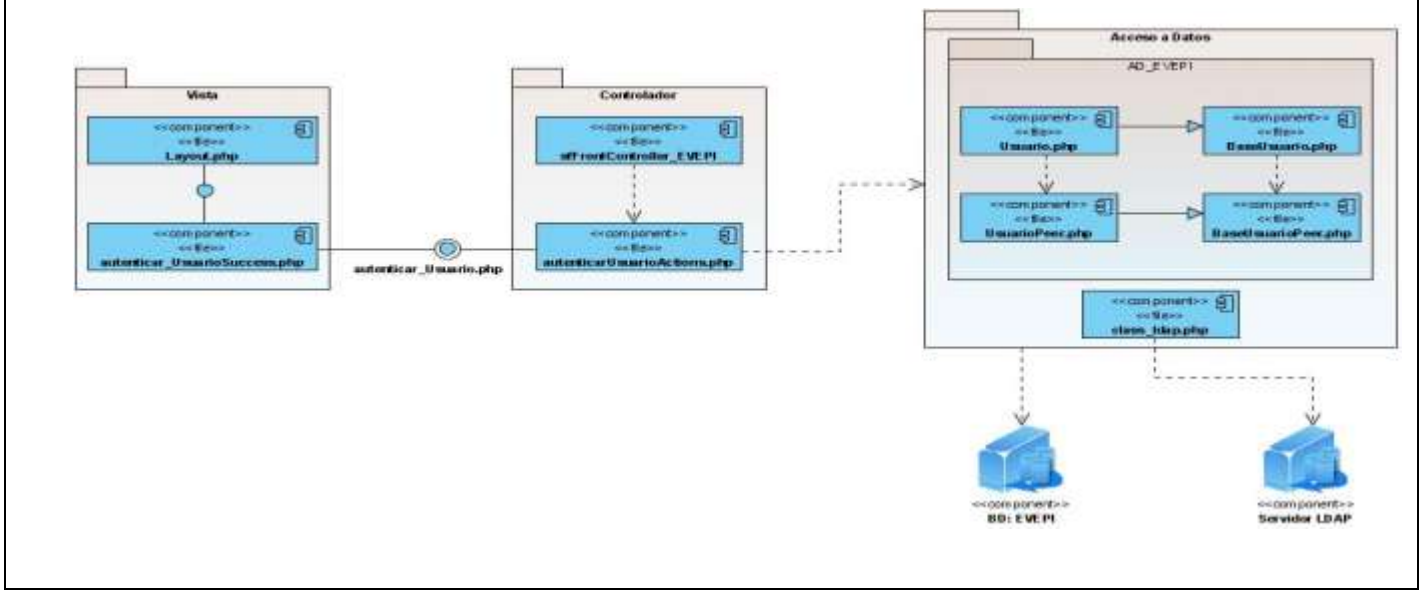


Fig. 4. 2 Diagrama de Componentes <Autenticar Usuario>

DIAGRAMA DE COMPONENTES
<CU_ Calcular Concordancia>

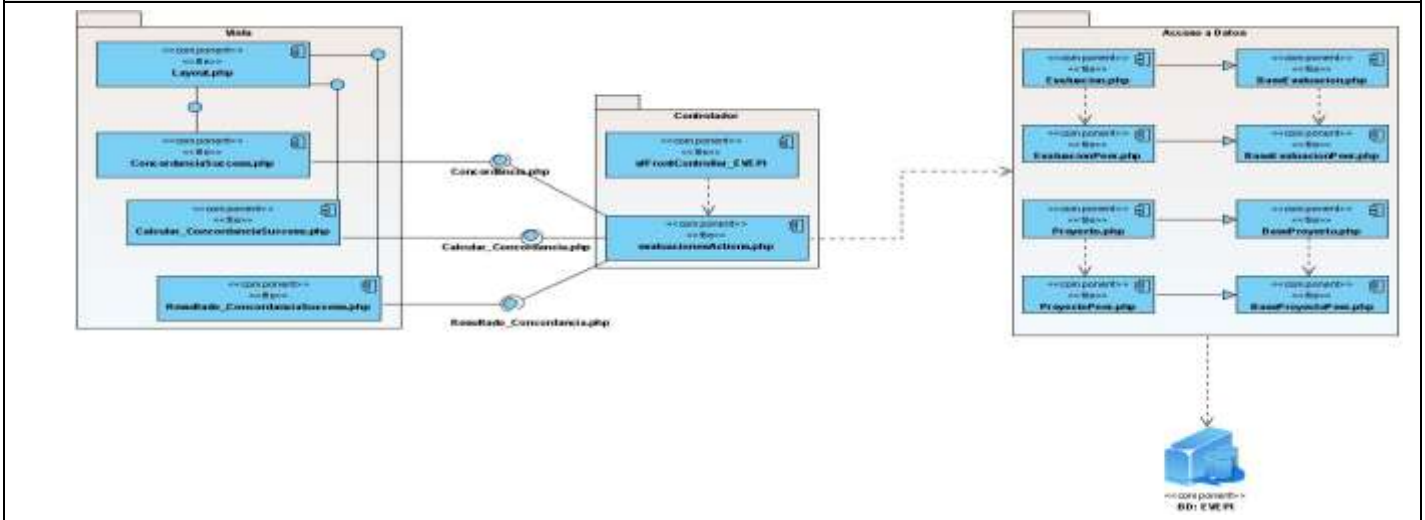


Fig. 4. 3 Diagrama de Componentes <Calcular Concordancia>

DIAGRAMA DE COMPONENTES
<CU_ Calcular Factibilidad>

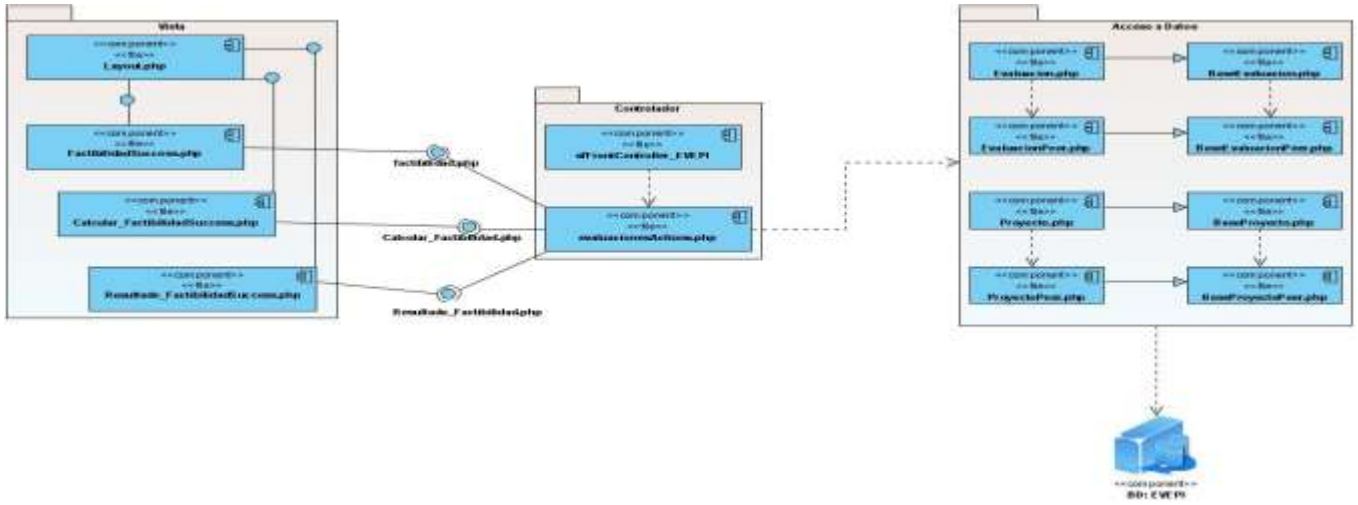


Fig. 4. 4 Diagrama de Componentes <Calcular Factibilidad>

DIAGRAMA DE COMPONENTES
<CU_ Aprobar Proyecto>

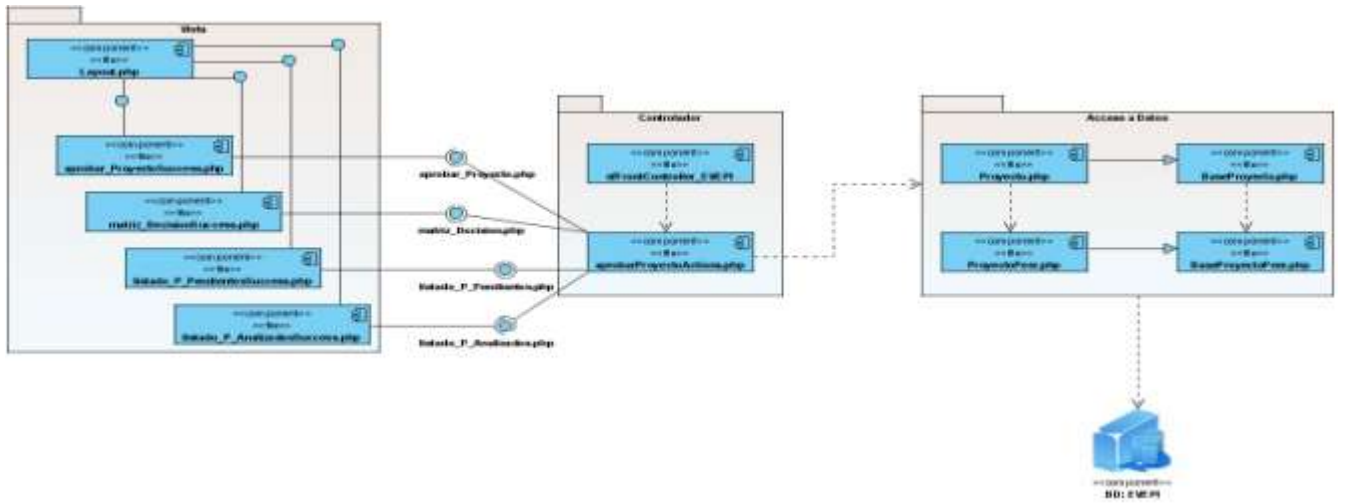


Fig. 4. 5 Diagrama de Componentes <Aprobar Proyecto>

DIAGRAMA DE COMPONENTES
<CU_ Gestionar Criterio>

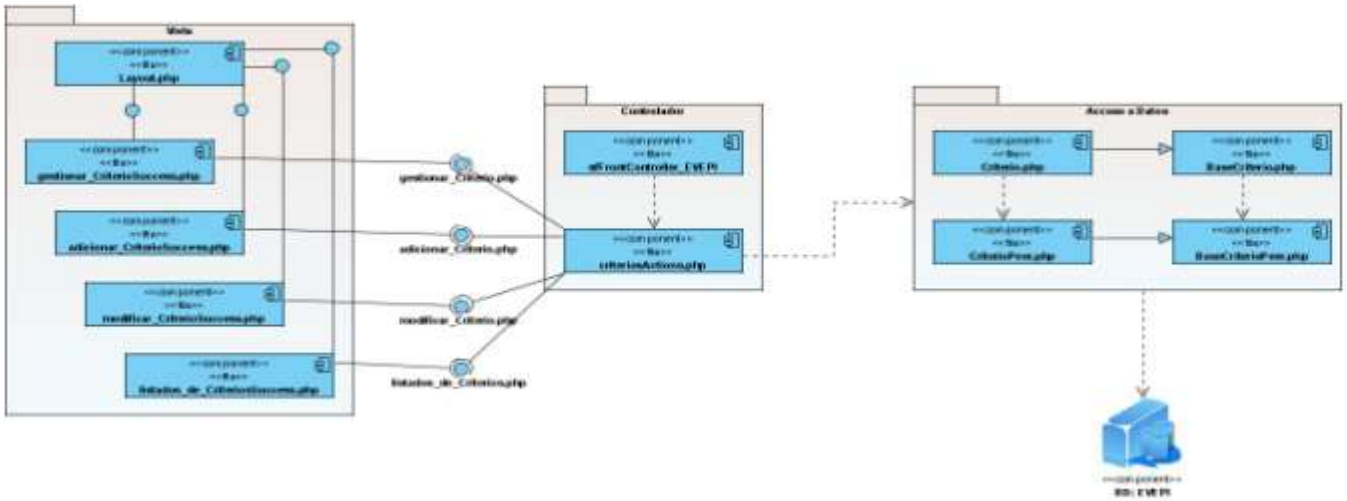


Fig. 4. 6 Diagrama de Componentes <Gestionar Criterio>

DIAGRAMA DE COMPONENTES
<CU_ Gestionar Criterio Propuesto>

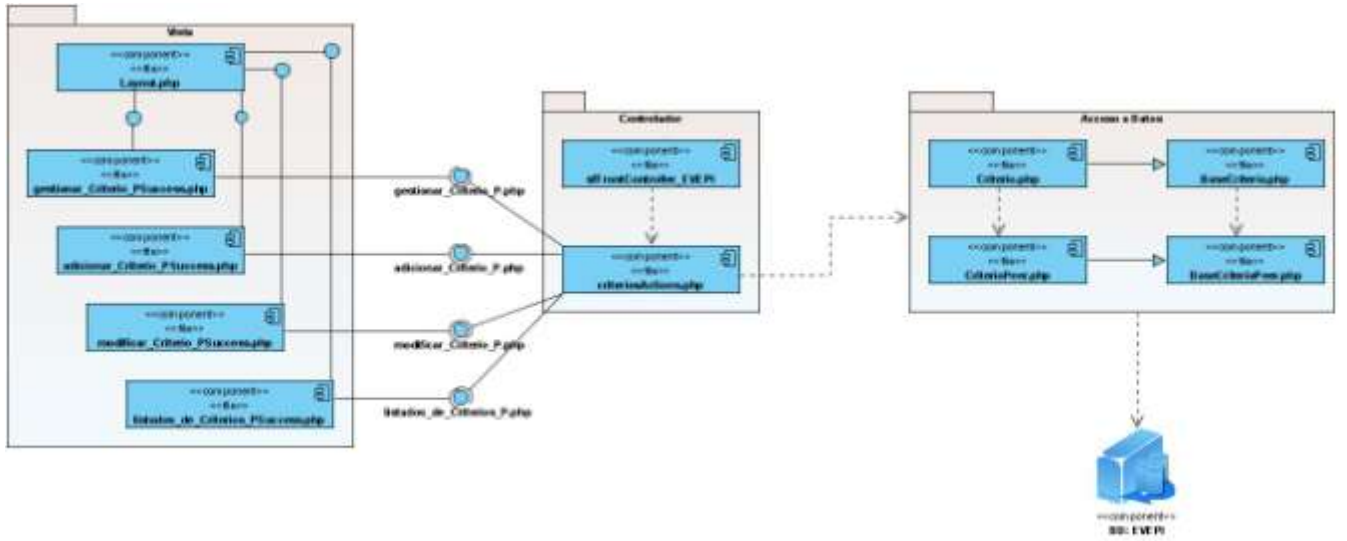


Fig. 4. 7 Diagrama de Componentes <Gestionar Criterio Propuesto>

DIAGRAMA DE COMPONENTES
<CU_ Gestionar Entidad>

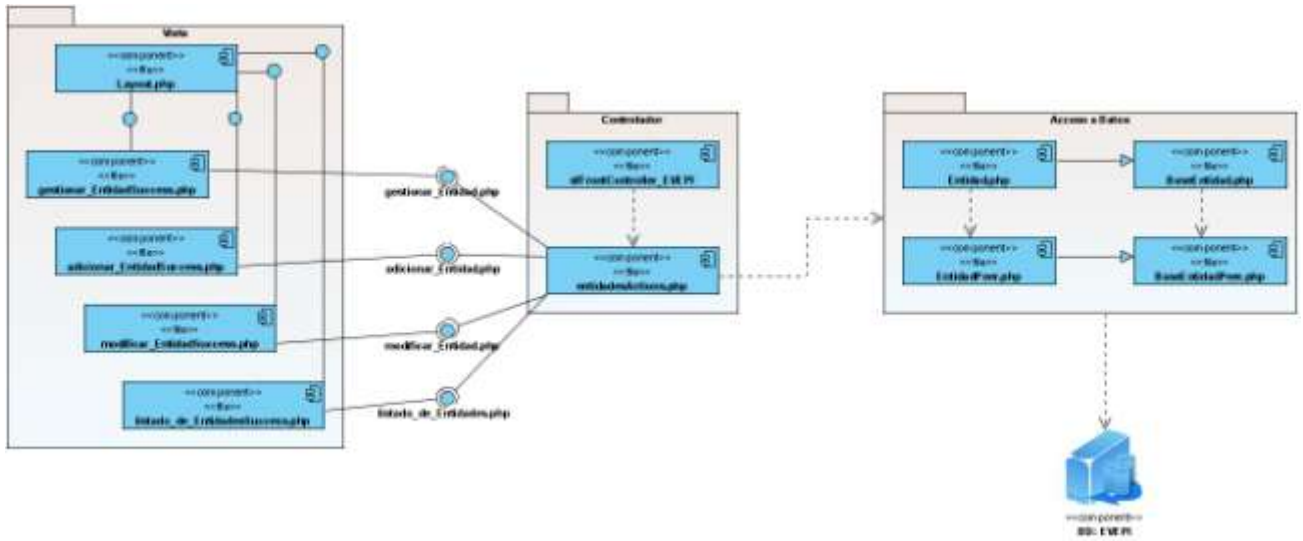


Fig. 4. 8 Diagrama de Componentes <Gestionar Entidad>

DIAGRAMA DE COMPONENTES
<CU_ Gestionar País>

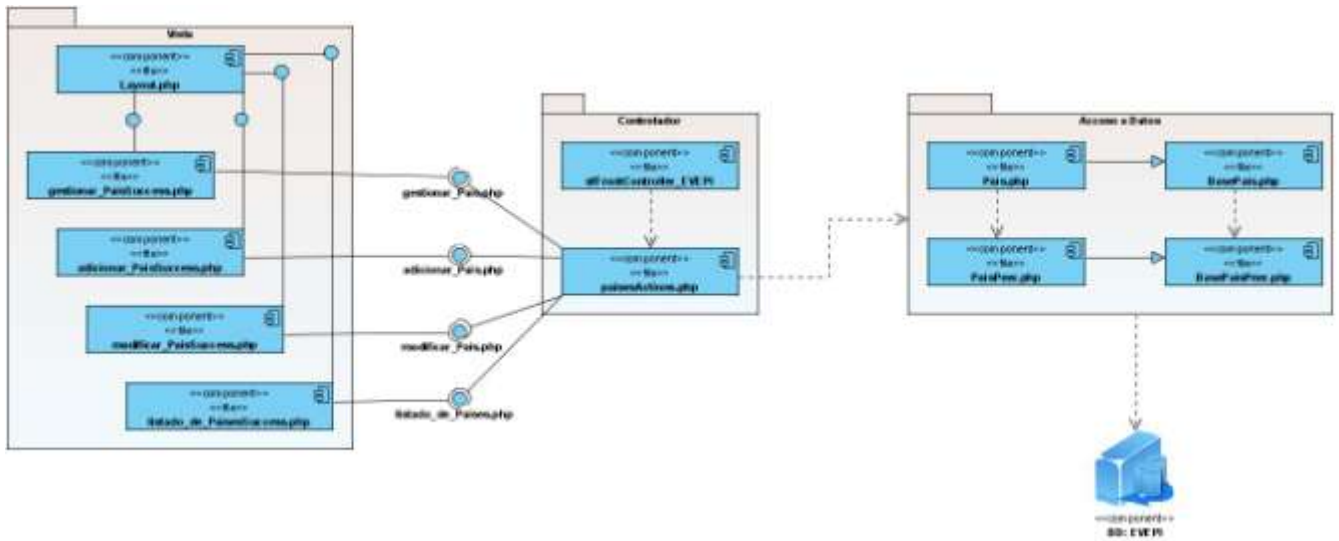


Fig. 4. 9 Diagrama de Componentes <Gestionar País>

DIAGRAMA DE COMPONENTES
<CU_ Gestionar Proyecto>

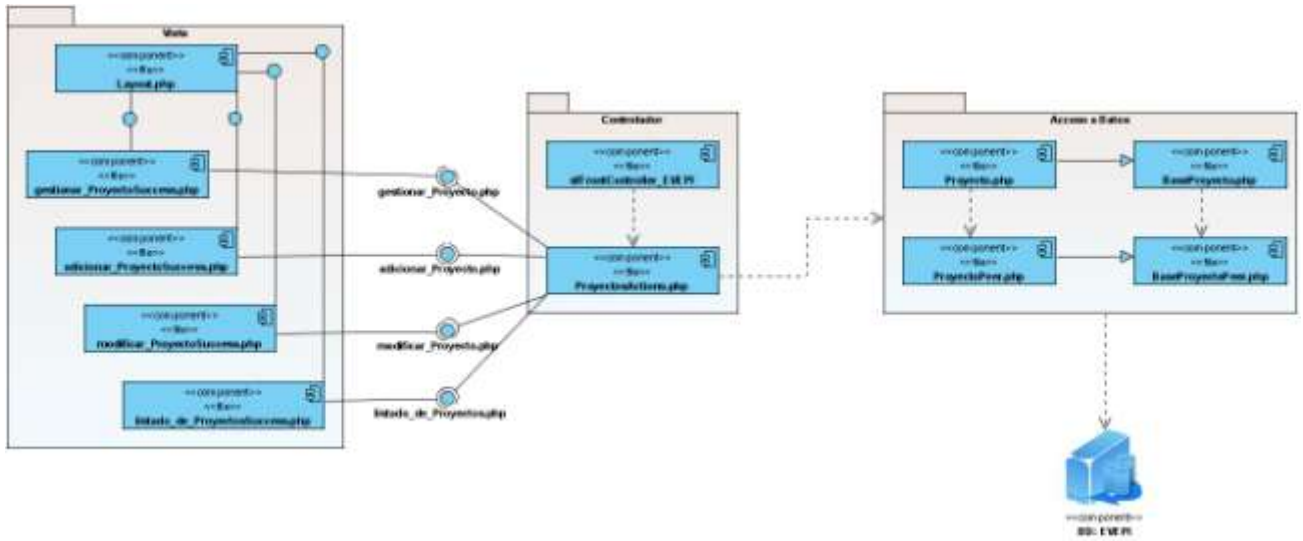


Fig. 4. 11 Diagrama de Componentes <Gestionar Proyecto>

DIAGRAMA DE COMPONENTES
<CU_ Gestionar Evaluación>

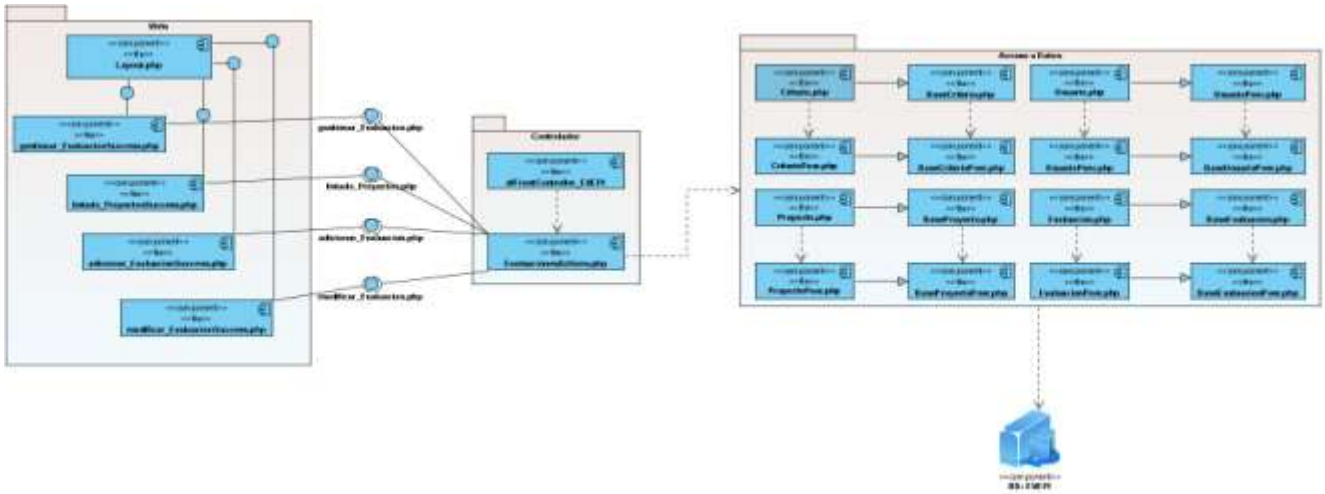


Fig. 4. 12 Diagrama de Componentes <Gestionar Evaluación>

DIAGRAMA DE COMPONENTES
<CU_ Gestionar Usuario>

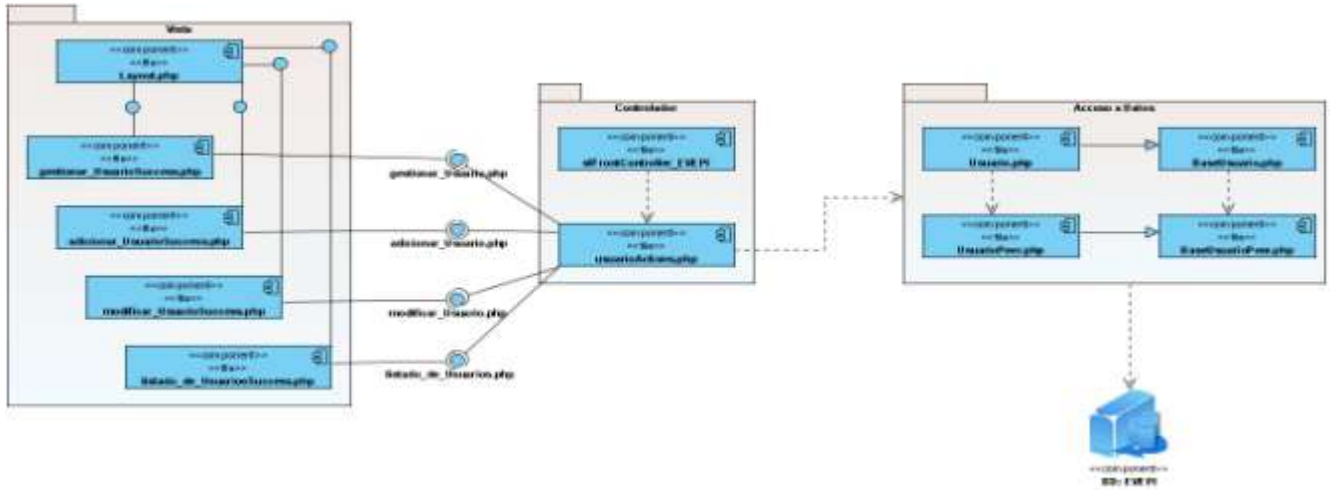


Fig. 4. 13 Diagrama de Componentes <Gestionar Usuario>

DIAGRAMA DE COMPONENTES
<CU_ Evaluar Criterio>

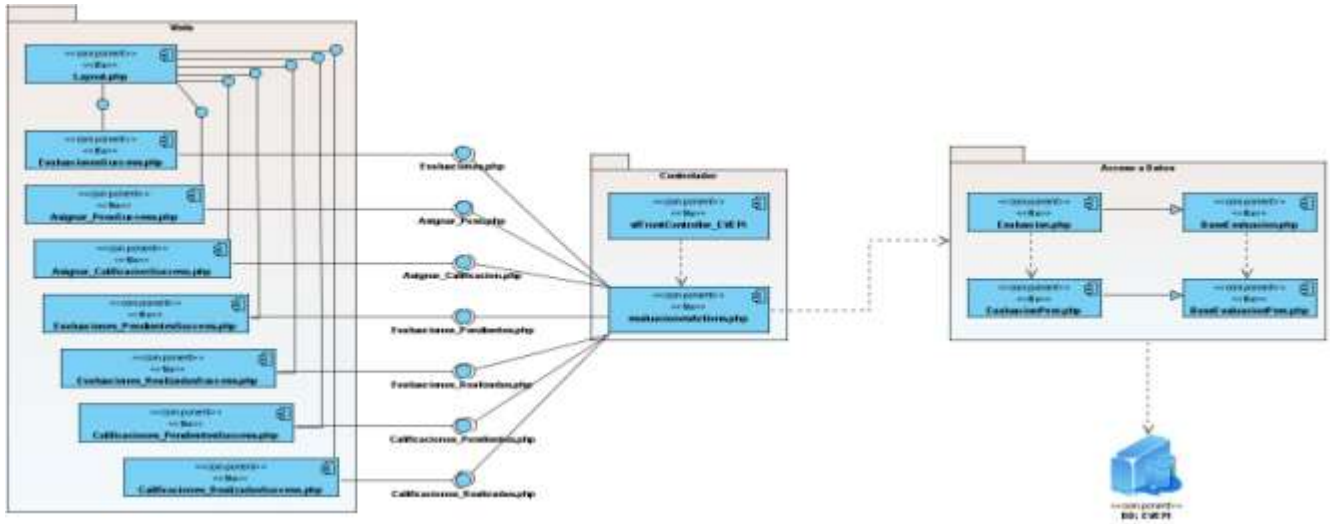


Fig. 4. 14 Diagrama de Componentes <Evaluar Criterio>

4.4 Conclusiones

En este capítulo se crearon los elementos correspondientes al Modelo de Implementación: los modelos de componentes por cada caso de uso así como el de despliegue. Al concluir el mismo, se completaron los artefactos principales que constituyen la base de la aplicación del sistema: “Entorno Virtual de Evaluación para Proyectos Informáticos (EVEPI)” dando cumplimiento de esta forma, a los pedidos y necesidades del cliente.

CAPÍTULO 5: FACTIBILIDAD DEL SISTEMA

5.1 Introducción

Ningún trabajo se obtiene sin esfuerzo, y sería arriesgado esfuerzo alguno si antes no se analiza cuánto demorará realizarlo y cuántas personas se necesitan. Se debe cuantificar: la complejidad del sistema, funcionalidad, complejidad técnica, el nivel de experiencia de las personas que integran el proyecto así como el tiempo necesario para producir una unidad de complejidad. En el presente capítulo, haciendo uso del método de estimación: Análisis de Puntos por Casos de Uso, se realizará la estimación del costo, el esfuerzo y el tiempo necesarios para obtener el software.

5.2 Método de estimación por Casos de Uso

La estimación mediante el análisis de Puntos por Caso de Uso, se trata de estimar el tiempo de desarrollo de un proyecto mediante la asignación de "pesos" a un cierto número de factores que lo afectan, para finalmente contabilizar el tiempo total estimado para el proyecto a partir de esos factores. La cantidad de ensayos hasta la fecha muestran que es aplicable satisfactoriamente en aplicaciones de negocio, como los sistemas de información. (40) A continuación, se detallan los pasos a seguir para la aplicación de éste método.

5.2.1 Cálculo de puntos de casos de uso sin ajustar

El primer paso a realizar será el cálculo de los Puntos de Casos de Uso sin ajustar, calculándose de la siguiente manera:

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 el Factor de Peso de los Actores sin ajustar se debe realizar un análisis de la cantidad de actores presentes en el sistema, así como la complejidad de cada uno de ellos.

Los criterios para el análisis se encuentran en la tabla siguiente:

Tipo de Actor	**Descripción**	**Peso**	**Cantidad * Peso**
Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación (*API*, *Application Programming Interface*).	1	0 * 1
Medio	Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto.	2	0 * 2
Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica.	3	4 * 3
∑(actores*Peso)			12

Tabla 5. 1 Factor de peso de los actores sin ajustar

El Factor de Peso de los Casos de Uso sin ajustar se calcula mediante un análisis de la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde una transacción se entiende como una secuencia de actividades atómica, es decir, se efectúa la secuencia de actividades completa, o no se efectúa ninguna de las actividades de la secuencia y está representada por uno o más pasos del flujo de eventos principal del Caso de Uso, pudiendo existir más de una transacción dentro del mismo Caso de Uso. Los criterios se muestran en la siguiente tabla:

Tipo de Caso de Uso	Descripción	Peso	Cantidad * Peso
Simple	El Caso de Uso contiene de 1 a 3 transacciones.	5	10 * 5
Medio	El Caso de Uso contiene de 4 a 7 transacciones.	10	2 * 10
Complejo	El Caso de Uso contiene más de 8 transacciones.	15	0 * 15
∑ **CU*Peso**			70

Tabla 5. 2 Factor de peso de los Casos de Uso sin ajustar

Una vez calculado el Factor de Peso de los Actores sin ajustar (**UAW**) y Factor de Peso de los Casos de Uso sin ajustar (**UUCW**) se procede al cálculo de los Puntos de Casos de Uso sin ajustar (**UUCP**)

UUCP= UAW + UUCW = 12+70 = **82**

5.2.2 Cálculo de puntos de casos de uso ajustados

Una vez que se tienen los Puntos de Casos de Uso sin ajustar, se debe ajustar éste valor mediante la siguiente ecuación:

UCP = UUCP * TCF * EF

Donde:

- **UCP**: Puntos de Casos de Uso ajustados.
- **UUCP**: Puntos de Casos de Uso sin ajustar.
- **TCF**: Factor de complejidad técnica.
- **EF**: Factor de ambiente.

Factor de complejidad técnica (**TCF**): Este coeficiente se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de los factores se cuantifica con

un valor de 0 a 5, donde 0 significa un aporte irrelevante y 5 un aporte muy importante. En la siguiente tabla se muestra el significado y el peso de cada uno de éstos factores:

Factor	Descripción	Peso	Peso * Valor
T1	Sistema distribuido.	2	2 * 1
T2	Objetivos de performance o tiempo de respuesta.	1	1 * 5
T3	Eficiencia del usuario final.	1	1 * 5
T4	Procesamiento interno complejo.	1	1 * 2
T5	El código debe ser reutilizable.	1	1 * 5
T6	Facilidad de instalación.	0.5	0.5 * 5
T7	Facilidad de uso.	0.5	0.5 * 3
T8	Portabilidad.	2	2 * 5
T9	Facilidad de cambio.	1	1 * 4
T10	Concurrencia.	1	1 * 5
T11	Incluye objetivos especiales de seguridad.	1	1 * 2
T12	Provee acceso directo a terceras partes.	1	1 * 5
T13	Se requieren facilidades especiales de entrenamiento a los usuarios.	1	1 * 1
Total			50

Tabla 5. 3 Factor de complejidad técnica

La ecuación para su cálculo es:

TCF = 0.6 + 0.01 * Σ (Peso i * Valor Asignado i)

Entonces, **TCF** = 0.6 + 0.01 * 50 = **1.1**

Factor de ambiente: Contempla las habilidades y el entrenamiento del grupo de desarrollo por su importancia en las estimaciones de tiempo. Al igual que el factor de complejidad técnica se cuantifican con valores de 0 a 5. La ecuación para su cálculo es:

EF = 1.4 - 0.03 * Σ (Peso i * Valor Asignado i).

Factor	Descripción	Peso	Peso * Valor
E1	Familiaridad con el modelo de proyecto utilizado.	1.5	1.5 * 3
E2	Experiencia en la aplicación.	0.5	0.5 * 1
E3	Experiencia en orientación a objetos.	1	1 * 4
E4	Capacidad del analista líder.	0.5	0.5 * 5
E5	Motivación.	1	1 * 5
E6	Estabilidad de los requerimientos.	2	2 * 5
E7	Personal part - time.	-1	-1 * 3
E8	Dificultad del lenguaje de programación.	-1	-1 * 3
Total			20.5

Tabla 5. 4 Factor de ambiente

Entonces, **EF** = 1.4 - 0.03 * 20.5 = **0.785**

Finalmente, **UCP (Puntos de Caso de Uso ajustados) =** UUCP * TCF * EF = 82 * 1.1 * 0.785 = **70.807**

5.2.3 Estimación del esfuerzo

El esfuerzo en horas - hombre viene dado por:

E = UCP * CF

Donde:

- **E**: Esfuerzo estimado en horas-hombre.
- **UCP**: Puntos de Casos de Uso ajustados.
- **CF**: Factor de conversión.

CF = 20 horas-hombre (si Total EF ≤ 2)

CF = 28 horas-hombre (si Total EF = 3 ó Total EF = 4)

CF = abandonar o cambiar proyecto (si Total EF ≥ 5)

Total EF = Cant EF < 3 (entre E1 –E6) + Cant EF > 3 (entre E7, E8)

Como **Total EF** = 1 + 0

Total EF = 1

CF = 20 horas-hombre (porque Total EF ≤ 2)

Luego **E** = 70.807 * 20 horas-hombre

E = 1416.14 horas-hombre

5.2.4 Distribución del esfuerzo entre las diferentes actividades

Para llevar a cabo una estimación más completa de la duración total del módulo, se agrega el esfuerzo de las demás actividades. Para ello se plantea la distribución del esfuerzo entre las diferentes actividades:

Actividad	Porcentaje	Horas-Hombre
Análisis	20%	708.07
Diseño	20%	708.07
Implementación	40%	1416.14
Pruebas	10%	354.035
Otras actividades	10%	354.035
Total	100%	3540.35

Tabla 5. 5 Distribución del esfuerzo

El Esfuerzo Total sería 3540.35 horas-hombre, si estimamos teniendo en cuenta que un mes tiene 260 horas laborables, pues se trabajan 10 horas diarias 26 días al mes aproximadamente, entonces el Esfuerzo Total en mes-hombre sería 13.61 mes-hombre.

5.2.5 Cálculo del tiempo de desarrollo de todo el proyecto

A partir de los datos conocidos anteriormente se podría asegurar que dos personas pueden realizar el trabajo en un tiempo aproximado de 7 meses, por lo que el Entorno Virtual de Evaluación para Proyectos Informáticos (EVEPI) debe ser realizado en un período de 7 meses.

5.3 Beneficios tangibles e intangibles

Los beneficios del Sistema EVEPI en la Universidad de las Ciencias Informáticas están relacionados directamente con ella misma, brindando un soporte informativo y de gestión de la factibilidad de los Proyectos de software que se contratarán. A través del mismo se podrá determinar si contratar un determinado proyecto informático resultará rentable o no. Esta propuesta de aplicación puede hacerse extensiva a otros centros y empresas dedicadas a la creación de software.

5.4 Análisis de costos y beneficios

Para la implementación del sistema se emplearon en su mayoría tecnologías totalmente libres, por lo que el pago de licencias de software afectará de forma mínima su costo. Los desarrolladores del sistema son estudiantes de la facultad dos de la Universidad de las Ciencias Informáticas por lo que no hay gastos en salario de profesionales. La aplicación va dirigida a la propia universidad, y su mayor beneficio se refleja en el mejoramiento de las estimaciones de factibilidades de proyectos de software, llegando a la conclusión con ello que es totalmente factible el desarrollo del sistema.

5.5 Conclusiones

Con el estudio de la factibilidad, la estimación, el cálculo del tamaño, el esfuerzo y el costo para el desarrollo del software conjuntamente con los beneficios tangibles e intangibles se evidencia la viabilidad del sistema así como sus beneficios en aras de lograr un elevado desarrollo productivo, requisito este fundamental para el avance económico de nuestro país.

CONCLUSIONES

Al concluir el presente trabajo con el desarrollo del sistema: Entorno Virtual de Evaluación para Proyectos Informáticos (EVEPI), la Universidad de las Ciencias Informáticas cuenta con un procedimiento informatizado para la evaluación de las premisas técnicas y de mercado que se deben tener en cuenta para aceptar o rechazar un proyecto de software, quedando resuelto de esta forma el problema antes existente sobre la insuficiente agilidad y veracidad de la información que generaba el procedimiento que se hacía en la misma para aceptar los proyectos y garantizar su contratación con alta probabilidad de éxito. Nada de esto hubiera sido posible sin:

- El estudio del proceso de contratación de proyectos de software.
- El estudio de las premisas técnicas y de mercado.
- La elaboración de forma general del marco teórico de la investigación.
- El estudio detallado de los procesos del negocio vinculados a dicho proyecto.
- La compresión de los casos de uso del sistema los cuales permitieron definir las clases de análisis y diseño; así como, la estructuración de sus diagramas respectivamente, elaborándose además el modelo de datos que garantiza que el sistema propuesto permita centrar, almacenar y recuperar la información que se genere en el desarrollo de las actividades de evaluación proyectos de software.

Por todo lo anterior expuesto se concluye que los objetivos propuestos para el presente trabajo han sido cumplidos satisfactoriamente.

RECOMENDACIONES

Con el objetivo de brindar una aplicación que incluya todos los procesos que se relacionan con la evaluación de premisas para la contratación de proyectos de software, se recomienda:

- Adicionarle al sistema EVEPI la implementación del procedimiento de factibilidad económica para así obtener en un mismo software un resultado completo de la gestión de factibilidad de proyectos informáticos.

- Extender el uso del sistema EVEPI a todos los centros productores de software del país con el objetivo de mejorar el proceso de contratacion de software a nivel nacional.

- Investigar de forma continua acerca de nuevas tecnologías informáticas, para garantizar mejoras en futuras versiones del sistema.

- Capacitar al personal para el trabajo con la aplicación.

- Elaborar el Manual de Usuario con el objetivo de bridarle a los usuarios un documento técnico que les ofrezca asistencia en cuanto al funcionamiento y características del software.

BIBLIOGRAFÍA

1. **Hernández, Rolando Alfredo.** *Curso Básico de Gestión de Proyectos.* Ciudad de La Habana : s.n., 2007.

2. **Cáceres, Luis Jesus Galindo.** *Sociedad, cultura y comunicación. Un punto de partida.* s.l. : Social Science, 1998. pág. 277. 8FZ7-ADO-8435.

3. **Meléndrez, Edelsys Hernández.** *Cómo escribir una tesis.* Ciudad de La Habana : s.n., 2006.

4. **Nuñes, Paulo.** knoow.net. [En línea] 28 de 04 de 2008. [Citado el: 02 de 02 de 2009.] http://www.knoow.net/es/cieeconcom/gestion/gestiondeproyectos.htm#plus.

5. **Coacyle.** Coacyle.com. [En línea] [Citado el: 02 de 02 de 2009.] http://www.coacyle.com/UserFiles/File/NORMAS%20PRESENTACION%20TRABAJOS%20%20%20FICHAS/Normas%20presentacion+Fichas%20Aprobadas%20JG(1).pdf.

6. **Javeriana.** Pontificia Universidad. [En línea] [Citado el: 02 de 02 de 2009.] http://med.javeriana.edu.co/pediatria/clases/salud/Capitulo%206.pdf.

7. **Thompson, J, Mónica.** PromonegocioS.net. [En línea] 09 de 2006. [Citado el: 02 de 04 de 2009.] http://www.promonegocios.net/proyecto/evaluacion-proyectos.html.

9. **Eribe, Roberto.** Estudio de Factibilidad. 2009.

10. **Ascisc.** Aldereteysocios. [En línea] [Citado el: 02 de 02 de 2009.] http://www.aldereteysocios.com/estfact.html.

11. **Iglesias, Arabel Moráguez.** GestioPolis. [En línea] 05 de 2006. [Citado el: 06 de 04 de 2009.] http://www.gestiopolis.com/canales6/eco/metodo-delphi-estadistica-de-investigacion-cientifica.htm.

12. **Fonseca, Raquel Zamora.** *Programa de formación de habilidades para la gestión de contenido en los profesores de la Universidad de Cienfuegos.* Cienfuegos : s.n., 2007.

13. **Humana, Esencia.** Esencia Humana. [En línea] 2008. [Citado el: 03 de 02 de 2009.] http://www.esenciahumana.com.mx/Servicios/AplicacionesWeb/VentajasBeneficiosAplicaciones.html.

14. **Roa, Oscar Baron.** onglasses.net. [En línea] 20 de 05 de 2008. [Citado el: 03 de 02 de 2009.] http://74.125.47.132/search?q=cache:4zmPgr7UCE0J:www.onglasses.net/default.aspx%3Farticuloid%3D329+Metodologia+de+desarrollo+de+software&hl=es&gl=cu&strip=1.

15. **José H. Canós, Patricio Letelier y Carmen Penadés.** willydev.net. [En línea] [Citado el: 03 de 02 de 2009.] http://www.willydev.net/descargas/prev/TodoAgil.pdf.

16. **Ivar Jacobson, Grady Booch, James Rumbaugh.** *El Proceso Unificado de Desarrollo de Software.* Madrid : Adison Wesley, 2000. 84-7829-036-2.

17. **James Rumbaugh, Ivar Jacobson, Grady booch.** *El Lenguaje Unificado de modelado.* s.l. : Addison-Wesley, 1999.

18. **Informática, Sub-Jefatura de.** *Herramientas case.* 1999.

19. **Taringa.net.** Taringa.net. [En línea] [Citado el: 21 de 03 de 2009.] http://www.taringa.net/posts/downloads/883942/Rational-Rose-Enterprise-Edition-2002.html.

20. **Rational.com.** Rational.com. [En línea] [Citado el: 21 de 03 de 2009.] http://www.rational.com.ar/herramientas/roseenterprise.html.

21. **Paradigm.com, Visual.** Visual Paradigm.com. [En línea] [Citado el: 21 de 03 de 2009.] http://www.visual-paradigm.com/product/vpuml/communityedition.jsp.

22. **Collector, Garbage.** Garbage Collector. [En línea] 01 de 11 de 2004. [Citado el: 22 de 02 de 2009.] http://www.error500.net/garbagecollector/archives/categorias/bases_de_datos/sistema_gestor_de_base_de_datos_sgbd.php.

23. **Pecos, Daniel.** netpecos.org. [En línea] [Citado el: 22 de 02 de 2009.] http://www.netpecos.org/docs/mysql_postgres/index.html.

24. **mailxmail.com.** mailxmail.com. [En línea] 08 de 03 de 2005. [Citado el: 23 de 02 de 2009.] http://www.mailxmail.com/curso/informatica/programacionestructurada/capitulo4.htm.

25. **Hinostroza, Raul Rodas.** linuxcentro.net. [En línea] 22 de 02 de 2007. [Citado el: 23 de 02 de 2009.] http://www.linuxcentro.net/linux/staticpages/index.php?page=CaracteristicasPHP.

26. **cibernetia.com.** cibernetia.com. [En línea] [Citado el: 24 de 02 de 2009.] http://www.cibernetia.com/manuales/instalacion_servidor_web/1_conceptos_basicos.php.

27. **Ciberaula.** Ciberaula.com. [En línea] 2006. [Citado el: 22 de 02 de 2009.] http://linux.ciberaula.com/articulo/linux_apache_intro/.

28. **DiarioLinux.** DiarioLinux. [En línea] 10 de 10 de 2007. [Citado el: 03 de 03 de 2009.] http://diariolinux.com/2007/10/10/beta-de-zend-studio-para-eclipse/.

29. **Eguiluz, Javier.** Maestros del Web. [En línea] 06 de 09 de 2007. [Citado el: 04 de 03 de 2009.] http://www.maestrosdelweb.com/editorial/el-framework-symfony-una-introduccion-practica-i-parte/.

30. **Lago, Ramiro.** proactiva-calidad.com. [En línea] 04 de 2007. [Citado el: 05 de 03 de 2009.] http://www.proactiva-calidad.com/java/patrones/mvc.html.
31. **Alvarez, Sara.** DesarrolloWeb.com. [En línea] 30 de 08 de 2007. [Citado el: 05 de 03 de 2009.] http://www.desarrolloweb.com/articulos/arquitectura-cliente-servidor.html.
32. **Siegel, S.** *Estadística no paramétrica aplicada a las ciencias de la conducta.* México D.F : Trillas, 1974. págs. 262-273.
33. **Manero, Rafael Arcia.** *Proyecto de Tutorial Electrónico para el Programa de Estudio de Operador de Microcomputadora.* Santa Clara : s.n., 2007.
34. **González, Anaisa Hernández.** IDENTIFICACIÓN DE PROCESOS DE NEGOCIO. [En línea] 12 de 2004. [Citado el: 20 de 03 de 2009.] http://www.cujae.edu.cu/ediciones/Revistas/Industrial/Vol-XXV/3-2004/83-88%20Identificaci%C3%B3n%20de%20procesos.pdf.
35. **MeRinde.** MeRinde. [En línea] [Citado el: 22 de 03 de 2009.] http://merinde.rinde.gob.ve/index.php?option=com_content&task=view&id=128&Itemid=294.
36. **Vilas, Ana Fernández.** tvdi.det.uvigo.es. [En línea] 2001. [Citado el: 26 de 03 de 2009.] http://tvdi.det.uvigo.es/~avilas/UML/node50.html.
37. **Diana Paola Hurtado Bustamante, Diana Patricia Gutiérrez Valencia, Juan Pablo Suárez Valencia, Gabriel Asakawa, Eudo Quevedo Pantoja.** *Patrones GRASP.*
38. **Vilas, Ana Fenández.** tvdi.det.uvigo.es. [En línea] 20 de 03 de 2001. [Citado el: 28 de 03 de 2009.] http://tvdi.det.uvigo.es/~avilas/UML/node49.html.
39. **Vila, Ana Fernández.** tvdi.det.uvigo.es. [En línea] 20 de 03 de 2001. [Citado el: 02 de 04 de 2009.] http://tvdi.det.uvigo.es/~avilas/UML/node50.html.
40. **Trejo, Natalia Bibiana.** Estimación de Esfuerzo con Casos de Uso. [En línea] [Citado el: 10 de 04 de 2009.] http://170.210.92.6/osofia/$IngSof/Clases/Clase05c.pdf.
41. **Cyta.com.ar.** *Cyta.com.ar.* [En línea] [Citado el: 02 de 02 de 2009.] http://www.cyta.com.ar/biblioteca/bddoc/bdlibros/proyectoinformatico/libro/c1/c1.htm.
42. programacionextrema.org. [En línea] 04 de 2003. [Citado el: 20 de 02 de 2009.] http://www.programacionextrema.org/articulos/newMethodology.es.html.

ANEXOS

Anexo: 1 Procedimiento que emplea ALBET en la evaluación de proyectos.

SECCIÓN	CONTENIDO
IDENTIFICACIÓN	Nombre de la compañía, dirección, números de teléfonos y fax, sitio web y dirección de correo electrónico. Incluye los datos de la casa matriz y sus subsidiarias.
DEPENDENCIAS	Tipo y localización de las instalaciones involucradas en el sector biotecnológico y médico-farmacéutico de la compañía.
HISTORIA	Información sobre fundadores, fecha de fundación, cambios de nombre, principales adquisiciones, subsidiarias anteriores o inversiones y membrecía a organizaciones, etc.
DIRECTIVOS	Relación de los principales ejecutivos y sus cargos; datos específicos; currículum y otras actividades actuales.
TIPO DE COMPAÑIA	Estado legal de la compañía (pública, privada, agencia gubernamental, fundación o centro de investigación universitaria, etc.)
EMPLEADOS	Número total de empleados.
PRINCIPAL INVERSIONISTA	Incluye el inversionista principal.
ESTRATEGIA DE NEGOCIOS	Refleja las direcciones estratégicas de la compañía.
ACUERDOS	Describe el contenido del acuerdo, la fecha, el nombre de la compañía con la que se firma el acuerdo.
I+D	Describe los intereses generales de investigación y las tecnologías trabajadas por la compañía o subsidiarias. Incluye también investigaciones hechas con otras compañías.

PRODUCTOS EN EL MERCADO	Describe los productos que comercializan.
PRODUCTOS EN DESARROLLO	Ej. Para el caso de la industria biofarmacéutica. Incluye el nombre genérico y la marca comercial de los productos, los usos esperados y la fase de desarrollo en que se encuentran. Para los productos terapéuticos humanos las fases son: · Preclinicals · IND (investigational new drug application) · Phase I · Phase II · Phase III · NDA (new drug application)- para productos biológicos, la PLA(product license approval) es análoga a la NDA · Market Para los instrumentos y diagnósticos para humanos, los estados son: · Clinical testing · 510K clearance or PMA (premarket approval) · Market
NOTICIAS DE PRENSA	Información de prensa que recoge las noticias y cambios relevantes en la evolución de la empresa.
ASPECTOS JUDICIALES	Litigios, sentencias y órdenes judiciales.
COMPETENCIA	Incluye los principales competidores.
PUBLICACIONES, SOFTWARE, BASE DATOS	Incluye las publicaciones, bases de datos y software de la compañía.

Anexo: 2 Plantilla de Criterios Técnicos.

Indicadores	Peso	Evaluación (0-5)
1. Calidad del personal directivo		
2. Calidad del personal directo		
3. Posibilidades de la tecnología disponible		
4. Disponibilidad de las herramientas de software necesarias		
5. Acceso al mercado de tecnologías y licencias necesarias		
6. Organización del proceso de producción		
7. Calidad del mantenimiento de la tecnología.		
8. Localización el proyecto		
9. Gestión de tiempo		
10. Disponibilidad de insumos		
11. Nivel de información del cliente		
12. Garantía de servicios necesarios		
13. Dependencia tecnológica		
14. Cumplimientos de la calidad del producto final		
15. Relación demanda/capacidad de producción		
16. Necesidad de nuevas inversiones		
17. Despliegue del producto		
18. Costo del proyecto		
19. Éxitos pasados		
20. Conocimiento de las legislaciones vigentes.		

Anexo: 3 Plantilla de Criterios de Mercado.

Indicadores	Peso	Evaluación (0-5)
1. Necesidad del producto		
2. Productos sustitutivos		
3. Productos similares		
4. Productos complementarios		
5. Universo de probables consumidores		
6. Capacidad de pago potencial de los clientes		
7. Restricciones legales de investigaciones		
8. Dificultades de acceso al mercado		
9. Demanda actual		
10. Demanda futura		
11. Cantidad de competidores		
12. Oferta actual		
13. Oferta futura		
14. Calidad del producto		
15. Servicio al cliente		
16. Protección al mercado		
17. Régimen de mercado		
18. Grado de receptibilidad del producto en el mercado		
19. Éxitos anteriores		
20. Novedad del producto		

Anexo 4: Plantilla de Proyectos.

Datos del Proyecto	
1. Nombre del Proyecto	
2. Fecha de registro	
3. Entidad	
4. País de procedencia	
5. Nivel de confianza de los Expertos	
6. Factibilidad Técnica	
7. Factibilidad de Mercado	
8. Descripción	

Anexo 5: Matriz de Decisión.

PC / Pt	Pc < 0,3	0,3 < Pc < 0,5	0,5 < Pc < 0,7	Pc > 0,7
Pt < 0,3	Abandonar	Analizar los indicadores de mercado y los técnicos profundamente	Analizar cómo mejorar los indicadores técnicos	Analizar cómo mejorar los indicadores técnicos
0,3 < Pt < 0,5	Analizar cómo mejorar los indicadores del mercado	Analizar cómo superar los indicadores de mercado o los técnicos	Adelante	Adelante
0,5 < Pt < 0,7	Analizar cómo mejorar los indicadores de mercado	Adelante cuidando el movimiento del mercado	Adelante	Adelante
Pt > 0,7	Analizar cómo mejorar los indicadores de mercado	Adelante cuidando el movimiento del mercado	Adelante	Adelante

Anexo 6: Descripción textual de los Casos de Uso del Sistema.

Nombre del Caso de Uso	**Gestionar usuario**
Actores	Administrador(Inicia)
Propósito	Mantener un control de todos los datos relacionados con los usuarios que utilizan el software.
Resumen	El caso de uso se inicia cuando el administrador desea realizar la gestión de un usuario. Puede **registrar, modificar y eliminar** un usuario. El caso de uso finaliza cuando el administrador realiza las acciones necesarias sobre el usuario.
Referencias	**R1** (1.1, 1.2, 1.3)
Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
	1 - El sistema muestra la **Pantalla1** con las opciones: Adicionar (A), Modificar (B) y Eliminar (C). a) Si elige adicionar usuario, ir a la sección: “Adicionar”. b) Si elige modificar, ir a la sección: “Modificar Usuario”. c) Si elige eliminar, ir a la sección: “Eliminar Usuario”.
Sección “Adicionar ”	
	2- El sistema muestra en **Pantalla2** los campos: Usuario (A), Rol (B), y el botón Aceptar (G).
3- El Administrador del Sistema entra los datos del usuario, para realizar su registro en la	3.1- El sistema verifica que los campos,

aplicación, presiona el botón Aceptar (G).	Usuario y Rol estén llenos. 3.2- El sistema verifica que este usuario no exista. 3.3- El sistema almacena los datos del usuario. 3.4- El sistema emite un mensaje informando que el usuario ha sido registrado y finaliza el caso de uso.
Curso Alternativo de los eventos	
	3.2- Se emite un mensaje para que llene los campos obligatorios.
4- El Administrador llena los campos vacíos y presiona el botón Aceptar (G).	4.1- Ir a 3.1.
	3.3- Si el usuario existe se muestra un mensaje informativo y finaliza el caso de uso.
Sección "Modificar"	
	2- El sistema muestra en **Pantalla3** con los campos: Usuario (A), Rol (B), llenos y el botón Aceptar (G).
3- El administrador del sistema realiza las modificaciones deseadas y presiona el botón Aceptar (G).	3.1- El sistema verifica que los campos queden llenos. 3.2- El sistema almacena la modificación de los datos del usuario. 3.3- El sistema emite un mensaje informando que los datos del usuario han

	sido modificados y finaliza el caso de uso.
Curso Alternativo de los eventos	
	3.2- El sistema emite un mensaje para que llene los campos obligatorios.
4- El Administrador llena los campos vacíos y presiona el botón Aceptar (G).	4.1- Ir a 3.1.
Sección "Eliminar"	
2-El Administrador presiona el botón eliminar.	2.1- El sistema muestra un mensaje de confirmación para eliminar.
3-El Administrador confirma que desea eliminar.	3.1- El sistema verifica que el usuario exista en la base de datos. 3.2- El sistema elimina el usuario de la base de datos y finaliza el caso de uso.
Curso Alternativo de los eventos	
	3.2- Si el usuario no existe el sistema emite un mensaje informativo y finaliza el caso de uso.

Nombre del Caso de Uso	**Autenticar usuario**
Actores	Usuario del Sistema (Inicia).
Propósito	Permitir a los usuarios acceder a la información que le corresponde, así como realizar las tareas correspondientes dentro del ámbito de sus privilegios.
Resumen	El Caso de Uso se inicia cuando el usuario introduce los datos que se le piden para acceder a la aplicación, estos se verifican y finaliza dándole los permisos y habilitándole la entrada.
Referencias	**R2** (2.1, 2.2, 2.3)

Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
	1- El sistema muestra la **Pantalla1** con los campos Usuario (A), Contraseña (B) y el botón Aceptar (C).
1- El usuario entra Usuario en (A), la Contraseña en (B) y presiona el botón Aceptar (C) para iniciar la sesión.	1.1- El sistema encripta la contraseña. 1.2- Verifica que el usuario y la contraseña sean correctos. 1.3- En caso de ser correcto se le asignan los permisos.
Curso Alternativo de los eventos	
	1.3- En caso de estar incorrecto el usuario o la contraseña el sistema envía un mensaje de aviso.

Nombre del Caso de Uso	**Gestionar Proyecto**
Actores	Analista (Inicia)
Propósito	Mantener un control sobre los datos de un proyecto de software.
Resumen	El caso de uso se inicia cuando el cliente le solicita al analista adicionar, modificar o eliminar los datos de un determinado proyecto de software.
Referencias	**R3** (3.1, 3.2, 3.3)
Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1- El Analista necesita realizar alguna acción referente a la gestión de Proyectos de software.	1.1- El sistema muestra en **Pantalla 1** las opciones: Adicionar Proyecto (A), Modificar Proyecto (B) y Eliminar Proyecto (C). a) Si elige adicionar proyecto, ir a la

	sección: “Adicionar”. b) Si elige modificar proyecto, ir a la sección: “Modificar”. c) Si elige eliminar proyecto, ir a la sección: “Eliminar”.
Sección: “Adicionar”	
	2- El sistema muestra en **Pantalla2** los campos: Nombre del Proyecto (A), Fecha de Registro (B), Entidad (C), País (D), Descripción (F) y el botón Aceptar (G).
3- El Analista llena los campos relacionados con el proyecto y para su registro en la aplicación presiona el botón Aceptar (G)	3.1 El sistema verifica que los campos: Nombre del Proyecto (A), Fecha de Registro (B), Entidad (C), País (D), Descripción (F) estén llenos. 3.2- El sistema almacena los datos del usuario.
Curso Alternativo de los eventos	
	3.2- Se emite un mensaje para que llene los campos obligatorios.
4- El Analista llena los campos que le faltaron y presiona el Botón Aceptar (G)	4.1 ir a 3.1
Sección “Modificar”	
	2- El sistema muestra en **Pantalla3** los campos: Nombre del Proyecto (A), Fecha de Registro (B), Entidad (C), País (D), Descripción (F) y el botón Aceptar (G).
3- El Analista del sistema realiza las	3.1- El sistema verifica que los campos

modificaciones deseadas y presiona el botón Aceptar (G).	queden llenos. 3.2- El sistema almacena la modificación de los datos del usuario. 3.3- El sistema emite un mensaje informando que los datos del usuario han sido modificados y finaliza el caso de uso.
Curso Alternativo de los eventos	
	3.2- El sistema emite un mensaje para que llene los campos obligatorios.
4- El Analista llena los campos vacíos y presiona el botón Aceptar (G).	4.1- Ir a 3.1.
Sección "Eliminar"	
2-El Analista selecciona el Proyecto que quiere eliminar y presiona el botón eliminar.	2.1- El sistema muestra un mensaje de confirmación para eliminar.
3-El Analista confirma que desea eliminar.	3.1- El sistema verifica que el Proyecto exista en la base de datos. 3.2- El sistema elimina el proyecto de la base de datos y finaliza el caso de uso.
Curso Alternativo de los eventos	
	3.2- Si el Proyecto no existe el sistema emite un mensaje informativo y finaliza el caso de uso.

Nombre del Caso de Uso	**Gestionar Evaluación**
Actores	Analista (Inicia)
Propósito	Crear la evaluación que se le realizará a un Proyecto de Software.
Resumen	El caso de uso se inicia cuando ya ha sido registrado: el Proyecto, Expertos y Criterios a evaluar y además el Analista desee realizar la evaluación donde podrá adicionar, eliminar criterios y expertos como estime conveniente así como seleccionar el proyecto al cual le va a realizar la evaluación.
Referencias	**R4** (4.1, 4.2, 4.3, 4.4, 4.5)
Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1- El Analista necesita realizar alguna acción referente a la gestión de la Evaluación.	1.1- El sistema muestra en **Pantalla 1** las opciones: Seleccionar Proyecto (A) y los Botones: Crear_Evaluación_Técnica (B) y Crear_Evaluación_Mercado (C). a) Si elige Crear_Evaluación_Técnica ir a la sección: “Crear_Eva_Técnica”. b) Si elige Crear_Evaluación_Mercado ir a la sección: “Crear_Eva_Mercado”.
Sección: “Crear_Eva_Técnica”	
	2- El sistema muestra en **Pantalla2** los listados: Criterios Técnicos (A) y Expertos (B) y el Botón Aceptar (G).
3- El Analista selecciona los criterios técnicos que el estime y los expertos que trabajarán en la evaluación y presiona el botón Aceptar (G)	3.1- El sistema verifica la selección del analista. 3.2- El sistema crea la Evaluación Técnica definida por el Analista, mostrándole la

	misma y la Opción: Eliminar (E). 3.3- Si elige la opción Eliminar (E) ir a la sección “Eliminar”.
Curso Alternativo de los eventos	
	3.2- El sistema emite un mensaje para que realice la selección.
4- El Analista realiza la selección y presiona el Botón Aceptar (G)	4.1- ir a 3.1
Sección “Crear_Eva_Mercado”	
	2- El sistema muestra en **Pantalla3** los listados: Criterios Mercado (A) y Expertos (B) y el Botón Aceptar (G).
3- El Analista selecciona los criterios de mercado que el estime y los expertos que trabajarán en la evaluación y presiona el botón Aceptar (G)	3.1- El sistema verifica la selección del analista. 3.2- El sistema crea la Evaluación de Mercado definida por el Analista, mostrándole la misma y la Opción: Eliminar (E). 3.3- Si elige la opción Eliminar (E) ir a la sección “Eliminar”.
Curso Alternativo de los eventos	
	3.2- El sistema emite un mensaje para que realice la selección.
4- El Analista realiza la selección y presiona el Botón Aceptar (G)	4.1- ir a 3.1

Sección "Eliminar"	
	2- El sistema muestra en la **Pantalla4** la Evaluación, con los listados de Expertos (A) y Criterios (B) el botón Eliminar (G).
3-El Analista selecciona el Experto(s) y/o Criterio(s) que quiere eliminar y presiona el botón eliminar (G).	3.1- El sistema muestra un mensaje de confirmación para eliminar.
4-El Analista confirma que desea eliminar.	4.1- El sistema elimina el/los Experto(s) y/o criterio(s).
Curso Alternativo de los eventos	
	4.1- Si el Experto o Criterio no existe el sistema emite un mensaje informativo y finaliza el caso de uso.

Nombre del Caso de Uso	**Gestionar Criterios**	
Actores	Analista (Inicia)	
Propósito	Adicionar una serie de criterios que el Analista considera que le hace falta para su Evaluación.	
Resumen	El caso de uso se inicia cuando el Analista desea registrar al sistema antes de realizar una Evaluación determinados criterios que estima que necesita la misma para un resultado más eficiente.	
Referencias	**R5** (5.1, 5.2, 5.3)	
Curso Normal de los eventos		
Acciones del Actor		**Respuesta del sistema**
1- El Analista necesita realizar alguna acción referente a la gestión de Criterios para la Evaluación de algún Proyecto de software.		1.1- El sistema muestra en **Pantalla 1** las opciones: Adicionar Criterio (A), Modificar Criterio (B) y Eliminar Criterio (C).

	a) Si elige adicionar criterio, ir a la sección: "Adicionar". b) Si elige modificar criterio, ir a la sección: "Modificar Criterio". c) Si elige eliminar criterio, ir a la sección: "Eliminar Criterio"
Sección: "Adicionar"	
	2- El sistema muestra en **Pantalla2** los campos: Nombre del Criterio (A), Tipo de Criterio (B), Descripción (C) y el botón Aceptar (G).
3- El Analista llena los campos relacionados con el criterio y para su registro en la aplicación presiona el botón Aceptar (G)	3.1 El sistema verifica que los campos: Nombre del Criterio (A), Tipo de Criterio (B), Descripción (C) estén llenos. 3.2- El sistema almacena los datos del Criterio.
Curso Alternativo de los eventos	
	3.2- Se emite un mensaje para que llene los campos obligatorios.
4- El Analista llena los campos que le faltaron y presiona el Botón Aceptar (G)	4.1 ir a 3.1
Sección "Modificar Criterio"	
	2- El sistema muestra en **Pantalla5** los campos: Nombre del Criterio (A), Tipo de Criterio (B), Descripción (C) y el botón Aceptar (G).
3- El Analista del sistema realiza las	3.1- El sistema verifica que los campos

modificaciones deseadas y presiona el botón Aceptar (G).	queden llenos. 3.2- El sistema almacena la modificación de los datos del Criterio. 3.3- El sistema emite un mensaje informando que los datos del Criterio han sido modificados y finaliza el caso de uso.
Curso Alternativo de los eventos	
	3.2- El sistema emite un mensaje para que llene los campos obligatorios.
4- El Analista llena los campos vacíos y presiona el botón Aceptar (G).	4.1- Ir a 3.1.
Sección "Eliminar"	
2-El Analista selecciona el Criterio que quiere eliminar y presiona el botón Eliminar.	2.1- El sistema muestra un mensaje de confirmación para eliminar.
3-El Analista confirma que desea eliminar.	3.1- El sistema verifica que el Criterio existe en la base de datos. 3.2- El sistema elimina el Criterio de la base de datos y finaliza el caso de uso.
Curso Alternativo de los eventos	
	3.2- Si el Criterio no existe el sistema emite un mensaje informativo y finaliza el caso de uso.

Nombre del Caso de Uso	**Calcular Concordancia**
Actores	Analista (Inicia)
Propósito	Determinar si hubo consistencia entre los Expertos en cuanto a las evaluaciones asignadas a los Criterios.

Resumen	El caso de uso se inicia cuando el Analista desea saber si existe o no concordancia entre los Expertos para poder continuar con la Evaluación.	
Referencias	**R6** (6.1)	
Curso Normal de los eventos		
Acciones del Actor		**Respuesta del sistema**
1- El Analista necesita realizar alguna acción referente al cálculo de concordancia.		1.1- El sistema muestra en **Pantalla 1** las opciones: Seleccionar Proyecto (A) y el Botón Calcular (B)
2- El Analista realiza la selección del Proyecto en (A) al que desea hallarle la concordancia de los Expertos y presiona el Botón Calcular(B)		2.1- El sistema verifica que se haya seleccionado el Proyecto en (A) que se está evaluando. 2.2- El sistema le ofrece el resultado al analista. Fin del caso de uso.
Curso Alternativo de los eventos		
		2.2 – El sistema lanza un mensaje para que se haga la selección del proyecto correcto.
3- El Analista realiza la selección del Proyecto.		3.1 ir a 2.1

Nombre del Caso de Uso	**Evaluar Criterio**	
Actores	Experto (Inicia)	
Propósito	Evaluar los criterios tanto del orden técnico como de mercado con el objetivo de gestionar la factibilidad du un proyecto de software.	
Resumen	El caso de uso se inicia cuando el Experto le asigna un peso o una calificación a un número determinado de criterios.	
Referencias	**R7** (7.1)	
Curso Normal de los eventos		

Acciones del Actor	**Respuesta del sistema**
1- El Experto necesita realizar alguna acción referente a Evaluar Criterio.	1.1- El sistema muestra en **Pantalla 1** la lista de criterios que debe evaluar (A), la sección evaluación (B) y el Botón Aceptar (C)
2- El Experto realiza la evaluación en (B) y presiona el Botón Aceptar(C)	2.1- El sistema verifica que se hayan llenado todos los campos en (B). 2.2- El sistema guarda la evaluación. Fin del caso de uso.
Curso Alternativo de los eventos	
	2.2 – El sistema lanza un mensaje para que se llenen los campos libres.
3- El Experto completa la Evaluación de los criterios.	3.1 ir a 2.1

Nombre del Caso de Uso	**Aprobar Proyecto**
Actores	Decisor (Inicia)
Propósito	La solución que da el Sistema debe estar supervisada por un persona capacitada, pues el mismo a veces puede dar una solución bastante puntual en los casos que no son extremos y no tener en cuenta otros factores.
Resumen	El caso de uso se inicia cuando todo el proceso de cálculos ha terminado y el Decisor se presenta a supervisar el resultado y tomar una decisión al respecto.
Referencias	**R8** (8.1, 8.2,)
Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**

1- El Decisor necesita realizar alguna acción referente a Aprobar el Proyecto que se ha evaluado.	1.1- El sistema muestra en **Pantalla 1** el listado de los proyectos existentes en el sistema (A)
2- El Decisor realiza la selección del Proyecto en (A) al que desea aprobar.	2.1- El sistema verifica que se haya seleccionado el Proyecto en (A) que se quiere aprobar. 2.2- El sistema le muestra en la **Pantalla2** la Matriz de Decisión al Decisor y el ListBox: Decisión junto al Botón: Aceptar (B)
3- El Decisor en dependencia del resultado mostrado y su juicio escoge aceptar el Proyecto considerando que es factible su contratación o escoge rechazarlo, presionando el Botón Aceptar.	3.1- El sistema Guarda la decisión del Decisor.
Curso Alternativo de los eventos	
	2.2 – El sistema lanza un mensaje para que se haga la selección del proyecto correcto.
3- El Decisor realiza la selección del Proyecto.	3.1 ir a 2.1

Nombre del Caso de Uso	**Calcular Factibilidad**
Actores	Analista (Inicia)
Propósito	Determinar la factibilidad de un Proyecto de Software para su futura contratación.
Resumen	El caso de uso se inicia cuando el Analista desea saber si el Proyecto de Software que ha sido evaluado es factible o no en los órdenes técnico y de mercado.

Referencias	**R9** (9.1)
Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1- El Analista necesita realizar alguna acción referente al cálculo de factibilidad.	1.1- El sistema muestra en **Pantalla 1** las opciones: Seleccionar Proyecto (A) y los botones: Factibilidad_Técnica (B) y Factibilidad_Mercado (C)
2- El Analista realiza la selección del proyecto en (A) al que desea hallarle la Factibilidad y presiona el Botón de acuerdo a lo que desee determinar (Técnico/mercado).	2.1- El sistema verifica que se haya seleccionado el Proyecto en (A) que se ha evaluado. 2.2- El sistema le ofrece el resultado al Analista y lo guarda en la base de datos.
Curso Alternativo de los eventos	
	2.2 – El sistema lanza un mensaje para que se haga la selección del proyecto correcto.
3- El Analista realiza la selección del Proyecto.	3.1 ir a 2.1

Nombre del Caso de Uso	**Gestionar País**
Actores	Administrador (Inicia)
Propósito	Adicionar los países con los que la Organización evaluadora tiene convenios y así facilitarle el trabajo al Analista.
Resumen	El caso de uso se inicia cuando el Administrador desea registrar al sistema determinados países con los que se tiene convenio.
Referencias	**R10** (10.1, 10.2, 10.3)
Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1- El Administrador necesita realizar alguna	1.1- El sistema muestra en **Pantalla 1** las

acción referente a la gestión de países.	opciones: Adicionar País (A), Modificar País (B) y Eliminar País (C). a) Si elige adicionar país, ir a la sección: "Adicionar." b) Si elige modificar país, ir a la sección: "Modificar País." c) Si elige eliminar país, ir a la sección: "Eliminar País."
Sección: "Adicionar"	
	2- El sistema muestra en **Pantalla2** los campos: Nombre del País (A), y el botón Aceptar (G).
3- El Administrador llena el campo relacionado con el País y para su registro en la aplicación presiona el botón Aceptar (G)	3.1 El sistema verifica que el campo: Nombre del País (A), esté lleno. 3.2- El sistema almacena los datos del País.
Curso Alternativo de los eventos	
	3.2- Se emite un mensaje para que llene el campo obligatorio.
4- El Administrador llena el campo que le faltó y presiona el Botón Aceptar (G)	4.1 ir a 3.1
Sección "Modificar País"	
	2- El sistema muestra en **Pantalla3** los campos: Nombre del País (A) y el botón Aceptar (G).
3- El Administrador del sistema realiza las modificaciones deseadas y presiona el botón	3.1- El sistema verifica que el campo quede

Aceptar (G).	lleno. 3.2- El sistema almacena la modificación del dato del País. 3.3- El sistema emite un mensaje informando que el dato del País ha sido modificado y finaliza el caso de uso.
Curso Alternativo de los eventos	
	3.2- El sistema emite un mensaje para que llene el campo obligatorio.
4- El Administrador llena el campo vacío y presiona el botón Aceptar (G).	4.1- Ir a 3.1.
Sección "Eliminar País"	
2-El Administrador selecciona el País que quiere eliminar y presiona el botón Eliminar.	2.1- El sistema muestra un mensaje de confirmación para eliminar.
3-El Administrador confirma que desea eliminar.	3.1- El sistema verifica que el País existe en la base de datos. 3.2- El sistema elimina el País de la base de datos y finaliza el caso de uso.
Curso Alternativo de los eventos	
	3.2- Si el País no existe el sistema emite un mensaje informativo y finaliza el caso de uso.

Nombre del Caso de Uso	**Gestionar Entidad**
Actores	Administrador (Inicia)
Propósito	Adicionar las Entidades con las que la Organización evaluadora tiene convenios y así facilitarle el trabajo al Analista.

Resumen	El caso de uso se inicia cuando el Administrador desea registrar al sistema determinadas Entidades con las que se tiene convenio.
Referencias	**R11** (11.1, 11.2, 11.3)
Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1- El Administrador necesita realizar alguna acción referente a la gestión de entidades.	1.1- El sistema muestra en **Pantalla 1** las opciones: Adicionar Entidades (A), Modificar Entidades (B) y Eliminar Entidades (C). a) Si elige adicionar entidades, ir a la sección: “Adicionar”. b) Si elige modificar entidades, ir a la sección: “Modificar Entidades”. c) Si elige eliminar entidades, ir a la sección: “Eliminar Entidades”
Sección: “Adicionar”	
	2- El sistema muestra en **Pantalla2** los campos: Nombre de la Entidad (A), y el botón Aceptar (G).
3- El Administrador llena el campo relacionado con la entidad y para su registro en la aplicación presiona el botón Aceptar (G)	3.1 El sistema verifica que el campo: Nombre de la Entidad (A), esté lleno. 3.2- El sistema almacena los datos de la Entidad.
Curso Alternativo de los eventos	
	3.2- Se emite un mensaje para que llene el campo obligatorio.
4- El Administrador llena el campo que le faltó	4.1 ir a 3.1

y presiona el Botón Aceptar (G)	
Sección "Modificar Entidades"	
	2- El sistema muestra en **Pantalla3** los campos: Nombre de la Entidad (A) y el botón Aceptar (G).
3- El Administrador del sistema realiza las modificaciones deseadas y presiona el botón Aceptar (G).	3.1- El sistema verifica que el campo quede lleno. 3.2- El sistema almacena la modificación de la Entidad. 3.3- El sistema emite un mensaje informando que el dato de la Entidad ha sido modificado y finaliza el caso de uso.
Curso Alternativo de los eventos	
	3.2- El sistema emite un mensaje para que llene el campo obligatorio.
4- El Administrador llena el campo vacío y presiona el botón Aceptar (G).	4.1- Ir a 3.1.
Sección "Eliminar Entidades"	
2-El Administrador selecciona la Entidad que quiere eliminar y presiona el botón Eliminar.	2.1- El sistema muestra un mensaje de confirmación para eliminar.
3-El Administrador confirma que desea eliminar.	3.1- El sistema verifica que la Entidad existe en la base de datos. 3.2- El sistema elimina la Entidad de la base de datos y finaliza el caso de uso.
Curso Alternativo de los eventos	

	3.2- Si la Entidad no existe el sistema emite un mensaje informativo y finaliza el caso de uso.

Nombre del Caso de Uso	**Gestionar Criterios Propuestos**
Actores	Administrador (Inicia)
Propósito	Adicionar, modificar o eliminar una serie de criterios que el Administrador considera que hace falta o dejaron de tener vigencia la evaluación de proyectos de software.
Resumen	El caso de uso se inicia cuando el Administrador desea registrar, nuevos criterios que por su importancia tienen carácter general en la Evaluación de Proyectos de Software. También el Administrador puede modificar y/o eliminar criterios del sistema debido a que los mismos ya no aporten una información integral en la Evaluación.
Referencias	**R12** (12.1, 12.2, 12.3)

Curso Normal de los eventos	
Acciones del Actor	**Respuesta del sistema**
1- El Administrador necesita realizar alguna acción referente a la gestión de Criterios Propuestos para la Evaluación de Proyectos de softwares.	1.1- El sistema muestra en **Pantalla 1** las opciones: Adicionar Criterio (A), Modificar Criterio (B) y Eliminar Criterio (C). a) Si elige adicionar criterio, ir a la sección: “Adicionar”. b) Si elige modificar criterio, ir a la sección: “Modificar Criterio”. c) Si elige eliminar criterio, ir a la sección: “Eliminar Criterio”

Sección: "Adicionar"	
	2- El sistema muestra en **Pantalla2** los campos: Nombre del Criterio (A), Tipo de Criterio (B), Descripción (C), Modificable (E) y el botón Aceptar (G).
3- El Administrador llena los campos relacionados con el criterio y para su registro en la aplicación presiona el botón Aceptar (G)	3.1 El sistema verifica que los campos: Nombre del Criterio (A), Tipo de Criterio (B), Descripción (C), Modificable (E) estén llenos. 3.2- El sistema almacena los datos del Criterio.
Curso Alternativo de los eventos	
	3.2- Se emite un mensaje para que llene los campos obligatorios.
4- El Administrador llena los campos que le faltaron y presiona el Botón Aceptar (G)	4.1 ir a 3.1
Sección "Modificar Criterio"	
	2- El sistema muestra en **Pantalla3** los campos: Nombre del Criterio (A), Tipo de Criterio (B), Descripción (C), Modificable (E) y el botón Aceptar (G).
3- El Administrador del sistema realiza las modificaciones deseadas y presiona el botón Aceptar (G).	3.1- El sistema verifica que los campos queden llenos. 3.2- El sistema almacena la modificación de los datos del Criterio. 3.3- El sistema emite un mensaje informando que los datos del Criterio han

	sido modificados y finaliza el caso de uso.
Curso Alternativo de los eventos	
	3.2- El sistema emite un mensaje para que llene los campos obligatorios.
4- El Administrador llena los campos vacíos y presiona el botón Aceptar (G).	4.1- Ir a 3.1.
Sección "Eliminar Criterio"	
2-El Administrador selecciona el Criterio que quiere eliminar y presiona el botón Eliminar.	2.1- El sistema muestra un mensaje de confirmación para eliminar.
3-El Administrador confirma que desea eliminar.	3.1- El sistema verifica que el Criterio existe en la base de datos. 3.2- El sistema elimina el Criterio de la base de datos y finaliza el caso de uso.
Curso Alternativo de los eventos	
	3.2- Si el Criterio no existe el sistema emite un mensaje informativo y finaliza el caso de uso.

Anexo 7: Diagramas de Interacción

DIAGRAMA DE SECUENCIA
<CU_ Autenticar Usuario>

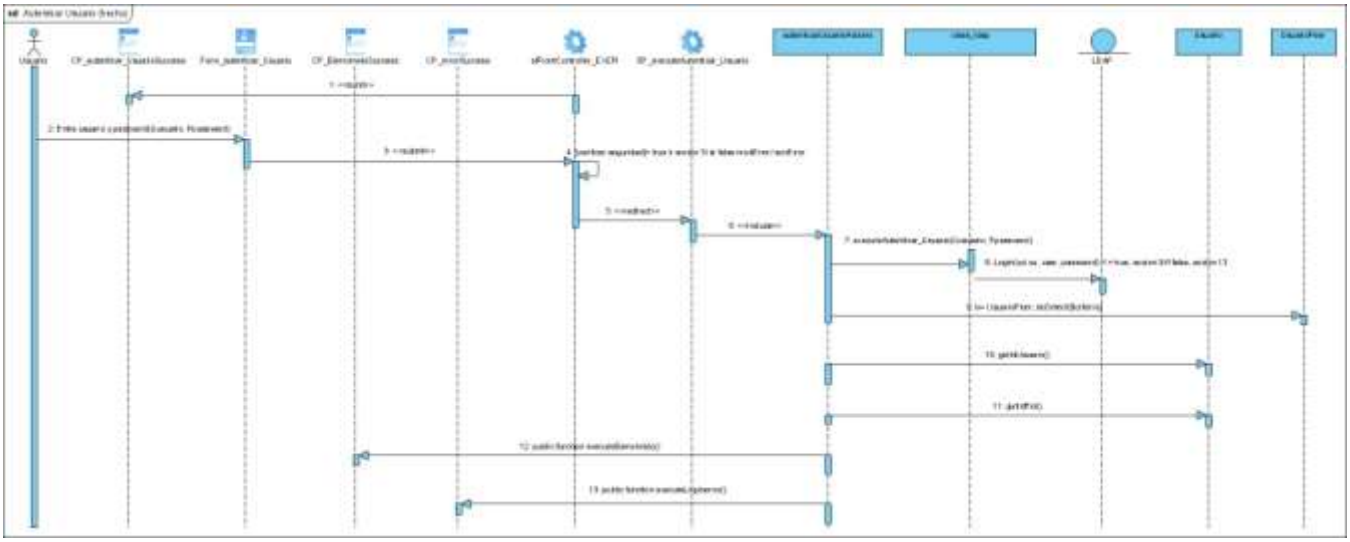


DIAGRAMA DE SECUENCIA
<CU_ Aprobar Proyecto>

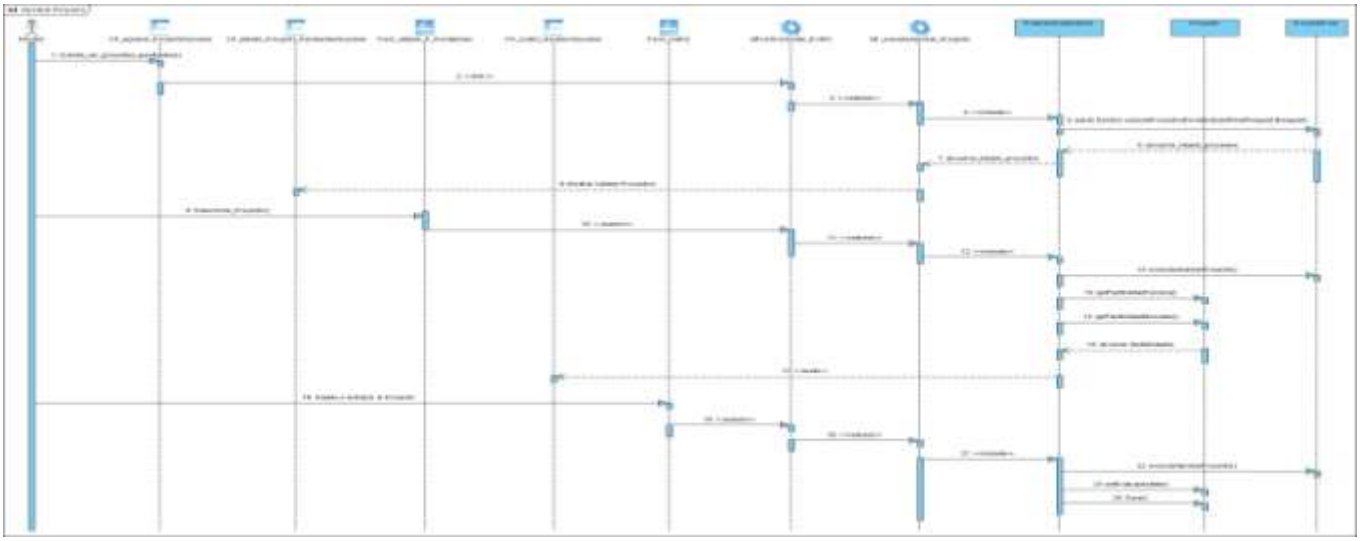


DIAGRAMA DE SECUENCIA
<CU_ Calcular Concordancia>

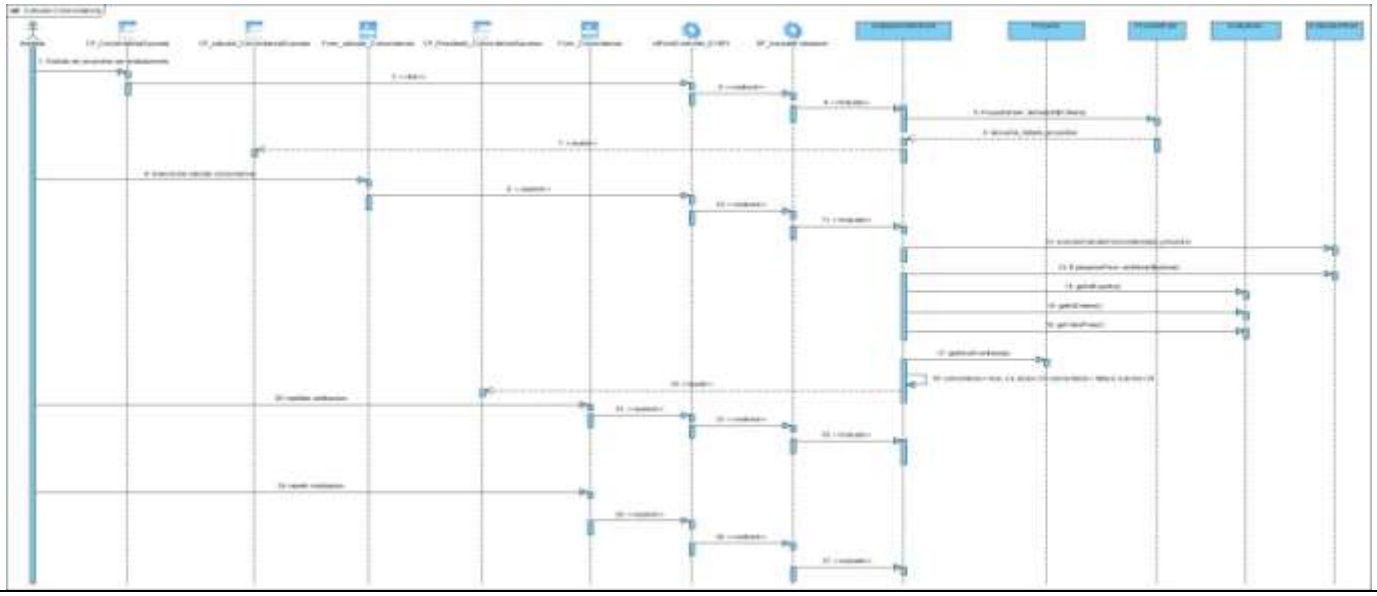


DIAGRAMA DE SECUENCIA
<CU_ Calcular Factibilidad>

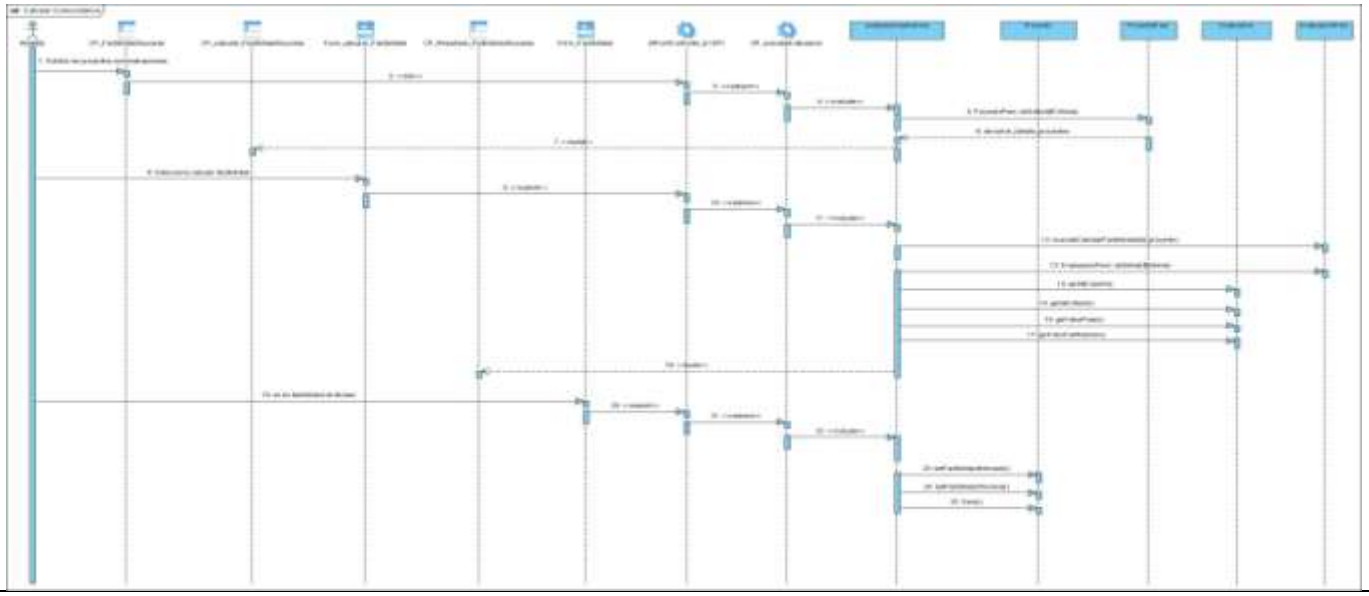


DIAGRAMA DE SECUENCIA
<CU_ Evaluar Criterio: Sección Calificaciones Pendientes >

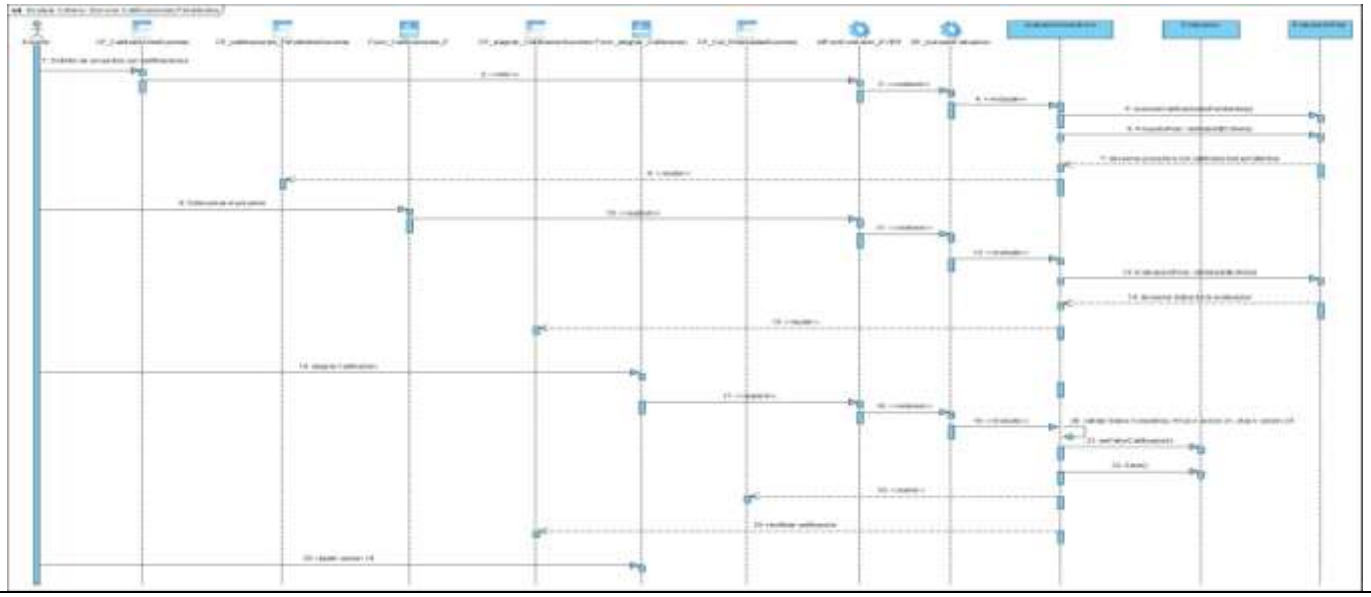


DIAGRAMA DE SECUENCIA
< CU_ Evaluar Criterio: Sección Evaluaciones Pendientes >

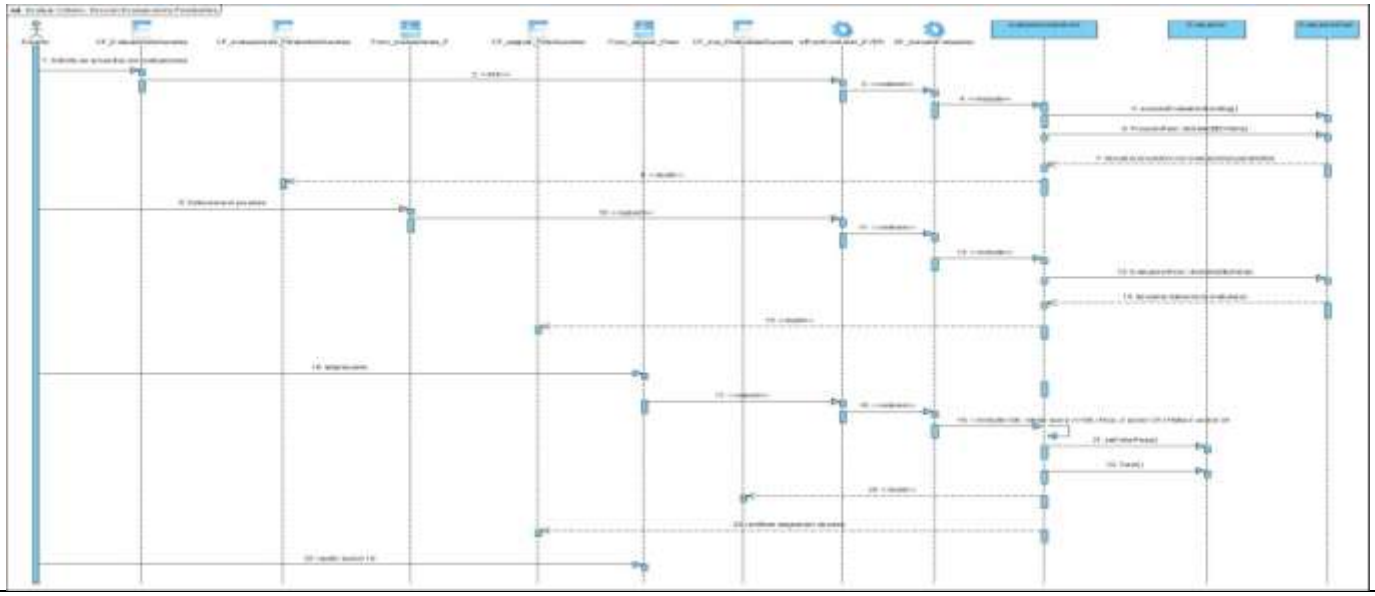


DIAGRAMA DE SECUENCIA
< CU_ Gestionar Criterio>

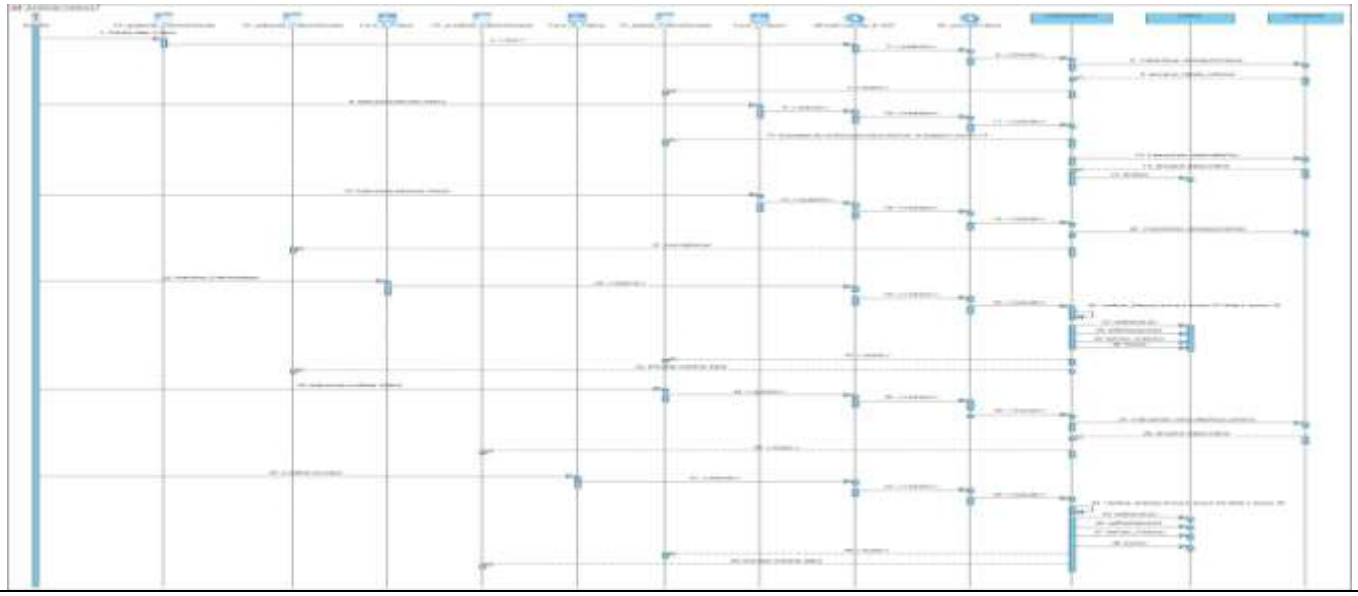


DIAGRAMA DE SECUENCIA
< CU_ Gestionar Criterio Propuesto>

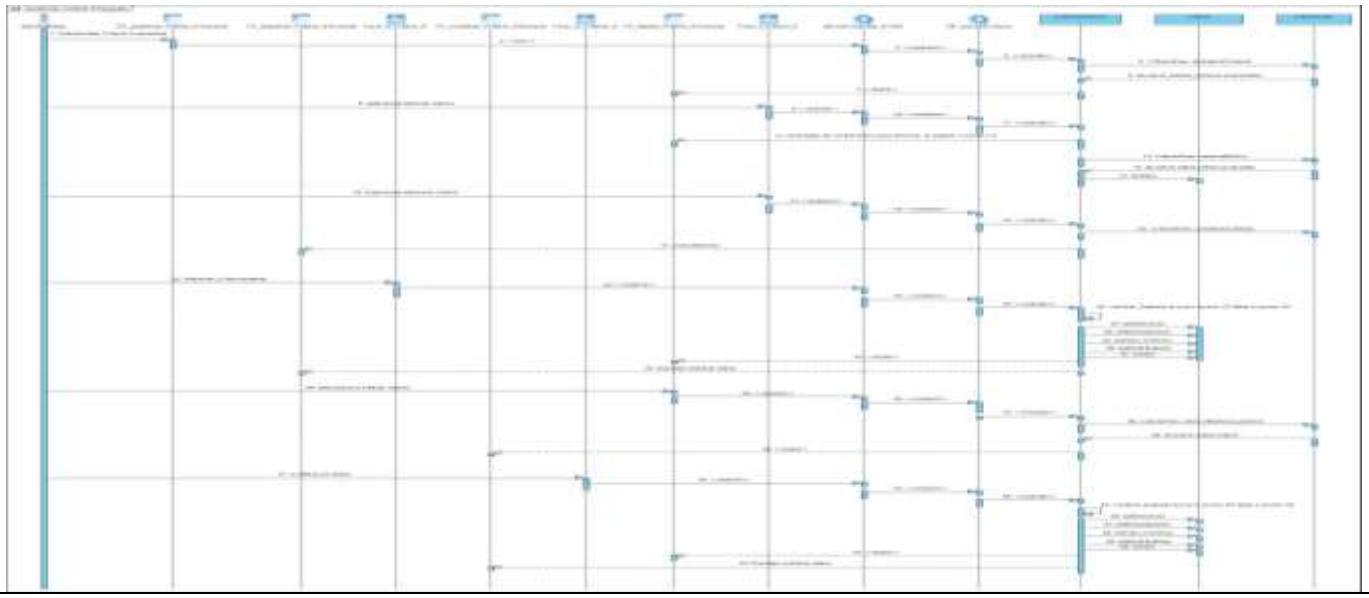


DIAGRAMA DE SECUENCIA
< CU_ Gestionar Entidad>

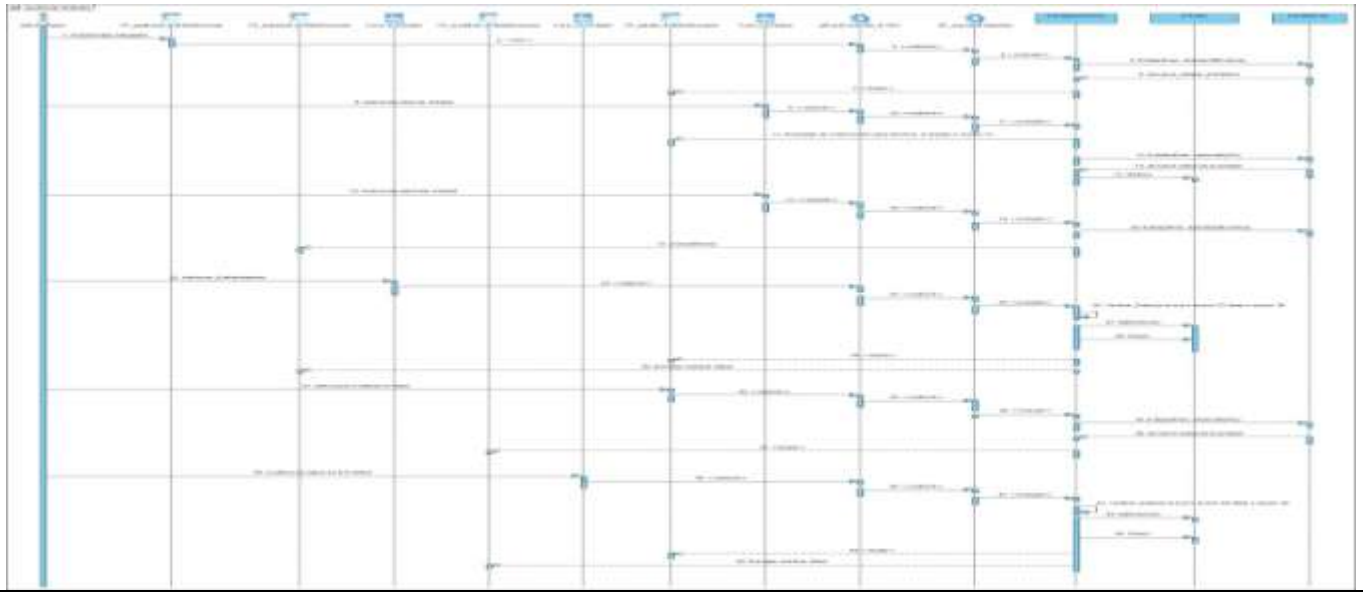


DIAGRAMA DE SECUENCIA
< CU_ Gestionar Evaluación>

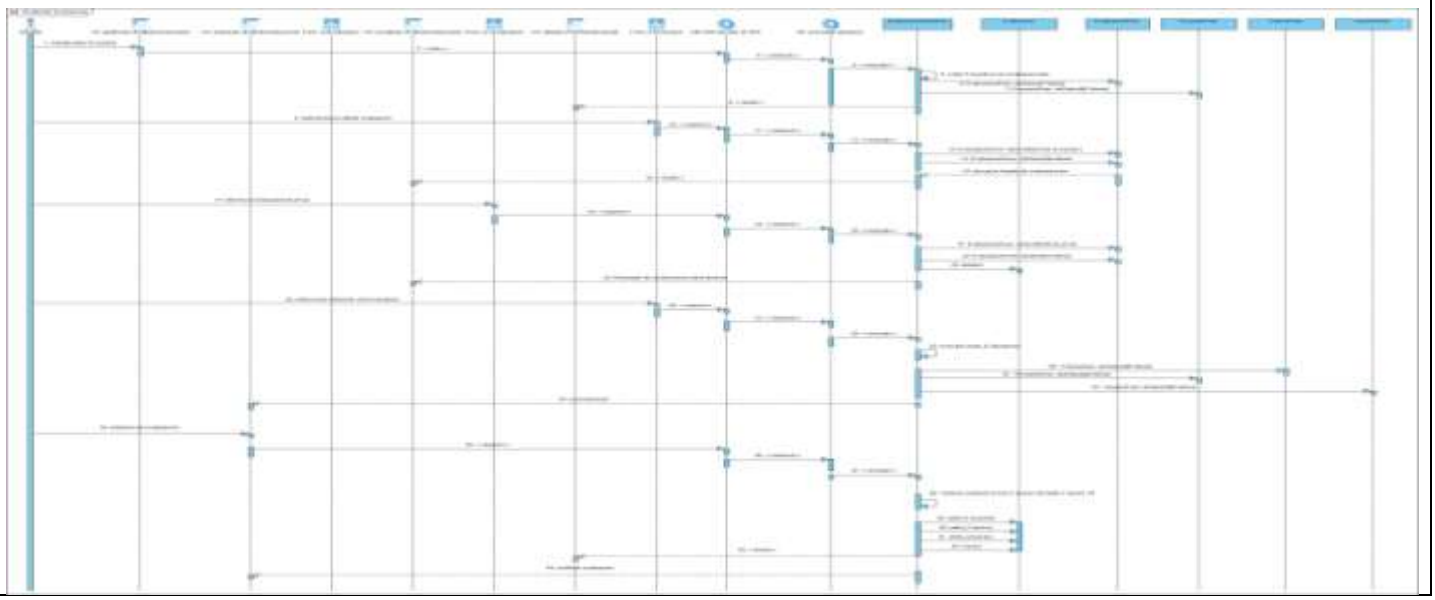


DIAGRAMA DE SECUENCIA
< CU_ Gestionar País>

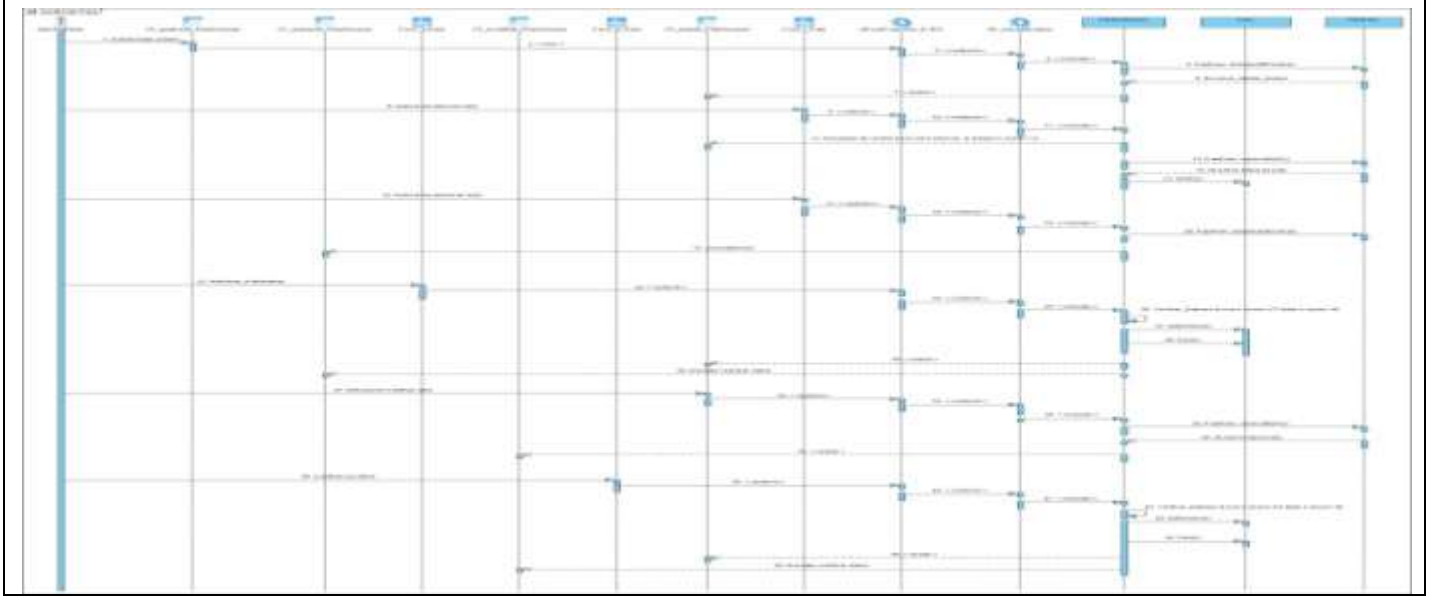


DIAGRAMA DE SECUENCIA
< CU_ Gestionar País>

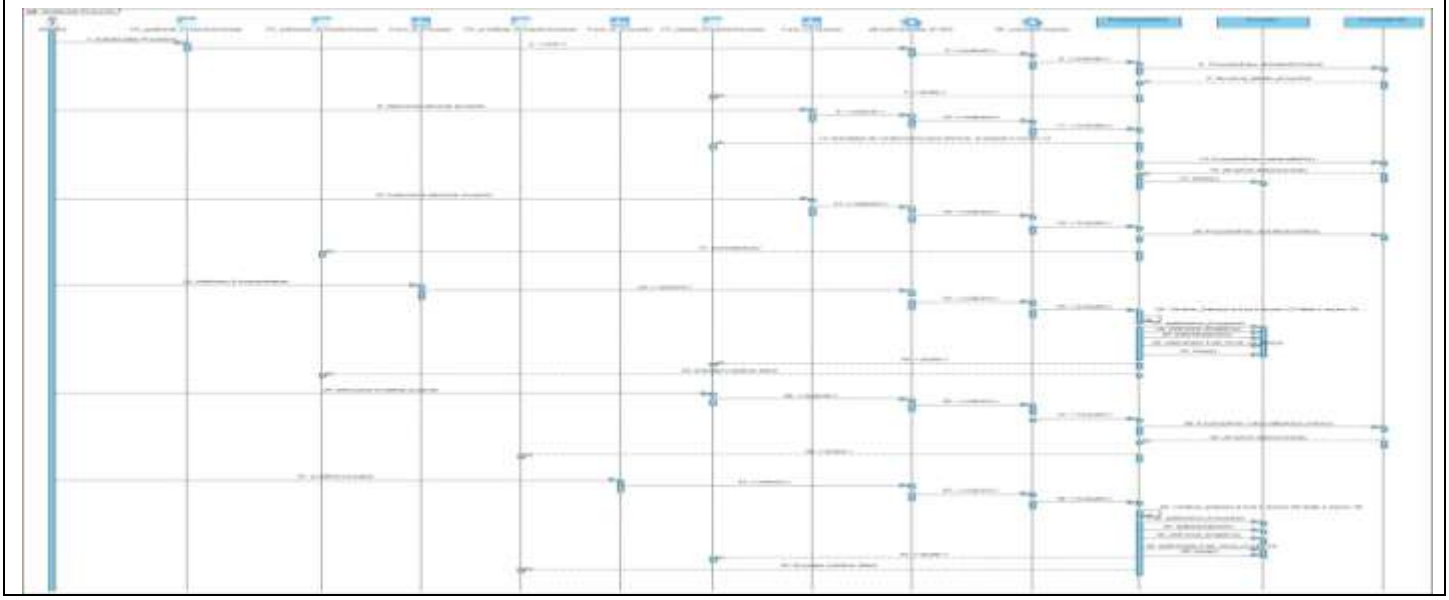
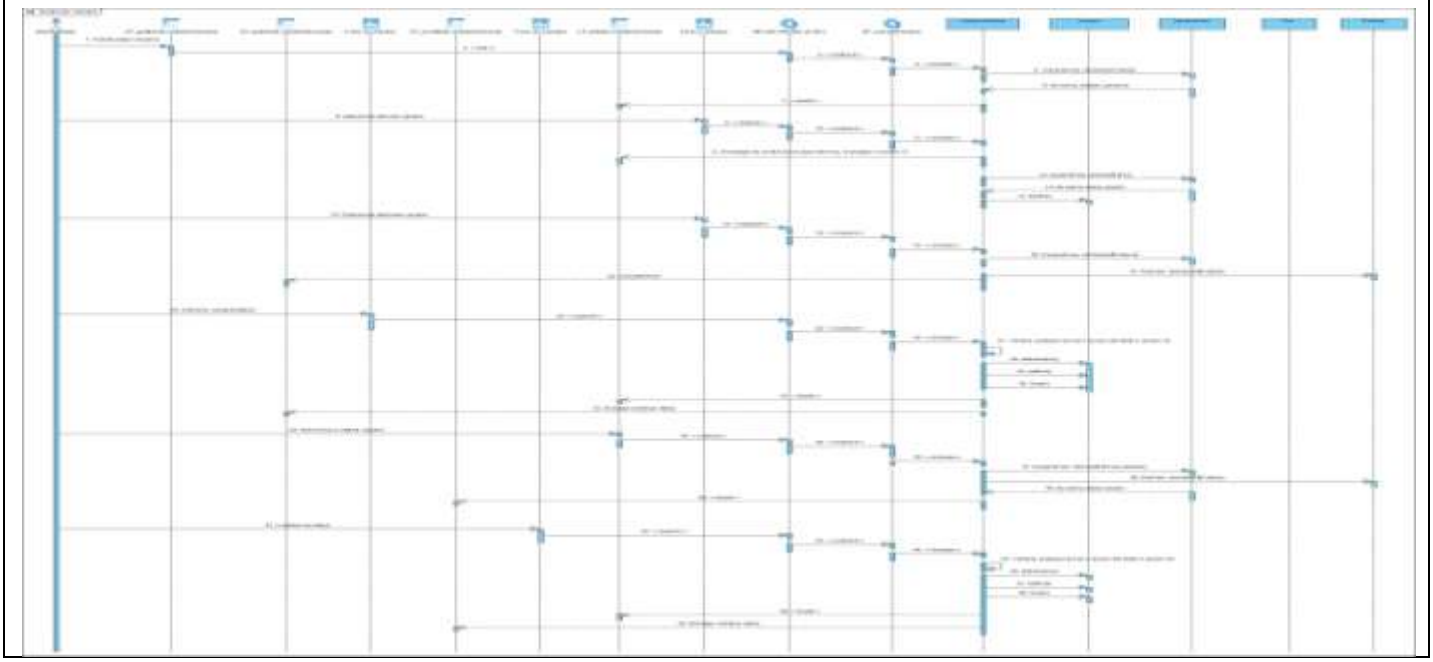


DIAGRAMA DE SECUENCIA
< CU_ Gestionar Usuario>



GLOSARIO DE TÉRMINOS

BD: (Base de Datos), es un conjunto de datos interrelacionados, almacenados con carácter más o menos permanente en la computadora, puede ser considerada una colección de datos variables en el tiempo.

CC: Estereotipo para identificar las clases controladoras que se encargan de dirigir y controlar el funcionamiento de una petición, decidiendo quien procesa y quien muestra.

CE: Estereotipo para identificar las clases entidades que contienen los atributos, según la interfaz.

CGI: son las siglas en inglés de Common Gateway Interface, en español, Interfaz común de puerta de enlace. Es el medio de comunicación que emplea un servidor Web para enviar información útil en ambos sentidos, entre el visualizador (browser) y su propio programa de cómputo.

CI: Estereotipo para identificar las clases interfaces. Encargadas de mostrar la información solicitada.

CP: Estereotipo para identificar las páginas clientes. Encargadas de mostrar la información solicitada.

EVEPI: Entorno Virtual de Evaluación para Proyectos Informáticos.

FRAMEWORK: es una estructura de soporte definida, mediante la cual otro proyecto de software puede ser organizado y desarrollado.

GNU: (General Public License), es una licencia creada por la Fundación de software Libre a mediados de los 80 y está orientada principalmente a proteger la libre distribución, modificación y uso de software.

GRASP: (General Responsibility Assignment Software Patterns), son patrones generales de software para asignación de responsabilidades.

JAVA: Lenguaje de programación de alto nivel orientado a objetos.

MySQL: Sistema de gestión de base de datos.

ORM: (*object-relational mapping*) o *"mapeo de objetos a bases de datos* es una interfaz que traduce la lógica de los objetos a la lógica relacional y está formada por objetos que permiten acceder a los datos y que contienen en sí mismos el código necesario para hacerlo.

PHP: Profesional Home Page Tools es un lenguaje de programación el cual se ejecuta en los servidores Web

Postgree: Sistema de gestión de base de datos.

RUP: (Rational Unified Process), es una metodología de desarrollo de software de tipo Robusta.

SGBD: Es el software que permite la utilización y/o la actualización de los datos almacenados en una o varias bases de datos por uno o varios usuarios.

SP: Estereotipo para identificar las páginas servidoras.

UML: Es un lenguaje gráfico para detallar, construir, visualizar y documentar las partes o artefactos (información que se utiliza o produce mediante un proceso de software). Pueden ser artefactos: un modelo, una descripción que comprende el desarrollo de software que se basen en el enfoque Orientado a Objetos.

WYSIWYG: Es el acrónimo de ***W****hat* ***Y****ou* ***S****ee* ***I****s* ***W****hat* ***Y****ou* ***G****et* (en inglés, "lo que ves es lo que obtienes"), Se aplica a los procesadores de texto y otros editores de texto con formato (como los editores de HTML) que permiten escribir un documento viendo directamente el resultado final, frecuentemente el resultado impreso.

XML: Siglas en inglés de *Extensible Markup Languaje* (leguaje de marcas extensibles), es una manera de definir lenguajes para diferentes necesidades.

XP: (eXtreme Programing), es una metodología de desarrollo de software de tipo ágil.