Universidad de las Ciencias Informáticas

Facultad 2

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

Herramienta para medir el estado de avance del flujo de implementación. (MetSoft)

Autor(es): Carlos César Alayón Cabezas.

Roberto Pérez Amaro.

Tutor: Ing. Rosario Rodríguez Torres.

Co-Tutor: Ing. Arturo César Arias Orizondo.

Sé que sólo hay una libertad: la de pensamiento.

Antoine de Saint-Exupery

Declaración de Autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Roberto Pérez Amaro

_____

Firma del Autor

Carlos César Alayón Cabezas

_____

Firma del Autor

Ing. Rosario Rodríguez Torres

_____

Firma del Tutor

Agradecimientos

Carlos César Alayon Cabezas.

Agradezco a mis padres que sin ellos hoy no hubiera podido estar aquí, a mis hermanos, a mis tíos y a todos mis familiares en general, a mi tutora que nos ayudo muchísimo, a mis amigos de siempre a yancy, al loquillo (Yusniel), al bafle (Osvaldo), al enano (Albín), al mono (daisel), a los marineros (Edel y Frank), a todas las chicas del aula, + los que menciono el Chino

A mi javita que me aguanto todo este tiempo y a todos los que de una manera han tenido han aportado un granito de arena para que este trabajo de diploma se pudiera realizar.

Roberto Pérez Amaro.

En primer lugar a mami y a pipo que fueron los más sacrificados en esta batalla, a mima por el apoyo durante todos estos años, solo yo sé todo lo que hicieron por mí.

A Naty por todo el amor, cariño, confianza y comprensión sobre todo en este último mes, a los suegros.

Saide, demás está decir todo lo que hay que agradecer, eres amigo papa!!!!!

Alejo, por lo último.

Gorda te quiero.

Rosario (tutora), de a corazón gracias.

A los amigos, Charro, Bu, Andrés, Repa, Vaquero, Made y Duany, Pina y María, Elio, a la prima, son una pila de gente que no me alcanza una hoja para esto….. A todos ellos + los que mencionó el Charly, gracias.

Dedicatoria

Roberto Pérez Amaro.

A mami y a pipo.

Charly te quiero macho.

Carlos César Alayón Cabezas.

A mami, ova y papi.

Chino te quiero macho.

Resumen

En la Universidad de las Ciencias Informáticas (UCI), la información relacionada con el estado de avance en la etapa de implementación de los proyectos productivos no es en ocasiones fiable, ya que no se cuenta con una aplicación que se adapte a las características específicas de la universidad, por lo que la toma de decisiones en cuanto a esfuerzo y tiempo se hace difícil e inexacta. En el presente Trabajo de Diploma titulado: “Métrica para el estado de avance en la fase de implementación” (MetSoft), bajo la disciplina de desarrollo propuesta por la metodología RUP, se llevan a cabo el modelamiento del negocio, el levantamiento de requisitos, el análisis y diseño, así como la implementación y prueba de un sistema que permite gestionar con mayor eficiencia el estado de avance referente a los proyectos productivos en la etapa de implementación, así como brindar una serie de reportes y gráficas del estado real de los mismos, basado en una métrica diseñada y aplicada en el proyecto SIGEP, la cual toma en cuenta características específicas en el marco de trabajo de la universidad. El valor práctico de MetSoft consiste en disponer de una herramienta web amigable y fácil de usar que aplique la métrica para la gestión de proyectos adaptada a las características de la UCI, siendo de ayuda tanto a los jefes de proyecto como a los directivos de la universidad en el control del seguimiento del desarrollo, los resultados de su aplicación servirían de base para futuras estimaciones y planificaciones y los resultados del avance tendrían una base objetiva basada en la medición del software.

Índice

Introducción

La Universidad de las Ciencias Informáticas (UCI) se adentra cada vez más en la producción de software empresarial con el reto de convertir a la industria del software cubano en un renglón fundamental de la economía del país. Una de sus misiones es la producción de software y servicios informáticos a partir de la vinculación estudio-trabajo como modelo de formación, logrando una fuerte relación Universidad-Empresa. Como universidad desarrolladora de software que pretende insertarse en el mercado internacional, la UCI debe lograr que la planificación se convierta en un elemento esencial a tener en cuenta en el proceso productivo, para obtener productos de alta calidad.

La gestión de proyectos de software es una actividad protectora dentro de la ingeniería del software, empieza antes de iniciar cualquier actividad técnica y continúa a lo largo de la definición, del desarrollo y del mantenimiento del software. (PRESSMAN 1998) La actividad de gestión del proyecto comprende medición y métricas, estimación, análisis de riesgos, planificación del programa, seguimiento y control. (PRESSMAN 1998) En la UCI la planificación de proyectos de software es una tarea que se consolida con la práctica.

Como universidad joven, el poco tiempo de experiencia no alcanza aún para obtener una base de registros históricos que permitan plantear una estrategia común para estimar y planificar con precisión todos los proyectos.

Existen en el mundo varios sistemas de métricas probadas y aceptadas en cientos de instituciones con importantes resultados, sin embargo en la UCI no se cuenta con un sistema de métricas que permita tener datos específicos de los proyectos de gestión de software. Por tanto, la institución necesita utilizar un sistema de medida que le permita cuantificar los procesos asociados a la gestión de proyectos. Para gestionar la calidad en proyectos existen diversas formas de gestionar esta calidad, una de ellas, quizás la más objetiva, es haciendo uso de las métricas del software.

Las métricas no son más que: medidas o colecciones de datos de las actividades de los proyectos y recursos. Estas deben ser: simples, objetivas, fáciles de coleccionar, fáciles de interpretar y difíciles de malinterpretar (RUP 2003). En la actualidad el uso de las métricas se extiende rápidamente entre la comunidad de desarrolladores de software debido a que las mismas producen indicadores a partir de los cuales se pueden tomar decisiones importantes (CAPUCHINO 1996). Gran parte del software que se desarrolla en la UCI es del tipo de gestión, que se define como aplicaciones informáticas o programas informáticos diseñados para facilitar al usuario la realización de un determinado tipo de trabajo.

Debido al desarrollo alcanzado por nuestra universidad en la producción de software y dada la necesidad de apoyarse en métricas que midan el avance de los mismos. Los principales desarrolladores son estudiantes los cuales muchas veces no cuentan con la experiencia ni el conocimiento necesario para desarrollar un software con calidad y en el tiempo requerido.

Problemas existentes en nuestra universidad que propiciaron la elaboración de este trabajo de diploma:

- ✓ No se cuenta con un registro que permita saber el estado de avance de cada proyecto en la fase de implementación en la universidad.
- ✓ Muchas veces es necesario preparar a los estudiantes antes de comenzar un ciclo de producción. .
- ✓ Debido a que los desarrolladores son estudiantes no pueden emplear el 100% de su tiempo en el desarrollo de software ya que tienen que cumplir con la docencia.

Por los problemas expuestos anteriormente se propone la creación de una herramienta que permita aplicar una métrica que mida el estado actual de los proyectos en la fase de implementación ,adaptada a las características de la universidad, donde los resultados de la aplicación servirán de base para futuras estimaciones y planificaciones; y los resultados del avance tendrán una base objetiva basada en la medición del software por medio del cumplimiento de tareas (actividades que se definen en la fase de implementación) y división de la medición por casos de uso teniendo en cuenta los factores de complejidad técnica (alta, media, baja), contribuyendo a la formación de datos históricos con los que hoy no cuenta la UCI.

La situación anteriormente expuesta lleva a definir que el **problema científico de la investigación** corresponde a: ¿Cómo mejorar el proceso de gestión de la información de los proyectos en la fase de implementación?,

Partiendo del problema planteado, el **objeto de estudio** se enfocará hacia los procesos de gestión de información de los proyectos productivos.

El **campo de acción** se encuentra delimitado por los procesos de gestión de información de los proyectos productivos en la fase de implementación desarrollados en la UCI.

La **idea a defender** se encaminará hacia el desarrollo de una aplicación informática, basada en una métrica que cuantifica el estado de avance de un proyecto en la fase de implementación, mejorando el control de la información referente a dicha etapa.

Como **objetivo general**: Desarrollar una herramienta que permita aplicar la métrica para medir el estado de avance de los proyectos en la fase de implementación.

Se espera como resultado: obtener un software en el que se aplique una métrica que permita conocer el estado de avance de la fase de implementación de un proyecto.

Objetivos específicos:

- ✓ Realizar un estudio del estado del arte sobre las principales tendencias de la gestión de información de los proyectos.
- ✓ Definir una metodología y seleccionar las herramientas óptimas para desarrollar el sistema de gestión propuesto.

Para lograr el objetivo trazado se llevaron a cabo las siguientes **tareas de la investigación**:

- ✓ Realizar un estudio sobre la existencia de aplicaciones o soluciones similares a las características del sistema a desarrollar.
- ✓ Realizar el análisis y diseño del software a desarrollar.
- ✓ Implementar el negocio centrado en los requisitos que debe cumplir el producto final.
- ✓ Definir las herramientas más apropiadas para el desarrollo del sistema, así como del lenguaje de programación y metodología a utilizar.
- ✓ Documentar todo el proceso de desarrollo del software sirviendo de base para futuras investigaciones.

Aportes prácticos esperados del trabajo

Entre los aportes prácticos que nos brindará este trabajo de diploma estarán:

- ✓ Creación de una aplicación que permita a los líderes de proyectos llevar un control detallado del trabajo realizado en la fase de implementación.
- ✓ Aportar datos estadísticos en el desarrollo de software en la UCI.
- ✓ Permitir a los líderes de proyectos adquirir experiencia en el desarrollo de software, ya que el uso de la herramienta les permitirá evaluar su trabajo gradualmente.
- ✓ Establecimiento de una métrica basada en el desarrollo de software en la UCI, con el fin de crear una base para el seguimiento de los proyectos tanto en la universidad como en el país.
- ✓ Creación de registros de todo el proyecto en la fase de implementación en que se utilice la métrica, esto permitirá que en condiciones similares se pueda definir el tiempo necesario para otro proyecto creando así una base de datos de registros de cada uno.
- ✓ Posibilitar a los diferentes directivos obtener importantes reportes del estado en que se encuentran los proyectos productivos en la UCI.
- ✓ Hacer más eficiente el control de la información en el proceso productivo de la UCI.

El trabajo se encuentra estructurado para su mejor comprensión en cinco capítulos:

Capítulo 1. Fundamentación Teórica: En este capítulo se exponen los elementos teóricos que soportan el trabajo realizado, siendo estos el estudio de las diferentes métricas que se utilizan a nivel mundial, una panorámica de las herramientas, lenguajes y metodologías empleadas durante el desarrollo de la propuesta de solución, acompañadas de una breve argumentación de su selección.

Capítulo 2. Características del Sistema: Se brinda una breve descripción de los procesos del negocio en cuestión, así como sus actores, trabajadores y casos de usos. Se presentan además el diagrama de casos de usos del negocio y el modelo de objetos y se definen tanto los requerimientos funcionales como los no funcionales, a la vez que se presentan los actores del sistema y el diagrama de casos de uso del mismo.

Capítulo 3. Análisis y Diseño del Sistema: Durante este capítulo se exponen los diagramas de clases del análisis y del diseño. Se muestran también el diseño de la base de datos y la descripción de cada una sus tablas.

Capítulo 4. Implementación y Pruebas: Se muestra cómo está implementado el sistema, a través de los diagramas de componentes y de despliegue, además se exponen y detallan las diferentes pruebas que se le realizan al mismo.

Capítulo 5. Estudio de Factibilidad. Durante este capítulo se realizará el estudio de factibilidad del proyecto a desarrollar y la aplicación de pruebas a la herramienta.

Capítulo 1: Fundamentación Teórica

1.1 Introducción

En los inicios del proceso de desarrollo, no se contaba con un método que permitiera medir el estado de avance del software. Con el transcurso del tiempo y la puesta en práctica de este método se fueron notando cambios satisfactorios en el desarrollo y avance de los proyectos.

En el presente capítulo, se muestran los principales conceptos y sistemas informatizados, vinculados al campo de acción. Se brinda una breve panorámica de las tendencias y tecnologías actuales que permitieron la selección de las herramientas, lenguajes de programación y metodologías, que fueron utilizados durante el desarrollo de la propuesta de solución para el problema planteado.

1.2 Estimación

La estimación de la duración de las actividades, que conforman el desarrollo de software, es un tema que concierne a la gestión y control de proyectos para asegurar el éxito. La estimación, como actividad antecesora y constante de lo que luego será la planificación, proporciona valores aproximados de costos, tiempos y esfuerzos que se necesitarán para el desarrollo del producto a construir (Fuentes, Rodríguez 2008).

Esta aproximación requiere experiencia, acceder a una buena información histórica y el coraje de confiar en predicciones cuantitativas (medidas), cuando todo lo que existe, son datos cualitativos; sin embargo, contar con dicha información en etapas tempranas de desarrollo permite, tomar decisiones importantes antes de llevarlo a cabo, es decir, que de cierta forma permite predecir el futuro (PRESSMAN 1998). La primera tarea en la gestión de proyectos, es la estimación (CAPUCHINO 1996), la cual presenta varias técnicas para el desarrollo de software, que tienen en común los siguientes aspectos:

- ✓ Se ha de establecer de antemano el ámbito del proyecto.
- ✓ Como base para la realización de estimaciones, se usan las métricas del software.
- ✓ El proyecto se desglosa en partes más pequeñas, que se estiman individualmente.

La estimación es siempre difícil de realizar por diversas razones, algunas de ellas son:

- ✓ No existe un modelo de estimación universal.
- ✓ Son muchas las personas implicadas en el proyecto, desde la alta dirección de la empresa a los ejecutivos del proyecto, que precisan de las estimaciones.
- ✓ La utilidad de una estimación, varía con la etapa de desarrollo en que se encuentra el proyecto.
- ✓ Las estimaciones precisas, son difíciles de formular, sobre todo al inicio del proyecto.
- ✓ La estimación, suele hacerse superficialmente, sin tener en cuenta el esfuerzo necesario para hacer el trabajo.
- ✓ La rapidez del cambio de las metodologías y las tecnologías no permiten la estabilización del proceso de estimación.
- ✓ Los estimadores, pueden no tener experiencias sobre aquello que pretenden estimar.
- ✓ El estimador, suele hacer la estimación en función del tiempo que a él le llevaría en realizar el trabajo, sin tener en cuenta la experiencia y formación de la persona que realmente lo realiza.
- ✓ El estimador, tiende a reducir en alguna medida sus estimaciones para hacer más aceptable su oferta.

Una de las formas más apropiadas de hacer estimaciones o pronósticos es mediante normas o estándares.

1.2.1 ¿Qué se entiende por norma o estándar?

Según ISO, un estándar es: “un conjunto de acuerdos documentados que contienen especificaciones técnicas u otros criterios precisos, para ser usados constantemente, como reglas, lineamientos o definiciones de características. Todo esto con la finalidad de asegurar que los materiales, productos, procesos y servicios son óptimos para su propósito”.

Un estándar es el documento aprobado por consenso por un organismo reconocido, que proporciona reglas, pautas y/o características para uso común, con el objeto de obtener un óptimo nivel de resultados en un contexto dado. Una norma (tecnología) o estándar es toda actividad documentada que norma el comportamiento de un grupo de personas. Los estándares dan los medios para que todos los procesos, se realicen siempre de la misma forma, mientras surjan ideas para mejorarlos. Son

la guía para la productividad y la calidad. Es decir, las normas son: un modelo, un patrón, ejemplo o criterio a seguir. Cubren todos los aspectos técnicos relacionados con la información, su producción y su gestión. Son coherentes y consistentes. Deben ser desarrolladas por comités técnicos bajo supervisión de especialistas.

1.3 Entidades para la estandarización

Han sido muchos los departamentos de universidades, organismos de normalización o investigación nacionales o internacionales, sociedades de profesionales, departamentos de defensa, departamentos de calidad y procesos de empresas, los que han ido generando normas y estándares, los mismos se denotan a continuación:

1.3.1 IEEE (Instituto de Eléctrica e Ingenieros Electrónicos)

El IEEE (Institute of Electrical and Electronics Engineers), el Instituto de Ingenieros Eléctricos y Electrónicos, una asociación técnico-profesional mundial dedicada a la estandarización, entre otras cosas, es la mayor asociación internacional formada por profesionales de las nuevas tecnologías, como ingenieros eléctricos, en sistemas e ingenieros en telecomunicación. Surgió en 1963 con la fusión del AIEE (Instituto Americano de Ingenieros Eléctricos) y el Instituto de Ingenieros de Radio (IRE). El IEEE es una autoridad líder y de máximo prestigio en las áreas técnicas derivadas de la eléctrica original: desde ingeniería computacional, tecnologías biomédica y aeroespacial, hasta las áreas de energía eléctrica, control, telecomunicaciones y electrónica de consumo, entre otras. Su trabajo es remover la creatividad, el desarrollo y la integración, compartir y aplicar los avances en las tecnologías de la información, electrónica y ciencias en general para beneficio de la humanidad y de los mismos profesionales.

1.3.2 SEI (Instituto de Ingenieros de Software)

El Departamento de Defensa de los Estados Unidos le preocupaba el establecimiento de métricas para identificar a los contratistas potenciales referentes a esta rama, por esta razón creó el Software Engineering Institute (SEI). El SEI es un centro de investigación y desarrollo sostenido por el Departamento de Defensa y operado por la Universidad Carnegie Mellon. El propósito principal de este Instituto es ayudar a las empresas a buscar mejoras en cuanto a la ingeniería de software basadas en Métricas.

1.3.4 ISO (Organización Internacional de Estandarización)

La Organización Internacional para la Estandarización o International Organization for Standarization (ISO), es fundada en 1947. Es una organización internacional no gubernamental que produce normas internacionales industriales y comerciales. Dichas normas se conocen como normas ISO. La misión de la ISO es promover el desarrollo de la estandarización y las actividades con ella relacionadas en el mundo, con el objetivo de facilitar el intercambio de servicios y bienes, y para promover la cooperación en la esfera de lo intelectual, científico, tecnológico y económico. Se basa principalmente en estimaciones y normas estandarizadas.

Para lograr una buena calidad del producto es necesario identificar las mediciones y los criterios que serán utilizados para identificar el nivel deseado de la calidad y determinar si se está alcanzando, siendo los métodos de estimación y las métricas de software una de las vías más utilizadas para cuantificar y valorar resultados en la industria del software.

1.4 Métricas de software

Cuando se planifica un proyecto de software es esencial hacer estimaciones: del esfuerzo humano requerido, de la duración cronológica del proyecto y del costo, a través de las mediciones de software. Para lograr una buena calidad del producto, es necesario identificar las mediciones y los criterios que serán utilizados para identificar el nivel deseado de la calidad y determinar si se está alcanzando, siendo los métodos de estimación y las métricas de software una de las vías más utilizadas para cuantificar y valorar resultados. Las mediciones de software, es una de las áreas en la ingeniería de software donde se ha investigado desde hace más de 30 años conociéndose también como métricas de software. Existe una confusión en la utilización de los términos métricas de software y mediciones de software, sin embargo, en la literatura los términos métrica, medida o medición son usados como sinónimos indistintamente. (ZUSE 1995)

“Las métricas son un buen medio para entender, monitorizar, controlar, predecir y probar el desarrollo del software y los proyectos de mantenimiento” (L.Briand 1996). Las métricas de software proveen la información necesaria para la toma de decisiones técnicas.

En la Figura 1 se ilustra una extensión de esta definición para incluir los servicios relacionados al software como la respuesta a los resultados del cliente.

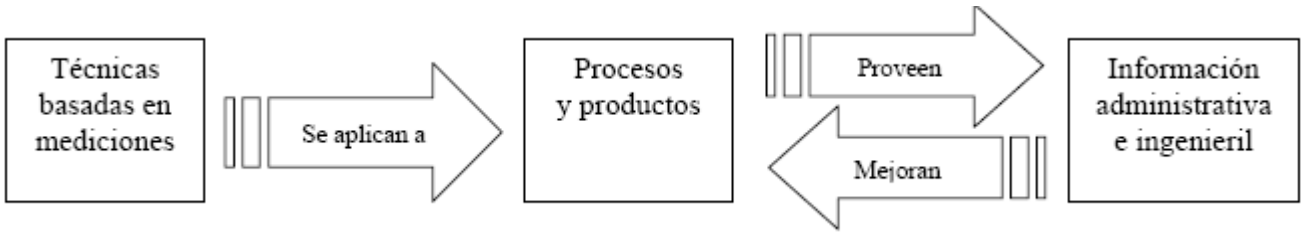

Figura # 1: Métricas de Software.

Las métricas son la maduración de una disciplina, ayudan a la evaluación de los modelos de análisis y de diseño, proporcionan una indicación de la complejidad de diseños procedimentales y de código fuente posibilitando diseño de pruebas más efectivas. Es por eso que propone un proceso de medición, el cual se puede caracterizar por cinco actividades:

- ✓ Formulación: La obtención de medidas y métricas del software apropiadas para la representación de software en cuestión.
- ✓ Colección: El mecanismo empleado para acumular datos necesarios para obtener las métricas formuladas.
- ✓ Análisis: El cálculo de las métricas y la aplicación de herramientas matemáticas.
- ✓ Interpretación: La evaluación de los resultados de las métricas en un esfuerzo por conseguir una visión interna de la calidad de la representación.
- ✓ Realimentación: Recomendaciones obtenidas de la interpretación de métricas técnicas trasmitidas al equipo de software. (NUSENOFF and BUNDE 1993)

1.4.1 Características de una buena métrica

Los principios fundamentales que deben seguir las métricas son (RUP 2003):

- ✓ Las métricas deben ser simples, objetivas, fáciles de coleccionar, fáciles de interpretar y difíciles de malinterpretar.
- ✓ La colección de las métricas debe ser automática y no intrusiva, o sea, no interferir en las actividades de los desarrolladores.

- ✓ Deben contribuir a la evaluación de la calidad temprana en el ciclo de vida, cuando los esfuerzos por mejorar la calidad del software son efectivos.
- ✓ Los valores absolutos y las tendencias de las métricas, deben ser usados activamente por el personal administrativo y el personal ingenieril, para comunicar progreso y calidad en un formato coherente.
- ✓ Empírica e intuitivamente persuasiva, la métrica debería satisfacer las nociones intuitivas del ingeniero de software sobre el atributo del producto en evaluación (por ejemplo: una métrica que mide la cohesión de un módulo debería aumentar su valor a medida que crece el nivel de cohesión).
- ✓ Independiente del lenguaje de programación: las métricas deberían apoyarse en el modelo de análisis, modelo de diseño o en la propia estructura del programa. No deben depender de la sintaxis o semántica del lenguaje de programación.

1.4.2. Clasificación de las métricas del software

Las métricas del software pueden ser clasificadas de la siguiente forma:

- ✓ **Métricas técnicas**: se centran en las características del software como: la complejidad lógica, el grado de modularidad, medir la estructura del sistema, el cómo está hecho, es decir, están centradas en las características del software más que en su proceso de desarrollo.
- ✓ **Métricas de calidad**: proporcionan una indicación de cómo se ajusta el software a los requisitos implícitos y explícitos del cliente es decir, cómo se va a medir para que el sistema se adapte a los requisitos que exige el cliente.
- ✓ **Métricas de productividad**: referidas al rendimiento del proceso de desarrollo como función del esfuerzo aplicado. Se centran en el rendimiento del proceso de la ingeniería del software.
- ✓ **Métricas orientadas al tamaño:** están orientadas para saber en qué tiempo se termina el software y cuántas personas se necesitan. Son medidas directas al software y al proceso por el cual se desarrolla.

- ✓ **Métricas orientadas a la función**: son medidas indirectas del software y del proceso por el cual se desarrolla. Las métricas orientadas a la función se centran en la funcionalidad o utilidad del programa.

- ✓ **Métricas orientadas a la persona**: proporcionan medidas e información sobre la forma que las personas desarrollan el software de computadoras y sobre todo el punto de vista humano de la efectividad de las herramientas y métodos.

1.5 Métrica para el avance de la fase de implementación

Basados en una propuesta implementada en el proyecto SIGET (Sistema Gestión Penitenciario) se explica en qué consiste la métrica a desarrollar.

1.5.1 Formulación de la propuesta

Para conocer el estado del software durante la implementación es necesario medir el avance a lo largo de este flujo. Este puede expresarse como el por ciento de cumplimiento de las tareas de la implementación. Por esta razón, para medir el avance es necesario, considerar las tareas de implementación de acuerdo al flujo de trabajo impuesto por la arquitectura del software, su complejidad y los casos de uso a implementar.

De acuerdo a la arquitectura definida para desarrollar una solución de software se definen los roles y actividades que se requieren para la implementación del producto. En una arquitectura por capas podrían ser necesarios, roles específico para implementar cada una de las capas arquitectónicas y en función de la complejidad de estas tareas, el tiempo que se requiere para implementar un caso de uso puede variar.

El proyecto de desarrollo que se toma como base para formular la propuesta utilizó una arquitectura de tres capas sobre frameworks de peso ligero (Spring, Hibernate) y sobre el diseño arquitectónico se definió un conjunto de tareas de implementación.

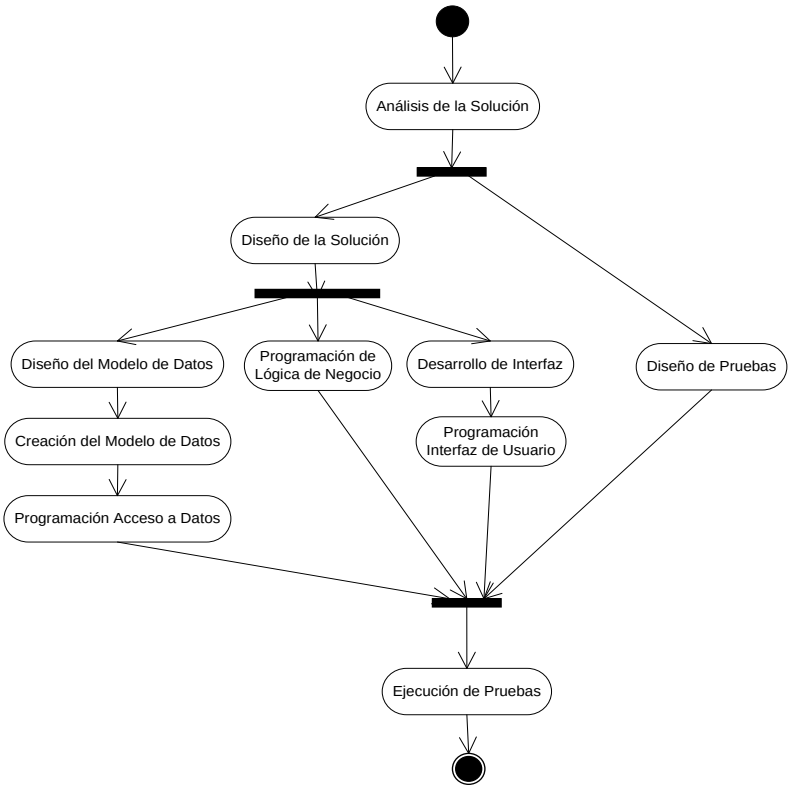

Figura # 2: Secuencia de tareas que definen el flujo de implementación.

A cada una de las tareas se le asignó un peso, que representa el por ciento del tiempo total que toma su ejecución como parte de la implementación de un caso de uso. Nótese por tanto, que la suma de todos los pesos resultaría un 100%. Estos pesos se fueron ajustando a lo largo de las iteraciones del desarrollo para que expresaran con mayor fidelidad la complejidad de las tareas y el esfuerzo en tiempo requerido para ejecutarlas.

Tarea de implementación	***Descripción***	***Involucrados***	**Peso**
Análisis de la solución	Consiste en el análisis de la documentación resultante de la captura de requisitos para conocer a profundidad el sistema que se desea construir. Concluye una vez que los involucrados en la implementación de las funcionalidades tengan claridad de lo que se desea implementar.	Analistas, Diseñador, Programadores, Diseñador de base de datos,	4%

		Diseñador de pruebas	
Diseño del modelo de datos	Partiendo de un modelo lógico obtenido durante la captura de requisitos, se definen las estructuras de base de datos que darán soporte de persistencia a la solución de software.	Analista, Diseñador, Diseñador de base de datos	7%
Creación del modelo de datos	Creación de las estructuras y objetos de base de datos en el sistema de gestión seleccionado.	Administrador de base de datos	2%
Desarrollo de la interfaz	Construcción del prototipo de interfaz de usuario a partir del prototipo resultante de la captura de requisitos y basado en las pautas de diseño establecidas. Constituye el punto de partida para la programación de interfaz	Realizador gráfico, Analista	11%
Diseño de la solución	Es considerada una macro actividad que agrupa actividades relacionadas con varios roles y áreas del desarrollo: ➢ Diseño de interfaces de negocio: Consiste en crear las interfaces que encapsulan la lógica de negocio sobre una o varias entidades. ➢ Diseño de entidades de dominio: Definir las entidades de dominio involucradas en la solución del problema. ➢ Diseño de interfaces de acceso a datos: Consiste en crear las interfaces de los DAOs, así como los métodos necesarios para responder a las funcionalidades. ➢ Diseño de la capa de presentación: Se definen las vistas y el flujo de navegación, las posibles peticiones del usuario y los controladores que las atienden. Finalmente se definen componentes del cliente, como son clases JavaScript, documentos HTML, imágenes, etcétera.	Diseñadores, Programadores de Interfaz de usuario, Programadores de acceso a datos	10%

Programación de acceso a datos	Consiste en implementar la capa de acceso a datos. Tiene que estar creada la base de datos y las entidades de dominio. Se concluye cuando se hayan realizado las siguientes actividades: ➢ Realizar los ficheros de mapeo: Consiste en crear los ficheros de mapeo del Framework Hibernate. ➢ Programación de interfaces: Consiste en crear las clases que implementan las interfaces de los DAOs. ➢ Realizar pruebas unitarias: Realizar pruebas unitarias a las implementaciones de los DAOs.	Programador de acceso a datos	9%
Programación de lógica de negocio	Se implementan los métodos definidos para cada una de las interfaces de los managers, ajustándose a las funcionalidades previstas.	Diseñador	9%
Programación de interfaz	Implementar los controladores de spring, validaciones, programar la lógica del cliente, realizar pruebas al funcionamiento de la interfaz.	Programador de interfaz de usuario	33%
Pruebas de calidad	Considerada una macro actividad debido al grupo de actividades que deben desarrollarse para darle cumplimiento. Estas actividades son: ➢ Estudio de la documentación: Consiste en el estudio de la documentación generada por el analista durante la captura de requisitos, hasta lograr ser un especialista del negocio. ➢ Diseño de Casos de Prueba: Construcción de todos los posibles caminos de ejecución, o escenarios, de cada caso de uso. Se obtiene como resultado un listado final con los casos de prueba identificados a partir de los posibles escenarios, los resultados esperados para cada caso y las condiciones o valores requeridos para la ejecución de los distintos escenarios. ➢ Diseño de Juego de Datos: Para cada escenario de	Diseñador de casos de prueba, Probador, Analista	14%

	caso de prueba se identifican sus valores de prueba, es decir, se precisan los datos de prueba. ➢ Ejecución de las Pruebas: Garantiza la calidad del sistema desarrollado, verificando que todas las funcionalidades han sido correctamente implementadas. No concluye hasta que las funcionalidades sean liberadas una vez resueltos los defectos detectados.		

Tabla # 1: Descripción y peso de las tareas de implementación.

La realización de estas tareas hace posible que se desarrollen los propósitos de la implementación. Desde el punto de vista de los casos de uso, ninguno habrá concluido hasta tanto no hayan sido realizadas cada una de las actividades anteriores.

Por este motivo a cada una de las tareas definidas se le otorga, para cada caso de uso, un estado de completamiento entre 0 y 1: 0 si aún no se ha realizado y 1 si ya la tarea se encuentra terminada. Un valor decimal que se encuentre entre estos dos números representa el progreso de la tarea dado por el responsable de ejecutarla y basado en las actividades definidas dentro de su flujo de trabajo que aún le restan por terminar (Fuentes, Rodríguez 2008).

Así el estado de avance de un caso de uso puede definirse como:

$$\textbf{Avance por Caso de Uso } (ACU_i) = \sum_{j=1}^{n} (\textbf{Estado de completamiento}_{ij} \times \textbf{Peso de la tarea }_j)$$

Donde: Peso de la tarea corresponde al valor asignado a la tarea (j).

Otra dimensión a analizar para establecer el avance de la implementación es la complejidad de los caso de uso a desarrollar. La existencia de casos de uso de diferente complejidad implica que el tiempo de realización de una tarea no sea el mismo para todos los casos. Por ello, para medir el avance de un módulo conformado por varios casos de uso, sería erróneo calcularlo como el promedio del avance de estos.

Los casos de uso se clasifican según su complejidad en: alto, medio o bajo. Esta clasificación es concedida por el Diseñador en función de los siguientes aspectos:

- ✓ Número de elementos de datos.

- ✓ Lógica de negocio.
- ✓ Interfaces externas, dígase dispositivos u otro sistema.

En un módulo solo se presenta una de las combinaciones posibles de complejidad de los casos de uso que lo conforman. Para cada combinación se estableció un peso específico, valor llamado Peso de la complejidad del caso de uso (Pccu) y es el que permite ponderar la diferencia de esfuerzo para desarrollar los casos de uso de diferente complejidad (Fuentes, Rodríguez 2008).

Caso	***Combinación posible***	***Peso CU complejidad alto (Pccu $_{alta}$)***	***Peso CU complejidad medio (Pccu $_{media}$)***	**Peso CU complejidad bajo (Pccu $_{baja}$)**
1	Existen casos de uso con complejidad alta, media y baja	0,5	0,3	0,2
2	Todos los casos de uso tienen complejidad alta	1	0	0
3	Existen casos de uso con complejidad alta y media	0,6	0,4	0
4	Existen casos de uso con complejidad alta y baja	0,8	0	0,2
5	Todos los casos de uso tienen complejidad media	0	1	0
6	Existen casos de uso con complejidad media y baja	0	0,6	0,4
7	Todos los casos de uso tienen complejidad baja	0	0	1

Tabla # 2: Relación de los Pesos de la complejidad del caso de uso según las combinaciones posibles.

Utilizando las posibles combinaciones de complejidad de los casos de uso que componen un módulo y partiendo del valor del avance de cada caso de uso, es posible determinar el avance total del módulo. Esto se obtendría mediante la suma del producto entre el Peso de complejidad y el promedio de avance de los casos de uso para cada uno de los niveles de complejidad, lo que quedaría representado por la siguiente expresión:

Avance por Módulo (AM)= Pccu $_{alto}$ x PAcu $_{alto}$ + Pccu $_{medio}$ x PAcu $_{medio}$ + Pccu $_{bajo}$ x PAcu $_{bajo}$

Donde PAcu alto es el promedio de avance de los caso de uso de complejidad alta, PAcu medio es el promedio de avance de los caso de uso de complejidad media y PAcu bajo es el promedio de avance de los caso de uso de complejidad baja.

Para determinar el avance de la implementación, se puede considerar aceptable, el promedio de avance de cada módulo.

$$Avance\ del\ Proyecto = \frac{\sum_{i=1}^{n} Avance\ del\ Módulo}{n},$$ **donde *n* representa la cantidad de módulos.**

La siguiente tabla muestra un resumen, para una semana dada, de la situación del proyecto, donde se puede observar la lista de los módulos que corresponden a la iteración y la cantidad de funcionalidades (casos de uso) a implementar en cada uno de ellos.

Total de semanas	**9**
Semana actual	**8**
Semanas que quedan	**1**

No	Módulos	Funcionalidades	Avance	Semanas para terminar	Estado del desarrollo
1	Requisas y decomisos	22	98%	0,2	En tiempo
2	Ubicación	3	98%	0,2	En tiempo
3	Pertenencias	7	96%	0,3	En tiempo
4	Novedades	10	97%	0,3	En tiempo
5	Medidas disciplinarias	6	90%	0,8	En tiempo
6	Capacidades	4	97%	0,3	En tiempo
7	Armamento	17	85%	1,4	Atrasado
8	Dotación del interno	5	95%	0,4	En tiempo
9	Educación	12	92%	0,7	En tiempo
10	Trabajo	6	86%	1,3	Atrasado
11	Deporte y Cultura	20	91%	0,8	En tiempo
12	Evaluaciones Técnicas	6	97%	0,3	En tiempo
13	Reportes	20	65%	4,4	Atrasado
		138	**91%**	**0,8**	En tiempo

Figura # 3: Resumen del avance por módulos para una semana dada.

.Se puede observar que la tabla expresa las semanas que necesita el módulo para terminar. De los datos anteriores se puede establecer la siguiente relación:

$$\frac{Total\ Semanas}{100\%} = \frac{Semana\ Actual}{Avance\ por\ Módulo}$$

Para obtener las semanas que quedan para terminar, suponiendo que se mantiene el ritmo de trabajo, se puede realizar la siguiente consideración:

$$Semanas\ para\ Terminar = \frac{Semana\ Actual \times 100\%}{Avance\ por\ Módulo} - Semana\ Actual$$

Nótese que existe otro campo en la tabla cuyo nombre es *Estado de Desarrollo*. El valor por módulo de este campo depende totalmente de los resultados arrojados en el cálculo anterior. Si el resultado obtenido en Semanas por Terminar supera las semanas que quedan para la fecha de entrega establecida por el cliente, el módulo está Atrasado, esto significa que deben tomarse decisiones derivadas de esta información que permitan efectuar la entrega en la fecha acordada. En el caso contrario, es decir, si el resultado obtenido es menor que las semanas que quedan para la fecha de entrega establecida entonces el módulo está En Tiempo (Fuentes, Rodríguez 2008).

1.5.2 Análisis de los resultados.

La forma más cómoda y visual de lograr un seguimiento del avance por módulos, es a través del análisis del gráfico del estado de avance. Este gráfico muestra claramente en qué estado de avance se encuentran los módulos y por tanto advierte al jefe del proyecto en cuáles tiene que volcar los mayores esfuerzos.

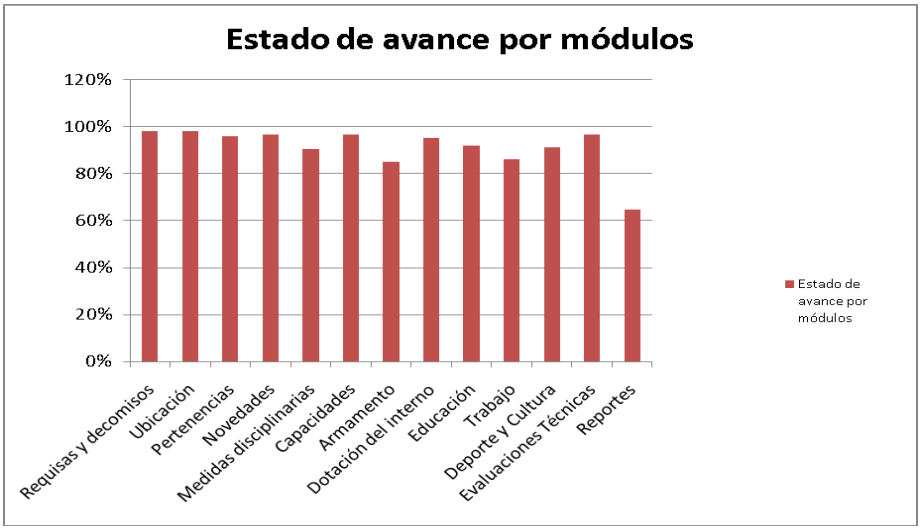

Figura # 4. Estado de avance por módulos.

Se puede visualizar además como se ha comportado semanalmente el avance de cada uno de los módulos. En la siguiente figura se muestra un histórico por semanas del progreso de los diferentes módulos y total del proyecto durante el proceso de implementación del software.

		semana1	semana2	semana3	semana4	semana5	semana6	semana7	semana8	semana 9
No	Módulos	7-3-2008	14-3-2008	21-3-2008	28-3-2008	4-4-2008	11-4-2008	18-4-2008	25-4-2008	2-5-2008
1	Requisas y decomisos	24%	34%	57%	62%	81%	89%	92%	92%	98%
2	Ubicación	13%	41%	42%	42%	79%	91%	93%	93%	98%
3	Pertenencias	48%	60%	77%	81%	93%	95%	96%	96%	96%
4	Novedades	44%	46%	53%	57%	73%	77%	85%	86%	97%
5	Medidas disciplinarias	30%	30%	41%	41%	50%	66%	82%	86%	90%
6	Capacidades	25%	47%	67%	75%	84%	92%	93%	93%	97%
7	Armamento	7%	18%	18%	19%	34%	58%	71%	86%	85%
8	Dotación del interno	11%	32%	55%	56%	93%	94%	95%	95%	95%
9	Educación	18%	48%	61%	76%	89%	89%	91%	92%	92%
10	Trabajo	15%	49%	49%	55%	77%	80%	86%	86%	86%
11	Deporte y Cultura	37%	61%	76%	79%	80%	85%	85%	91%	91%
12	Evaluaciones Técnicas	26%	53%	78%	83%	88%	91%	91%	91%	97%
13	Reportes					26%	31%	46%	65%	65%
		25%	43%	56%	61%	73%	80%	85%	89%	91%

Figura # 5. Histórico del avance por semanas.

Una representación que resume el comportamiento semanal del avance de cada módulo quedaría de la siguiente manera:

El avance general del proyecto a partir de los datos de cada semana se representaría gráficamente de la siguiente manera.

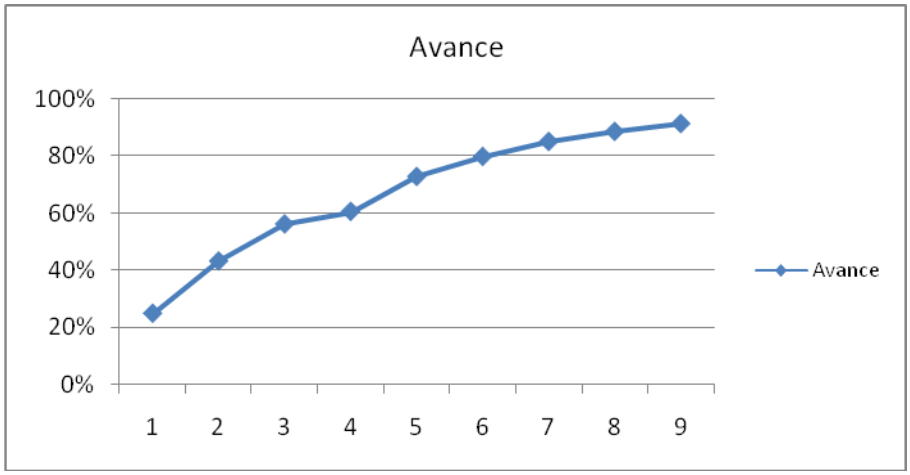

Figura # 6. Histórico del avance del proyecto.

Esta gráfica permite observar las tendencias a lo largo del tiempo. Nótese que durante las primeras semanas del desarrollo existe un avance relativo alto. Luego se observa un menor avance en la semana 4 respecto a la semana 3, período durante el cual se efectuaban en la Universidad los IV Juegos Deportivos, los que afectaron como mostró la métrica, el proceso de desarrollo, debido a que la mayoría de los miembros del equipo de proyecto son estudiantes. Luego se observa nuevamente un avance relativamente alto hasta aproximadamente el 80% punto en el cual comienza la etapa de pruebas y donde el avance fue más lento.

Un comportamiento ideal estaría representado por una línea recta en la gráfica del avance general del proyecto; pero esto solo es posible bajo condiciones ideales, las cuales son muy difíciles de obtener debido entre múltiples factores, a que los miembros del grupo de desarrollo no tienen la experiencia suficiente y no se dedican al trabajo a tiempo completo. Además las variables de ajustes introducidas al método deben ser evaluadas para ganar en exactitud. Un ejemplo de esto lo demuestra la desaceleración del avance durante la etapa de pruebas, lo que podría indicar que el peso dado a esta fase debería ser mayor con respecto al otorgado al resto de las tareas de implementación.

Lo anterior determina que el método propuesto no es exacto, ya que en él influyen múltiples factores y las variables son cambiantes; sin embargo, su aplicación arrojó resultados que en la práctica no se alejaron de la realidad y el proyecto cumplió en el tiempo previsto sus compromisos productivos, en parte porque pudieron ser tomadas a tiempo las medidas gerenciales pertinentes cuando se detectaron desviaciones.

1.6 Tendencias y tecnologías actuales

Para la realización de cualquier trabajo es preciso hacer un estudio de las tendencias y tecnologías similares y existentes a nivel mundial con el fin de seleccionar las necesarias para realizar un trabajo que presente la calidad exigida, a continuación se presenta una breve panorámica de dichas tendencias y tecnologías.

1.6.1 Metodologías a usar

En la ingeniería de software de un proyecto, la metodología de desarrollo de software define quién debe hacer qué, cuándo y cómo debe hacerlo. Una metodología es un proceso que se encargada de elaborar estrategias de desarrollo de software, entre ellas se encuentran las de tipo Agiles y las de tipo Tradicionales.

1.6.1.1 Metodologías ágiles

Durante los últimos años han surgido las llamadas metodologías ágiles. Estas aportan nuevas técnicas y métodos de trabajo para el desarrollo de cada etapa de un software. En general esta metodología hace un balance entre los procesos y el esfuerzo, ya que tratan de centrarse en las cuestiones necesarias sin perderse en las burocráticas. Estas metodologías tienen como prioridad satisfacer al cliente mediante tempranas y continúas entregas del software [8]. Dentro de las más mencionadas de estas metodologías, se encuentran Extreme Programing (XP) y Microsoft Solution Framework (MSF).

1.6.1.2 Metodologías Tradicionales

Las Metodologías Tradicionales imponen una disciplina de trabajo sobre el proceso de desarrollo del software, haciendo énfasis en la previa y total planificación del proyecto a desarrollar. Una vez detallado todo el trabajo a realizar es que comienza el ciclo de desarrollo del software. RUP es la metodología más conocida dentro de esta clasificación.

- **Rational Unified Process (RUP)**

La metodología RUP es el resultado de varios años de desarrollo y uso práctico en el que se han unificado técnicas de desarrollo, a través del UML, y trabajo de muchas metodologías utilizadas por los clientes. Como proceso en sí, define como sus elementos más importantes [1]:

Los trabajadores, que indican el “quién”. Define el comportamiento y responsabilidades (rol) de un individuo, grupo de individuos, sistema automatizado o máquina, que trabajan en conjunto como un equipo.

- ✓ Actividades, que definen el “cómo”, una tarea que tiene un propósito claro, es realizada por un trabajador y manipula elementos.
- ✓ Los artefactos, que no son más que el “qué”, productos tangibles del proyecto que son producidos, modificados y usados por las actividades (modelos, elementos dentro del modelo, código fuente y ejecutables). (Rumbaugh, James.)

Por último el Flujo de actividades, que identifica el “Cuándo”, secuencia de actividades realizadas por trabajadores y que produce un resultado de valor observable.

RUP, tiene como características fundamental en su ciclo de vida ser:

- ✓ Dirigido por casos de uso.
- ✓ Centrado en la arquitectura.
- ✓ Iterativo e Incremental.

Divide en 4 fases el desarrollo del software, como se muestra en la Figura 7.

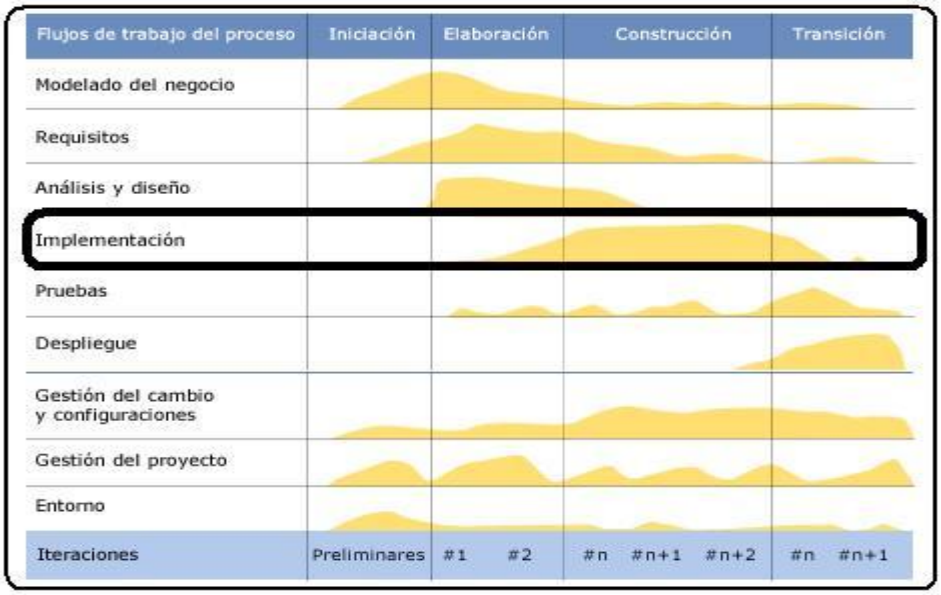

Figura # 7: RUP en dos dimensiones.

- ✓ Inicio

- ✓ Elaboración
- ✓ Construcción
- ✓ Transición

Cada una de estas etapas es desarrollada mediante el ciclo de iteraciones, la cual consiste en reproducir el ciclo de vida en cascada a menor escala. Los objetivos de una iteración se establecen en función de la evaluación de las iteraciones precedentes. Es el más usado a nivel mundial por su completa forma de organización y resultados.

El ciclo de vida que se desarrolla por cada iteración, es llevado bajo dos disciplinas:

Disciplina de Desarrollo:

- ✓ Ingeniería de Negocios
- ✓ Requerimientos
- ✓ Análisis y Diseño
- ✓ Implementación
- ✓ Pruebas

Disciplina de Soporte:

- ✓ Configuración y administración del cambio
- ✓ Administrando el proyecto
- ✓ Ambiente
- ✓ Distribución

La metodología RUP se ha unificado a través del UML, a la hora de construir sistemas por medio de conceptos orientados a objetos la forma más factible y usada es a través del Lenguaje Unificado de Modelado.

1.7 Lenguaje Unificado de Modelado (UML)

UML, son las siglas de Lenguaje Unificado de Construcción de Modelos, fue creado por los miembros de Rational: Grady Booch, James Rumbaugh e Iván Jacobson. Es un lenguaje para describir principalmente sistemas orientados a objetos independientes de cualquier lenguaje de programación específico, es simple de aprender, bastante flexible y consistente desde el planeamiento hasta el despliegue. Los beneficios de usar UML incluyen la trazabilidad mejorada, inteligibilidad entre los usuarios y un mantenimiento realmente simplificado. Sirve para el modelado completo de sistemas complejos, tanto en el diseño de los sistemas software como para la arquitectura hardware donde se ejecuten. Proporciona valor conceptual y visual, facilidad para extender el lenguaje y para representar elementos particulares a determinados tipos de aplicaciones. Mediante este lenguaje se hace posible además de la visualización, la especificación, construcción y documentación de sistemas que involucran gran cantidad de software con tecnología orientada a objetos. [2]

El UML está compuesto por diversos elementos gráficos, que se combinan para conformar diagramas y proporciona un estándar que permite al analista de sistemas, generar un anteproyecto de varias facetas que sean comprensibles para los clientes, desarrolladores y todos aquellos que estén involucrados en el proceso de desarrollo.

La finalidad de los diagramas UML, es presentar diversas perspectivas de un sistema, a las cuales se les conoce como modelo. Es importante destacar que un modelo UML describe lo que supuestamente hará un sistema, pero no dice cómo implementar dicho sistema.

A través de un conjunto de símbolos y diagramas del UML los creadores de sistemas pueden generar diseños que capturen sus ideas en una forma convencional y fácil de comprender para comunicarlas a otras personas que estén involucradas en el proceso de desarrollo de los sistemas. El modelo gráfico de UML tiene un vocabulario que se presenta en la Figura 3 a continuación:

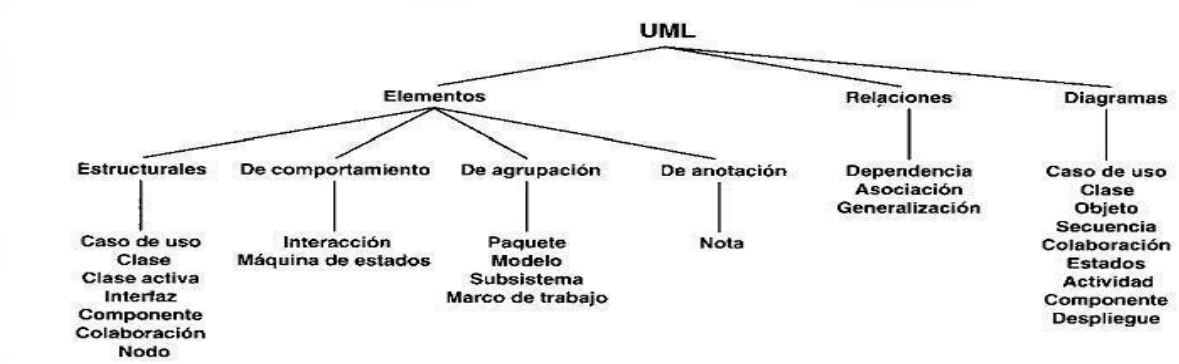

Figura # 8: Modelo gráfico de UML.

Existen herramientas que nos pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software, en tareas como el proceso de realizar un diseño del proyecto, calculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras, estas son las herramientas CASE, las cuales son muy utilizadas a la hora de realizar el modelado con UML, abarcan todos los pasos del proceso de software, y también aquellas actividades generales que se aplican a lo largo de todo el proceso.

1.8 Herramientas CASE de modelado con UML

Las herramientas CASE como su nombre lo indica “Computer Aided Software”, en español: “Ingeniería de Software Asistida por Computadora”, constituyen diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software, reduciendo el coste de las mismas en términos de tiempo y de dinero. Son un complemento de la caja de herramientas del ingeniero del software, además ayudan a asegurar que la calidad sea algo diseñado antes de llegar a construir el producto. [3]

Todas las herramientas CASE prestan soporte a un lenguaje de modelado para acompañar la metodología y un alto porcentaje de ellas soportan UML, debido al alto grado de aceptación que ha tenido este lenguaje desde su lanzamiento.

Entre las herramientas CASE más conocidas y aplicadas y se encentran:

- ✓ Visual Paradigm.
- ✓ Rational Rose Enterprise Edition.

1.8.1 Visual Paradigm

Es una herramienta CASE que utiliza "UML": como lenguaje de modelado. Es una Suite de herramientas CASE, dotada de una buena cantidad de productos o módulos para facilitar el trabajo durante la confección de un software, lo cual garantiza la calidad del producto final.

Para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases.

Es fácil de usar, es amigable, soporta ingeniería inversa, generación de código, importación desde Rational Rose, exportación/importación XMI, generador de informes, editor de figuras, etc.

Posibilita crear un gran conjunto de artefactos de las distintas fases del desarrollo del software. Presenta sincronización entre diagramas de entidad-relación y diagramas de clases, interoperabilidad con otras aplicaciones, facilidad para redactar especificaciones de casos de uso del sistema, generación de documentos y de código ORM; disponibilidad de múltiples versiones, para cada necesidad, disponibilidad de integrarse en los principales IDEs y en múltiples plataformas.

1.8.2 Rational Rose Enterprise Edition

Rational Rose Enterprise Edition en una herramienta CASE que permite el diseño detallado del software y la generación de código fuente de programas y bases de datos e ingeniería inversa (obtención del diseño a partir del código fuente), basado en modelos con soporte UML. [4]

El navegador UML de Rational Rose, permite establecer una trazabilidad real entre el modelo (análisis y diseño) y el código ejecutable. Facilita el desarrollo de un proceso cooperativo en el que todos los agentes tienen sus propias vistas de información (vista de Casos de Uso, vista Lógica, vista de Componentes y vista de Despliegue), pero utilizan un lenguaje común para comprender y comunicar la estructura y la funcionalidad del sistema en construcción. [4]

Esta herramienta, propone la utilización de cuatro tipos de modelos para realizar el 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. [4]

Desde los pequeños programas hasta los sistemas más grandes poseen una estructura y un comportamiento que los hace clasificables según su "arquitectura".

1.9 La Arquitectura de Software (AS)

La Arquitectura, es una base de la aplicación, que es analizada desde varios puntos de vista. Representa la organización fundamental de un sistema, constituida en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución. Constituye la nueva visión que los desarrolladores tienen de los sistemas de software en la última década.

Este nuevo aspecto hace posible el estudio de sistemas ya implementados así como el desarrollo de nuevos, según la categoría del problema que resuelven y el tipo de "arquitectura". Los distintos niveles de abstracción de la funcionalidad de los sistemas, están asociados con diferentes aspectos y componentes de su "arquitectura". Esta asociación se manifiesta en forma clara, en las distintas etapas del proceso de desarrollo de software.

Existen distintos tipos de arquitecturas de sistemas que se clasificación en patrones de diseño y que tienen aplicaciones a lo largo del proceso de desarrollo, entre ellas se encuentran:

1.9.1 Arquitectura n-Capas

Con el objetivo de resolver los problemas de escalabilidad, disponibilidad, seguridad e integración que pueden presentar las aplicaciones compactas, se ha generalizado la división de las aplicaciones en capas. La capa que se agrega es la que surge de separar definitivamente las reglas de negocio de la de Acceso a Datos. Esta arquitectura trae consigo la ventaja de aislar la lógica de negocios de todo lo que tenga que ver con el origen de datos, de modo que ante cualquier eventual cambio, solo se deberá tocar un módulo específico, así como al momento de plantear la escalabilidad del sistema no se afrontarán grandes modificaciones. Dentro de este tipo de arquitectura una de las divisiones muy utilizada en el desarrollo de sistemas es la arquitectura en tres capas.

1.9.2 Arquitectura en tres capas

La arquitectura en tres capas cuenta con una interfaz gráfica que facilita al usuario el uso del sistema

(**Capa 1:** Capa de Presentación.), con una capa para centralizar la lógica de negocio

(**Capa 2:** Lógica de Negocio.) Y por último una capa que servirá para guardar los datos

(**Capa 3:** Base de Datos.).[5]

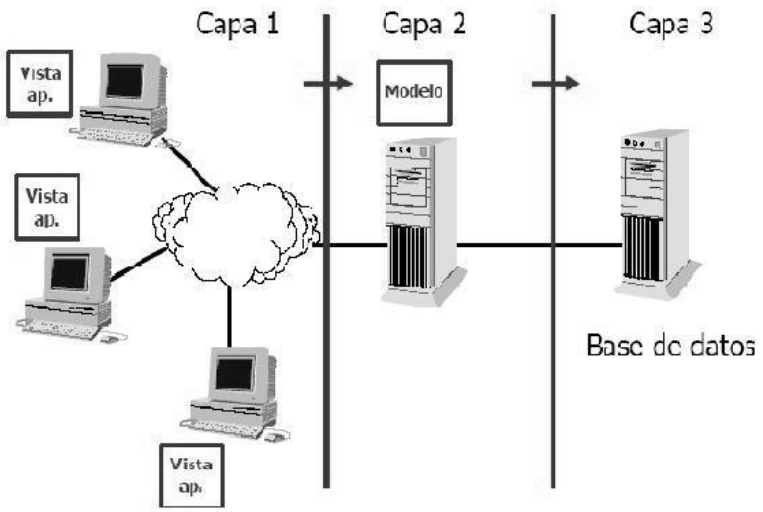

Figura # 9: Arquitectura en tres capas.

Esta arquitectura otorga algunas ventajas:

- ✓ Centralización de los aspectos de seguridad, que serían responsabilidad del modelo.
- ✓ No replicación de lógica de negocio en los clientes: esto permite que las modificaciones y mejoras sean automáticamente aprovechadas por el conjunto de los usuarios, reduciendo los costes de mantenimiento.
- ✓ Mayor sencillez de los clientes. [5]

1.9.3 Patrón de arquitectura Modelo Vista-Controlador (MVC)

El MVC (Model/View/Controller) fue introducido como parte de la versión Smalltalk-80 del lenguaje de programación Smalltalk. Fue diseñada para reducir el esfuerzo de programación necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos. Sus características principales son que el Modelo, las Vistas y los Controladores se tratan como entidades separadas; esto hace que cualquier cambio producido en el Modelo se refleje automáticamente en cada una de las Vistas.

MVC es un patrón de arquitectura de software clásico que está formado por tres niveles:

- ✓ **El modelo,** que representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.
- ✓ **La vista,** que transforma el modelo en una página Web que permite al usuario interactuar con ella.
- ✓ **El controlador,** que se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

En la Figura 5 se puede observar el patrón MVC.

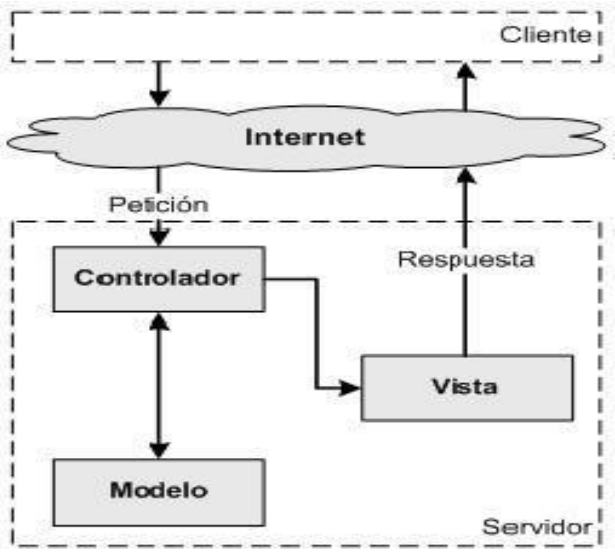

Figura # 10: Modelo Vista-Controlador.

El patrón MVC separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. Se ve frecuentemente en aplicaciones Web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página. En MVC, la capa de presentación está partida en controlador y vista. La principal separación es entre presentación y dominio; la separación entre vista y controlador es menos clara.

Este modelo de arquitectura presenta varias ventajas:

- ✓ Hay una clara separación entre los componentes de un programa; lo cual permite implementarlos por separado.

- ✓ Hay un API muy bien definido; cualquiera que use el API, podrá reemplazar el Modelo, la Vista o el Controlador, sin aparente dificultad.
- ✓ La conexión entre el Modelo y sus Vistas es dinámica; se produce en tiempo de ejecución, no en tiempo de compilación.

Al incorporar el modelo de arquitectura MVC a un diseño, las piezas de un programa se pueden construir por separado y luego unirlas en tiempo de ejecución. Si uno de los componentes, posteriormente, se observa que funciona mal, puede reemplazarse sin que las otras piezas se vean afectadas.

El patrón de arquitectura “Modelo Vista Controlador” provee las características apropiadas para darle uso en el diseño y desarrollo de la herramienta propuesta por las características antes expuestas.

En el mundo del desarrollo del software y más específico a la hora de guardar, manejar e interactuar con los datos son muchos los servidores Web y tipos de lenguajes de programación que se pueden utilizar por sus disimiles características, además que es indispensable el uso de un gestor de base de datos [6].

1.10 Los Servicios Web

Los Servicios Web (en inglés Web Service) son un conjunto de aplicaciones o de tecnologías con capacidad para interactuar en la Web. Estas aplicaciones o tecnologías intercambian datos entre sí mediante una colección de protocolos y estándares, con el objetivo de ofrecer unos servicios.

Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web. Un Servidor Web, es un tipo de software que funciona en un ordenador y maneja la entrega de los componentes de las páginas como respuesta a peticiones de los navegadores de los clientes. Básicamente, sirve contenido estático a un navegador, carga un archivo y lo sirve a través de la red al navegador de un usuario. Este intercambio es mediado por el navegador y el servidor que hablan el uno con el otro mediante el protocolo HTTP. Los archivos para cada sitio de Internet se almacenan y se ejecutan en el servidor. [7]

Entre los servidores web existentes el más apropiado para la realización de la herramienta a realizar es el servidor Web Apache por lo expuesto a continuación.

1.10.1 Servidor Web Apache

Apache, es el servidor Web hecho por excelencia. Presenta características que hacen que cada vez millones de servidores reiteren su confianza en este programa, destacándose: su configuración, robustez y estabilidad. Provee un alto grado de calidad y fortaleza para las implementaciones que utilizan el protocolo HTTP (de páginas Web). Comenzó a desarrollarse en febrero de 1995, por Rob McCool. [8]

Es una tecnología gratuita, de código fuente abierta, la cual permite hacer lo que se quiera con el código fuente, siempre que se les reconozca su trabajo. Corre en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal (Ejemplo: Linux, Novell, Unix y Windows). El servidor Web Apache:

- ✓ Tiene interfaz de autenticación con todos los sistemas.
- ✓ Facilita la integración como "plugins" de lenguajes de programación de páginas Web dinámicas.
- ✓ Tiene integración en estándar del protocolo de seguridad SSL.
- ✓ Provee una interfaz a todas las bases de datos.
- ✓ Tiene una potente configuración en la creación y gestión de logs, ya que 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.
- ✓ . Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurarlo para que ejecute un determinado script cuando ocurra un error en concreto.
- ✓ Es un servidor altamente configurable de diseño modular. Es muy sencillo ampliar las capacidades de este servidor, ya que se pueden escribir módulos para realizar determinadas funciones, lo que implica que haya gran cantidad de ellos disponibles para su utilización.

Los módulos de Apache pueden clasificarse en tres categorías:

- ✓ . **Módulos Base:** Módulo con las funciones básicas del Apache.

- ✓ **Módulos Multiproceso** (para manejar las peticiones)**:** Son los responsables de la unión con los puertos de la máquina, aceptando las peticiones y enviando a los hijos a atender a las peticiones. Se han diseñado varios módulos multiprocesos para cada uno de los sistemas operativos sobre los que se ejecuta el Apache, optimizando el rendimiento y rapidez del código.
- ✓ **Módulos Adicionales:** Cualquier otro módulo que le añada una funcionalidad al servidor. Simplemente hay que añadirle un módulo, de forma que no es necesario volver a instalar el software.

1.11 Lenguajes de programación para el desarrollo de Aplicaciones Web

Un lenguaje de programación, es un medio necesario para que las máquinas, en particular las computadoras, puedan interpretar las instrucciones que se les dan y para que se pueda además controlar su comportamiento. Consiste en un grupo de reglas sintácticas que definen su estructura y en un grupo de reglas semánticas que definen el significado de sus elementos. Estas reglas son utilizadas por el programador a través de las cuales crea un programa o subprograma. Por otra parte, las instrucciones que forman dicho programa son conocidas como código fuente. Existen los lenguajes del lado del servidor y los de lado cliente.

1.11.1 Lenguajes del lado servidor

Los lenguajes del lado del servidor, son aquellos que son reconocidos, ejecutados e interpretados por el propio servidor y que se envían al cliente en un formato comprensible para él. Son independientes del cliente por lo que es mucho menos rígido respecto al cambio de un navegador a otro o respecto a las versiones del mismo. Por otra parte, los scripts son almacenados en el servidor, quien los ejecuta y traduce a HTML, por lo que permanecen ocultos para el cliente. Este hecho puede resultar a todas luces una forma legítima de proteger el trabajo intelectual realizado, el lenguaje de programación ideal por ser la herramienta a realizar una aplicación Web es PHP, que sus características se muestran a continuación [6]

1.11.2 Hypertext Preprocessor (PHP)

PHP, es un lenguaje de programación interpretado de alto nivel, embebido en páginas HTML y ejecutado en el servidor, gratuito, independiente de plataforma, rápido ya que generalmente es utilizado como módulo de Apache, con una gran librería de funciones y mucha documentación.

La meta del lenguaje es permitir rápidamente a los desarrolladores la generación dinámica de páginas. Este lenguaje posee como una de sus características más importantes el soporte para:

- ✓ (a) La gran cantidad de gestores de bases de datos, entre los que pueden mencionarse InterBase, mSQL, MySQL, Oracle, Informix, PostgreSQL, etc.
- ✓ (b) La mayoría de los servidores Web de hoy día, incluyendo Apache, Microsoft Internet Information Server, Personal Web Server, Netscape y Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd y muchos otros.

Ofrece además la integración con las varias bibliotecas externas, que permiten que el desarrollador haga casi cualquier cosa desde generar documentos en PDF hasta analizar código XML y brinda una solución simple y universal para las paginaciones dinámicas del Web de fácil programación. Ofrece la integración con bibliotecas externas para el trabajo con gráficos, así como la creación y juego con los mismos, entre algunas de esas bibliotecas se encuentran:

- ✓ JPGraph, es una librería PHP que te permitirá crear gráficos matemáticos y estadísticos de manera sencilla y con absoluto control. Esta librería soporta una amplia variedad de tipos de gráficos para todas las necesidades.
- ✓ Libchart, es una librería PHP para crear gráficos de manera rápida y sencilla. Es compatible con PHP4/5.
- ✓ GraPHPico, es de código abierto, la información técnica está disponible en inglés y español y aunque solo tiene gráficos de: porcentaje, barras, torta y colores, es fácil de implementar y funciona muy bien.
- ✓ PostGraph, es una librería en PHP para crear diagramas de barras. Esto le permite crear gráficos profesionalmente en pocos pasos.

Su diseño elegante lo hace perceptiblemente más fácil de mantener y ponerse al día que el código comparables en otros lenguajes. Debido a su amplia distribución está perfectamente soportado por una gran comunidad de desarrolladores. Como producto de código abierto, goza de la ayuda de un gran grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y se reparen rápidamente.

El código se pone al día continuamente, con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP.

1.11.3 Lenguajes del lado cliente

Los lenguajes de lado cliente son aquellos que pueden ser directamente "digeridos" por el navegador y no necesitan un tratamiento previo. Son totalmente independientes del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio. HTML es un lenguaje simple de marcado utilizado para crear documentos de hipertexto para Web y JavaScript permite incluir macros en páginas Web. [9]

1.11.3.1 Hypertext Markup Language (HTML)

HTML, es un conjunto de símbolos o palabras que definen varios componentes de un documento Web; estos se pueden ver siempre dentro de las etiquetas "<", ">". Es un lenguaje simple de marcado utilizado para crear documentos de hipertexto para WWW. Fue creado por Tim Berners-Lee en 1991; en un principio con objetivos divulgativos, sin dar respuesta a todos los posibles usos que se le iba a dar y a todos los colectivos de gente que lo utilizarían en un futuro. Sin embargo, pese a esta deficiente planificación, si que se han ido incorporando modificaciones con el tiempo, estos son los estándares del HTML. [10]

En la actualidad, es el lenguaje que utilizan todos los navegadores para mostrar la información final. Algunas ventajas muy importantes que posee el HTML es la de ser muy fácil de aprender, lo que permite que cualquier persona, pueda enfrentarse a la tarea de crear una Web; la facilidad con que se pueden actualizar los contenidos y que permite utilizar estilos en formato CSS en las páginas para una mayor facilidad en su modificación.

HTML, no solo permite establecer hiperenlaces entre diferentes documentos, sino que es un lenguaje de descripción de independiente de la plataforma en que se utilice. Es decir, un documento HTML contiene toda la información necesaria sobre su estructura y con el usuario, es luego el browser que se utilice el responsable de asegurar que el documento tenga un aspecto coherente, independiente del tipo de máquina desde donde se acceda al documento. [10]

Para el uso de los diferentes lenguajes de programación Web que se proponen para el desarrollo de la herramienta se debe utilizar un entorno de desarrollo integrado y el que se adapta para el proceso de creación del sistema es Zend Studio.

1.11.3.2 JavaScript

JavaScript, es un lenguaje interpretado que permite incluir macros en páginas Web. Estas macros se ejecutan en el ordenador del visitante de nuestras páginas, y no en el servidor. JavaScript es también un lenguaje de scripts desarrollado por Netscape para incrementar las funcionalidades del lenguaje HTML que comenzó a ofrecerlo como parte de su Navegador v.2.0. JavaScript sólo sirve para incluirse en documentos HTML y fuera de ellos no tiene ninguna vigencia, no crea aplicaciones autónomas. Se utiliza embebido en el código HTML, entre las tags <script> y </script>. Proporciona los medios para:

- ✓ Controlar las ventanas del navegador y el contenido que muestran.
- ✓ Evitar depender del servidor Web para cálculos sencillos.
- ✓ Capturar los eventos generados por el usuario y responder a ellos sin salir a Internet
- ✓ . Comprobar los datos que el usuario introduce en un formulario antes de enviarlos.
- ✓ Comunicarse con el usuario mediante diversos métodos.

La característica de JavaScript que más simplifica la programación es que, aunque el lenguaje soporta cuatro tipos de datos, no es necesario declarar el tipo de las variables, argumentos de funciones ni valores de retorno de las funciones.

1.12 Entorno de desarrollo integrado ('IDE')

Un entorno de desarrollo integrado (IDE), es un entorno de programación que ha sido empaquetado como un programa de aplicación, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI. Proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación.

1.12.1 Zend Studio

Zend Studio, es un programa de la casa Zend (uno de los mayores impulsores de de la tecnología de servidor PHP), orientada a desarrollar aplicaciones Web con PHP. Proporciona un buen número de ayudas (creación y gestión de proyectos, depuración del código), además de servir de editor de texto para páginas PHP. Son muchos los desarrolladores que trabajan con él, por lo que se considera uno de los mejores IDE.

El estar escrito en Java, le ha permitido a Zend lanzar versiones del producto para Windows, Linux y MacOS, con relativa facilidad. Zend Studio contiene una ayuda contextual con todas las librerías de funciones del lenguaje que asiste en todo momento ofreciendo nombres de las funciones y parámetros que deben recibir.

Dispone de herramientas para gestionar los proyectos, muy útiles para mejorar la productividad en la programación. Consta de dos partes en las que se dividen las funcionalidades de parte del cliente y las del servidor. Permite además hacer depuraciones simples de scripts.

Para simplificar el desarrollo de una aplicación, se utilizan Framework, los cuales mediante la automatización de algunos de los patrones utilizados para resolver tareas comunes facilitan los trabajos. Además proporciona estructura al código fuente, forzando al desarrollador a crear código más legible, más fácil de mantener y facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas. [11]

1.12.1 Symfony

Symfony, es un framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones Web. Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación Web. Este framework proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación Web compleja. Además automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. Symfony está desarrollado completamente en PHP5. Ha sido probado en numerosos proyectos reales y en sitios Web de comercio electrónico de primer nivel.

Es compatible con la mayoría de gestores de base de datos como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas Unix, Linux, como en plataformas Windows.

Como software libre, cuenta con una gran comunidad en varios idiomas, incluyendo español, y está seriamente respaldado por sus creadores (Sensio Labs), quienes además de usarlo en sus aplicaciones, dan conferencias a lo largo de todo el mundo.

Actualmente está en la versión estable 1.011 y en beta 1.1. Symfony, está basado en un patrón clásico del diseño Web, conocido como arquitectura MVC, toma lo mejor de esta arquitectura y la implementa de forma que el desarrollo de aplicaciones sea rápido y sencillo. El controlador frontal es

un componente que sólo tiene código relativo al MVC, por lo que no es necesario crear uno, ya que Symfony lo genera de forma automática. Las clases de la capa del modelo, también se generan automáticamente, en función de la estructura de datos de la aplicación. La librería Propel se encarga de esta generación automática, ya que crea el esqueleto o estructura básica de las clases y genera el código necesario. Cuando Propel encuentra restricciones de claves foráneas (o externas) o cuando encuentra datos de tipo fecha, crea métodos especiales para acceder y modificar esos datos. La abstracción de la base de datos es completamente invisible al programador, ya que la realiza otro componente específico llamado Creole. Así, si se cambia el sistema gestor de bases de datos en cualquier momento, no se debe reescribir ni una línea de código, ya que tan sólo es necesario modificar un parámetro en un archivo de configuración. Por último, la lógica de la vista se puede transformar en un archivo de configuración sencillo, sin necesidad de programarla (Potencier, Fabien and Zaninotto, François).

En la Figura 6 se puede observar el flujo de trabajo de Symfony

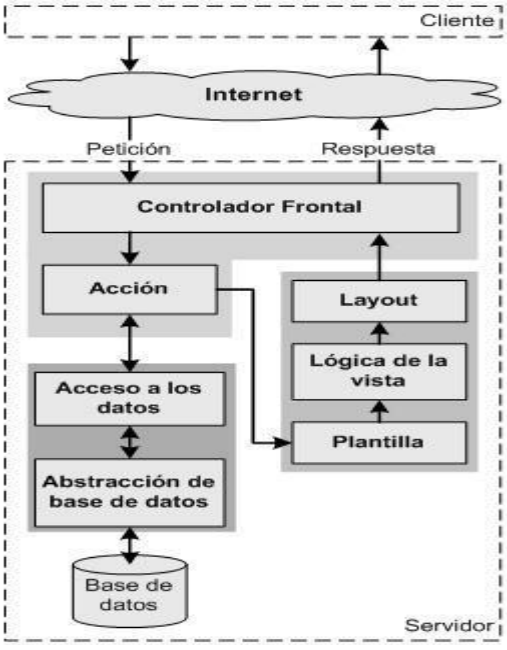

Figura # 11: Flujo de trabajo de Symfony.

1.13 Sistemas Gestores de Base de Datos (SGBD)

Un Sistema Gestor de Base de Datos, es un conjunto de programas que permiten crear y mantener una base de datos, asegurando su integridad, confidencialidad y seguridad; además de coordina la

organización, almacenamiento y extracción de los datos, e interpreta las consultas a la base de datos. [5]

Entre los más conocidos están:

1.13.1 PostgreSQL

PostgreSQL, es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (ORDBMS) de código abierto más avanzada del mundo. Comenzó como un proyecto denominado Ingres en la Universidad Berkeley de California. Ingres, fue más tarde desarrollado comercialmente por la Relational Technologies/Ingres Corporation. En 1986 otro equipo dirigido por Michael Stonebraker de Berkeley continuó el desarrollo del código de Ingres para crear un sistema de bases de datos objeto-relacionales llamado Postgres. En 1996, debido a un nuevo esfuerzo de código abierto y a la incrementada funcionalidad del software. Postgres fue renombrado a PostgreSQL, tras un breve periplo como Postgres95. El proyecto PostgreSQL sigue actualmente un activo proceso de desarrollo a nivel mundial gracias a un equipo de desarrolladores y contribuidores de código abierto.

PostgreSQL, posee muchas características que normalmente sólo se encontraban en las bases de datos comerciales tales como DB2 u Oracle. La siguiente es una breve lista de algunas de esas características, a partir de PostgreSQL. [12] (Douglas, Korry and Douglas, Susan)

1. DBMS Objeto-Relacional: PostgreSQL, aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multa-versión, soporte multa-usuario, optimización de consultas, herencia, y arrays. [13]

2. Altamente Extensible: PostgreSQL, soporta operadores, funcionales métodos de acceso y tipos de datos definidos por el usuario. [13]

3. Soporte_SQL_Comprensivo: PostgreSQL, soporta la especificación SQL99 e incluye características avanzadas tales como las uniones (joins). [13]

4. Integridad Referencial: PostgreSQL, soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos. [13]

5. API Flexible: La flexibilidad del API de PostgreSQL ha permitido a los vendedores proporcionar soporte al desarrollo fácilmente para el RDBMS PostgreSQL. Estas interfaces incluyen Object Pascal, Python, Perl, PHP, ODBC, Java/JDBC, Ruby, TCL, C/C++, y Pike. [13]

6. Lenguajes Procedurales: PostgreSQL, tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL. Otra ventaja de PostgreSQL es su habilidad para usar Perl, Python. [13]

7. MVCC: MVCC, o Control de Concurrencia Multi-Versión (Multi-Version Concurrency Control), es la tecnología que PostgreSQL usa para evitar bloqueos innecesarios. MVCC está considerado mejor que el bloqueo a nivel de fila porque un lector nunca es bloqueado por un escritor. En su lugar, PostgreSQL mantiene una ruta a todas las transacciones realizadas por los usuarios de la base de datos. PostgreSQL es capaz entonces de manejar los registros sin necesidad de que los usuarios tengan que esperar a que los registros estén disponibles. [13]

8. Cliente/Servidor: PostgreSQL, usa una arquitectura proceso-por-usuario cliente/servidor. Esta es similar al método del Apache 1.3.x para manejar procesos. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL. [13]

9. Write Ahead Logging (WAL): La característica de PostgreSQL, conocida como "Write Ahead Logging" incrementa la dependencia de la base de datos al registro de cambios antes de que estos sean escritos en la base de datos. Esto garantiza que en el hipotético caso de que la base de datos se caiga, existirá un registro de las transacciones a partir del cual podremos restaurar la base de datos. [13]

1.13.2 MySQL

MySQL, un sistema de gestión de base de datos desarrollado por MySQL AB como software libre en un esquema de licenciamiento dual. MySQL AB fundada por David Axmark, Allan Larsson y Michael Widenius en 1995 en Suecia pertenece a Sun Microsystems desde enero de 2008, quien posee el copyright de la mayor parte del código. Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero las empresas que quieran incorporarlo en productos privativos pueden comprar a la empresa una licencia específica que les permita este uso.

MySQL, es muy utilizado en aplicaciones Web como MediaWiki o Drupal, en múltiples plataformas, incluyendo: AIX/GNU-Linux/Windows 95/Windows 98/Windows NT/Windows 2000/Windows XP/ Windows Vista y otras versiones de Windows/Mac OS X/Solaris/SunOS-Apache-MySQL-PHP/Perl/Python) y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación Web está muy ligada a PHP, que a menudo aparece en combinación con MySQL.

1.13.3 Oracle

Oracle, es básicamente una herramienta que se basa en la tecnología cliente/servidor para la gestión de bases de datos. Surgió entre finales de los años 70 y principio de los años 80, desarrollado por la compañía “Oracle Corporación” y conocida entonces como “Relational Software”. En la actualidad es un producto vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hacen que sólo se vea en empresas muy grandes y multinacionales, por norma general. En el desarrollo de páginas Web pasa lo mismo: como es un sistema muy caro no está tan extendido como otras bases de datos.

1.14 Conclusiones Parciales

En el presente capítulo, se presentaron los principales conceptos y sistemas informatizados vinculados al campo de acción, se brinda una breve panorámica de las tendencias y tecnológicas actuales que permitieron la selección de las herramientas, lenguajes de programación y metodologías, que fueron seleccionadas para el desarrollo de la propuesta del trabajo de diploma.

Se decidió, seleccionar para el desarrollo y construcción del sistema, como lenguajes de programación del lado cliente, HTML, que es el lenguaje de definición de estructura más difundida y para incrementar sus funcionalidades, JavaScript. Del lado servidor, PHP, partiendo de que es "open source" y puede ser utilizado en cualquiera de los principales sistemas operativos. Posee además, una de las comunidades más grandes de desarrolladores en el mundo, incluyendo la UCI; un código simple que facilita su trabajo, una gran librería de funciones y mucha documentación. Una de las características que se tuvieron en cuenta es la facilidad de PHP, para funcionar con el servidor Web Apache, ya que se integra como un módulo de este y justifica la selección del mismo, ambos poseen gran compatibilidad y por su parte Apache es también gratuito, multiplataforma y presenta gran modularidad que lo hace personalizable a la vez que configurable y flexible. Como gestor de base de datos PostgreSQL se adaptó perfectamente a las necesidades planteadas.

Para efectuar la implementación de la aplicación, nos apoyamos en el IDE Zend Studio, ya que está orientado a desarrollar aplicaciones Web con PHP. La metodología de desarrollo de software escogida fue RUP, ya que abarca de manera organizada todo el ciclo de vida del software y permite obtener un producto más eficiente y predecible, basado en la previa y total planificación del proyecto, lo que posibilita además, contar con una detallada y amplia documentación del proyecto. La metodología RUP utiliza el UML para preparar todos los esquemas de un sistema software. Se utiliza este lenguaje para el modelado del sistema, ya que constituye un estándar a nivel internacional para especificar, visualizar, construir y documentar artefactos de un software. Además, es el lenguaje para describir principalmente sistemas orientados a objetos. Visual Paradigm es una herramienta CASE para el modelado con UML y que integra todos los elementos que propone la metodología RUP para cubrir todo el ciclo de vida de un proyecto. Se aprovecharon los beneficios brindados por Symfony, framework que se adaptó perfectamente, a las necesidades reales presentadas para la propuesta de solución, software libre, que cuenta con una gran comunidad en varios idiomas, incluyendo español y que permitió agilizar el trabajo. Basado en el patrón de arquitectura MVC, de aquí que este, haya sido el implementado en el sistema desarrollado, conjuntamente con las distintas ventajas que nos brinda su uso: el poder separar los componentes de un sistema, permitiendo implementarlos por separado y la conexión dinámica entre el Modelo y sus Vistas.

Capítulo 2: Características del Sistema

2.1 Introducción

Es de suma importancia detenerse a analizar las características y organización de los procesos que se desarrollan en el área que se pretende automatizar, esto es un aspecto fundamental antes de empezar a desarrollar el sistema y se hace con el fin de lograr una mayor comprensión del problema que se desea resolver. Con este propósito se realiza la modelación del negocio y también con la intención de establecer el común entendimiento entre clientes y desarrolladores.

En el presente capítulo se describe el objeto de estudio, se realiza la modelación del negocio, y además se analizan tanto los requerimientos funcionales como los no funcionales.

2.2 Descripción general de los procesos involucrados en el negocio

La UCI como institución ejecutora en desarrollo de software, necesita establecer mecanismos fiables para mantener el control de sus proyectos. El seguimiento del cronograma de trabajo, controlando por fechas el cumplimiento de sus principales hitos, constituye una técnica empleada para estos fines. Sin embargo, cuando aún la UCI no cuenta con un registro histórico de tiempos asociados al desarrollo, los plazos impuestos en los cronogramas suelen ser empíricos y no necesariamente realistas. Para la crítica etapa de implementación, en la que muchos se interesan por conocer su avance, este problema se agudiza cuando no existen tareas “definidas”, ni relaciones entre sí y se dificulta el seguimiento del progreso de las mismas. Como consecuencia, la información del avance de la implementación que se brinda es subjetiva, los problemas no se detectan a tiempo y los proyectos concluyen, casi siempre, fuera de fecha. Por esta razón, el presente trabajo propone el desarrollo de una herramienta que permita aplicar una métrica mediante la cual el líder de proyecto cuantifica objetivamente el estado de avance del software durante esta etapa. La herramienta toma en consideración por un lado, los casos de uso a desarrollar valorando su complejidad y por otro, las tareas para implementarlos de acuerdo al flujo de trabajo establecido por la arquitectura del software, teniendo en cuenta que estas tareas tienen un peso específico dentro del tiempo total necesario para implementar un caso de uso. De la aplicación de esta métrica se obtiene cuantitativamente el estado de avance del desarrollo, así como su evolución en el tiempo, resultando en información precisa, necesaria para la toma de decisiones gerenciales del proyecto. Adicionalmente, el registro histórico de datos que la soportan, constituye una base objetiva para futuras estimaciones.

Se propone la creación de una aplicación en la que se aplique esta métrica.

2.3 Información que se maneja en el negocio

- Un Proyecto está compuesto por:
  - ✓ Listado de Tareas del Proyecto: Tareas que define el jefe de proyecto.
  - ✓ Listado de Módulos del Proyecto: Módulos en los cuales se divide el proyecto.
  - ✓ Listado de CU del Proyecto: CU que componen los módulos.

2.4 Objeto de automatización

Se desean automatizar los procesos de registro de todos los proyectos productivos de la UCI y como se comportan en el flujo de implementación según su avance.

2.5 Propuesta del sistema

Se propone implementar un sistema con el que se pretende facilitar el registro y control de la información que se manipula en los proyectos productivos en la etapa de implementación y que brinde además un número de reportes de cómo se comportan los mismos según su avance, proporcionando así grandes beneficios para aquellas personas que deseen información al respecto.

El sistema permite crear reportes de estado del avance del proyecto, así como saber el estado de avance de cada módulo y caso de uso, además Graficar ese avance para un mejor entendimiento del mismo.

Para todo esto el sistema brinda la posibilidad de que el jefe de proyecto registre el nombre del proyecto, las tareas, los módulos y los CU que componen los mismos.

2.6 Modelamiento del negocio

Durante el flujo de trabajo modelamiento del negocio, se logra una mayor comprensión del problema a resolver, mediante el estudio de la estructura y dinámica de la organización; donde se identifican sus procesos, roles y responsabilidades mediante los modelos de casos de uso del negocio y de objetos, se garantiza el entendimiento común entre desarrolladores y usuarios finales del software que se pretende desarrollar.

Dentro del modelamiento del negocio es importante definir los actores y trabajadores ya que son ellos los que interactúan con el de forma directa o indirecta.

2.6.1 Actores y trabajadores del negocio

Lo que se modela como un actor del negocio es, el rol que se juega al interactuar con el negocio para beneficiarse de sus resultados. Es un individuo o grupo de individuos, organización, máquina o sistema de información externo al negocio y que interactúa con él. Por otra parte, un trabajador del negocio define el rol de un individuo o grupo de individuos, máquina o sistema automatizado que participan directamente en los procesos que se llevan a cabo en el negocio, ya que son los que realizan las actividades y son propietarios de elementos pero no obtienen ningún beneficio con los resultados del proceso.

Las **Tablas 1** y **2** muestran, los actores y trabajadores del negocio respectivamente y sus justificaciones correspondientes:

Actores del Negocio	Justificación
Líder de Proyecto	El Líder de Proyecto es el encargado de definir las tareas, módulos y CU del proyecto en el flujo de trabajo de implementación y realizar los cálculos correspondientes para cuantificar el avance.

Tabla # 3: Descripción de los actores del negocio.

Trabajadores del Negocio	Justificación
Implementador	El implementador es el encargado de darle cumplimiento a los CU.

Tabla # 4: Descripción de los trabajadores del negocio.

2.6.2 Modelo de casos de uso del negocio

El modelo de casos de uso del negocio, se describe mediante diagramas de casos de uso. Es una técnica que permite la descripción de los procesos de negocio de una organización en términos de casos de uso y

actores del negocio ya definidos anteriormente, que se corresponden con los procesos del negocio y los clientes, respectivamente.

A continuación se muestra la **Figura 7** correspondiente al diagrama de casos de uso del negocio:

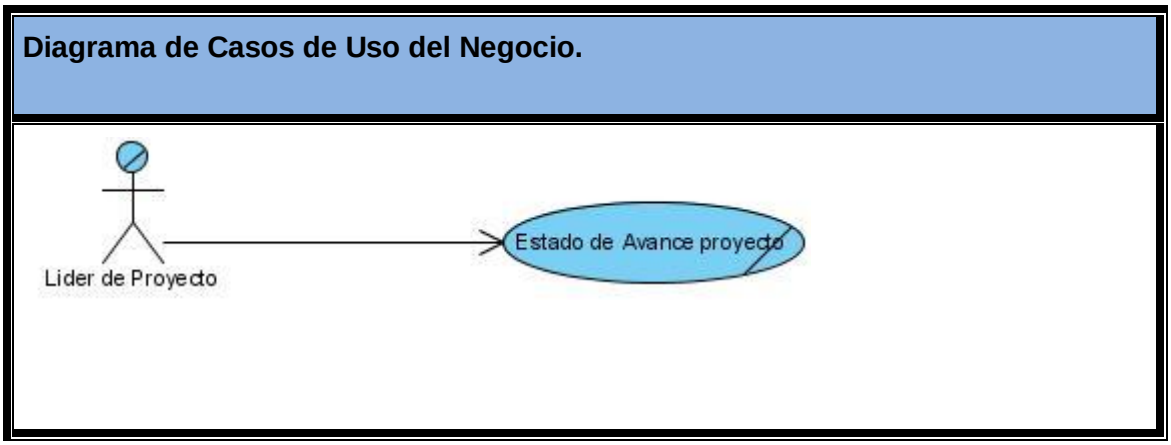

Figura # 12: Diagrama de CU del Negocio.

El modelado de los casos de uso brinda una panorámica del proceso como tal del negocio pero para detallar el fuljo de sucesos así como la forma de interactuar de los actores con proceso en sí, es preciso la realización de los casos de uso del negocio.

2.6.3 Realización de los casos de uso del negocio

Al detallar los casos de uso del negocio, se sabe comienza un caso de uso y cómo termina, pasando por la forma de interactuar con los actores, brindando así una mayor comprensión de los procesos del negocio a clientes y desarrolladores.

Se describe de forma textual como se llevan a cabo las actividades dentro del negocio y quienes las realizan. Además, por medio de los diagramas de actividades se muestra el flujo de los procesos de manera gráfica. A través de las calles, se especifican las responsabilidades de los actores y trabajadores del negocio y a través del flujo de objetos cómo se utilizan las entidades del negocio.

Descripciones textuales de los casos de uso del negocio y sus diagramas de actividades correspondientes:

Caso de Uso:	Cuantificar Estado de Avance

Actores:	Directivos de la UCI
Trabajadores:	Líder de Proyecto
Resumen:	El directivo de la UCI le pide al líder de proyecto saber en qué estado de avance esta su proyecto en la etapa de implementación y para esto el líder de proyecto define las tareas, los módulos y los CU, además de hacer los cálculos pertinentes para saber el estado de avance.
Flujo normal de los eventos.	
Acción del actor	**Respuesta del negocio**
	1. El líder de proyecto crea las tareas, los módulos, los CU específicos para cada módulo, la estimación del tiempo (semanas) que demorara el proyecto y realiza los cálculos pertinentes.
2 El directivo solicita información del proyecto.	2.1 El líder de proyecto le pregunta qué tipo de información en específico necesita.
3. El directivo detalla la información que desea obtener.	3.1 El líder le muestra la información del tipo solicitada.
4 El directivo recibe la información	

solicitada.	
Mejoras:	Se obtendrá un reporte digital sobre el estado de avance del flujo de implementación lo que traerá consigo mejoras en cuanto a tiempo y control.

Tabla # 5: Descripción textual del caso de uso del negocio “Estado de Avance del Proyecto”.

Diagrama de actividades del caso de uso del negocio

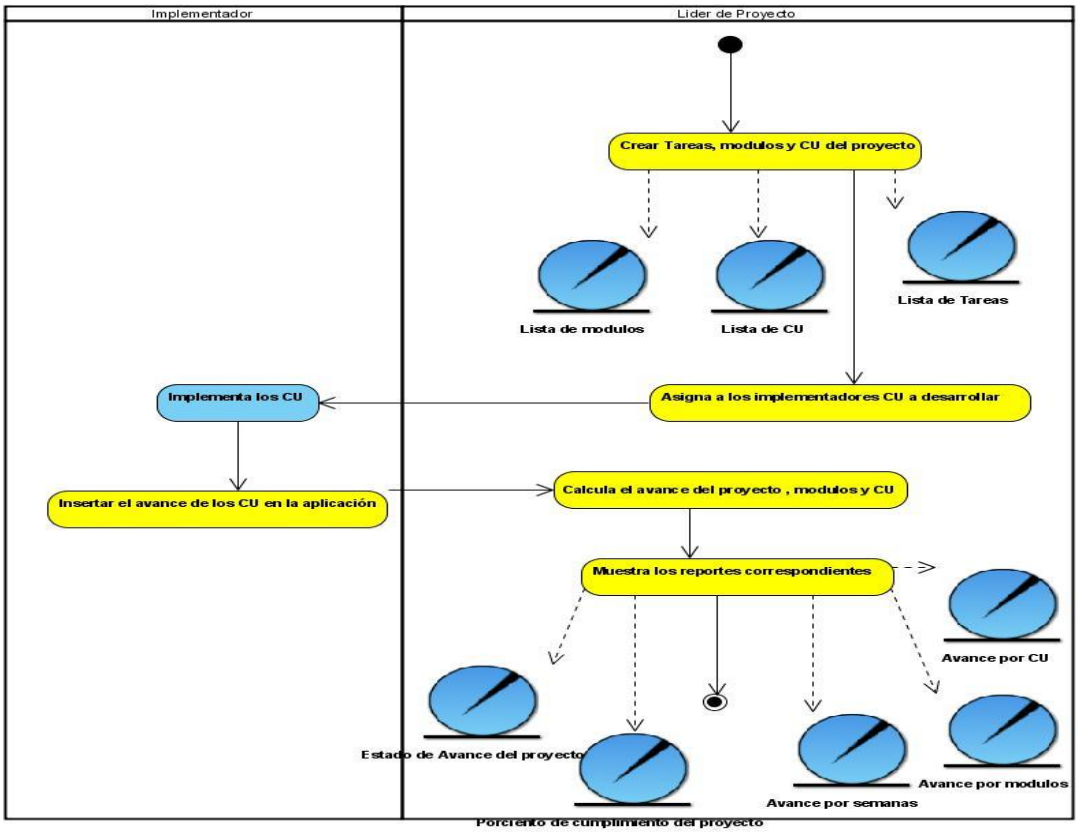

Figura # 13: Diagrama de actividades de CU.

Para describir cómo cada caso de uso del negocio es llevado a cabo por parte de un trabajador se utiliza un conjunto de entidades del negocio y unidades de trabajo, esta descripción es el modelo de objetos del negocio como se puede observar en la **Figura 14**.

2.6.4 Modelo de objetos del negocio

El modelo de objetos del negocio, es un modelo interno a un negocio, muestra la relación entre los trabajadores y entidades del negocio dentro del flujo de trabajo de modelamiento del negocio.

La **Figura 14** es la correspondiente al modelo de objetos del negocio, donde una vez realizado el modelo de objetos es preciso definir las reglas que regirán el negocio, que no son más que las restricciones que existen en un negocio dado; entiéndase por esto las acciones no válidas que la aplicación debe controlar para que el negocio no colapse.

Modelo de Objetos del Negocio.

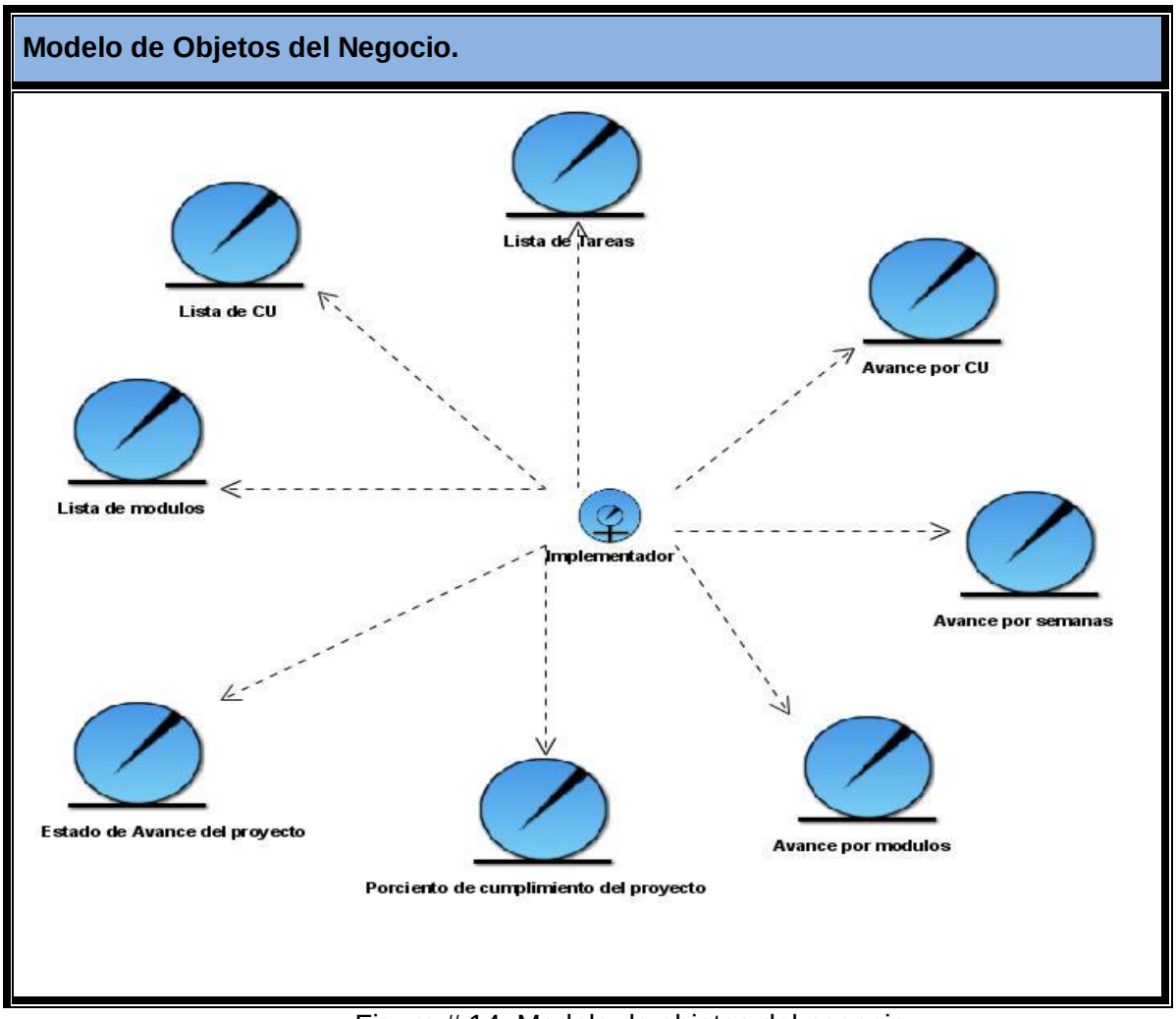

Figura # 14: Modelo de objetos del negocio.

2.6.5 Reglas del negocio

Las reglas del negocio regulan aspectos del negocio, ya que describen políticas que deben cumplirse o condiciones que deben ser satisfechas.

A continuación se listan las reglas identificadas en el presente negocio:

- Un proyecto puede tener solo un líder de proyecto.
- Las tareas definidas son las mismas para todos los módulos.

Una vez terminada toda la fase del modelamiento del negocio se hace indispensable la captura de requisitos, porque en él se establece qué tiene que hacer exactamente el sistema y como, también todo lo relacionado con la integridad del sistema.

2.7 Captura de requisitos

El flujo de trabajo de requerimientos es uno de los más importantes. Pudiera verse a los requisitos como al contrato que se debe cumplir, de modo que los usuarios finales tienen que comprender y aceptar los requisitos que se especifiquen. Los requisitos se dividen en dos grupos.

Los requisitos funcionales y los no funcionales:

2.7.1 Requisitos funcionales

Los requerimientos funcionales son aquellas capacidades o condiciones que el sistema debe cumplir. Ellos definen qué es lo que el sistema debe hacer y permiten identificar las funcionalidades requeridas.

No alteran la funcionalidad del producto, es decir, los requerimientos funcionales se mantienen invariables sin importarle con que propiedades o cualidades se relacionen.

A continuación se enumeran las funcionalidades que el sistema debe cumplir:

- ✓ **RF 1 Gestionar proyectos**
  - 1.1 Registrar un nuevo Proyecto. (Nombre Proyecto, Facultad a la que pertenece el proyecto, Cantidad de semanas en que se debe realizar el proyecto).
  - 1.2 Modificar los datos del proyecto excepto la fecha de inicio.

- ✓ **RF 2 Gestionar Tareas**
  - 2.1 Crear tareas de un proyecto.
  - 2.2 Asignar peso de la tarea.
  - 2.3 Ver tareas creadas.
  - 2.4 Modificar Pesos.
- ✓ **RF 3 Crear Módulos**
  - 3.1 Crear Módulos.
- ✓ **RF 4 Crear CU**
  - 4.2 Crear CU.
- ✓ **RF 5 Graficar**
  - 5.1 Graficar Estado de Avance Por Módulos.
  - 5.2 Graficar el estado de avance del proyecto por semanas.
- ✓ **RF 6 Gestionar usuarios**
  - 6.1 Adicionar un nuevo usuario.
  - 6.2 Eliminar un usuario.
  - 6.3 Cambiar el líder de proyecto de un proyecto dado.
- ✓ **RF 7 Mostrar Información**
  - 7.1 Conocer el estado de avance en que se encuentra el proyecto.
  - 7.2 Conocer el por ciento de avance del proyecto.
  - 7.3 Conocer el por ciento de avance de cada módulo por semana.

- 7.4 Conocer el estado de avance en que se encuentra un módulo del proyecto.
- 7.5 Conocer el por ciento de avance de cada módulo del proyecto.
- 7.6 Conocer el por ciento de completamiento de cada CU del proyecto.

✓ **RF 8 Completar CU**

- 8.1 Asignar el estado de completamiento de un CU con respecto a las tareas.

2.7.2 Requisitos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Son esas características que posibilitan que el producto sea atractivo, usable, rápido, confiable, etc. En muchos casos los requerimientos no funcionales son fundamentales en el éxito del producto. Normalmente están vinculados a requerimientos funcionales, es decir una vez que se conozca lo que el sistema debe hacer, se puede determinar cómo ha de comportarse, qué cualidades debe tener o cuán rápido o grande debe ser.

- **Requerimientos de Software:**

RNF 1: Navegador Internet Explorer v6.0 o superior, Mozilla Firefox v2.x.

RNF 2: Servidor Web Apache v2.x o superior con módulo PHP 5.2.3 disponible, este debe estar configurado con las extensiones “pgsql”, “ldap”.

RNF 3: Como Sistema Gestor de Base de Datos: PostgreSQL preferiblemente v8.x en adelante.

- **Requerimientos de Hardware:**

RNF 4: Tarjeta de red.

RNF 5: Para los servidores tanto Web como SGBD: PENTIUM IV o superior con 256 MB de RAM o más.

RNF 6: Capacidad de disco duro, preferiblemente mayor a los 10 GB.

- **Restricciones para el diseño e implementación:**

RNF 7: Utilizar como lenguaje del lado del servidor al PHP v5.2.3 o superior y del lado del cliente: HTML y JavaScript.

RNF 8: Para la programación en PHP se recomienda el IDE: Zend Studio, con la ayuda del Framework Symfony.

RNF 9: Se recomienda el uso de la arquitectura MVC.

RNF 10: El código deberá ser reutilizable.

- **Requerimientos de Apariencia o interfaz externa:**

RNF 11: Estará diseñado para resolución de 1024x768, aunque deberá verse en 800x600 o cualquier resolución superior a esta.

RNF 12: La interfaz debe ser agradable para el usuario, que combine correctamente los colores, tipo de letra y tamaño, que los iconos estén en correspondencia con lo que representan no debe contener muchas imágenes que demoren las respuestas al usuario.

RNF 13: La navegabilidad debe ser sencilla.

- **Requerimientos de Seguridad:**

RNF 14: Se podrá acceder al sistema solamente después de autenticarse.

RNF 15: Se podrá acceder al sistema solamente después de autenticarse.

RNF 16: Autorización (Atribución a los usuarios respecto a sus funciones de trabajo.)

RNF 17: Contar con un sistema de permisos y usuarios para el acceso a la información.

RNF 18: Chequeo de seguridad sobre las operaciones no reversibles (adicionar, modificar, eliminar).

RNF 19: El sistema debe estar disponible las 24 horas del día (disponibilidad).

- **Requerimientos de Usabilidad:**

RNF 20: El sistema debe permitir el acceso concurrente de diferentes usuarios, en dependencia del nivel de usabilidad que este definida para cada uno.

RNF 21: El sistema podrá ser usado por cualquier persona que posea conocimientos básicos de computación y trabajo en la Web.

- **Requerimientos de Rendimiento:**

RNF 22: El sistema deberá ser capaz de gestionar toda la información y dar respuesta a las solicitudes lo más rápido posible.

- **Requerimientos de Soporte:**

RNF 23: Contar con una etapa de prueba con vista a eliminar la mayor cantidad de errores.

- **Requerimientos de Portabilidad:**

RNF 24: El sistema deberá ser multiplataforma, haciendo énfasis en Linux y Windows.

- **Requerimientos Políticos Culturales:**

RNF 25: El sistema solo podrá ser utilizado en territorio cubano.

RNF 26: El sistema debe tener una interfaz que esté acorde con el lugar donde se implantará, es decir, que refleje los ideales de la organización y que se amena, cumpliendo con las pautas de diseño.

RNF 27: El producto no debe contener palabras en otros idiomas.

RNF 28: El producto debe respetar los términos empleados normalmente por los especialistas en el tema de la esfera que se automatiza.

- **Requerimientos Legales:**

RNF 29: La mayoría de las herramientas de desarrollo son libres y sus licencias están avaladas.

RNF 30: Reconocido y autorizado por instancias superiores tales como la dirección de producción de la UCI (facultad).

RNF 31: Documentación legal de uso como Declaración de Autoría.

RNF 32: El sistema podrá ser ejecutado en diferentes plataformas.

- **Requerimientos de Confiabilidad:**

RNF 33: Los reportes que se obtendrán deben ser 100% precisos y reales.

RNF 34: Deben establecerse los mecanismos necesarios para el restablecimiento del sistema ante fallos de comunicación u otros.

RNF 35: Garantiza un control estricto sobre el tráfico de información.

RNF 36: Chequeo constante de la integridad y consistencia en los datos.

Ya identificados y definidos los requisitos tanto funcionales como no funcionales que debe cumplir el sistema y haciendo uso de ellos se comienza la etapa del modelado del sistema.

2.8 Modelado del sistema

El modelado del sistema se divide en varias secciones, entre las que se encuentran, la definición de los actores de sistema, el modelado de casos de uso del sistema y la descripción de los casos de uso del sistema, todo esto se hace con el fin de tener una visión objetiva del sistema, no es más que una definición del quién y el cómo lo hace.

2.8.1 Definición de los actores del sistema

Los actores del sistema representan terceros fuera del sistema que colaboran con él. Estos pueden ser tanto los usuarios como los sistemas externos de un sistema que interactúan con él y que pueden ser expresados mediante uno o más actores del sistema.

La **Tabla 6** muestra los actores del sistema y su correspondiente justificación:

Actor del sistema	**Justificación**

Usuario	Son todos los que se autentican dentro del sistema puede comportarse como un Líder de Proyecto, un Administrador del Sistema, un Directivo de la UCI o un implementador.
Implementador	Es el que se encarga de darle cumplimiento a los CU con respecto a las tareas.
Líder de Proyecto	Es el encargado de gestionar toda la información con respecto a su proyecto.
Directivo de la UCI	Obtiene información referente a los proyectos como conocer, en qué estado de avance se encuentra un proyecto, conocer el por ciento de avance del proyecto, conocer en qué estado de avance se encuentra un proyecto por semana, conocer el estado de avance en que se encuentra un módulo, conocer el porciento de completamiento de cada Módulo, conocer el porciento de completamiento de cada CU.
Administrador del sistema	Administra toda la información referente al sistema, como gestionar usuarios y administrar la base de datos.

Tabla # 6: Actores del sistema.

2.8.2 Modelo de casos de uso del sistema

La descripción de los casos de uso del sistema, detallan las acciones que tienen lugar durante la interacción actor-sistema, es decir, describe el flujo de actividades que realiza el actor al hacer uso del sistema y las correspondientes respuestas del mismo. La **Figura 15** muestra el diagrama de casos de uso del sistema.

2.8.3 Descripción de los casos de uso del sistema

La descripción de los casos de uso del sistema, detallan las acciones que tienen lugar durante la interacción actor-sistema, es decir, describe el flujo de actividades que realiza el actor al hacer uso del sistema y las correspondientes respuestas del mismo.

Diagrama de casos de uso del sistema.

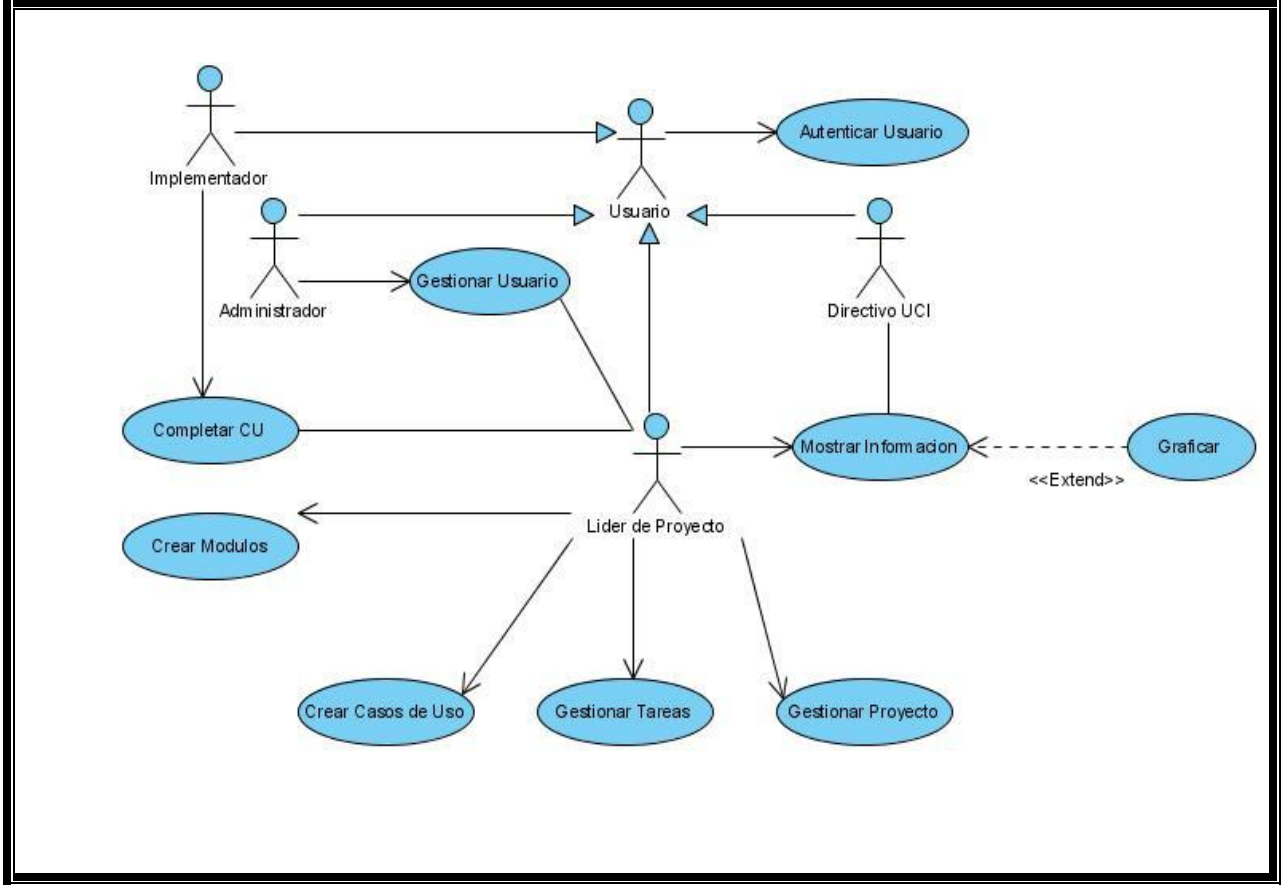

Figura # 15: Diagrama de CU del sistema.

Descripciones de los casos de uso del sistema:

2.8.3.1 CU: Gestionar usuarios

Nombre del CU:	Gestionar usuarios.
Actores:	Administrador del Sistema (Inicia).
Propósito:	Permite al Administrador del Sistema insertar, modificar o eliminar los usuarios otorgándole los permisos individuales a cada usuario sobre el sistema.
Resumen:	El CU comienza cuando el Administrador del Sistema gestiona el acceso de usuarios al sistema, el Sistema muestra una lista de las opciones que puede realizar con respecto a los usuarios y el CU termina cuando el Administrador del Sistema actualiza la lista de usuarios.
Referencias:	RF 6.
Precondiciones :	Los usuarios tienen que pertenecer al dominio UCI.
Curso normal de eventos	
Acción del Actor	**Respuesta del Sistema**
1. El Administrador del Sistema se autentica.	1.1 Muestra una lista con las posibles opciones a las que puede acceder: ✓ Insertar usuario. ✓ Eliminar usuario. ✓ Modificar líder de proyecto.

	.
Sección: Insertar usuario	
Curso normal de los eventos	.
Acción del Actor	**Respuesta del Sistema**
2. El Administrador del Sistema selecciona la opción ¨ Insertar usuario¨.	2.1 Muestra un formulario donde se insertaran los datos del usuario así como los permisos que le serán concedidos, que son de: ✓ Líder de proyecto. ✓ Directivo UCI. ✓ Administrador del sistema.
3. El Administrador del Sistema inserta los datos solicitados por el Sistema: ✓ Usuario. ✓ Permisos. ✓ Contraseña del administrador.	3.1 “Esta seguro que desea enviar”.
4. El Administrador del Sistema selecciona la opción aceptar.	4.1 El Sistema verifica que los datos insertados por el Administrador del Sistema, sin son válidos inserta el usuario en la base de datos y re direcciona al administrador para la página inicio.
Cursos Alternos	
Acción del Actor	**Respuesta del Sistema**
3.a	3.1 1 El sistema muestra el formulario con los datos que el directivo pretende insertar.
	Si los datos introducidos por el

	Administrador del Sistema no se corresponden con los pedidos muestra un mensaje de error al introducir los datos. "Nombre de usuario o contraseña incorrectos". "El usuario insertado ya existe en la base de datos".
4. a Si el Administrador del Sistema selecciona la opción de ¨Cancelar¨.	4.1 El sistema re direcciona a la página de inicio.
Postcondiciones:	Se creó una instancia Usuario.rol.
Sección: Eliminar usuarios.	
Curso normal de los eventos.	.
Acción del Actor	**Respuesta del Sistema**
2. El Administrador del Sistema selecciona la opción ¨ Eliminar usuarios¨.	2.1 Muestra una lista de todos los usuarios registrados en la base de datos.
3. El Administrador del Sistema selecciona el usuario a eliminar.	3.1 "Está seguro que desea enviar"
4.1 El Administrador del Sistema selecciona la opción ¨ Aceptar¨.	4.1 El Sistema elimina al usuario en la base de datos y re direcciona para la misma página.
Cursos Alternos	
Acción del Actor	**Respuesta del Sistema**

4. a Si el Administrador del Sistema selecciona la opción de ¨Cancelar¨.	4.1 El Sistema re direcciona al Administrador del Sistema a la página de inicio.
Postcondiciones:	Se eliminó una instancia Usuario.rol.
Sección: Insertar Implementador	
Curso normal de los eventos	.
Acción del Actor	**Respuesta del Sistema**
1. El Líder de proyecto selecciona la opción ¨ Insertar Desarrollador¨.	1.1 Muestra un formulario donde se insertaran los datos del usuario.
2. El Líder de proyecto inserta los datos solicitados por el Sistema: ✓ Usuario. ✓ Contraseña del administrador.	2.1 “Está seguro que desea enviar”.
3. El Líder de proyecto selecciona la opción aceptar.	3.1 El Sistema verifica los datos insertados por el Líder de proyecto, si son válidos inserta el usuario en la base de datos y re direcciona al Líder para la página inicio.
Cursos Alternos	
Acción del Actor	**Respuesta del Sistema**
2.a	2.1 1 El sistema muestra el formulario con los datos que el directivo pretende insertar.

	Si los datos introducidos por el Líder de proyecto no se corresponden con los pedidos muestra un mensaje de error al introducir los datos. "Nombre de usuario o contraseña incorrectos". "El usuario insertado ya existe en la base de datos".
3. a Si el Líder de proyecto selecciona la opción de ¨Cancelar¨.	3.1 El sistema re direcciona a la página de inicio.
Postcondiciones:	Se creó una instancia CU implementador.
Sección: Modificar Líder	
Curso normal de los eventos	.
Acción del Actor	**Respuesta del Sistema**
2. El Administrador del Sistema selecciona la opción ¨ Modificar Líder¨.	2.1 Muestra un formulario donde se muestran los proyectos con su respectivo Líder.
3. El Administrador del Sistema selecciona el proyecto al cual quiere modificarle el Líder.	3.1 Muestra un formulario donde se insertaran los datos del usuario así como la contraseña de administrador

4. El Administrador del Sistema inserta los datos solicitados por el Sistema: ✓ Usuario. ✓ Contraseña del administrador.	4.1 “Esta seguro que desea enviar”.
5. El Administrador del Sistema selecciona la opción aceptar.	5.1 El Sistema verifica que los datos insertados por el Administrador del Sistema, sin son válidos inserta el usuario en la base de datos y re direcciona al administrador para la página inicio.
Cursos Alternos	
Acción del Actor	**Respuesta del Sistema**
	5.1 a Si los datos introducidos por el Administrador del Sistema no se corresponden con los pedidos muestra un mensaje de error al introducir los datos. “Nombre de usuario o contraseña incorrectos”. “El usuario insertado ya existe en la base de datos”.
5. a Si el Administrador del Sistema selecciona la opción de ¨Cancelar¨.	5.1 El sistema re direcciona a la página de inicio.
Postcondiciones:	Se creó una instancia Usuario.rol.

Tabla # 7: Descripción del caso de uso del sistema “Gestionar usuarios del sistema”.

Interfaz (Insertar Usuario)

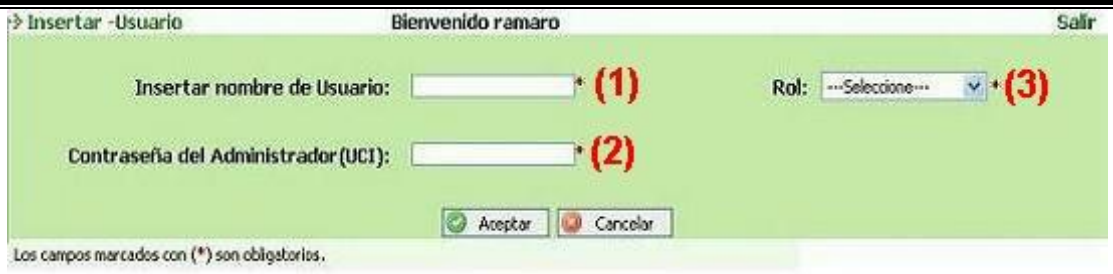

Figura # 16: Interfaz (Insertar Usuario).

1. Usuario del dominio (UCI) del usuario a insertar.
2. Contraseña del administrador del Sistema
3. Rol que va a jugar dentro de la aplicación (Directivo, Administrador Líder).

Interfaz (Eliminar Usuario)

Figura # 17: Interfaz (Eliminar Usuario).

1. Usuario a Eliminar.

Interfaz (Insertar Implementador)

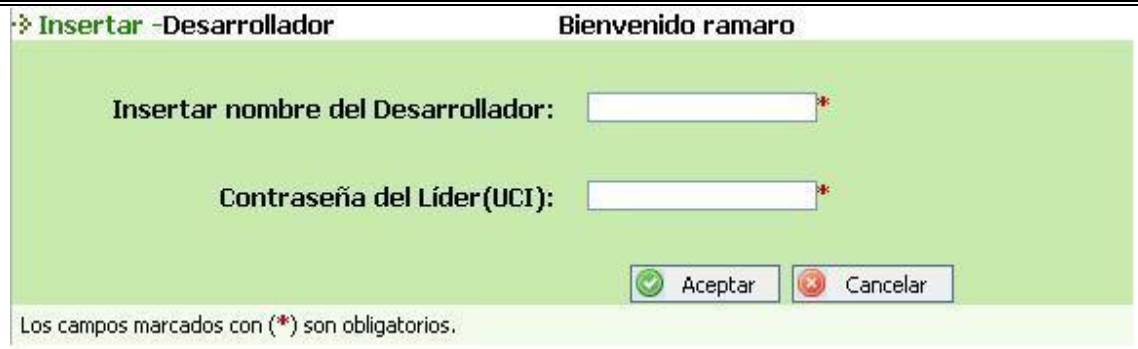

Figura # 18: Interfaz (Insertar Implementador).

1. Usuario del dominio (UCI) del desarrollador.
2. Contraseña del Líder.

Interfaz (Modificar Líder)

Figura # 19: Interfaz (Modificar Líder).

1. Usuario del dominio (UCI) del Líder.
2. Contraseña del Líder.

2.8.3.2 CU Autenticar

Nombre del CU:	Autentificar Usuario.
Actores:	Usuario (Inicia).
Propósito:	Permitir a los usuarios autenticarse y hacer uso del sistema según los permisos que estos posean y al terminar el trabajo cerrar sesión.
Resumen:	El caso de uso se inicia cuando un usuario del sistema desea hacer uso del mismo. Para esto es necesario realizar una autenticación previa y el sistema da el acceso a los recursos de acuerdo al usuario identificado. Termina el caso de uso cuando el usuario entra al sistema o se le deniega el acceso.
Referencias :	
Precondiciones:	El usuario debe pertenecer al dominio UCI, y estar previamente insertado en la base de datos.

<table>
<tr><td colspan="2">Curso normal de los eventos</td></tr>
<tr><td>Acción del actor:</td><td>Respuesta del Sistema</td></tr>
<tr><td>1. El usuario accede a la página principal de la aplicación.</td><td>1.1 El sistema muestra el formulario en el cual el usuario debe introducir los datos
necesarios para ser identificado :
✓ Nombre de usuario
✓ Contraseña</td></tr>
<tr><td rowspan="2">2. El usuario introduce los datos que ha solicitado el sistema (usuario y contraseña para ser identificado como usuario del sistema) y presiona el botón “Aceptar”.</td><td>2.1 El sistema verifica usuario y contraseña introducidos</td></tr>
<tr><td>2.2 Si los datos introducidos por el usuario se corresponden con los de un usuario registrado en la base de datos el sistema le permite acceder al sistema de acuerdo a los permisos que posea.</td></tr>
<tr><td colspan="2">Cursos Alternos</td></tr>
<tr><td>Acción del actor</td><td>Respuesta del Sistema</td></tr>
<tr><td>2. a</td><td>2.2 Si los datos introducidos por el usuario no se corresponden con los de un usuario registrado en la base de datos el sistema muestra un mensaje de error y le indica al usuario que no es válido (le deniega el acceso).
“Nombre de usuario o contraseña</td></tr>
</table>

	incorrecto". "Su usuario no está registrado en la base de datos¨.
Postcondiciones:	El Usuario.rol accedió al sistema.
Prioridad:	Secundario.

Tabla # 8: Descripción del caso de uso del sistema "Autentificar Usuario".

Interfaz (Autenticarse)

Figura # 20: Interfaz (Autenticarse).

1. Usuario del dominio (UCI) del Líder.
2. Contraseña del Líder.

2.8.3.3 CU Gestionar Proyecto

Nombre del CU:	Gestionar Proyecto.
Actores:	Líder del Proyecto (inicia).
Propósito:	Permitir registrar y modificar los datos de

	un proyecto.
Resumen:	El caso de uso se inicia cuando el Líder de Proyecto selecciona la opción "Registrar Proyecto" el sistema muestra una interfaz con todos los tipos de gestiones que se pueden efectuar: registrar un nuevo proyecto o modificar los datos del proyecto. El Líder de Proyecto escoge la opción que desea realizar. El caso de uso culmina cuando el Líder de Proyecto termina con la opción seleccionada.
Referencias :	RF 1.
Precondiciones:	1. Tiene que haberse autenticado correctamente con los permisos necesarios.
	2. La opción registrar nuevo proyecto solo se ejecutará si el Líder de proyecto no ha registrado un proyecto anteriormente.
Curso normal de los eventos	
Acción del actor:	**Respuesta del Sistema**
1. Si el Líder de Proyecto no ha creado ningún proyecto.	1.1 El sistema muestra la opción de: ✓ Registrar proyecto.

Sección: Registrar proyecto.	
Curso normal de los eventos	
Acción del Actor	**Respuesta del Sistema**
2. El Líder de Proyecto selecciona la opción Registrar un nuevo Proyecto.	2.1 El sistema muestra el formulario para introducir los datos necesarios para adicionar. ✓ Nombre del proyecto. ✓ Cantidad de semanas. ✓ Facultad a la que pertenece.
3. El Líder de Proyecto introduce los datos requeridos por el sistema para la adición de un nuevo proyecto y selecciona la opción “Aceptar”.	3.1 ¨Está seguro que desea enviar.¨ 4.1 El sistema verifica la validez de los datos introducidos
3. El líder selecciona la opción “Aceptar”.	a. Si los datos son válidos, el sistema los adiciona en la base de datos y crea el proyecto.
Postcondiciones:	Se creó una instancia Proyecto.

Cursos Alternos	
Acción del actor	**Respuesta del Sistema**
1. a Si el Líder de Proyecto ya ha creado un proyecto.	El sistema muestra las opción de: ✓ Modificar datos del proyecto.
Sección: Modificar datos del proyecto.	
Curso normal de los eventos	
Acción del Actor	**Respuesta del Sistema**
2. a El Líder de Proyecto selecciona la opción Modificar datos del proyecto.	2.1 El sistema muestra el formulario para introducir los datos necesarios para modificar. ✓ Nombre del proyecto. ✓ Cantidad de semanas. ✓ Facultad a la que pertenece.
3. a El Líder de Proyecto introduce los datos requeridos por el sistema para la modificación del proyecto y luego selecciona la opción "Aceptar".	3.1 Está seguro que desea enviar.
4 .a El Líder de Proyecto selecciona la opción "Aceptar".	4.1 Si los datos son válidos, el sistema los modifica en la base de datos y actualiza el proyecto.

Cursos alternos	
Acción del Actor	**Respuesta del Sistema**
4.b	4.2 Si los datos no son válidos el sistema muestra un mensaje de error indicando que los datos no son válidos. "Proyecto ya existente".
4. c El Líder de Proyecto decide que no desea modificar los datos de un proyecto y presiona el botón "Cancelar".	4. 3 El sistema re direcciona al Líder de proyecto para la página de Inicio.
Postcondiciones:	Se definió para Proyecto.nombre el atributo nombre, para Proyecto.semanas el atributo semanas y para Proyecto.facultad el atributo facultad.
Prioridad:	Critico.

Tabla # 9: Descripción del caso de uso del sistema "Gestionar Proyectos Productivos".

Interfaz (Gestionar Proyecto)

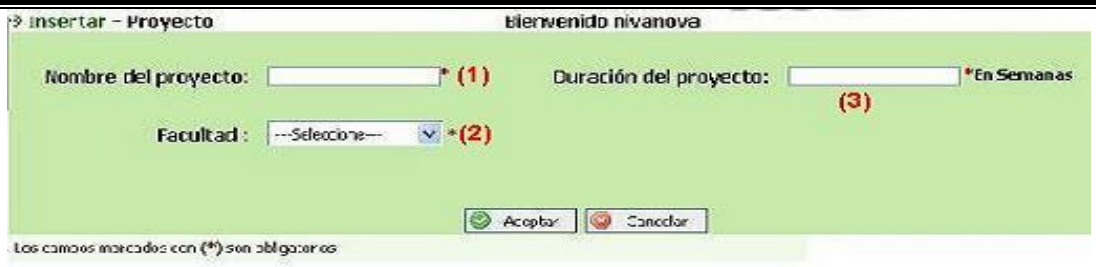

Figura # 21: Interfaz (Registrar Proyecto).

1-Nombre del Proyecto.

2-Facultad a la que pertenece.

3-Duración del proyecto (semanas).

2.8.3.4 CU Gestionar Tareas

Nombre del CU:	Gestionar Tareas.
Actores:	Líder de Proyecto. (Inicia).
Propósito:	Permitir al Líder de proyecto definir las tareas de su proyecto.
Resumen:	El caso de uso se inicia cuando el líder selecciona la opción ✓ Crear Tareas. ✓ Eliminar Tareas. ✓ Ver Tareas Creadas. ✓ Modificar Pesos.
Referencias :	RF 2.
Precondiciones:	Para crear una tarea el proyecto debe estar creado. La suma de todos los pesos de las tareas debe ser menor a 100.

Curso normal de los eventos	
Acción del actor:	**Respuesta del Sistema**
1. El líder de proyecto selecciona la opción ¨Crear Tareas¨.	1.1 El sistema muestra un formulario donde pide el nombre de la tarea y el peso.
2. El Líder de Proyecto introduce los datos que ha solicitado el sistema (Nombre y Peso) y presiona el botón “Aceptar”.	2.1 ¨Está seguro que desea enviar.¨
	2.2 Si los datos introducidos por el Líder de Proyecto son válidos el sistema guarda la tarea en la base de datos y mantiene la misma página hasta que las tareas no sean correctamente creadas.
Cursos Alternos	
2.a Si selecciona la opción ¨Cancelar¨	2.1 Re direcciona al Líder de Proyecto a la página de inicio.
2.b	2.2 Si los pesos introducidos por el Líder de Proyecto son mayores que 100, el sistema re direcciona para la página Modificar Pesos, en la cual el Líder de Proyecto deberá modificar los pesos de las tareas.
Postcondiciones	Se creó una instancia de tareas
Sección Eliminar Tareas	
Precondiciones:	Para Eliminar una tarea el proyecto debe estar creado.

	Deben de existir tareas creadas.
Acción del actor	**Respuesta del Sistema**
1. El líder de proyecto selecciona la opción ¨Eliminar Tareas¨.	1.1 Muestra todas las tareas creadas hasta el momento.
2. El líder de proyecto selecciona un tarea y presiona el botón “Aceptar”	2.1 ¨Está seguro que desea enviar¨.
3 Si presiona el botón “Aceptar”	3.1 El sistema elimina la tarea seleccionada y mantiene la misma página por si desea eliminar otra.
Cursos Alternos	
2.a El líder de proyecto selecciona un tarea y presiona el botón “Cancelar”	2.2 Mantiene al líder en la misma página.
3.b El líder de proyecto selecciona un tarea y presiona el botón “Cancelar”	3.2 Re direcciona al Líder de Proyecto a la página de inicio.
Postcondiciones	Se eliminó una instancia de tareas.
Sección Ver tareas Creadas	
Precondiciones:	Para ver tareas creadas el proyecto debe estar creado. Deben de existir tareas creadas.
Acción del Actor	**Respuesta del Sistema**

1 El líder de proyecto selecciona la opción Ver Tareas Creadas.	1.1 Muestra una lista con todas las tareas creadas hasta el momento con sus pesos correspondientes.
Postcondiciones	Se mostraron todas las instancias de tareas.
Sección Modificar Pesos	
Precondiciones:	Para modificar los pesos de una tarea el proyecto debe estar creado. Deben de existir tareas creadas.
Acción del actor:	**Respuesta del Sistema**
1. El líder de proyecto selecciona la opción ¨Modificar Pesos¨.	1.1 Muestra una lista con todas las tareas creadas hasta el momento con sus pesos correspondientes.
2. El Líder de Proyecto introduce los pesos que desea modificar y presiona el botón “Aceptar”.	2.1 ¨Está seguro que desea enviar.¨
	2.2 Si los datos introducidos por el Líder de Proyecto son válidos el sistema modifica el peso la tarea en la base de datos y mantiene la misma página hasta que las tareas no sean correctamente creadas.
Cursos Alternos	
2.a Si selecciona la opción ¨Cancelar¨	2.1 Re direcciona al Líder de Proyecto a la página de inicio.

2.b	2.2 Si los pesos introducidos por el Líder de Proyecto son mayores que 100, el sistema muestra un error¨ La suma de los pesos debe ser igual a 100¨.
Postcondiciones:	Se definió para tareas.peso el valor de peso.
Prioridad:	Secundario.

Tabla # 10: Descripción del caso de uso del sistema “Crear tareas”.

Interfaz (Crear Tarea)

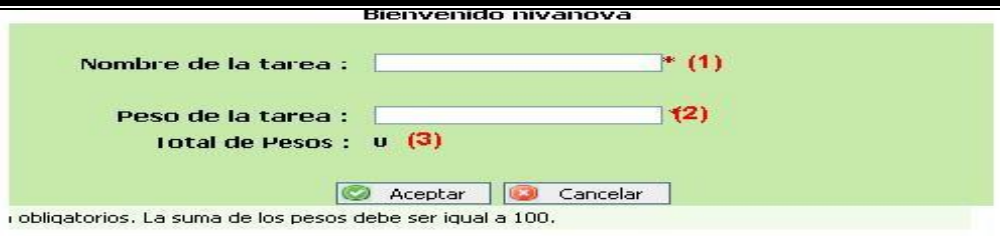

Figura # 22: Interfaz (Crear Tarea).

1-Nombre de la tarea

2-Peso de la tarea

3-Total de peso

Interfaz (Eliminar Tarea)

Figura # 23: Interfaz (Eliminar Tarea).

1-Nombre de la tarea a eliminar.

Interfaz (Modificar Peso)

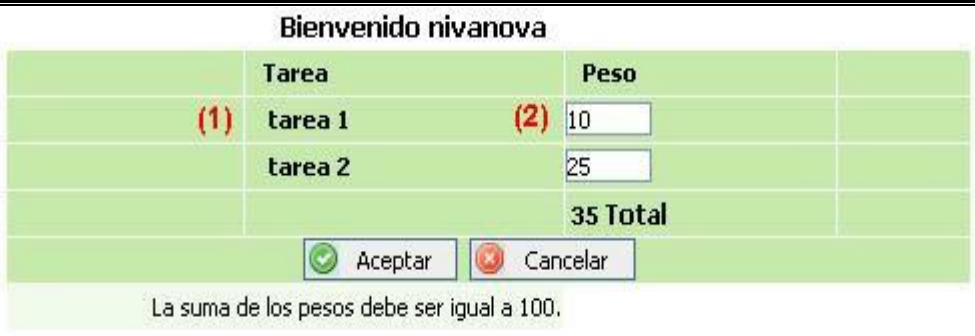

Figura # 24: Interfaz (Modificar Peso).

1-Nombre de la tarea.

2-Peso de la tarea.

Interfaz (Modificar Peso)

Figura # 25: Interfaz (Modificar Peso).

1-Nombre de la tarea.

2-Peso de la tarea.

2.8.3.5 CU Crear Módulos

Nombre del CU:	Crear Módulos.
Actores:	Líder de Proyecto. (Inicia).
Propósito:	Permitir al Líder de proyecto definir los Módulos de su proyecto.

Resumen:	El caso de uso se inicia cuando el líder de proyecto selecciona la opción ¨Crear Módulos¨. ✓ Nombre del módulo.
Referencias :	RF 3.
Precondiciones:	El proyecto tiene que estar creado.
Curso normal de los eventos	
Acción del actor:	**Respuesta del Sistema.**
1. El líder de proyecto selecciona la opción ¨Crear Módulos¨.	1.1 El sistema muestra un formulario donde pide el nombre del módulo.
2. El Líder de Proyecto introduce los datos que ha solicitado el sistema (Nombre del módulo) y presiona el botón “Aceptar”.	2.1 Esta seguro que desea enviar. 2.2 Si los datos introducidos por el Líder de Proyecto son válidos el sistema guarda el módulo en la base de datos y mantiene la misma página.
Cursos Alternos.	
Acción del actor.	**Respuesta del Sistema.**
2. a Si el líder de proyecto selecciona la opción ¨Cancelar.¨	2.2 Re direcciona a la página de inicio.
Postcondiciones:	Se creó una instancia de módulo.

Prioridad:	Crítico.

Tabla # 11: Descripción del caso de uso del sistema “Crear Módulos”.

Interfaz (Crear Módulo)

Figura # 26: Interfaz (Crear Módulo).

1-Nombre del Módulo.

2.8.3.6 CU Crear CU

Nombre del CU:	Crear CU.
Actores:	Líder de Proyecto. (Inicia).
Propósito:	Permitir al Líder de proyecto definir los CU de cada Módulo.
Resumen:	El caso de uso se inicia cuando el líder de proyecto selecciona la opción ¨Crear CU¨.

	✓ Nombre del CU. ✓ Complejidad del CU. ✓ Módulo al que pertenece. ✓ Desarrollador.
Referencias :	RF 4.
Precondiciones:	Los módulos tienen que estar previamente creados.
Curso normal de los eventos.	
Acción del actor:	**Respuesta del Sistema.**
1. El líder de proyecto selecciona la opción ¨Crear CU¨.	1.1 El sistema muestra un formulario donde pide al usuario que llene los datos de CU. ✓ Nombre del CU. ✓ Complejidad. ✓ Módulo al que pertenece. ✓ Desarrollador.
2. El Líder de Proyecto introduce los datos que ha solicitado el sistema (Nombre, complejidad y Módulo al que pertenece) y presiona el botón “Aceptar”.	2.1 Está seguro que desea enviar. 2.2 Si los datos introducidos por el Líder de Proyecto son válidos el sistema guarda el CU en la base de datos y muestra la misma página.

Cursos Alternos	
Acción del actor	**Respuesta del Sistema**
2. a El Líder de Proyecto decide que no desea adicionar el CU creado presiona el botón “Cancelar”.	2.2 Re direcciona a la página de inicio.
Postcondiciones:	Se creó una instancia de CU.
Prioridad:	Secundario.

Tabla # 12: Descripción del caso de uso del sistema “Crear CU”.

Interfaz (Crear CU)

Figura # 27: Interfaz (Crear CU).

1-Nombre del CU.

2-Módulo que pertenece.

3-Complejidad del CU.

4-Nombre del desarrollador.

2.8.3.7 CU Graficar

Nombre del CU:	Graficar.
Actores:	Líder de Proyecto y Directivo de la UCI. (Inicia).
Propósito:	Permitir al Líder de proyecto obtener gráficos relacionados con el avance del proyecto
Resumen:	El caso de uso se inicia cuando el líder de proyecto selecciona la opción ¨Graficar Estado de Avance¨.
Referencias :	RF 5.
Precondiciones:	El por ciento de avance tiene que ser mayor que 0.
Curso normal de los eventos	
Acción del actor:	**Respuesta del Sistema**
1. El líder de proyecto selecciona la opción ¨Graficar Estado de Avance¨.	1.1 El sistema muestra las opciones de Graficar. ✓ Graficar Estado de Avance del proyecto por módulos. ✓ Graficar Estado de Avance del proyecto por semanas.

Sección: Graficar Estado de Avance por Módulos	
Curso normal de los eventos	
Acción del Actor	**Respuesta del Sistema**
2 El líder de proyecto o el Directivo de la UCI selecciona la opción ¨Graficar Estado de avance Por Módulos¨.	2.1 El sistema muestra una gráfica de barras donde se representa el por ciento de avance por módulos.
3 El Líder de proyecto o el Directivo de la UCI recibe la gráfica.	1.1
Postcondiciones:	Se mostró el estado de avancemódulos.
Sección: Graficar Estado de Avance del proyecto por semanas.	
Curso normal de los eventos	
Acción del Actor	**Respuesta del Sistema**
2 El líder de proyecto selecciona la opción: ¨ Graficar Estado de Avance del proyecto por semanas¨.	2.1 El sistema muestra una gráfica de líneas donde se representa el por ciento de avance del proyecto por semanas.
3 El Líder de proyecto recibe la gráfica.	1.1
Postcondiciones:	Se mostró el estado de avanceproyecto.
Prioridad:	Secundario.

Tabla # 13: Descripción del caso de uso del sistema “Graficar”.

Interfaz (Graficar)

Figura # 28: Interfaz (Graficar).

1-Avance de los Módulos.

2-Avance de los CU.

2.8.3.8 CU Mostrar Información

Nombre del CU:	Mostrar Información.
Actores:	Líder de Proyecto y el Directivo de la UCI. (Inicia).
Propósito:	Permitir al Líder de Proyecto obtener información sobre el estado de avance, el por ciento de avance y el estado de avance por semana de su proyecto así como el estado de avance en que se

	encuentra un módulo, el porciento de completamiento de cada módulo y el estado de completamiento de cada CU de su proyecto. El Directivo de la UCI tendrá que seleccionar un proyecto y podrá obtener toda la información antes expuesta.
Resumen:	El caso de uso se inicia cuando un usuario se autentica, ya sea un Líder de Proyecto o un Directivo de la UCI, si es un Líder de Proyecto y este ya ha creado un proyecto podrá tener acceso a toda la información del mismo, pero si el usuario es un Directivo de la UCI tendrá que seleccionar un proyecto y desde ese momento podrá obtener toda la información del avance referente al mismo.
Referencias :	RF 7.
Precondiciones:	El proyecto debe estar creado.
Curso normal de los eventos.	
Acción del actor:	**Respuesta del Sistema**
1. Si el usuario que se registra es un Directivo de la UCI.	1.1 El sistema muestra un listado con todos los proyectos existentes en la base de datos y la opción de ¨Seleccionar Proyecto ¨ .

2. El Directivo selecciona un proyecto.	2.1 El sistema muestra un listado de la opciones que puede seleccionar: ✓ Estado de avance del proyecto. ✓ Estado de avance se del proyecto por semanas. ✓ Estado de avance de los módulos del proyecto. ✓ Estado de avance de los CU del proyecto.
Sección: Estado de avance del proyecto	
Curso normal de los eventos	
Acción del Actor	**Respuesta del Sistema**
3 El Directivo selecciona la opción ¨Estado de avance del proyecto¨.	3.1 El sistema muestra en qué estado de avance se encuentra el proyecto, que no es más que saber si el proyecto está Atrasado o en Tiempo y su por ciento de avance.
Postcondiciones:	Se mostró avanceproyecto.
Sección: Estado de avance del proyecto por semanas.	
Curso normal de los eventos	

Acción del Actor	**Respuesta del Sistema**
1 El Directivo selecciona la opción ¨Estado de avance del proyecto por semana ¨.	1.1 El sistema muestra en qué estado de avance se encuentran los módulos del proyecto por semana.
Postcondiciones:	Se mostró avanceproyectoporsemanas.
Sección: Estado de avance de los módulos.	
Curso normal de los eventos	
Acción del Actor	**Respuesta del Sistema**
1 El Directivo selecciona la opción ¨Estado de avance de los un módulos ¨.	1.1 El sistema muestra en qué estado de avance se encuentran los módulos del proyecto.
Postcondiciones:	Se mostró avancemódulos.
Sección: Estado de avance de los CU del proyecto.	
Curso normal de los eventos	
Acción del Actor	**Respuesta del Sistema**
1 El Directivo selecciona la opción ´Estado de avance de los CU del proyecto ¨.	1.1 El sistema muestra en el Porciento de completamiento que esta cada CU del proyecto.
Pos condiciones:	Se mostró avanceCU.
Cursos Alternos.	

Acción del actor:	**Respuesta del Sistema**
1. Si el usuario que se registra es un Líder de Proyecto.	1.1 El sistema muestra un listado de la opciones que puede seleccionar: ✓ Estado de avance del proyecto. ✓ Estado de avance se del proyecto por semanas. ✓ Estado de avance de los módulos del proyecto. ✓ Estado de avance de los CU del proyecto. .
Sección: Estado de avance del proyecto	
Curso normal de los eventos	
Acción del Actor	**Respuesta del Sistema**
1 El Líder selecciona la opción ¨Estado de avance del proyecto¨.	1.1 El sistema muestra en qué estado de avance se encuentra el proyecto, que no es más que saber si el proyecto está Atrasado o en Tiempo y su por ciento de avance.
Pos condiciones:	Se mostró avanceproyecto.
Sección: Estado de avance del proyecto	

por semanas.	
Curso normal de los eventos	
Acción del Actor	**Respuesta del Sistema**
1 El Líder selecciona la opción ¨Estado de avance del proyecto por semana ¨.	1.1 El sistema muestra en qué estado de avance se encuentran los módulos del proyecto por semana.
Pos condiciones:	Se mostró avanceproyectoporsemanas.
Sección: Estado de avance de los módulos.	
Curso normal de los eventos	
Acción del Actor	**Respuesta del Sistema**
1 El Líder selecciona la opción ¨Estado de avance de los un módulos ¨.	1.1 El sistema muestra en qué estado de avance se encuentran los módulos del proyecto.
Pos condiciones:	Se mostró avancemódulos.
Sección: Estado de avance de los CU del proyecto.	
Curso normal de los eventos	
Acción del Actor	**Respuesta del Sistema**
1 El Líder selecciona la opción ´Estado de avance de los CU del proyecto ¨.	1.1 El sistema muestra en el Porciento de completamiento que esta cada CU del proyecto.

Pos condiciones:	Se mostró avanceCU.

Tabla # 14: Descripción del caso de uso del sistema “Mostrar Información”.

Interfaz (Mostrar Información Sección Avance de los Módulos)

Figura # 29: Interfaz (Avance por módulos).

1-Nombre del módulo.

2-Por ciento de Avance.

3-Estado de Avance.

Interfaz (Mostrar Información Sección Avance de los Módulos por semanas)

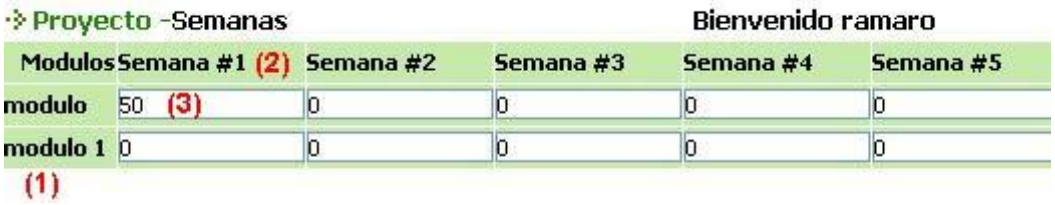

Figura # 30: Interfaz (Mostrar Información Sección Avance de los Módulos por semanas).

1-Nombre del módulo.

2-Numero de la semana.

3- Por ciento de Avance.

Interfaz (Mostrar Información Sección Avance del proyecto)

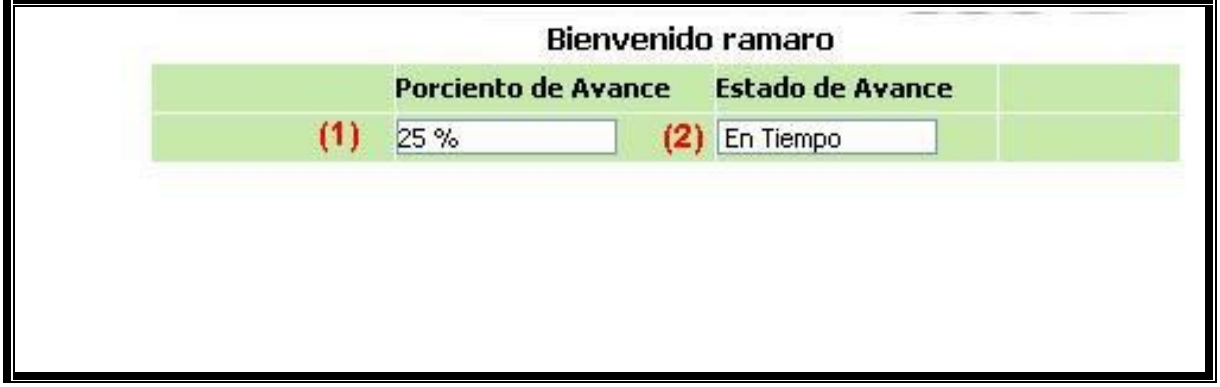

Figura # 31: Interfaz (Mostrar Información Sección Avance del proyecto).

1-Porciento de Avance.

2-Estado de Avance.

Interfaz (Mostrar Información Sección Avance de los CU)

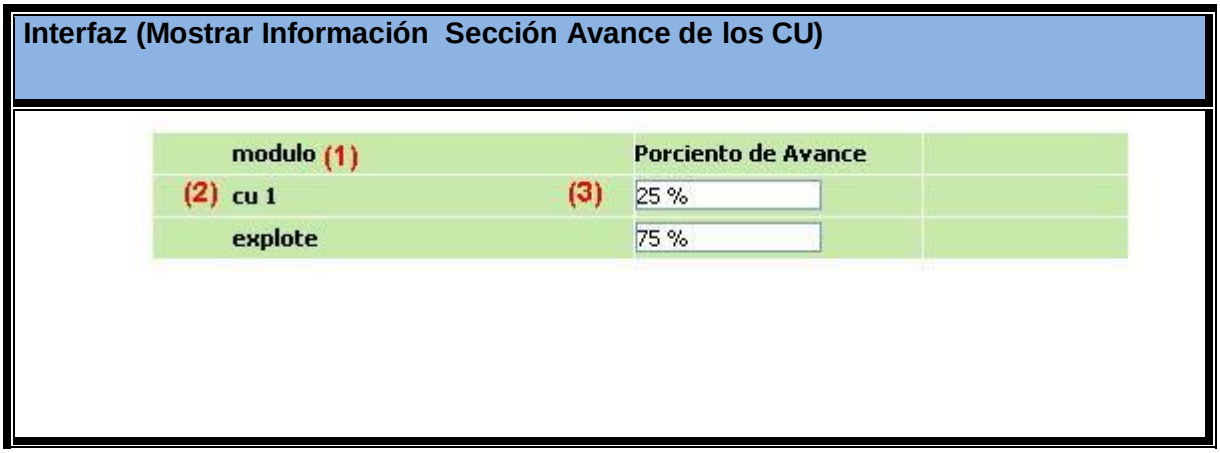

Figura # 32: Interfaz (Mostrar Información Sección Avance de los CU).

1-Nombre del módulo.

2-Nombre del CU.

3-Porciento de Avance.

Interfaz (Mostrar Información Sección Datos de Proyecto)

Figura # 33: Interfaz (Mostrar Información Sección Datos de Proyecto).

1-Nombre del Proyecto.

2-Nombre del Líder.

3-Facultad a la que pertenece.

4-Fecha de Inicio.

5-Duración del proyecto (semanas).

2.8.3.9 CU Complementar CU

Nombre del Caso de Uso:	Complementar CU.
Actores:	Implementador (inicia).
Propósito:	El Implementador y el Líder serán capaces de saber en qué estado de completamiento tiene cada CU con respecto a cada una de las tareas.
Resumen:	El CU inicia cuando el Implementador o el Líder eligen la opción de ¨Dar Cumplimiento a los CU¨, el sistema muestra todos los CU dentro de cada módulo y todas las tareas que tienen que cumplir los CU para estar realizados, este estado de cumplimiento está entre 0 y 1, ellos modifican y guardan los datos en la base de datos.
Referencias:	RF 8.
Precondiciones:	Tiene que estar el Proyecto creado con todas sus Tareas, Módulos y CU.
Curso normal de los eventos	
Acción del Actor	**Respuesta del Sistema**
1 El Implementador selecciona la opción ¨Dar cumplimiento a los CU¨.	1.1 El sistema muestra una tabla con todos los módulos, dentro de cada módulo los CU existentes y una lista de las tareas a realizar.
2. El Implementador modifica los valores del estado de cumplimiento de los CU	2.1 Está seguro que desea enviar

con que desea trabajar ejecuta la opción ¨Aceptar¨.	
3. El Implementador selección la opción ¨Aceptar¨.	2.2 El sistema registra esos cambios en la base de datos.
Cursos alternos	
Acción del Actor	**Respuesta del Sistema**
2. a Selecciona la opción ¨Cancelar¨.	2.1 El sistema muestra la misma página.
3. a Selecciona la opción ¨Cancelar¨.	3.1 El sistema re direcciona a la página de inicio.
Postcondiciones	Se creó una instancia de cumplimientoCU.
Sección: Líder da cumplimiento a los CU	
Curso normal de los eventos	
Acción del Actor	**Respuesta del Sistema**
1 El Líder de Proyecto selecciona la opción ¨Dar cumplimiento a los CU¨.	1.1 El sistema muestra una tabla con todos los módulos, dentro de cada módulo los CU existentes y una lista de las tareas a realizar.
2 El Líder de Proyecto selecciona el módulo que desea completar	2.1 El sistema muestra una tabla con todos los datos insertados hasta el momento.

3. El Líder de Proyecto modifica los valores del estado de cumplimiento de los CU en caso que lo necesite y pulsa la opción ¨Aceptar¨.	3.1 Está seguro que desea enviar
4. El Líder de Proyecto selección la opción ¨Aceptar¨.	4.1 El sistema registra esos cambios en la base de datos.
Cursos alternos	
Acción del Actor	**Respuesta del Sistema**
3. a Selecciona la opción ¨Cancelar¨.	3.1 El sistema muestra la misma página.
4. a Selecciona la opción ¨Cancelar¨.	4.1 El sistema re direcciona a la página de inicio.
Postcondiciones	Se creó una instancia de cumplimientoCU.
Prioridad:	Secundaria.

Tabla # 15: Descripción del caso de uso del sistema “Complementar los CU”.

Interfaz (Mostrar Información Sección Datos de Proyecto)

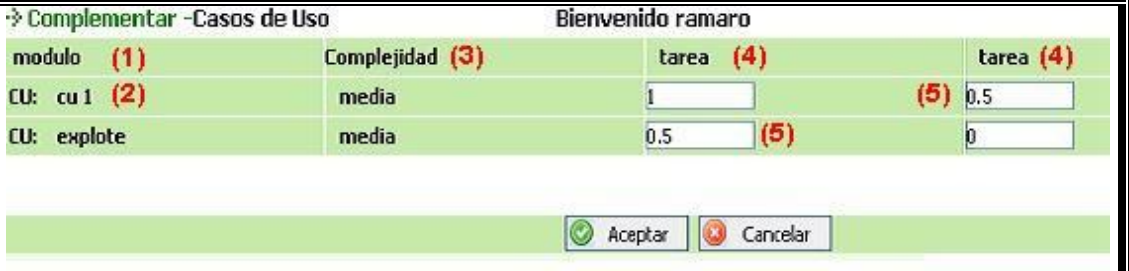

Figura # 34: Interfaz (Cumplir CU).

1-Nombre del Módulo

2-Nombre del CU.

3-Complejidad del CU.

4-Nombre de las Tareas

5-Porciento de cumplimiento (Valor entre 0 y 1).

2.9 Conclusiones parciales

En este capítulo, a partir de la comprensión de los procesos de negocio, se definieron las principales funcionalidades que debe tener el sistema a desarrollar, estructurándose en casos de uso. Se elaboró el diagrama de casos de uso del sistema y se describieron textualmente cada uno ellos. Con todo lo antes expuesto se tiene una noción de que es lo que se debe hacer y cómo, en la realización de la herramienta propuesta.

Capítulo 3: Análisis y Diseño.

3.1 Introducción

En este capítulo se muestran los resultados arrojados durante el flujo de trabajo de Análisis y Diseño del sistema a través de los diagramas de clases del análisis y del diseño, además de los diagramas de interacción y del diseño de la base de datos.

3.2 Análisis del sistema

El objetivo fundamental del flujo de análisis consiste en obtener una visión del sistema orientada en ver qué hace, de modo que sólo se interesa por los requisitos funcionales. Las actividades de análisis se desarrollan para facilitar la entrada al diseño, por lo que se convierten en un paso inicial y en una primera aproximación conceptual donde una vez comprendido los requisitos a este nivel, se aumenta la especificidad en aras de garantizar el cubrimiento de los requisitos funcionales y no funcionales. Para un mejor entendimiento es fundamental realizar el diagrama de clases del análisis.

3.2.1 Diagramas de clases del análisis

Un diagrama de clases del análisis representa los conceptos en un dominio del problema, las cosas del mundo real, no de la implementación automatizada de estas cosas. A continuación se muestran los diagramas de clases del análisis:

Diagramas de clases del análisis:

Diagrama de clases del análisis CU: Autenticarse.

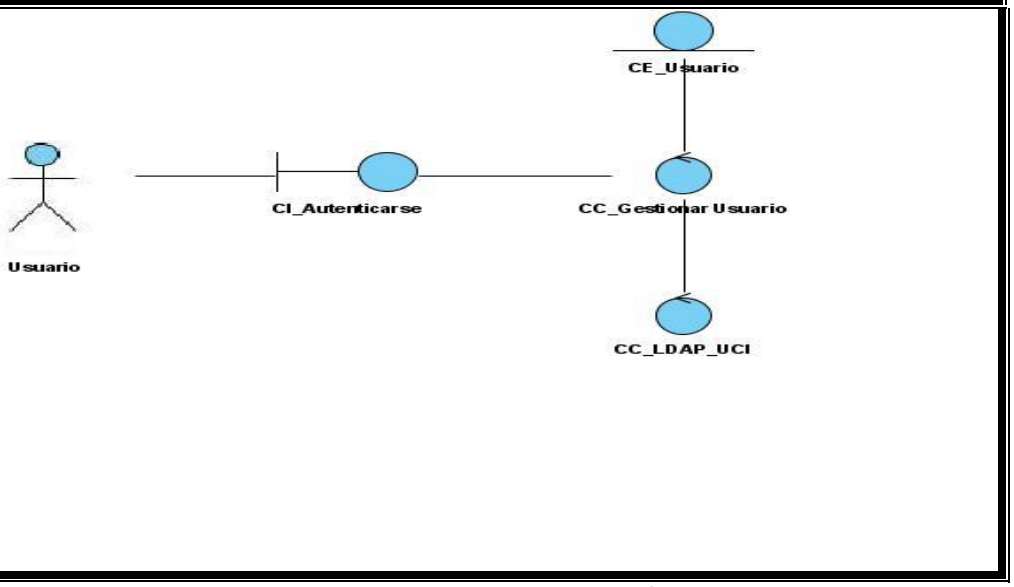

Figura # 35: Diagrama de clases del Análisis CU: Autenticar.

Diagrama de clases del análisis CU: Gestionar Usuario.

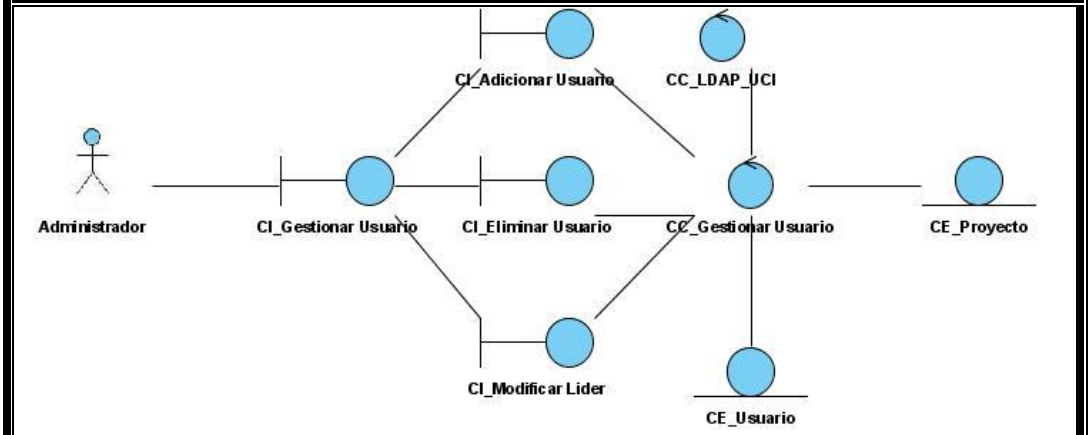

Figura # 35: Diagrama de clases del Análisis CU: Gestionar Usuario.

Diagrama de clases del análisis CU: Gestionar Proyecto.

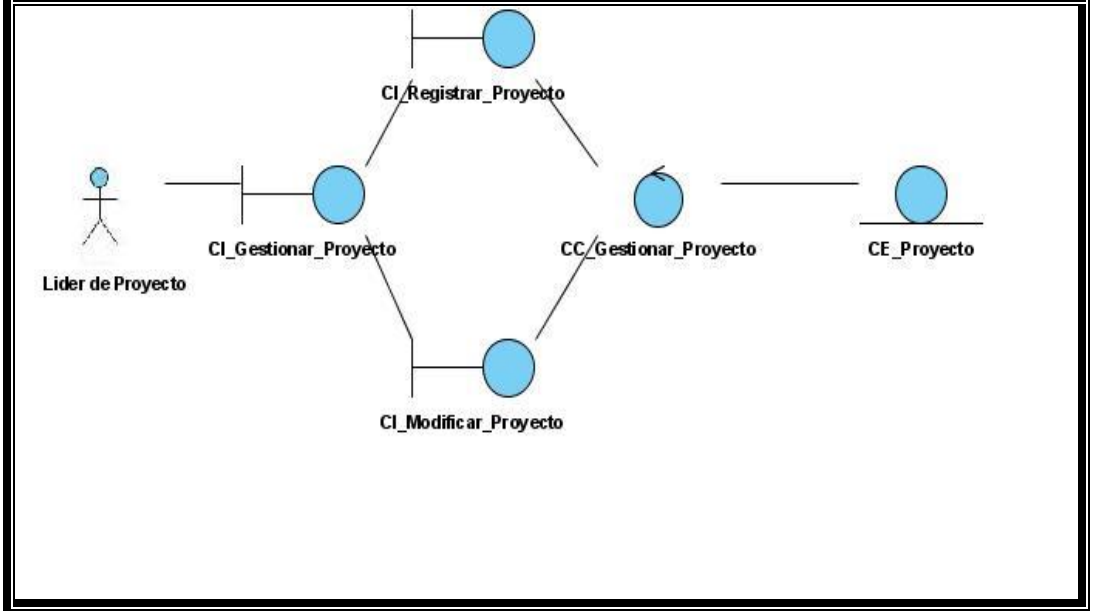

Figura # 36: Diagrama de clases del Análisis CU: Gestionar Proyecto.

Diagrama de clases del análisis CU: Gestionar Tareas.

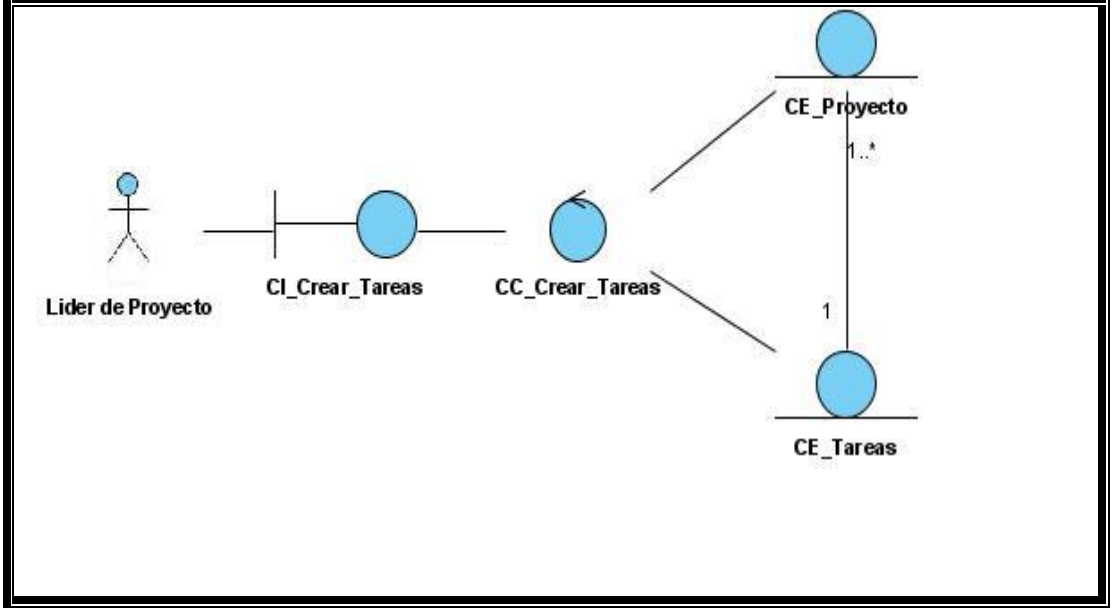

Figura # 37: Diagrama de clases del Análisis CU: Gestionar Tareas.

Diagrama de clases del análisis CU: Crear Módulos.

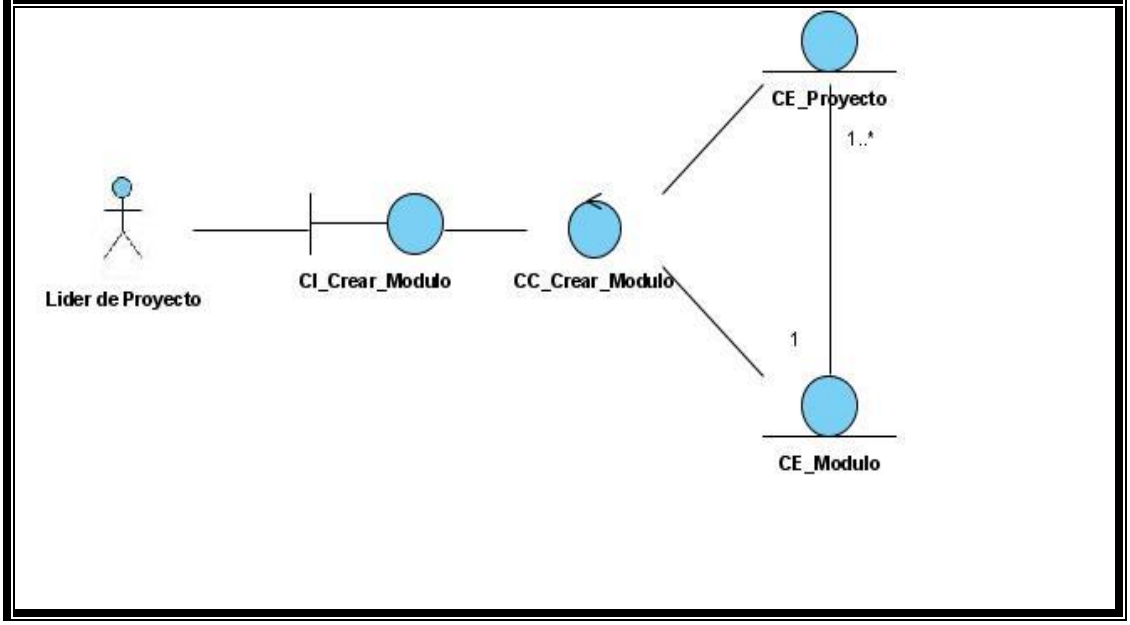

Figura # 38: Diagrama de clases del Análisis CU: Crear Módulos.

Diagrama de clases del análisis CU: Crear CU.

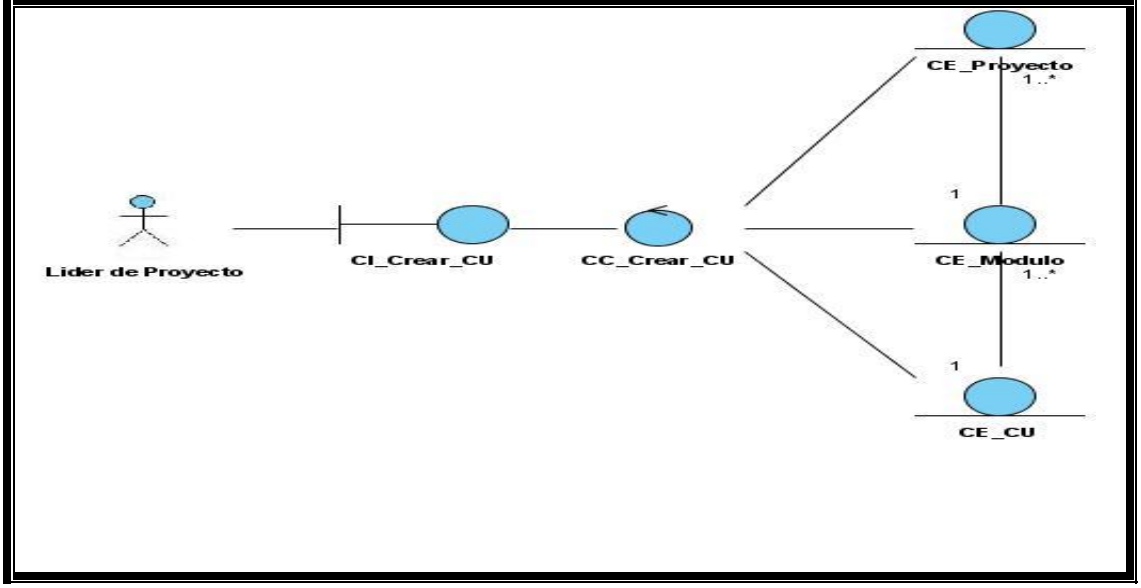

Figura # 39: Diagrama de clases del Análisis CU: Crear CU.

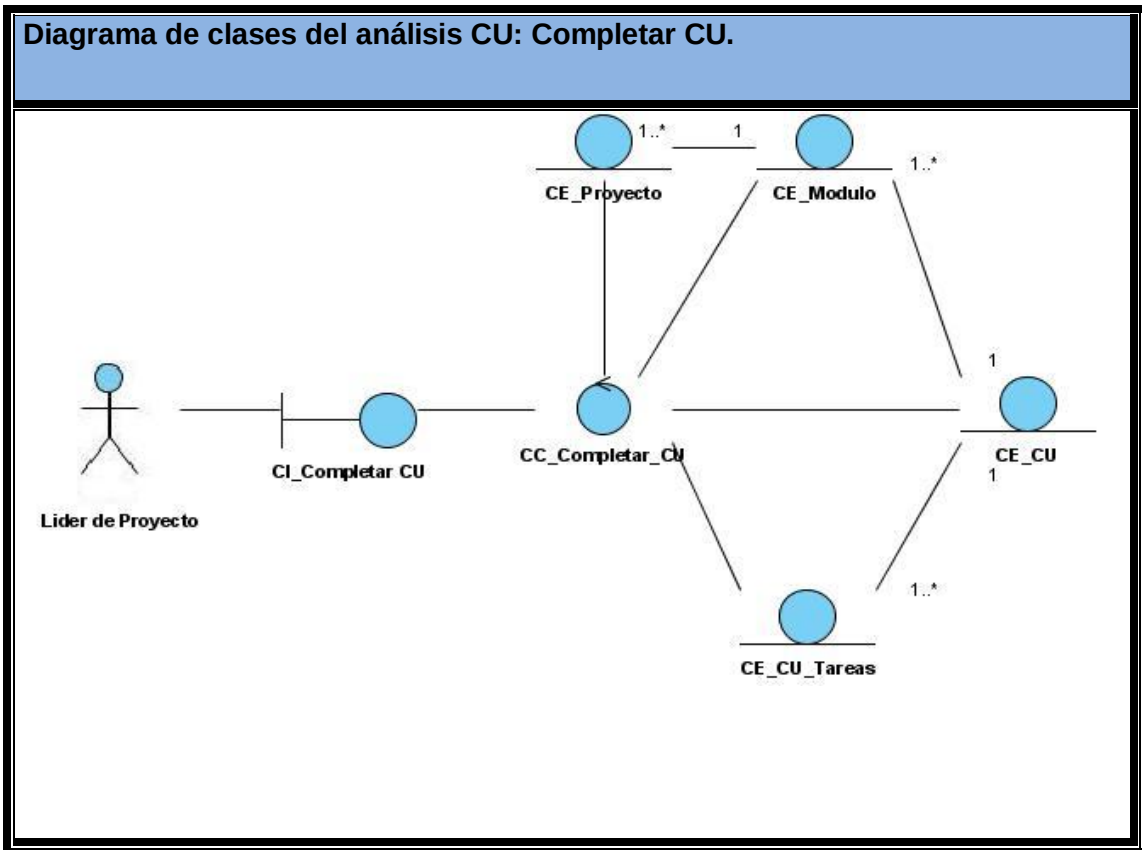

Figura # 40: Diagrama de clases del Análisis CU: Completar CU.

Una vez construidos los diagramas de clases del análisis se puede especificar el diseño del sistema, factor fundamental en este capítulo, que no es más que transformar los requerimientos en un diseño de cómo debe ser el sistema, desarrollar una robusta arquitectura del sistema y adaptar el diseño para que se corresponda con el entorno de implementación, diseñando sus funcionalidades.

3.3 Diseño del Sistema

Mediante el diseño se realiza un refinamiento del análisis, teniendo en cuenta los requisitos no funcionales, que no es más que ver cómo cumple el sistema sus objetivos y considerando además el entorno de implementación (Lenguaje de programación, plataforma, Sistemas heredados con los cuales interactuar,

etc.), los principales aspectos que se deben tener en cuenta a la hora de realizar el diseño del sistema están, los diagramas de clases del diseño, los diagramas de Interacción, el diseño de la Base de Datos, las definiciones de diseño que se apliquen, el tratamiento de errores, la seguridad y la interfaz.

Una parte muy importante a tener en cuenta a la hora de realizar el diseño del sistema es aplicar la arquitectura correctamente. Para la realización del sistema MetSoft específicamente se aplicó el patrón arquitectónico modelo-vista-controlador como se muestra a continuación:

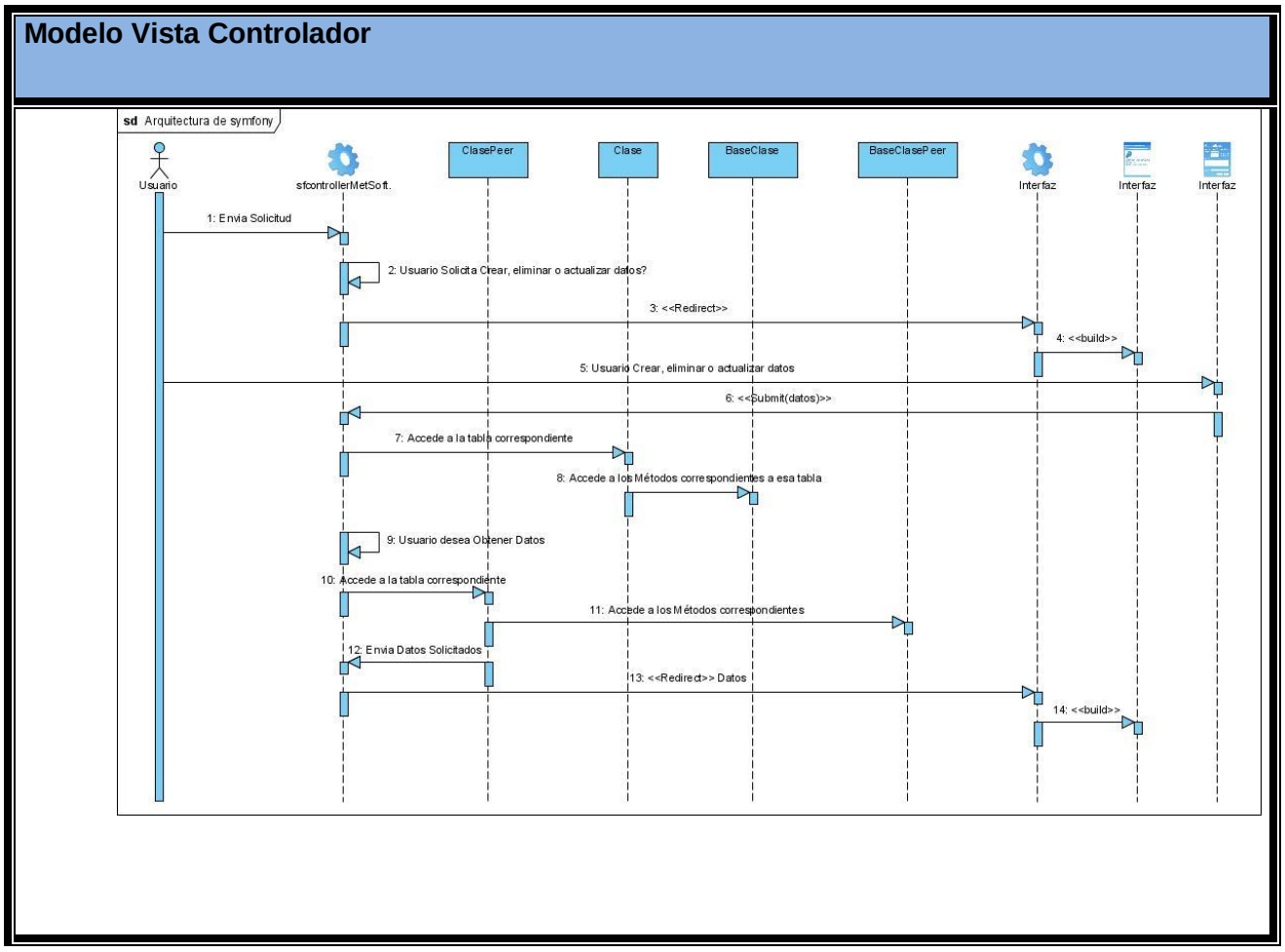

Figura # 41: Modelo Vista Controlador.

A continuación se muestra como es el funcionamiento del controlador:

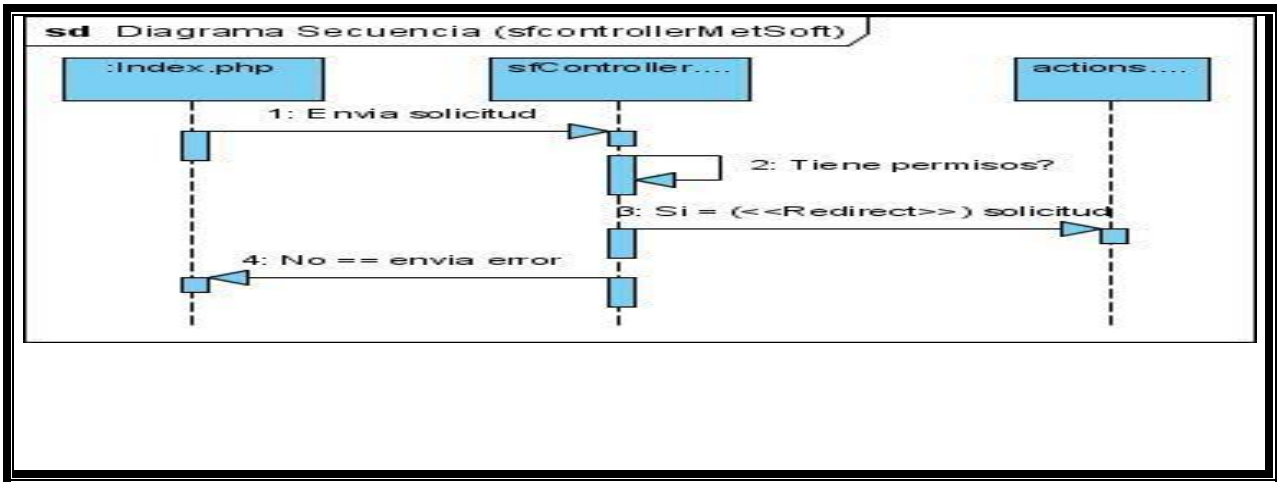

Figura # 42: Diagrama de Secuencia: sfcontrollerMetSoft.

3.3.1 Diagramas de clases del diseño

Un diagrama de clases muestra un conjunto de clases, interfaces y colaboraciones con sus relaciones estructurales y de herencia. Gráficamente es una colección de vértices y arcos. En el caso de las aplicaciones Web, representa las colaboraciones que ocurren entre las páginas, donde cada página lógica puede ser representada como una clase. En este tipo de aplicaciones son más importantes la modelación de la lógica y estado del negocio que los detalles de presentación.

Diagramas de clases del diseño Web:

Diagramas de clases del diseño CU: Autenticarse

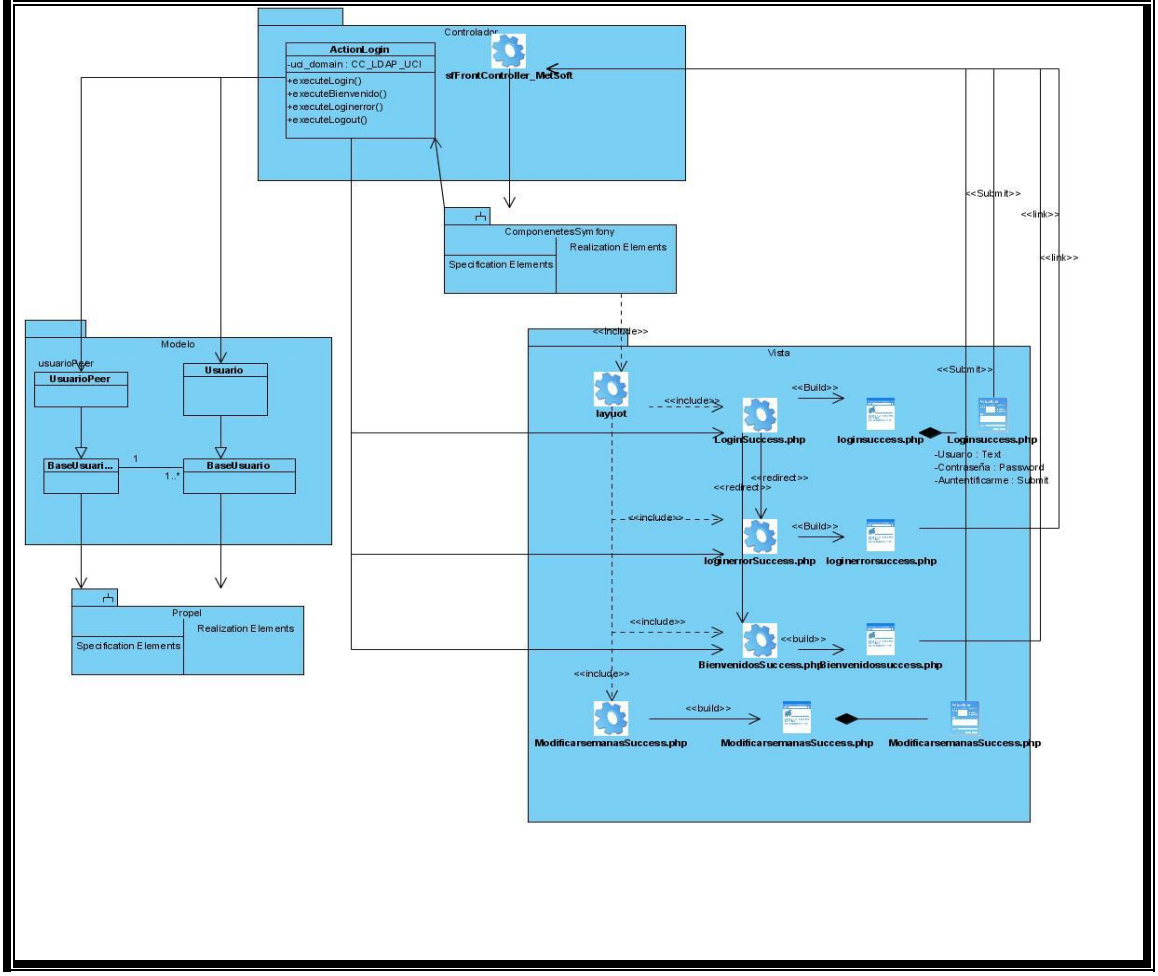

Figura # 43: Diagrama de clases del diseño CU: Autenticarse.

Diagramas de clases del diseño CU: Gestionar Usuarios.

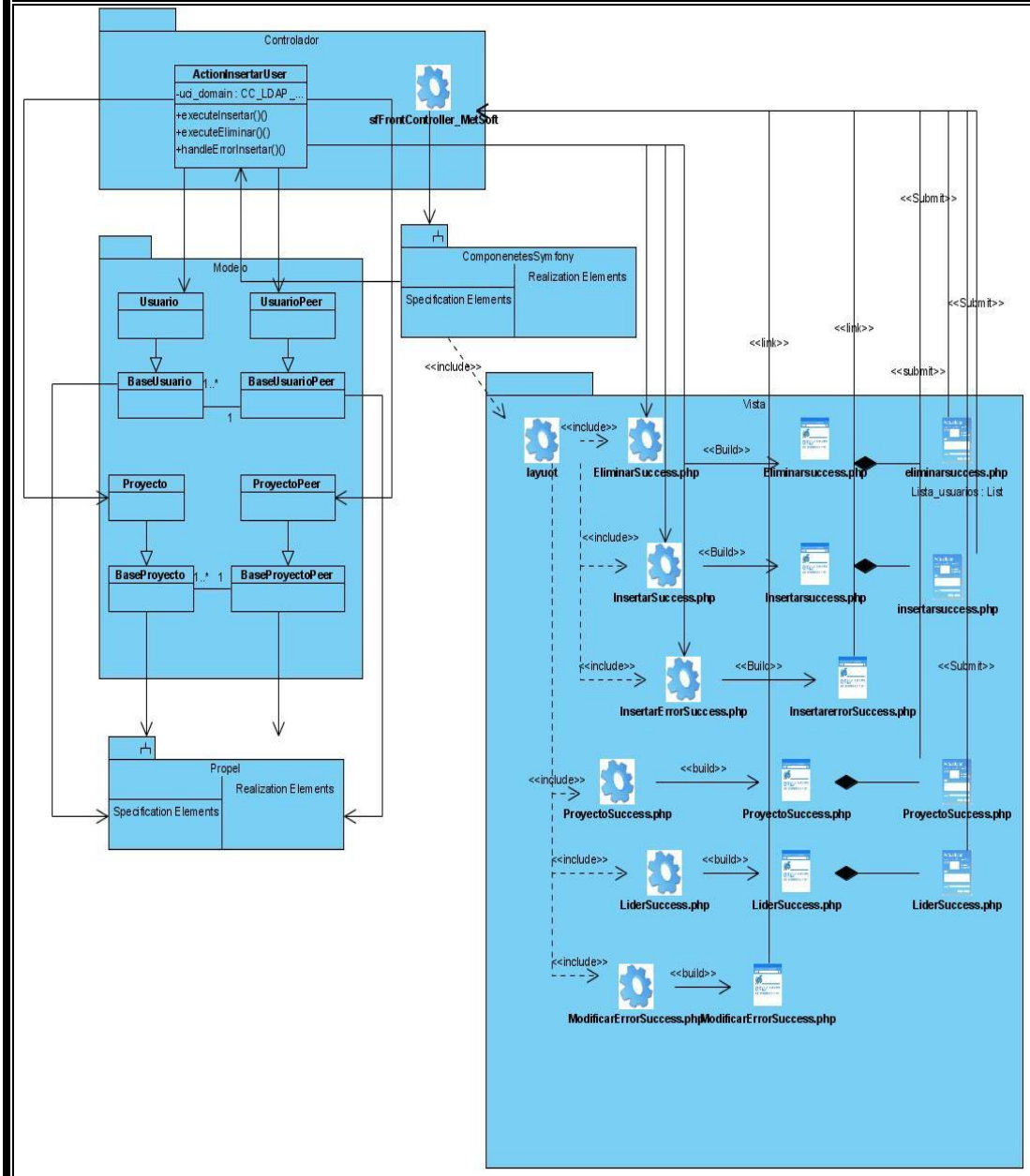

Figura # 44: Diagrama de clases del diseño CU: Gestionar Usuarios.

Diagramas de clases del diseño CU: Gestionar Proyecto.

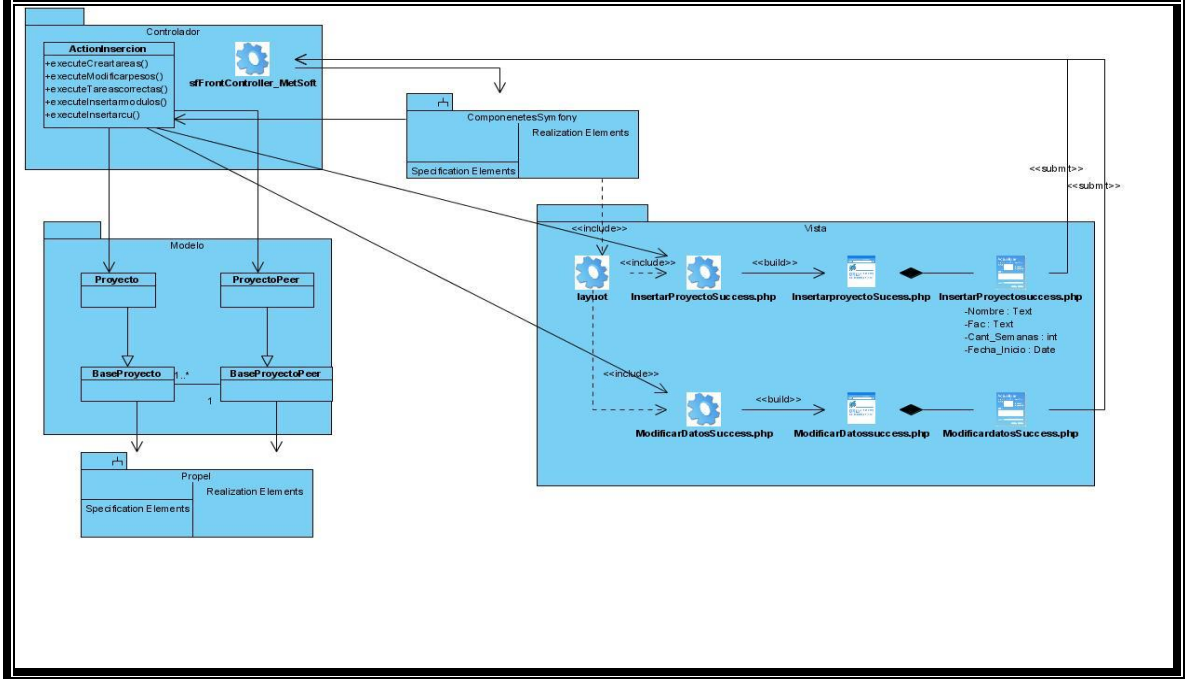

Figura # 45: Diagrama de clases del diseño CU: Gestionar Proyecto.

Diagramas de clases del diseño CU: Gestionar Tarea.

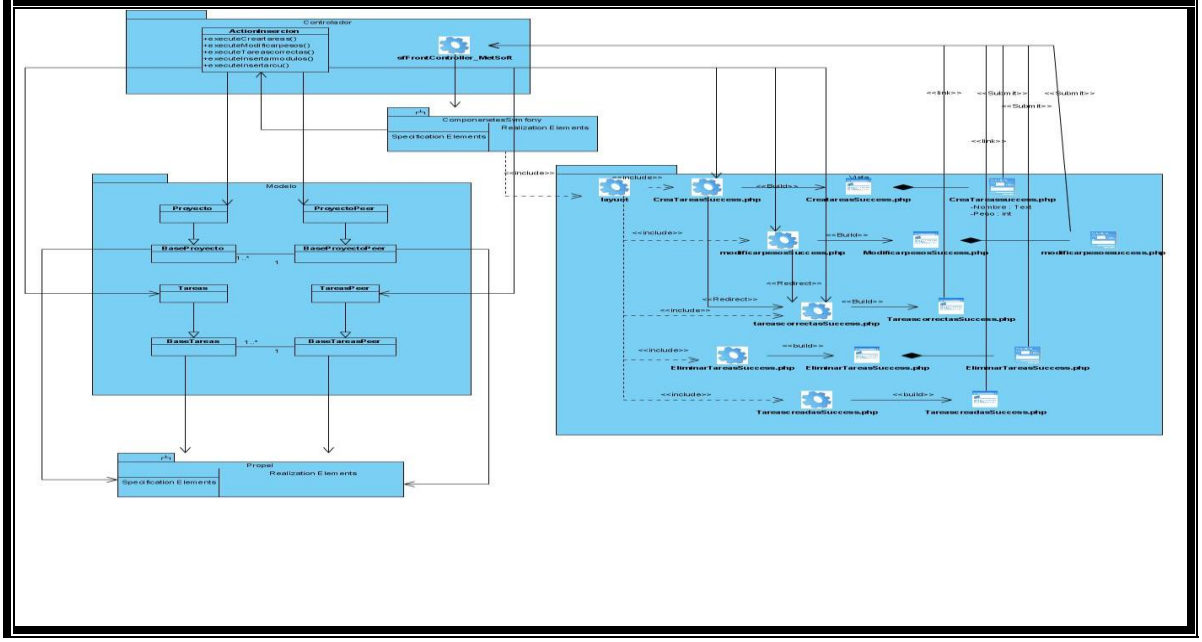

Figura # 46: Diagrama de clases del diseño CU: Gestionar Tarea.

Diagramas de clases del diseño CU: Crear Módulos.

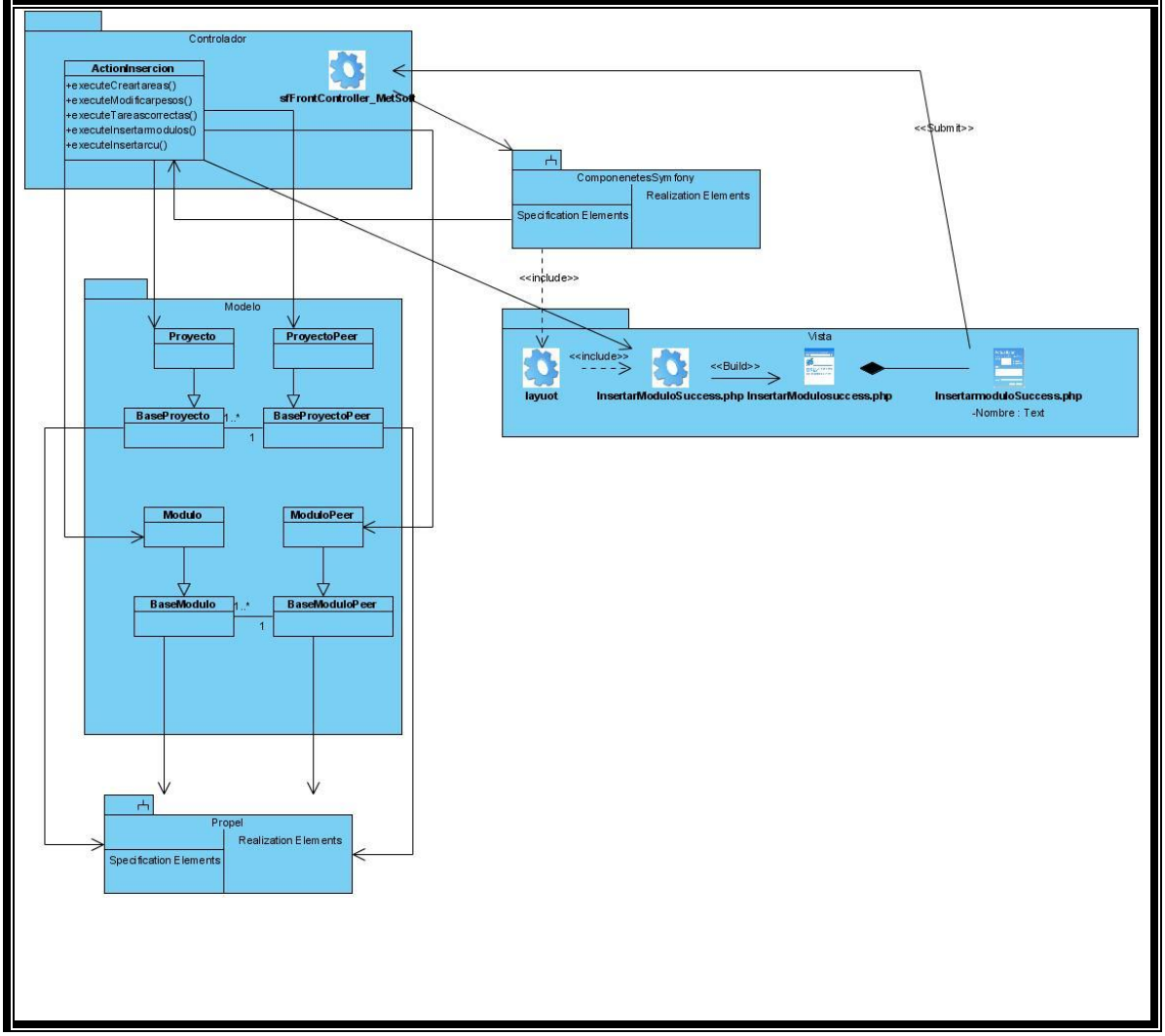

Figura # 47: Diagrama de clases del diseño CU: Crear Módulos.

Diagramas de clases del diseño CU: Crear CU.

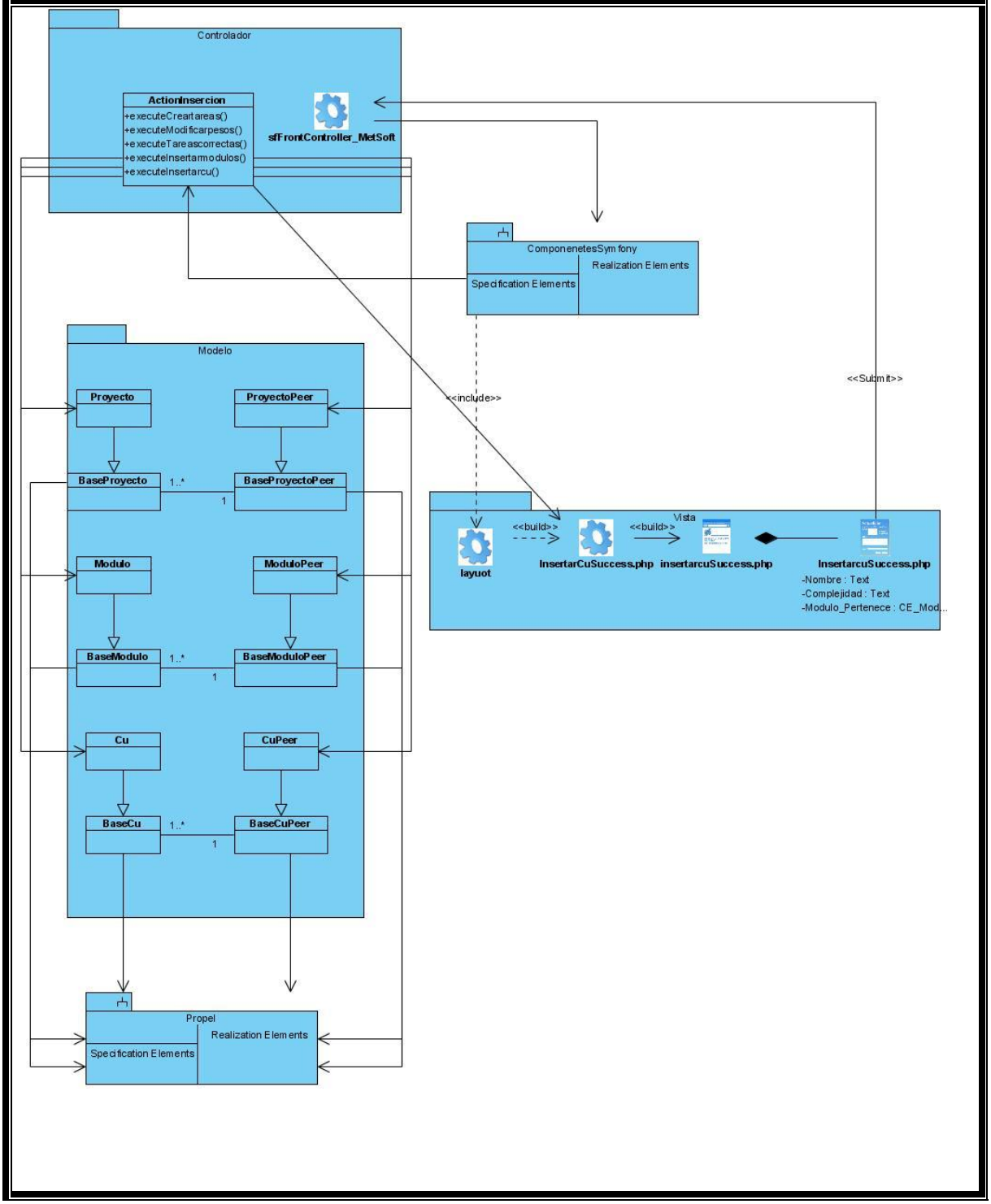

Figura # 48: Diagrama de clases del diseño CU: Crear CU.

Diagramas de clases del diseño CU: Completar CU.

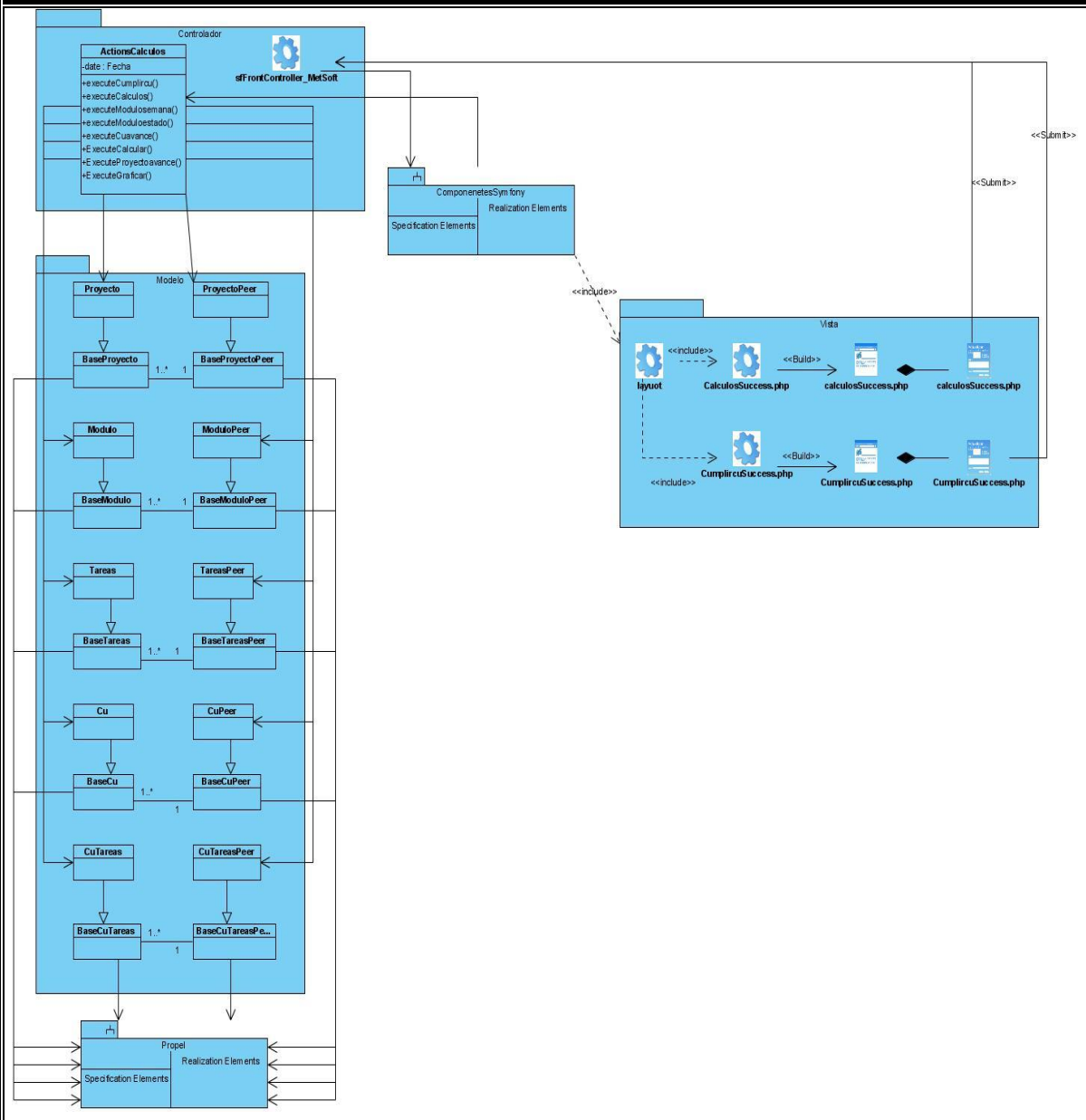

Figura # 49: Diagrama de clases del diseño CU: Completar CU.

3.3.2 Diagramas de Interacción

Los diagramas de secuencia y colaboración (diagramas de interacción) son diagramas de UML utilizados para modelar los aspectos dinámicos de un sistema y para construir sistemas ejecutables por medio de ingeniería directa e inversa.

Un diagrama de secuencia es un diagrama de interacción que destaca la ordenación temporal de los mensajes y muestra interacciones basados en tiempo entre los objetos. Gráficamente, es una tabla que representa objetos, dispuestos a lo largo del eje X, y mensajes, ordenados según se suceden en el tiempo, a lo largo del eje Y.

3.3.3 Diseño de la Base de Datos

Existen distintos modos de organizar la información y representar las relaciones entre los datos en una base de datos. Los sistemas administradores de bases de datos convencionales usan uno de los tres modelos lógicos de bases de datos para hacer seguimiento de las entidades, atributos y relaciones. Los tres modelos lógicos principalmente de bases de datos son el jerárquico, de redes y el relacional. Cada modelo lógico tiene ciertas ventajas de procesamiento y también ciertas ventajas de negocios.

Para realizar el diseño de la base de datos hay que tener en cuenta, el diagrama de clases persistentes, el modelo entidad relación y la descripción de las tablas de la base de datos todo esto para que tener una visión de que se debe hacer.

3.3.3.1 Diagrama de clases persistentes

Un diagrama de clase persistentes es un tipo de diagrama que muestra un conjunto de objetos capaces de almacenar su estado en un medio permanente como una base de datos relacional.

Diagrama de clases persistentes del sistema:

Diagrama de Clases Persistentes.

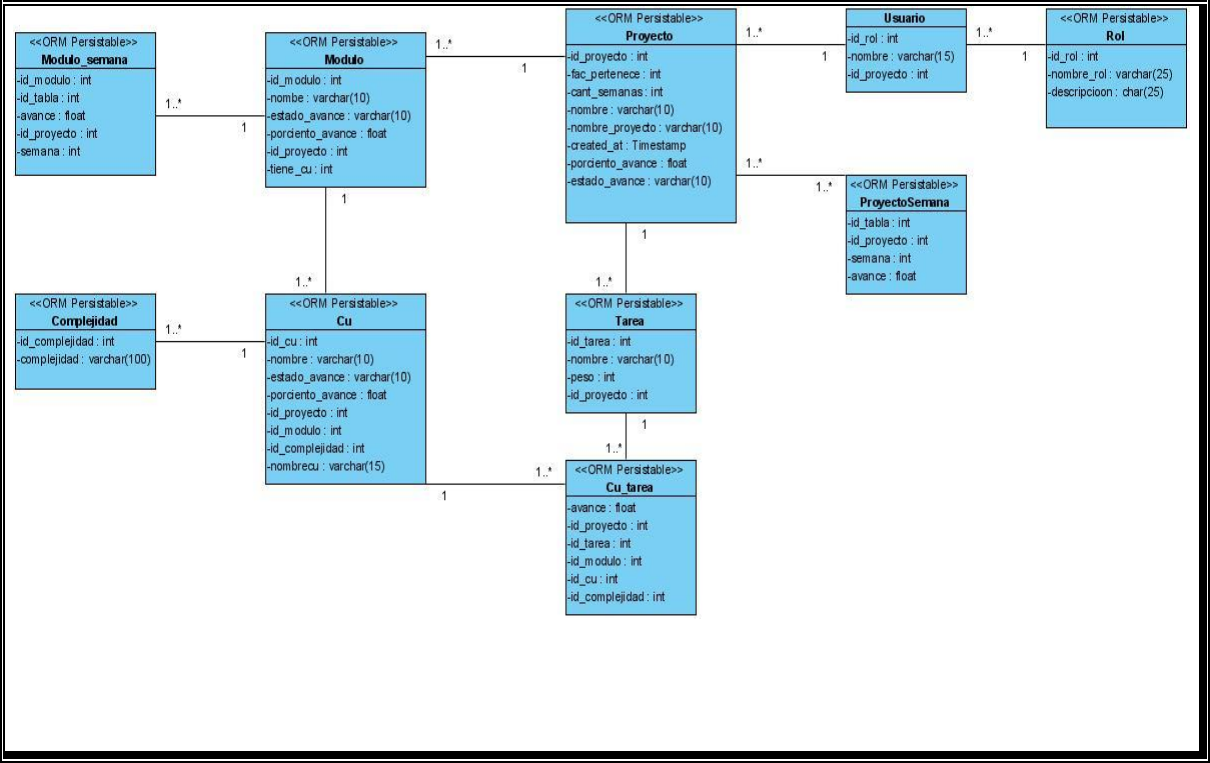

Figura # 50: Diagrama de clases persistentes.

Una vez realizado el diagrama de clases persistentes se puede entonces construir el modelo entidad relación (E R) que no es más que uno de los varios modelos conceptuales existentes para el diseño de bases de datos. Se basa en una percepción de un mundo real. Los elementos esenciales del modelo son las entidades, los atributos y las relaciones entre las entidades. Su propósito es simplificar el diseño de bases de datos a partir de descripciones textuales de los requerimientos.

3.3.3.2 Modelo entidad-relación

Está formado por un conjunto de conceptos que permiten describir la realidad mediante un conjunto de representaciones gráficas y lingüísticas.

Originalmente, el modelo entidad-relación sólo incluía los conceptos de entidad, relación y atributo. Más tarde, se añadieron otros conceptos, como los atributos compuestos y las jerarquías de generalización, en lo que se ha denominado modelo entidad-relación extendido.

Modelo entidad-relación del sistema:

Modelo Entidad Relación.

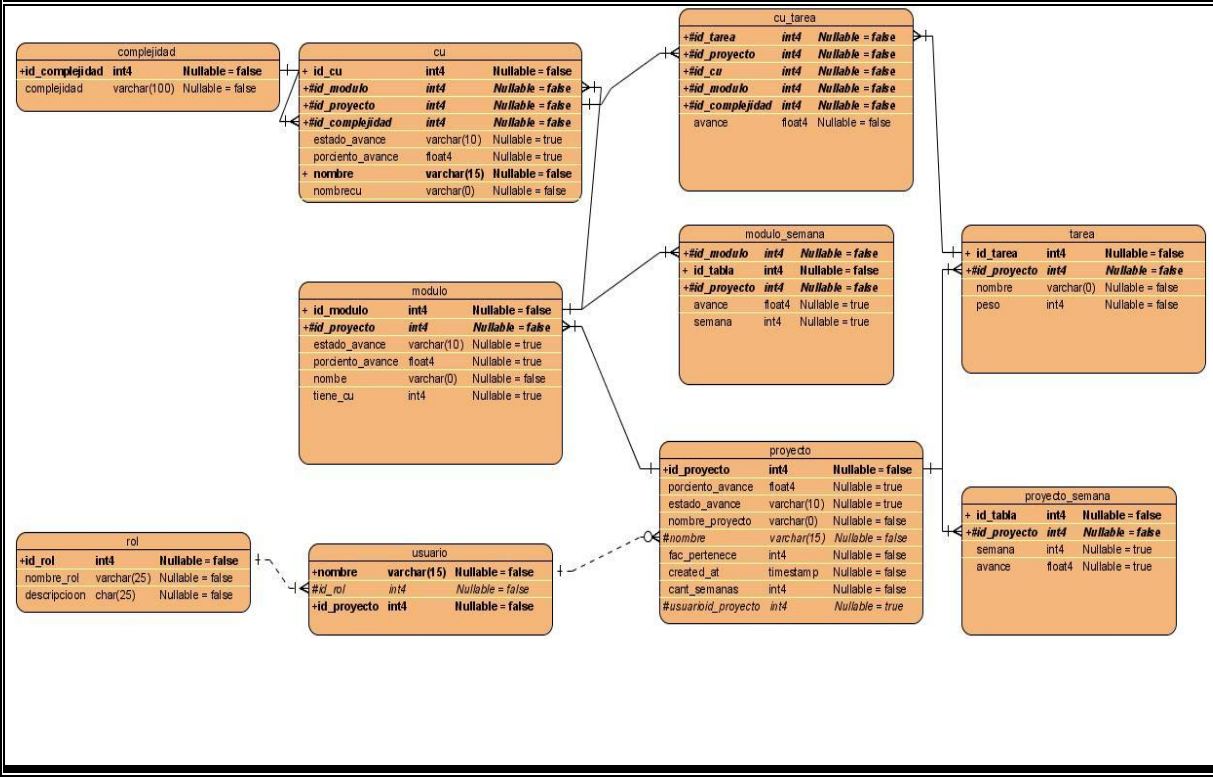

Figura # 51: Modelo entidad relación.

3.3.3.3 Descripción de las tablas de la base de datos

Nombre: rol.		
Descripción: Almacena los datos del rol.		
Atributo	**Tipo**	**Descripción**
id_rol	Integer	Identificador del rol.
nombre_rol	Varchar	Especifica tipo de rol.

descripción	Varchar	Describe lo que hace el rol.

Tabla # 16: Rol

Nombre: usuario.		
Descripción: Almacena los datos de un usuario.		
Atributo	**Tipo**	**Descripción**
nombre	Varchar	Identificador de usuario.
id_rol	Integer	Especifica tipo de rol que posee el usuario.
Id_proyecto	Varchar	Especifica el proyecto del implementador

Tabla # 17: Usuario

Nombre: proyecto.		
Descripción: Almacena los datos del proyecto.		
Atributo	**Tipo**	**Descripción**
id_proyecto	Serial	Identificador del proyecto.
porciento_avance	Real	Por ciento de avance.
estado_avance	Varchar	Especifica estado de avance.
nombre_proyecto	Varchar	Nombre del proyecto.
nombre	Varchar	Determina el nombre del líder del

		proyecto.
fac_pertenece	Integer	Facultad a la que pertenece.
created_at	TIMESTAMP	Fecha de inicio del proyecto.
cant_semanas	Integer	Cantidad de semanas de proyecto.

Tabla # 18: proyecto

Nombre: tarea.		
Descripción: Almacena los datos de las tareas.		
Atributo	**Tipo**	**Descripción**
id_tarea	Serial	Identificador de la tarea.
id_proyecto	Integer	Identificador del proyecto al que pertenece.
nombre	Varchar	Nombre de la tarea.
peso	Integer	Peso de la tarea.

Tabla # 19: Tarea

Nombre: módulo.		
Descripción: Almacena los datos de un módulo.		
Atributo	**Tipo**	**Descripción**
id_proyecto	Integer	Identificador del proyecto al que pertenece.
id_módulo	Serial	Identificador del módulo.

porciento_avance	Real	Por ciento de avance.
estado_avance	Varchar	Especifica estado de avance.
nombe	Varchar	Nombre del módulo.
tiene_cu	Integer	Especifica si en el módulo existen cu.

Tabla # 20: Módulo

Nombre: complejidad.		
Descripción: Almacena los datos de la complejidad.		
Atributo	**Tipo**	**Descripción**
complejidad	Varchar	Describe la complejidad.
id_complejidad	Integer	Especifica el valor de la complejidad.

Tabla # 21: Complejidad.

Nombre: cu.		
Descripción: Almacena los datos de un caso de uso.		
Atributo	**Tipo**	**Descripción**
id_proyecto	Integer	Identificador del proyecto al que pertenece.
id_cu	Serial	Identificador del cu.

id_módulo	Integer	Identificador del módulo al que pertenece.
porciento_avance	Real	Por ciento de avance.
estado_avance	Varchar	Especifica estado de avance.
nombrecu	Varchar	Nombre del cu.
id_complejidad	Integer	Identificador de la complejidad que posee.
nombre	Varchar	Nombre del implementador del cu.

Tabla # 22: Cu.

Nombre: cu_tarea.		
Descripción: Almacena los datos de un caso de uso con respecto a una tarea.		
Atributo	**Tipo**	**Descripción**
id_proyecto	Integer	Identificador del proyecto al que pertenece.
id_cu	Serial	Identificador del cu.
id_módulo	Integer	Identificador del módulo al que pertenece.
id_tarea	Integer	Identificador de la tarea.
avance	Real	Especifica estado de avance.
id_complejidad	Integer	Identificador de la complejidad que posee.

Tabla # 23: Cu_tarea.

Nombre: módulo_semana.		
Descripción: Almacena los datos de un caso de un módulo con respecto a una semana.		
Atributo	**Tipo**	**Descripción**
id_proyecto	Integer	Identificador del proyecto al que pertenece.
id_tabla	Serial	Identificador de la tabla.
id_módulo	Integer	Identificador del módulo al que pertenece.
avance	Real	Avance en la semana.
semana	Integer	Identificador de la semana.

Tabla # 24: Módulo_semana

Nombre: proyecto_semana.		
Descripción: Almacena los datos de un caso de un proyecto con respecto a una semana.		
Atributo	**Tipo**	**Descripción**
id_proyecto	Integer	Identificador del proyecto al que pertenece.
id_tabla	Serial	Identificador de la tabla.
avance	Real	Avance en la semana.
semana	Integer	Identificador de la semana.

Tabla # 25: Proyecto_semana

3.3.4 Definiciones de diseño que se apliquen

Symfony cumple con una gran gama de patrones de diseño que lo hacen muy robusto tanto en su código fuente como en las aplicaciones que se implementan con él. A continuación se presentan una serie de patrones utilizados en el diseño de la aplicación:

- **Experto:** Este patrón define que la responsabilidad de creación de un objeto debe recaer sobre la clase que conoce toda la información necesaria para crearlo. Esto se manifiesta ya que en la implementación del modelo-vista-controlador hecha por el framework la capa del controlador se divide en dos capas, la del controlador frontal que se encarga mediante la clase sfController de decodificar la petición y transferirla hacia el controlador de la de la acción correspondiente, el cual se encuentra en el módulo que maneja la información relacionada en la clase nombreMóduloActions [6].

  La capa del modelo, la cual se divide también en capa de acceso a datos manejada por Propel a través de las clases del modelo que encapsula toda la lógica de datos y la capa de abstracción de datos manejada por Creole encargada de manipular la conexión.

  Por último la capa de vista, en la cual se manejan los datos referentes a la petición así como la respuesta, de lo cual se encargan las clases sfRequest y sfResponse.

- **Adapter:** Symfony implementa el este patrón ya que es posible cambiar en cualquier de base de datos, simplemente configurando unos archivos y este cambio no sería problema para las clases que usan el modelo, ya que estas últimas brindan una interfaz común independientemente del ambiente en el cual se ejecute [6].

- **Singleton:** En Symfony existe un Singleton del contexto mediante el cual se puede acceder a todos los objetos del núcleo del Framework a través del método getContext () en las acciones y de esta forma obtener cualquier tipo de información relacionada con la petición [6].

- **Decorator:** Este patrón se manifiesta en la capa de la vista la cual se compone de un layout general que se puede definir para la aplicación entera o para un determinado módulo y que decora o que se compone en su interior de las plantillas [6].

- **Creador:** Este patrón nos ayuda a identificar quien es el responsable de la creación de una instancia. El mismo se observa en la clase del controlador del módulo nombreMóduloActions, en el

cual se encuentran definidas las acciones correspondientes a dicho módulo. En estas acciones se definen los objetos necesarios para realizar la lógica particular de la acción [6].

- **Front-Controller:** Todas las peticiones deben ser manejadas por un único controlador frontal, que es el punto de entrada a la aplicación. Este es el caso de front-controller definido por Symfony para cada entorno. Quien se encarga de recibir y delegar al controlador del módulo todas las peticiones así como de mostrar las respuestas de dichas peticiones [6].

- **Alta cohesión:** La información de una clase debe ser coherente y estar relacionada con la clase. La clase nombreMóduloActions agrupa las acciones relacionadas con el fin particular del módulo. En ellas se definen variables para las plantillas y funcionalidades con un propósito a fin, lo que les da cierta independencia permitiendo así mayor flexibilidad ante los cambios y una mayor usabilidad y reutilización [6].

3.3.5 Tratamiento de errores

Para garantizar un correcto funcionamiento de cualquier sistema es muy importante identificar y controlar los posibles errores que se puedan producir al interactuar con él. Symfony incluye un completo sistema de validación de errores, ofreciendo la posibilidad de realizar este proceso de múltiples formas. La vía seleccionada para el tratamiento de errores del sistema fue la utilización de archivos YML, en vez de usar código PHP en la acción. Este método es más factible, al ser reutilizable y más sencillo de programar, ya que solo se realizan configuraciones. De esta manera sólo se efectúa la validación del lado del servidor, siendo esta la más importante. No se realiza la validación del lado del cliente.

3.3.6 Seguridad

La seguridad es de suma importancia en todo sistema para proteger la información que en él se almacena de accesos no autorizados. Se requiere que cada usuario ingrese a la aplicación con los permisos que posee y que de esta misma forma haga su uso. Para garantizar que determinada información presente en el sitio sólo se muestre a los usuarios registrados y con autorización previa, se ha programado un módulo encargado de la seguridad. La implementación del mismo se ha hecho aprovechando las posibilidades que brinda el framework Symfony, el cual permite establecer la seguridad por niveles, basados en las credenciales del usuario, las que establecen el acceso que tiene el mismo a las diferentes funcionalidades.

También se utiliza LDAP para la autenticación obteniéndose un sistema más centralizado ya que se integra a los servicios que se brindan en la Universidad.

3.3.7 Interfaz

El sistema MetSoft posee una interfaz sencilla, legible sin abuso de colores, basándose principalmente en el rendimiento de la aplicación y brindando una vista agradable al usuario. Uso de los estándares CSS para manejar la presentación así como un código HTML semántico y organizado. Interfaz compatible con varios navegadores diferentes como Mozilla Firefox e Internet Explorer.

3.4 Conclusiones parciales

Durante este capítulo se llevó a cabo el flujo de trabajo Análisis y Diseño propuesto por la metodología RUP, mediante la elaboración de cada uno de los diagramas correspondientes. Contando como resultado final del diseño, con un plano del modelo de implementación, que permite pasar a la construcción de la propuesta de solución.

Capítulo 4: Implementación y Pruebas.

4.1 Introducción

Después de haber obtenido los resultados del diseño se comienza el flujo de trabajo de implementación, donde se muestra como están relacionadas las partes de código y la estructura del sistema en ejecución mediante los diagramas de componentes y despliegue respectivamente. Además de una breve descripción de las pruebas realizadas al sistema.

4.2 Implementación del sistema

El flujo de trabajo de implementación se comienza con el resultado del diseño. Describe en términos de componentes cómo son implementados los elementos del modelo de diseño y cómo estos se organizan de acuerdo a los nodos específicos en el modelo de despliegue. Los diagramas de despliegue y componentes (artefactos generados en este flujo de trabajo), conforman el modelo de implementación, al describir los componentes a construir y su organización y dependencia entre nodos físicos en los que funcionará la aplicación. El propósito principal de la implementación es desarrollar la arquitectura y el sistema como un todo. [1]

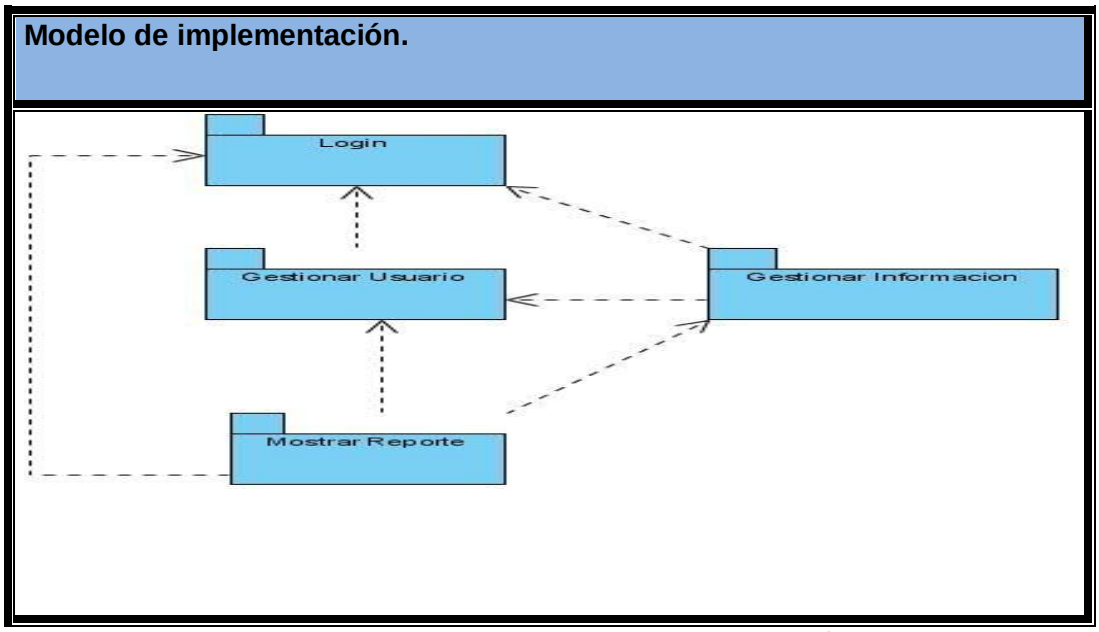

Figura # 52: Modelo de implementación.

4.2.1 Diagrama de despliegue

Un diagrama de despliegue es un grafo de nodos unidos por conexiones de comunicación que muestra las relaciones físicas entre los componentes hardware y software en el sistema final. Se compone por nodos, dispositivos y conectores; donde los nodos son elementos de procesamiento con al menos un procesador, memoria, etc.; los dispositivos son nodos estereotipados sin capacidad de procesamiento en el nivel de abstracción que se modela y los conectores expresan el tipo de conector o protocolo utilizado entre el resto de los elementos del modelo. Mediante el diagrama de despliegue se captura la configuración de los elementos de procesamiento y sus conexiones y se visualiza la distribución de los componentes de software en los nodos físicos. [14]

A continuación se muestra el diagrama de despliegue:

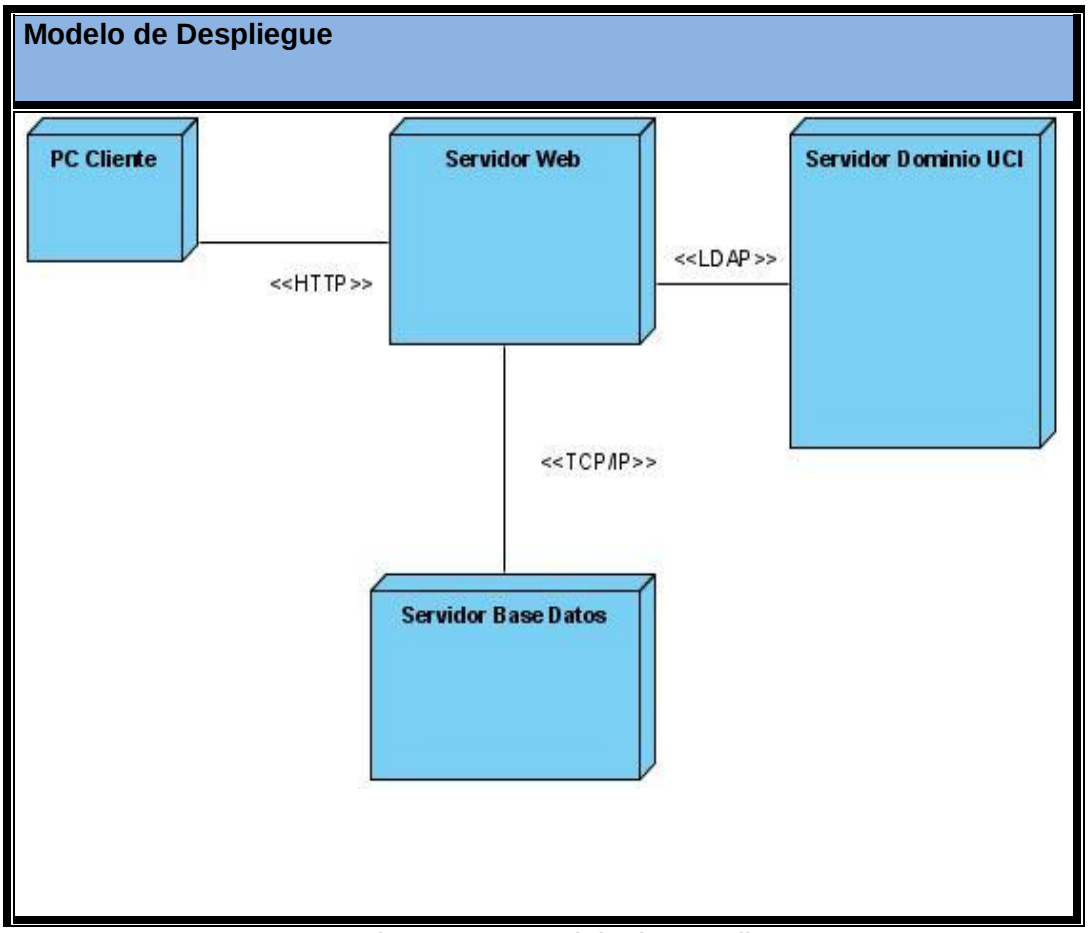

Figura # 53: Modelo de Despliegue.

4.2.2 Diagramas de Componentes

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. [15]

Diagramas de componentes:

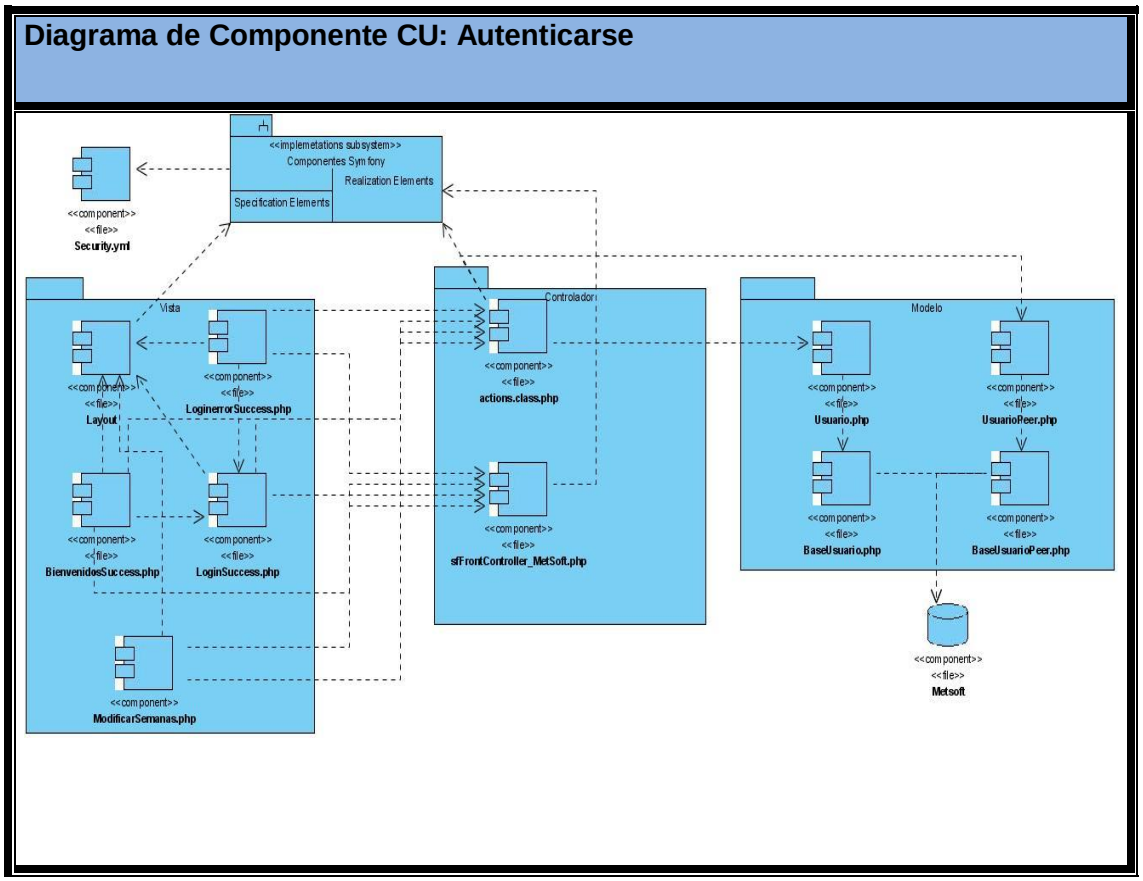

Figura # 54: Diagrama de Componente CU: Autentificarse.

Diagrama de Componente CU: Gestionar Usuario

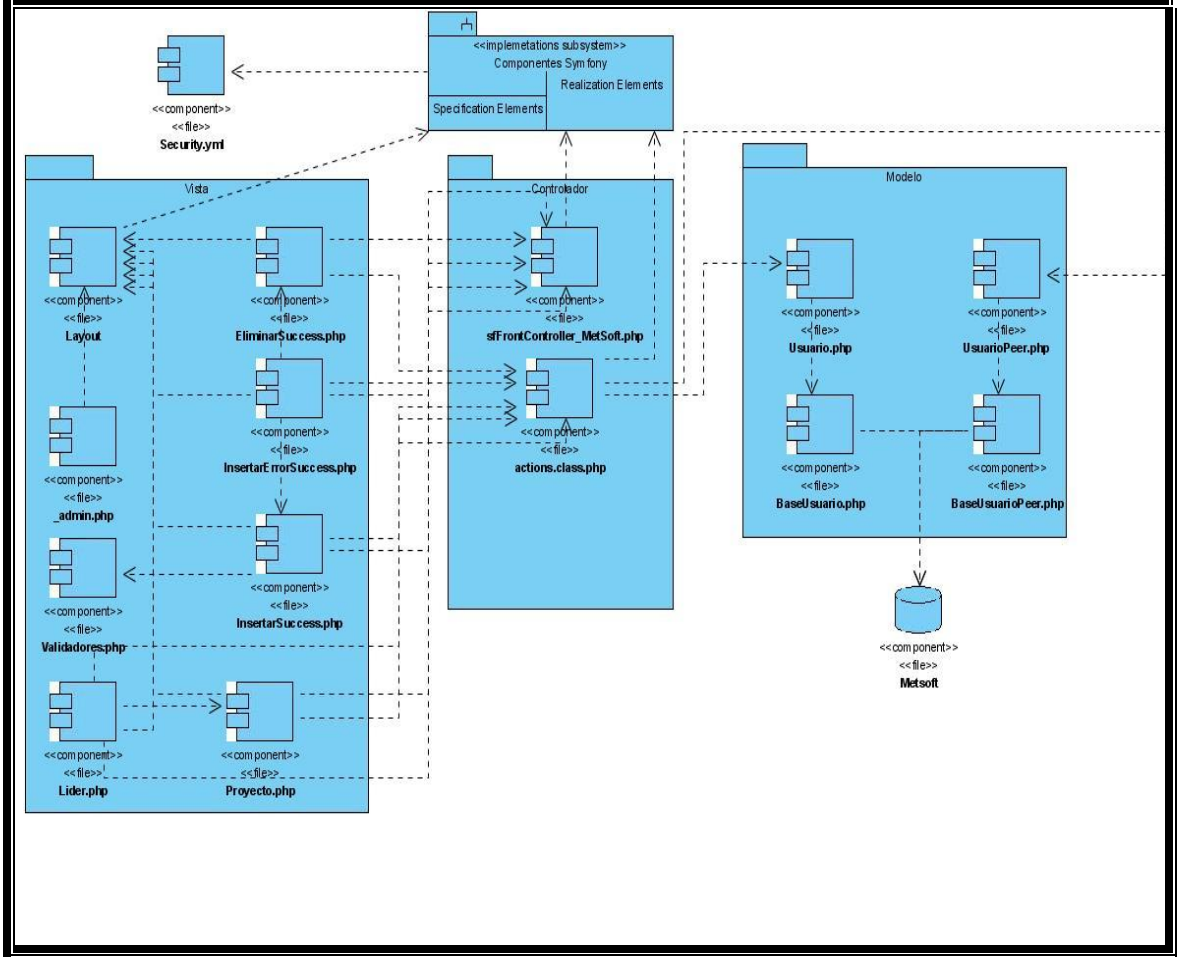

Figura # 55: Diagrama de Componente CU: Gestionar Usuario.

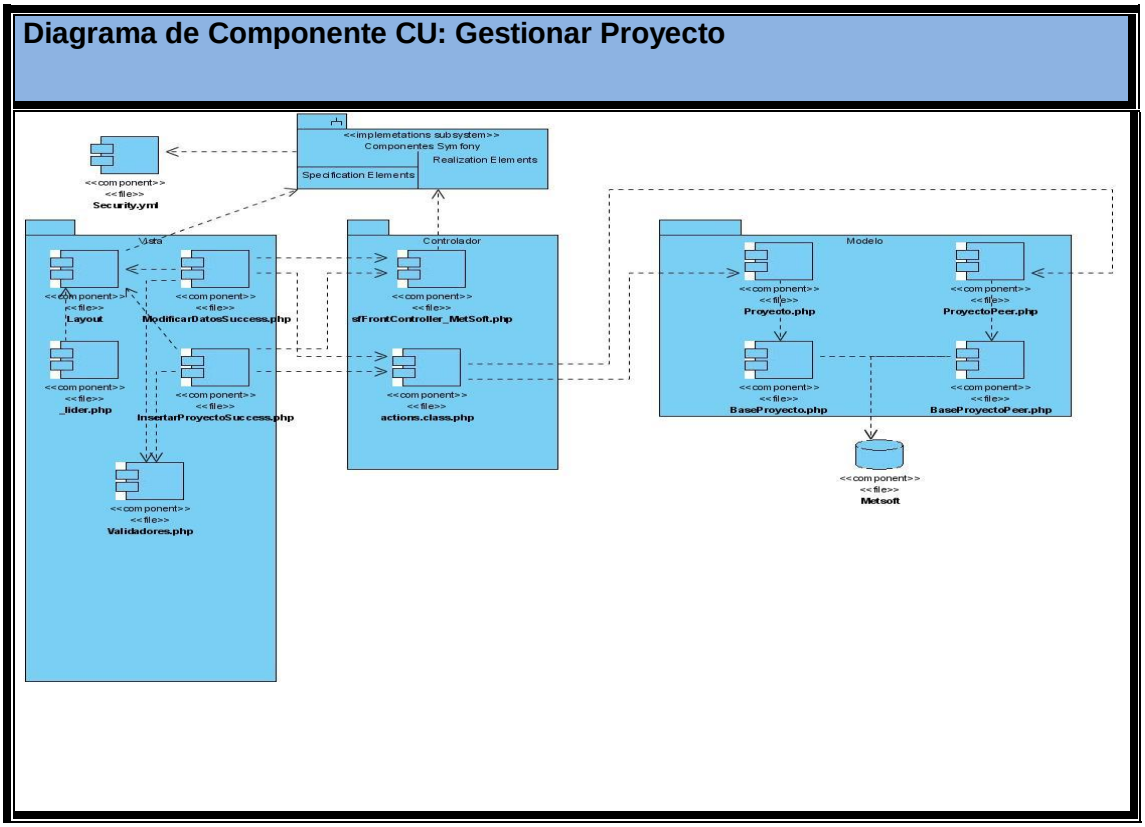

Figura # 56: Diagrama de Componente CU: Gestionar Proyecto.

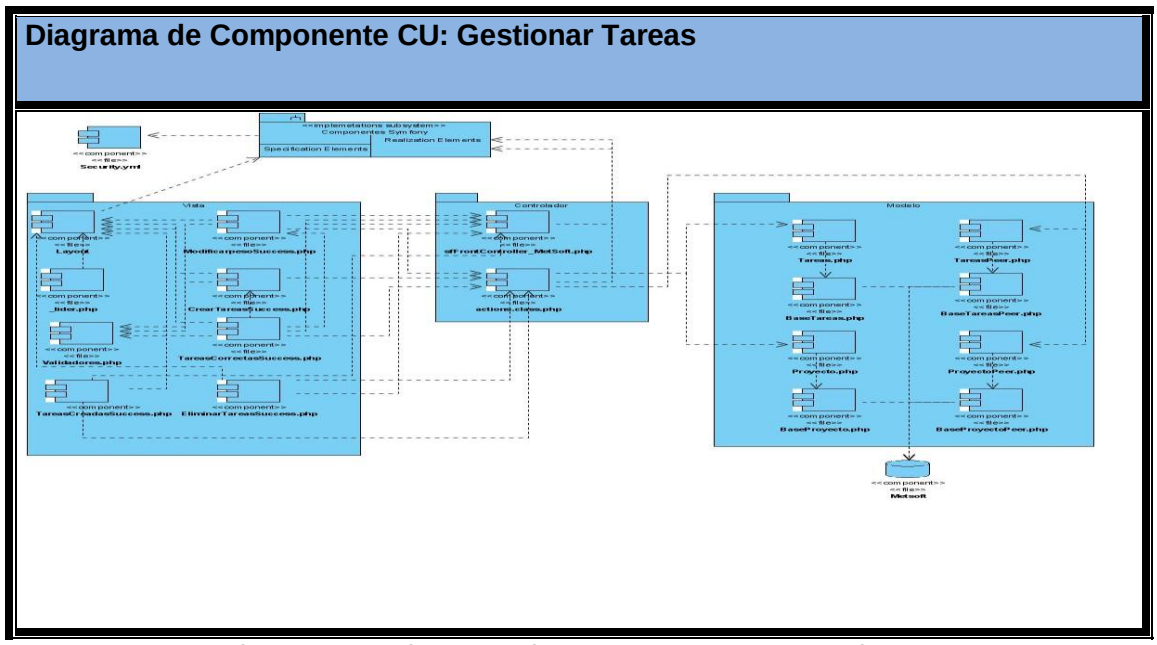

Figura # 57: Diagrama de Componente CU: Gestionar Tareas.

Diagrama de Componente CU: Crear Módulo

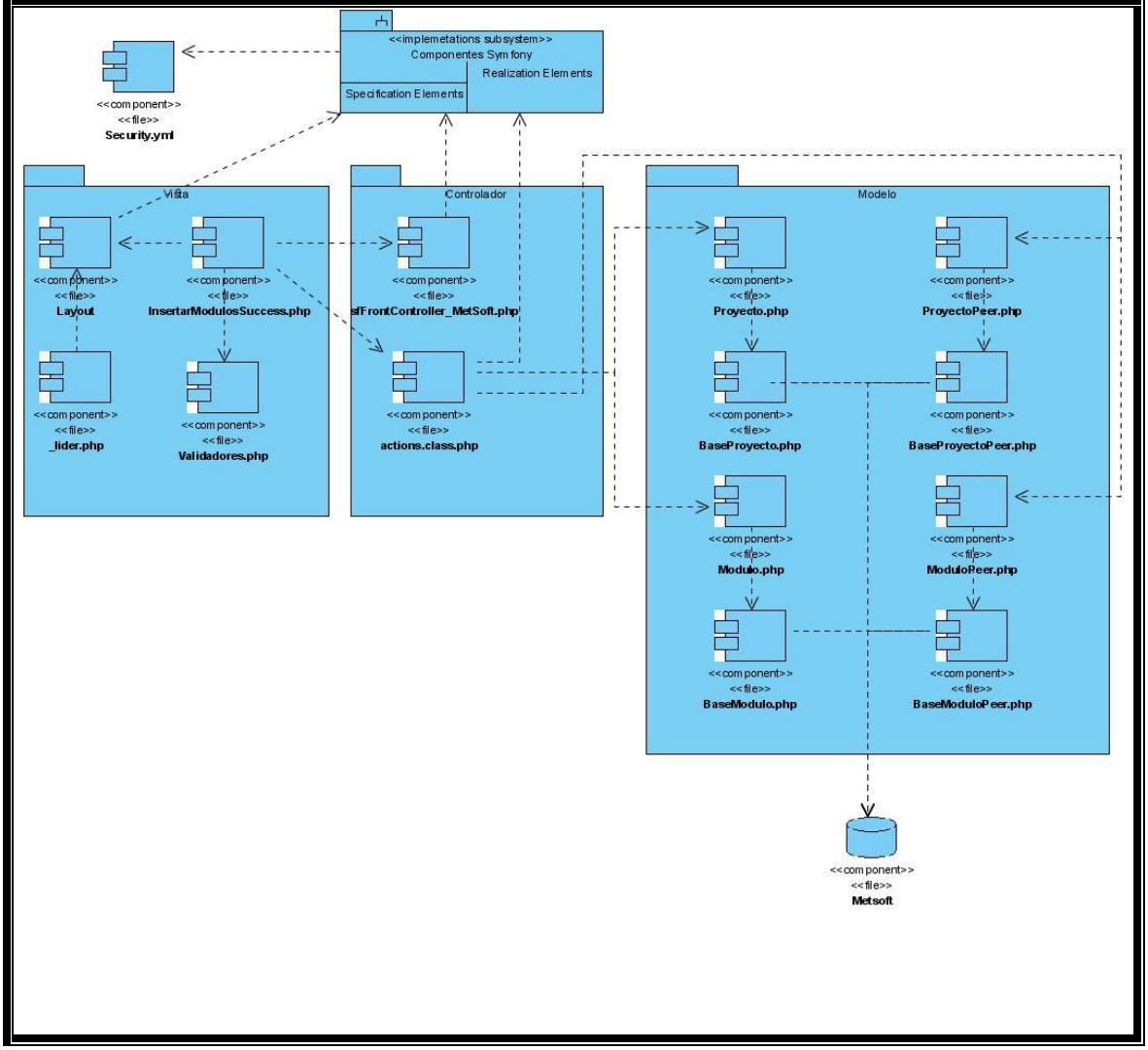

Figura # 58: Diagrama de Componente CU: Crear Módulo.

Diagrama de Componente CU: Crear Caso de Uso

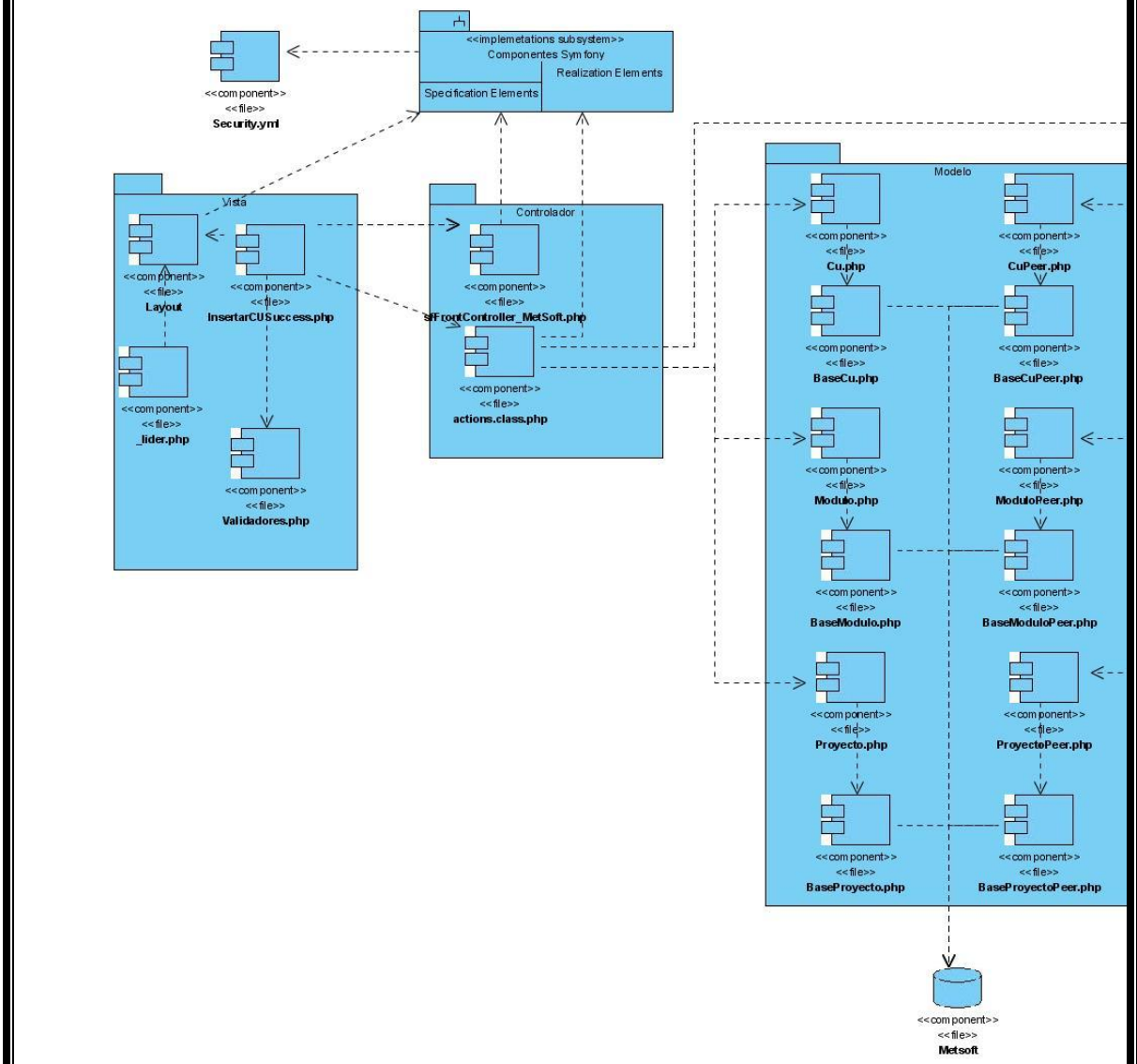

Figura # 59: Diagrama de Componente CU: Crear Caso de Uso.

Diagrama de Componente CU: Completar Caso de Uso

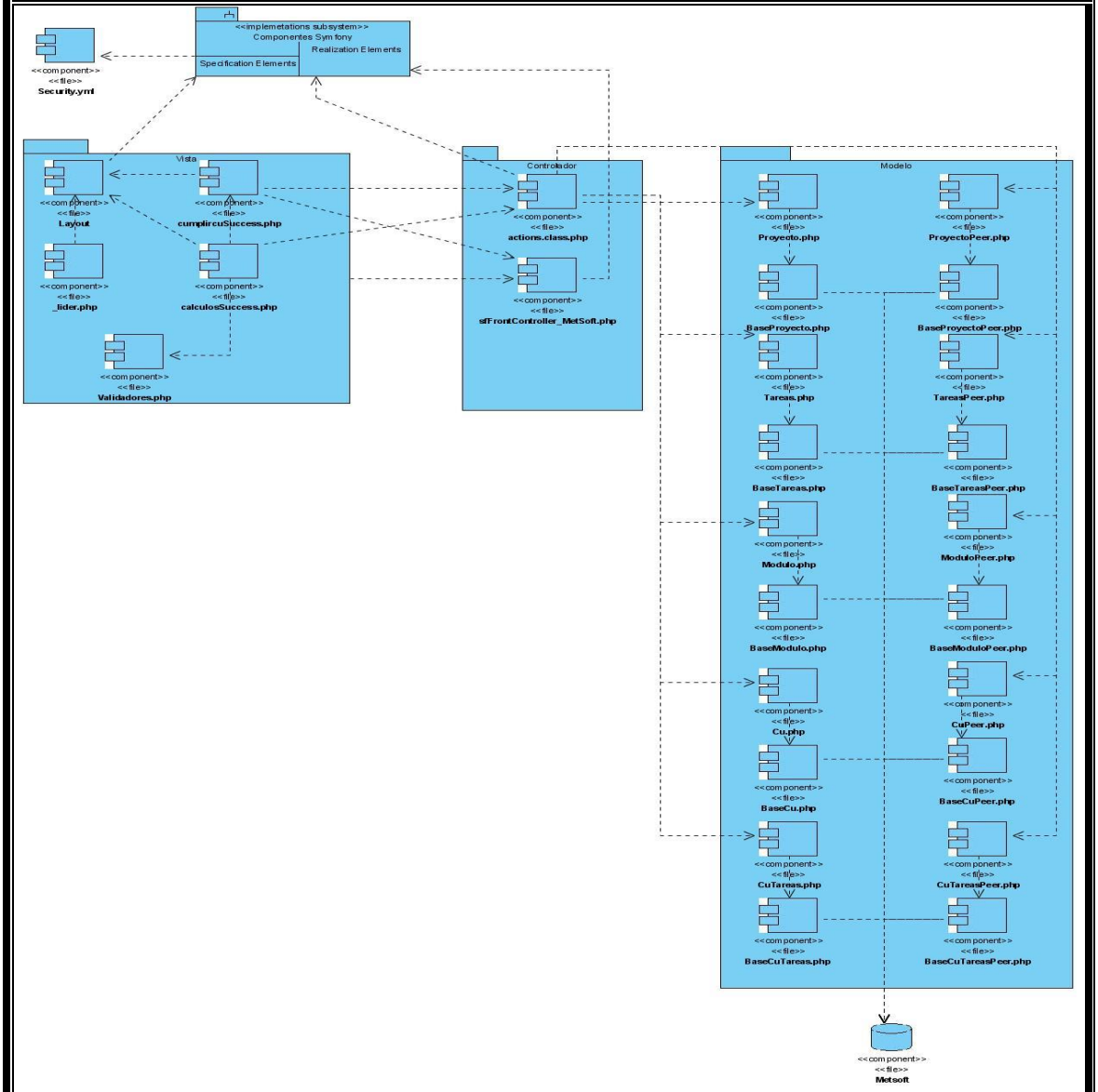

Figura # 60: Diagrama de Componente CU: Completar Caso de Uso.

4.3 Pruebas del Sistema

La prueba es un proceso de ejecución de un programa con la intención de descubrir errores. El objetivo de la prueba es diseñar pruebas que saquen a la luz diferentes clases de errores con la menor cantidad de tiempo y espacio. (PRESSMAN 2005)

Cualquier proceso de ingeniería puede ser probado de una de dos formas:

- ✓ Se pueden llevar a cabo pruebas que demuestren que cada función es completamente operativa.
- ✓ Se pueden desarrollar pruebas que aseguren que la operación interna se ajusta a las especificaciones y que todos los componentes internos se han comprobado de forma adecuada.

La primera aproximación se denomina prueba de la caja negra y la segunda prueba de caja blanca.

4.3.1 Prueba de Caja Negra

Las pruebas se llevan a cabo sobre la interfaz del software, y es completamente indiferente el comportamiento interno y la estructura del programa. Los casos de prueba de la caja negra pretende demostrar que:

Las funciones del software son operativas.

- ✓ La entrada se acepta de forma adecuada.
- ✓ Se produce una salida correcta.
- ✓ La integridad de la información externa se mantiene.

Se derivan conjuntos de condiciones de entrada que ejerciten completamente todos los requerimientos funcionales del programa.

La prueba de la caja negra intenta encontrar errores de las siguientes categorías:

Funciones incorrecta o ausente.

- ✓ Errores de interfaz.
- ✓ Errores en estructuras de datos o en accesos a bases de datos externas.
- ✓ Errores de rendimiento.

- ✓ Errores de inicialización y de terminación.

4.3.2.1 Caso de prueba 1: Autenticarse

1- Condiciones de ejecución

Es necesario que el usuario inserte el usuario y la contraseña del dominio UCI y que haya estado registrado anteriormente en la base de datos de la aplicación.

1.1 Requisitos a probar

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Autentificarse	Permite que el usuario acceda al sistema según el privilegio del mismo.	EP 1.1: El usuario se autenticó correctamente.	- El usuario inserta usuario y contraseña. - Se selecciona el botón aceptar. - El sistema accede a la página bienvenidos según el rol que tenga.
		EP 1.2: No se especificó correctamente el usuario o la contraseña de acceso.	- El sistema muestra un mensaje" Verifique nombre de usuario o contraseña".

		EP 1.3: El usuario especificado no estaba registrado en la base de datos.	- El sistema muestra un mensaje "El usuario no está registrado en la base de datos consulte a un administrador".

Tabla # 26: Requisitos de prueba (CP 1).

1.2 Descripción de variables

Condición de entrada	Clasificación	Puede ser nulo	Descripción
Usuario	Cadena de Caracteres	no	Se introduce el usuario. Pueden ser números y letras.
Contraseña	Cadena de Caracteres	no	Se introduce la contraseña. Pueden ser números y letras.

Tabla # 27: Descripción de variables (CP 1).

1.3 Juego de datos de prueba

Id del escenario	Escenario	Variable Usuario	Variable Contraseña	Respuesta del sistema
EP 1.1	El usuario se autenticó correctamente.	V (ccabezas)	V (carlos)	Entra a la página bienvenido
EP 1.2	No se especificó correctamente el usuario o la contraseña de acceso.	I(weda,12)	I(sfsdfsa)	El sistema muestra un mensaje"Verifique nombre de usuario o contraseña".
EP 1.3	El usuario especificado no	I(ramaro)	v(chino)	El sistema muestra un

	estaba registrado en la base de datos.			mensaje “El usuario especificado no está registrado en la base de datos consulte a un administrador”.

Tabla # 28: Juego de datos de prueba (CP 1).

4.3.2.2 Caso de prueba 2: Gestionar Usuario

2- Condiciones de ejecución

Es necesario que el usuario inserte el usuario y la contraseña del dominio UCI y que haya estado registrado anteriormente en la base de datos de la aplicación.

2.1- Requisitos de prueba

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario

2: Gestionar Usuario	Permite insertar, eliminar y modificar usuario en la base de datos.	EP 2.1: El administrador inserta un usuario correctamente.	- Referencia Caso de Uso autentificarse. - El administrador selecciona opción insertar usuario. - El administrador inserta usuario a adicionar, el rol y contraseña del administrador. - Se selecciona el botón aceptar. - El sistema muestra un mensaje "Está seguro que los datos son correctos." - El sistema inserta el usuario.
		EP 2.2: No se especificó correctamente el usuario o la contraseña a insertar.	- El sistema muestra un mensaje "EL usuario introducido o su contraseña es incorrecta".
		EP 2.3: Deja vacío el campo nombre de Usuario.	- El sistema muestra un mensaje "Entre nombre de usuario".
		EP 2.4: Deja vacío el campo contraseña de	- El sistema muestra un mensaje "La contraseña es

		administrador.	obligatoria".
		EP 2.5: Deja vacío el campo rol.	- El sistema muestra un mensaje "Debe seleccionar un rol".
		EP 2.6: El administrador elimina un usuario correctamente.	- Referencia Caso de Uso autentificarse. - El administrador selecciona opción eliminar usuario. - El administrador selecciona el usuario que va a eliminar usuario. - Se selecciona el botón aceptar. - El sistema muestra un mensaje "Está seguro que los datos son correctos." - El sistema elimina el usuario.
		EP 2.7: Deja vacío el campo usuario a eliminar.	- El sistema muestra un mensaje "Debe seleccionar un usuario".

		EP 2.8: El administrador modifica un usuario correctamente.	- Referencia Caso de Uso autentificarse. - El administrador selecciona opción modificar usuario. - El selecciona el proyecto al cual va modificar el usuario. - El administrador inserta usuario a modificar y contraseña del administrador. - Se selecciona el botón aceptar. - El sistema muestra un mensaje “Está seguro que desea modificar”. - El sistema modifica el usuario.
		EP 2.9: No se especificó correctamente el usuario o la contraseña a modificar.	- El sistema muestra un mensaje “Verifique usuario o contraseña”.
		EP 2.10: Deja vacío el campo nombre de usuario.	- El sistema muestra un mensaje” Debe introducir un usuario”.

		EP 2.11: Deja vacío el campo contraseña.	- El sistema muestra un mensaje" La contraseña es obligatoria".
		EP 2.12: El líder inserta un implementador correctamente.	- Referencia Caso de Uso autentificarse. - El Líder selecciona opción insertar desarrollador. - El Líder inserta desarrollador a adicionar y la contraseña del Líder. - Se selecciona el botón aceptar. - El sistema muestra un mensaje "Esta seguro que los datos son correctos." - El sistema inserta el desarrollador.
		EP 2.13: No se especificó correctamente el desarrollador o la contraseña a insertar.	- El sistema muestra un mensaje "EL usuario introducido o su contraseña es incorrecta".

		EP 2.14: Deja vacío el campo nombre de desarrollador.	- El sistema muestra un mensaje" Entre nombre de usuario".
		EP 2.15: Deja vacío el campo contraseña del Líder.	- El sistema muestra un mensaje" La contraseña es obligatoria".

Tabla # 29: Requisitos de prueba (CP 2).

2.2- Descripción de variables

Condición de entrada	Clasificación	Puede ser nulo	Descripción
Usuario	Cadena de Caracteres	no	Se introduce el usuario. Pueden ser números y letras.
Contraseña de administrador	Cadena de Caracteres	no	Se introduce la contraseña del administrador. Pueden ser números y letras.
Rol	Cadena de Caracteres	no	Se introduce el rol que va a tener el usuario en el proyecto en caso que se inserte Administrador, Líder o Directivo.

Tabla # 30: Descripción de variables (CP 2).

2.3 Juego de datos de prueba

Id del escenario	Escenario	Variable Usuario	Variable Contraseña	Variable Rol	Respuesta del sistema

			de administrador		
EP 2.1	El administrador inserta un usuario. correctamente	V(ccabezas)	V (carlos)	V (Líder)	Adiciona usuario en la base de datos.
EP 2.2	No se especificó correctamente el usuario o la contraseña a insertar.	I(weda,12)	I(sfsdfsa)	V (Líder)	El sistema muestra un mensaje “EL usuario introducido o su contraseña es incorrecta“.
EP 2.3	Deja vacío el campo nombre de Usuario	I()	v(chino)	V (Líder)	El sistema muestra un mensaje ”Entre nombre de usuario”
EP 2.4	Deja vacío el campo contraseña de administrador.	V(ccabezas)	I ()	V (Líder)	El sistema muestra un mensaje “La contraseña es obligatorio”.
EP 2.5	Deja vacío el campo rol.	V(ccabezas)	V (carlos)	I ()	El sistema muestra un mensaje “Debe seleccionar un rol”.
EP 2.6	El administrador elimina un usuario correctamente.	V(ccabezas)	-	-	El sistema elimina usuario.
EP 2.7	Deja vacío el campo usuario a	I()	-	-	El sistema muestra un mensaje ”Debe

	eliminar.				seleccionar un usuario”
EP 2.8	El administrador modifica un usuario correctamente.	V(ccabezas)	V (carlos)	-	El sistema modifica el usuario.
EP 2.9	No se especificó correctamente el usuario o la contraseña a modificar.	I(weda,12)	I(sfsdfsa)	-	El sistema muestra un mensaje “Verifique usuario o contraseña”
EP 2.10	Deja vacío el campo nombre de usuario.	I()	V (carlos)	-	El sistema muestra un mensaje “Debe introducir un usuario”.
EP 2.11	Deja vacío el campo contraseña.	V(ccabezas)	I()	-	El sistema muestra un mensaje “La contraseña es obligatoria”.
EP 2.12	El Líder inserta un desarrollador correctamente.	V(ccabezas)	V (carlos)	V (Líder)	Adiciona desarrollador en la base de datos.
EP 2.13	No se especificó correctamente el desarrollador o la contraseña a insertar.	I(weda,12)	I(sfsdfsa)	V (Líder)	El sistema muestra un mensaje “El usuario introducido o su contraseña es incorrecta“.
EP 2.14	Deja vacío el campo nombre del	I()	v(chino)	V (Líder)	El sistema muestra un mensaje ”Entre nombre de

	desarrollador.				usuario”
EP 2.15	Deja vacío el campo contraseña del Líder.	V(ccabezas)	I ()	V (Líder)	El sistema muestra un mensaje “La contraseña es obligatorio”.

Tabla # 31: Juego de datos de prueba (CP 2).

4.3.2.3 Caso de prueba 3: Gestionar Proyecto

3- Condiciones de ejecución

Es necesario que el usuario sea líder de proyecto.

3.1- Requisitos de prueba

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
3: Gestionar Proyecto	Permite registrar y modificar un proyecto en la base de datos.	EP 3.1: El líder registra un proyecto correctamente.	- Referencia Caso de Uso autentificarse. - El líder selecciona opción registrar proyecto. - El líder inserta el nombre del proyecto, la duración del proyecto en semanas y selecciona la facultad que pertenece.

			- Se selecciona el botón aceptar. - El sistema muestra un mensaje “Está seguro que los datos son correctos”. - El sistema inserta el proyecto.
		EP 3.2: Deja vacío el campo nombre del proyecto.	- El sistema muestra un mensaje “Debe introducir un proyecto”.
		EP 3.3: Deja vacío el campo facultad.	- El sistema muestra un mensaje “Debe introducir una facultad”.
		EP 3.4: Deja vacío el campo duración del proyecto.	- El sistema muestra un mensaje “Inserte cantidad de semanas”.
		EP 3.5: El líder modifica un usuario correctamente.	- Referencia Caso de Uso autentificarse. - El líder selecciona opción modificar proyecto.

			- El líder inserta el nombre del proyecto, la facultad que pertenece y la duración del proyecto en semanas. - Se selecciona el botón aceptar. - El sistema muestra un mensaje" Está seguro que los datos son correctos". - El sistema modifica proyecto.

Tabla # 32: Requisitos de prueba (CP 3).

3.2- Descripción de variables

Condición de entrada	Clasificación	Puede ser nulo	Descripción
Nombre del proyecto	Cadena de Caracteres	no	Se introduce el nombre. Pueden ser números y letras. Es validado.
Facultad	Cadena de Caracteres	no	Se introduce la Facultad. Solo se pueden seleccionar las que aparecen.
Duración en semanas	número	no	Se introduce la cantidad de semanas que va durar el

			proyecto. Solo pueden ser números enteros.

Tabla # 33: Descripción de variables (CP 3).

3.3- Juego de de prueba

Id del escenario	Escenario	Variable Nombre proyecto	Variable Facultad	Variable Duración	Respuesta del sistema
EP 3.1	El líder inserta un proyecto correctamente.	V(calidad)	V (2)	15	Inserta proyecto en la base de datos
EP 3.2	EP 3.2: Deja vacío el campo nombre del proyecto.	I()	V (2)	15	El sistema muestra un mensaje “Debe introducir un proyecto”.
EP 3.3	Deja vacío el campo facultad.	V(calidad)	I()	15	El sistema muestra un mensaje “Debe introducir una facultad”
EP 3.4	Deja vacío el campo duración del proyecto.	V(calidad)	V (2)	I()	El sistema muestra un mensaje “Inserte cantidad de semanas”.
EP 3.5	El líder modifica un usuario correctamente.	V(iptv)	v(3)	15	Modifica proyecto en la

					base de datos.

Tabla # 34: Juego de prueba (CP 3).

4.3.2.4 Caso de prueba 4: Gestionar Tarea

4- Condiciones de ejecución

Es necesario que el usuario sea líder de proyecto. Además la suma de los pesos de todas las tareas insertadas debe ser igual a 100 y una vez que estén todas insertadas no se pueden modificar.

4.1- Requisitos de prueba

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
4: Gestionar Tarea	Permite crear, eliminar y modificar el peso de una tarea en la base de datos.	EP 4.1: El líder crea una tarea correctamente.	- Referencia Caso de Uso autentificarse. - El líder selecciona opción crear tarea. - El líder inserta nombre de la tarea y peso. - Se selecciona el botón aceptar - El sistema muestra un mensaje “Está seguro que los datos son correctos”. - El sistema inserta la tarea.
		EP 4.2: El peso insertado es mayor que 100.	- El sistema muestra un mensaje “El peso de la tarea es Incorrecto”.

		EP 4.3: La suma de los pesos es mayor que 100.	- El sistema muestra un mensaje ”Está seguro que los datos son correctos” - El sistema muestra un mensaje “La tarea no puedo ser insertada Modifique el peso”.
		EP 4.4: Dejo vacío el campo nombre de la tarea.	- El sistema muestra un mensaje “Debe introducir el nombre de la tarea”
		EP 4.5: Dejo vacío el campo peso.	- El sistema muestra un mensaje “Introduzca peso de la tarea”.
		EP 4.6: El líder elimina una tarea correctamente.	- Referencia Caso de Uso autentificarse - El líder selecciona opción eliminar tarea. - El líder selecciona la tarea que va a eliminar. - Se selecciona el botón eliminar. - El sistema muestra un mensaje” Está seguro que los datos son correctos”. - El sistema elimina la tarea

		EP 4.6: Deja vacío el campo duración del proyecto.	- El sistema muestra un mensaje" Debe seleccionar una tarea".
		EP 4.7: El líder modifica un peso correctamente.	- Referencia Caso de Uso autentificarse. - El líder selecciona opción modificar peso. - El líder inserta el peso de la tarea. - Se selecciona el botón aceptar. - El sistema muestra un mensaje" Está seguro que los datos son correctos". - El modifica el peso de la tarea.
		EP 4.8: La suma de los pesos insertado es mayor que 100.	- El sistema muestra un mensaje "La suma de los pesos debe ser igual a 100".

Tabla # 35: Requisitos de prueba (CP 4).

4.2- Descripción de variables

Condición de entrada	Clasificación	Puede ser nulo	Descripción
Nombre de la tarea	Cadena de Caracteres	No	Se introduce el nombre Pueden ser números y letras.

Peso	número	No	Se introduce el peso. Solo pueden ser números enteros entre 1 y 100.

Tabla # 36: Descripción de variables (CP 4).

4.3- Juego de datos de prueba

Id del escenario	Escenario	Variable Nombre de la tarea	Variable peso	Respuesta del sistema
EP 4.1	El líder inserta una tarea correctamente.	V(Tarea1)	V(10)	Adiciona una tarea en la base de datos.
EP 4.2	El peso insertado es mayor que 100.	V(tarea1)	I(200)	El sistema muestra un mensaje “El peso de la tarea es Incorrecto”.
EP 4.3	La suma de los pesos es mayor que 100.	V(tarea1) V(tarea2)	V(90) I(90)	El sistema muestra un mensaje “La tarea no puedo ser insertada Modifique el peso”.
EP 4.4	Dejó vacío el campo nombre de la tarea.	I()	V(10)	El sistema muestra un mensaje “Debe introducir el nombre de la tarea”.
EP 4.5	Dejó vacío el campo peso.	V(Tarea1)	I()	El sistema muestra un mensaje “Introduzca peso de la tarea”.
EP 4.6	El líder elimina una tarea correctamente.	V(Tarea1)	-	Elimina una tarea de la base de datos.
EP 4.7	El líder modifica una tarea correctamente.	V(Tarea1)	V(20)	Modifica una tarea en la base de datos
EP 4.8:	La suma de los pesos insertado es mayor que	V(Tarea1)	I(200)	El sistema muestra un mensaje “La suma de

	100.			los pesos debe ser igual a 100".

Tabla # 37: Juegos de datos de prueba (CP 4).

4.3.2.5 Caso de prueba 5: Crear Módulo

5- Condiciones de ejecución

Es necesario que el usuario sea líder de proyecto.

5.1 Requisitos de prueba

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
5: Crear Módulo	Permite crear un módulo.	EP 5.1: El líder crea un módulo Correctamente.	- Referencia Caso de Uso autentificarse. - El líder selecciona opción crear módulo. - El líder inserta nombre del módulo - Se selecciona el botón aceptar. - El sistema muestra un mensaje" Está seguro que los datos son correctos". - El sistema inserta el módulo.
		EP 5.2: Deja vacío el campo nombre del módulo.	- El sistema muestra un mensaje" Debe introducir el nombre del módulo".

Tabla # 38: Requisitos de prueba (CP 5).

5.2 Descripción de variables

Condición de entrada	Clasificación	Puede ser nulo	Descripción
Nombre del módulo tarea	Cadena de Caracteres	No	Se introduce el nombre Pueden ser números y letras.

Tabla # 39: Descripción de variables (CP 5).

5.3 Juego de datos de prueba

Id del escenario	Escenario	Variable Nombre del módulo	Respuesta del sistema
EP 5.1	El líder inserta un módulo correctamente.	V(módulo1)	Adiciona una tarea en la base de datos.
EP 5.2	Deja vacío el campo nombre del módulo.	I()	El sistema muestra un mensaje “Debe introducir el nombre del módulo”.

Tabla # 40: Juego de datos de prueba (CP 5).

4.3.2.6 Caso de prueba 6: Crear Caso de Uso

6- Condiciones de ejecución

Es necesario que el usuario sea líder de proyecto. Además que ya hayan sido creados los módulos

6.1- Requisitos a probar

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario

6: Crear Caso de Uso	Permite crear casos de uso.	EP 6.1: El líder inserta un caso de uso correctamente.	- Referencia Caso de Uso autentificarse. - El líder selecciona opción crear CU. - El líder inserta el nombre del caso de uso, selecciona el módulo que pertenece, el nombre del desarrollador y la complejidad del mismo. - Se selecciona el botón aceptar. - El sistema muestra un mensaje" Está seguro que los datos son correctos". - El sistema inserta el caso de uso.
		EP 6.2: Deja vacío el campo nombre del CU.	- El sistema muestra un mensaje" Debe introducir el nombre del CU".
		EP 6.3: Deja vacío el campo módulo que pertenece.	- El sistema muestra un mensaje" Debe seleccionar un módulo".
		EP 6.4: Deja vacío el campo complejidad del CU.	- El sistema muestra un mensaje" Debe seleccionar una complejidad".

		EP 6.5: Deja vacío el campo desarrollador del CU.	- El sistema muestra un mensaje" Debe seleccionar un desarrollador".

Tabla # 41: Requisitos de prueba (CP 6).

6.2- Descripción de variables

Condición de entrada	Clasificación	Puede ser nulo	Descripción
Nombre del caso de uso	Cadena de Caracteres	No	Se introduce el nombre Pueden ser números y letras.
Complejidad	Cadena de Caracteres	No	Se selecciona la complejidad. Solo pueden ser las que aparecen(alta, media o baja)
Módulo que pertenece	Cadena de Caracteres	No	Se selecciona el módulo. Solo pueden ser los que aparecen
Desarrollador del CU	Cadena de Caracteres	No	Se selecciona el desarrollador. Solo pueden ser los que aparecen

Tabla # 42: Descripción de variables (CP 6).

6.3- Juego de datos de prueba

Id del escenario	Escenario	Variable Nombre del caso de uso	Variable Nombre del módulo	Variable Nombre del desarrollador	Variable Complejidad	Respuesta del sistema

EP 6.1	El líder inserta un caso de uso correctamente.	V(cu1)	V(módulo 1)	V(ccabezas)	V(alta)	Adiciona un caso de uso en la base de datos.
EP 6.2	Deja vacío el campo nombre del CU.	I ()	V(módulo 1)	V(ccabezas)	V(alta)	El sistema muestra un mensaje" Debe introducir el nombre del CU".
EP 6.3 :	Deja vacío el campo módulo que pertenece.	V(cu1)	I ()	V(ccabezas)	V(alta)	El sistema muestra un mensaje" Debe seleccionar un módulo".
EP 6.4	Deja vacío el campo complejidad del CU.	V(cu1)	V(módulo 1)	V(ccabezas)	I ()	El sistema muestra un mensaje "Debe seleccionar una complejidad".
EP 6.5	Deja vacío el campo desarrollador del CU.	V(cu1)	V(módulo 1)	I ()	V(alta)	El sistema muestra un mensaje "Debe seleccionar un desarrollador".

Tabla # 43: Juego de datos de prueba (CP 6).

4.3.2.7 Caso de prueba: Completar CU

7- Condiciones de ejecución

- Es necesario que el usuario sea líder de proyecto. Además los valores insertados para completar los casos de uso deben estar entra 0 y 1.

7.1 Requisitos a probar

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario

7: Completar Caso de Uso	Permite modelar como se han completado los casos de uso con respecto a las tareas	EP 7.1:El implementador completa los casos de uso correctamente	– Referencia Caso de Uso autentificarse – El implementador selecciona opción completar casos de uso – El implementador inserta los valores correspondientes – Se selecciona el botón aceptar – El sistema muestra un mensaje ”Esta seguro que desea Insertar” – El sistema inserta el avance
		EP 7.2: El valor insertado es mayor que 1	- El sistema muestra un mensaje “el avance debe de estar entre 0 y 1”
		EP 7.3:El líder completa los casos de uso correctamente	- Referencia Caso de Uso autentificarse - El líder selecciona opción completar casos de uso - El líder inserta los valores correspondientes - Se selecciona el botón aceptar - El sistema muestra un

			mensaje ”Esta seguro que desea Insertar” - El sistema inserta el avance

Tabla # 44: Requisitos de prueba (CP 7).

7.2 Descripción de variables

Condición de entrada	Clasificación	Puede ser nulo	Descripción
avance	número	No	Se introduce el avance de los casos de uso con respecto a las tareas. Estos valores deben de estar entre 0 y1.

Tabla # 45: Descripción de variables (CP 7).

7.3 Juego de datos a probar

Id del escenario	Escenario	Variable peso	Respuesta del sistema
EP 7.1	El implementador completa los casos de uso correctamente	V(0.2)	Adiciona una tarea en la base de datos
EP 7.2	El peso insertado es mayor que 1	I(2)	El sistema muestra un mensaje “el avance debe de estar entre 0 y 1”

EP 7.1	El líder completa los casos de uso correctamente	V(0.2)	Adiciona una tarea en la base de datos

Tabla # 46: Juego de datos de prueba (CP 7).

4.3.2.8 Caso de prueba 8: Mostrar información

8- Condiciones de ejecución

- El usuario debe ser líder de proyecto o directivo de la UCI.

8.1 Requisitos a probar

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
8:Mostrar Información	Muestra todos los reportes tanto al líder de proyecto como al directivo de la UCI (Avance del proyecto, Avance por módulo, Avance de los casos de Uso y Avance del proyecto semanal).	EP 8.1: Mostrar Avance por módulos.	- Referencia Caso de Uso autentificarse. - El líder o el directivo seleccionan la opción Avance por módulos. - El sistema muestra el avance por módulo.
		EP 8.2:Mostrar Avance por casos de uso	- Referencia Caso de Uso autentificarse - El líder o el directivo seleccionan la opción Avance casos de uso - Seleccionan de que módulo es que desean ver los casos de uso

			- El sistema muestra el avance por casos de uso.
		EP 8.3:Mostrar Avance del proyecto	- Referencia Caso de Uso autentificarse - El líder o el directivo seleccionan la opción Avance del proyecto - El sistema muestra el avance del proyecto
		EP 8.4:Mostrar Avance del proyecto semanal	- Referencia Caso de Uso autentificarse - El líder o el directivo seleccionan la opción Avance del proyecto semanal - El sistema muestra el Avance del proyecto semanal

Tabla # 47: Requisitos de prueba (CP 8).

4.3.2.9 Caso de prueba 9: Graficar

9- Condiciones de ejecución

El usuario debe ser líder de proyecto o directivo de la UCI.

9.1 Requisitos a probar

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
9: Graficar	Permite Graficar (Avance de los módulos y Estado de avance del proyecto semanal)	EP 9.1:Graficar Avance de los módulos	- Referencia Caso de Uso autentificarse - El líder o el directivo Graficar Avance de los módulos - El sistema muestra el grafico Avance de los módulos
		EP 9.2:Graficar avance del proyecto semanal	- Referencia Caso de Uso autentificarse - El líder o el directivo avance del proyecto semanal. - El sistema muestra el grafico avance del proyecto semanal

Tabla # 48: Requisitos de prueba (CP 9).

4.4 Conclusiones parciales.

Al terminar este capítulo, se cuenta con los diagramas de despliegue y componentes, así como los resultados arrojados de las pruebas realizadas al sistema que se propone.

Capítulo 5: Estudio de Factibilidad

5.1 Introducción

El objetivo fundamental de la estimación, es determinar la posibilidad de llevar adelante el proyecto (el estudio de la factibilidad), de acuerdo a diferentes restricciones, dadas por características propias del mismo, como pueden ser: equipo, organizativas, económicas, técnicas, de tiempo, etc. En este capítulo se estima el esfuerzo y se analizan los beneficios del sistema propuesto (MetSoft), utilizando el método de estimación por Puntos de Casos de Uso. Obteniendo importantes indicadores como son: esfuerzo y tiempo requerido de desarrollo.

5.2 Estimación de Esfuerzo

La estimación mediante el análisis de Puntos de Casos de Uso es un método de estimación del 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. A continuación, se realizará la estimación del Sistema para calcular el estado de avance de los proyectos de la UCI durante el flujo de implementación (MetSoft).

5.2.1 Paso 1. Identificar los Puntos de Casos de Uso Desajustados

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.

Factor de Peso de los Actores sin Ajustar (UAW). -Los actores del sistema se clasifican en:

Simple: Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación.

Medio: Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto.

Complejo: Una persona que interactúa con el sistema mediante una interfaz gráfica.

-El factor de peso de cada actor está dado por su clasificación:

- ✓ Para un actor simple el factor de peso es 1.
- ✓ Para un actor medio el factor de peso es 2.
- ✓ Para un actor complejo el factor de peso es 3.

El sistema MetSoft cuenta con tres actores (Administrador del Sistema, Directivo de UCI y Líder de Proyecto). Estas son personas que interactúan con el sistema mediante una interfaz gráfica. Por tanto estos actores se clasifican en: Complejos y poseen factor de peso 3.

Para calcular UAW

Tipo de Actor.	**Descripción.**	**Peso.**	**Cant. Actores.**	**Cant. Actores * Peso.**	**Resultado.**
Simple.	Sistema que interactúa con MetSoft mediante un API.	1	0	1*0	0
Medio.	Sistema que interactúa con MetSoft mediante un protocolo o una interfaz basada en texto.	2	0	2*0	0
Complejo.	Persona que interactúa con el sistema mediante una interfaz gráfica.	3	3	3*3	9
Total:			**9**		

Tabla # 49: Factor de peso de los actores sin ajustar.

Respuesta: UAW=6 ***Factor de Peso de los Casos de Uso sin ajustar (UUCW).*** -Los casos de uso del sistema se clasifican en:

Simple: El Caso de Uso contiene de 1 a 3 transacciones.

Medio: El Caso de Uso contiene de 4 a 7 transacciones.

Complejo: El Caso de Uso contiene más de 8 transacciones. -El factor de peso de cada caso de uso está dado por su clasificación:

- ✓ Para un caso de uso simple el factor de peso es 5.
- ✓ Para un caso de uso medio el factor de peso es 10.
- ✓ Para un caso de uso complejo el factor de peso es 15.

Nombre de los casos de uso de MetSoft----------Cantidad de transacciones

1. Autenticar Usuario----------3
2. Gestionar Usuario ----------6
3. Gestionar Proyecto Productivo----------6
4. Crear Casos de Uso----------3
5. Crear Módulos----------3
6. Crear Tareas----------3
7. Completar Casos de Uso----------6
8. Mostrar Información----------12
9. Graficar----------12

Para calcular UUCW

Tipo de Caso de Uso.	**Descripción.**	**Peso.**	**Cant. De Casos de uso de ese tipo.**	**Cant. De Casos de uso de ese tipo * Peso.**	**Resultado.**
Simple	De 1-3	5	4	5*4	20

	transacciones.				
Medio	De 4-7 transacciones.	10	3	10*3	30
Complejo	De 8 transacciones en adelante.	15	2	15*2	30
Total					**80**

Tabla # 50: Factor de peso de los casos de uso sin ajustar.

Calculando UUCP:

UUCP = UAW + UUCW

UUCP= 6+80=86

Resultado del Paso 1:

UUCP =86

5.2.2 Paso 2. Ajustar los Puntos de Casos de Uso.

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)

Para Calcular TCF:

TCF = 0.6 + 0.01 * Σ (Pesoi * Valori)

Donde:

Valor: es un número del 0 al 5.

-Significado de los valores:

0: No presente o sin influencia.

1: Influencia incidental o presencia incidental.

2: Influencia moderada o presencia moderada.

3: Influencia media o presencia media.

4: Influencia significativa o presencia significativa.

5: Fuerte influencia o fuerte presencia.

Factor	Descripción	Peso	Valor	Comentario	Σ (Pesoi * Valori)
T1	Sistema distribuido.	2	0	Sistema centralizado.	0
T2	Objetivos de performance o tiempo de respuesta.	1	4	Velocidad bastante rápida. Uso de caché.	4
T3	Eficiencia del usuario final.	1	4	Necesidad de eficiencia.	4
T4	Procesamiento interno complejo.	1	0	No hay cálculos complejos.	0
T5	El código debe ser reutilizable.	1	4	Código reutilizable.	4
T6	Facilidad de instalación.	0.5	4	Debe ser fácil de instalar.	2
T7	Facilidad de uso.	0.5	4	Debe ser fácil de usar.	2
T8	Portabilidad.	2	4	Sistema portable.	8
T9	Facilidad de cambio.	1	4	Se requiere un costo de mantenimiento bajo.	4

T10	Concurrencia.	1	4	Sí.	4
T11	Incluye objetivos especiales de seguridad.	1	3	Seguridad normal.	3
T12	Provee acceso directo a terceras partes	1	4	Los usuarios tienen acceso directo	4
T13	Se requieren facilidades especiales de entrenamiento a los usuarios.	1	1	Pocos usuarios internos, sistema fácil de usar.	1
Total:					**40**

Tabla # 51: Σ (Pesoi * Valori) para el factor de complejidad técnica.

TCF = 0.6 + 0.01 * Σ (Pesoi * Valori)

Respuesta: TCF= 0.6+0.01*40=1

Factor de ambiente (EF)

Para Calcular EF

EF = 1.4 - 0.03 * Σ (Pesoi * Valori)

Donde: **Valor**: es un número del 0 al 5.

Factor	**Descripción**	**Peso**	**Valor**	**Comentario**	**Σ (Pesoi * Valori)**
E1	Familiaridad con el modelo de proyecto utilizado.	1.5	4	El equipo está familiarizado con el modelo.	6
E2	Experiencia en la aplicación.	0.5	4	El equipo ha trabajado en este tipo de sistema.	2

E3	Experiencia en orientación a objetos.	1	4	El equipo tiene experiencia en la programación OO.	4
E4	Capacidad del analista líder.	0.5	4	Tiene conocimientos de ISW.	2
E5	Motivación.	1	5	El equipo está altamente motivado.	5
E6	Estabilidad de los requerimientos.	2	4	No se esperan cambios.	8
E7	Personal part-time.	-1	0	El equipo trabaja a tiempo completo.	0
E8	Dificultad del lenguaje de programación.	-1	2	Se programará en PHP.	-2
Total:					**25**

Tabla # 52: Σ (Pesoi * Valori) para el factor ambiente.

EF = 1.4 - 0.03 * Σ (Pesoi * Valori)

Respuesta: EF=1.4-0.03*25=0.65

UCP = UUCP * TCF * EF

UCP =86*1*0.65=55.9

Resultado del Paso 2

UCP=55.9

5.2.3 Paso 3. Calcular esfuerzo de FT Implementación

E = UCP * **CF**

Donde:

E: esfuerzo estimado en horas-hombre.

UCP: Puntos de Casos de Uso ajustados.

CF: factor de conversión.

Factor de conversión (CF)

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)

Para calcular CF

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

Total EF =2+0=2

Respuesta:

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

E = UCP * **CF**

E=55.9*20 horas-hombre=1118 horas-hombre

Resultado del Paso 3: E=1118 horas-hombre

5.2.4 Paso 4. Calcular esfuerzo de todo el proyecto

-El valor del esfuerzo calculado representa el esfuerzo del Flujo de Trabajo de Implementación, por comparación salen el resto de los esfuerzo y la suma de ellos es el esfuerzo total (ET).

Actividad	**% esfuerzo**	**Valor esfuerzo**
Análisis.	10%	280 horas-hombre
Diseño.	20%	559 horas-hombre
Implementación.	40%	1118 horas-hombre
Prueba.	15%	419 horas-hombre
Sobrecarga.	15%	419 horas-hombre
Total:	100%	2795 horas-hombre

Tabla # 53: Esfuerzo de los flujos de trabajo.

Un mes tiene como promedio 30 días y se supone que una persona trabaja 8 horas por día; la cantidad de horas que puede trabajar una persona en 1 mes es 240 horas.

ET = 2795 horas-hombre y por cada 240 horas se tiene 1 mes, por tanto

ET = 12 mes-hombre

Esto quiere decir que 1 persona puede realizar el sistema (MetSoft) en aproximadamente 1 año. Como este sistema estará desarrollado por dos personas, entonces MetSoft puede llevarse a cabo aproximadamente en 6 meses.

5.3 Beneficios tangibles e intangibles

El Sistema para medir el estado de avance de los proyectos en la UCI (MetSoft) tiene como principal objetivo controlar y medir el estado de avance de los proyecto en la UCI. Por lo que contará con el constante intercambio de usuarios, de ahí su interfaz gráfica sencilla y amigable. Los beneficios más destacables brindados por este sistema son principalmente intangibles:

- ✓ Permitirá que los líderes de proyectos lleven un control detallado y minucioso del trabajo realizado en la fase de implementación.
- ✓ Aportará datos estadísticos en el desarrollo de software en la UCI.
- ✓ Permitirá que los líderes de proyectos puedan adquirir una experiencia sustancial en el desarrollo de software, ya que les permitirá evaluar su trabajo semana tras semana.
- ✓ Establecerá una métrica basada en el desarrollo de software en la UCI, con el fin de crear una base para el seguimiento de los proyectos tanto en la escuela como en el país.
- ✓ Creará registros de todo el trabajo en la fase de implementación de cualquier proyecto de software en que se utilice la métrica, esto permitirá que en condiciones similares o parecidas se pueda definir el tiempo necesario para otro proyecto creando así una base de datos de registros de cada proyecto.
- ✓ Posibilitará a los diferentes directivos obtener importantes reportes del estado en que se encuentran los proyectos productivos en la UCI.
- ✓ Hará más eficiente el control de la información en el proceso productivo de la UCI.

Entre los beneficios tangibles se encuentran que le permitirá a todos los directivos de la UCI tener conocimiento del estado de los proyecto.

5.4 Análisis de costos y beneficios

El desarrollo de este sistema no requiere grandes gastos de recursos, ni de tiempo; los servidores que existen en la Universidad son capaces de soportar la base de datos que contiene la información, así como el software en su totalidad. Estará desarrollado casi en su totalidad por software libre. No reportará gastos por concepto de entrenamiento a los usuarios pues será sencillo y de fácil manejo. Pondrá en las manos de la UCI una herramienta para la gestión de sus proyectos productivos mejorando un grupo de problemas (mencionados en la introducción a este Trabajo de Diploma). Teniendo en cuenta todos los beneficios antes mencionados y que el esfuerzo necesario para el desarrollo de MetSoft es de 12 mes/hombre, se considera factible el desarrollo de esta aplicación.

5.5 Conclusiones parciales

A partir de este capítulo se obtuvo una estimación del tiempo y el costo de este proyecto, obteniéndose además la cantidad de horas hombre que se necesita para realizar el mismo y después de hacer todo este análisis se llego a la conclusión de que el sistema es factible ya que la cantidad de esfuerzo requerido es aceptable para dos estudiantes, con el software y hardware instalado fue suficiente para la implementación del mismo, por lo que no se necesita de mayor inversión.

Conclusiones.

Una vez definida y aplicada la métrica se arribaron a las siguientes conclusiones:

1- Se obtuvo una herramienta en su versión 1.0 capaz de aplicar la métrica en un ambiente web.

2- La documentación obtenida en el proceso de desarrollo de la herramienta constituye una guía para futuros desarrollos y constituye la base para futuras mejoras y extensiones de la herramienta obtenida.

3- La aplicación diseñada permitió obtener cuantitativamente el estado de avance de la implementación del software y constituye en sí un registro histórico de la productividad del trabajo. Mientras mayor nivel de detalle se tenga de las tareas de implementación y le sean asociadas a estas un peso que se corresponda con la fracción del tiempo total que se necesita para ejecutarse, mayor exactitud tendrá el valor del avance que genera la métrica.

4- Se creó un punto de partida para cuantificar el estado de avance de los proyectos.

5- Se sentaron las bases para comenzar a obtener registros históricos de los proyectos desarrollados en la UCI.

6- La aplicación puede ser utilizada en varios proyectos de desarrollo de software, redefiniendo las tareas de implementación y su peso específico dentro del tiempo total de desarrollo.

Recomendaciones

Para un posterior desarrollo y perfeccionamiento del sistema implementado se recomienda:

1- Agregar nuevas funcionalidades para lograr una mejora en la facilidad de utilización de la herramienta.

2- Continuar perfeccionando la herramienta desarrollada a partir de los nuevos requisitos que puedan surgir como resultado de su explotación.

3- Utilizar la herramienta en aquellos proyectos que precisen de ella.

Bibliografía

Autores Varios. Ingeniería de Software 1. [Online] 2008. http://ing.de.soft1.googlepages.com/home

CAPUCHINO, A. M. M. S. “Control y gestión de proyectos software, Unidad 4: Estimación de Proyectos Software". 1996. P.

Douglas, Korry and Douglas, Susan. PostgreSQL. The compressive guide to building, programming, and administering PostgreSQL databases. s.l.: Sams Publishing, 2006.

Díaz Fuentes Daily, Rodríguez Torres Rosario, “*Métrica para el estado de avance del flujo de implementación del SIGEP*”, Universidad de Ciencias Informáticas, Facultad 4, Junio del 2008

Jacobson, Ivar, Booch, Grady and Rumbaugh, James. El Proceso Unificado de Desarrollo de Software. Madrid : PEARSON EDUCACION, 2000.

Monroe, Robert, Melton, Ralph and Garlan, David**.** *Architectural Styles, design patterns, and objects.* s.l. : IEEE Software, 1997.

Molpeceres, Alberto. Procesos de desarrollo: RUP, XP y FDD. 2002

NUSENOFF, R. E. and D. C. BUNDE. ” A Guide Book and Spedsheet Tool for a Corporate Metrics Program”. Holanda, 1993. P.

Potencier, Fabien and Zaninotto, François. Symfony, la guía definitiva. Librosweb.es. [Online] 2007. http://www.librosweb.es/symfony.

Rumbaugh, James. The Unified Modeling Language Reference Manual. s.l. : Addison Wesley, 1999.

RUP, Rational Unified Process. 2003. P.

ZUSE, H. “History of Software Measurement”. 1995. p.

Referencias Bibliográficas

[1] https://pid.dsic.upv.es/C1/Material/Documentos%20Disponibles/Introducción%20a%20RUP.doc

[2] Gracia, Joaquín, (Mayo del 2005)UML: Diagramas UML. ¿Qué es UML? [Online].

http://www.ingenierosoftware.com/analisisydiseno/uml.php

[3] Herramientas Case [Online]

http://www.cyta.com.ar/biblioteca/bddoc/bdlibros/proyectoinformatico/libro/c5/c5.htm

[4] Quintero, J. B. (2005). Un estudio comparativo de herramientas para el modelado con UML. Revista Universidad de EAFIT. , 60-75.

[5] González, C. S. (s.f.). ONess. Recuperado el 21 de Marzo de 2008, de ONess: http://oness.sourceforge.net/proyecto/html/ch03s02.html

[6] La guía definitiva de Symfony [Online]
http://www.librosweb.es/symfony_1_2/

[7] Álvarez, R. (s.f.). desarrolloWeb.com. Recuperado el 18 de Marzo de 2008, de desarrolloWeb.com: http://www.desarrolloWeb.com

[8] W3C WORLD WIDE WEB. (Abril del 2008). Recuperado el 18 de Marzo de 2008, de W3C: http://www.w3c.es/divulgacion/guiasbreves/serviciosWeb/

[9] Álvarez, R. (s.f.). desarrolloWeb.com. Recuperado el 18 de Marzo de 2008, de desarrolloWeb.com: http://www.desarrolloWeb.com/articulos/(239)(1325).php

[10] Vieyra, G. E. (23 de Mayo de 2000). Conceptos de HTML. Recuperado el 18 de Marzo de 2008, de Conceptos de HTML: http://www.fismat.umich.mx/~elizalde/tesis/node49.html

[11] Alvarez Miguel Angel, "*Zend Studio: Editor web orientado a la programación de páginas PHP, con ayudas en la gestión de proyectos y depuración de código*", Junio del 2003. [Online]
http://www.desarrolloweb.com/articulos/1178.php

[12] Worsley, J. (2001). Proyecto S.O.B.L. Traducciones. Recuperado el 21 de Marzo de 2008, de Proyecto S.O.B.L. Traducciones.: http://www.sobl.org/traducciones/practical-postgres/node12.html

[13] Worsley, J. (2001). Proyecto S.O.B.L. Traducciones. Recuperado el 21 de Marzo de 2008, de Proyecto S.O.B.L. Traducciones.: http://www.sobl.org/traducciones/practical-postgres/node19.html

[14] Diagramas de Despliegue [Online]

http://tvdi.det.uvigo.es/~avilas/UML/node50.html

[15] Vilas, A. F. (20 de 03 de 2001). Recuperado el 18 de Mayo de 2008, de

http://www-gris.det.uvigo.es/~avilas/UML/node49.html

Glosario de Términos.

AIEE: Instituto Americano de Ingenieros Eléctricos.

API: Applications Programming Interface / Interfaz de Programación de Aplicaciones.

AS: (Arquitectura de Software) es el esqueleto o base de una aplicación, que es analizada desde varios puntos de vista. Representa la organización fundamental de un sistema, constituida en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución.

CASE: Computer Aided Software / Ingeniería de Software Asistida por Computadora.

CC: Clase controladora.

CE: Clase entidad.

CF: Factor de conversión.

CI: Clase interfaz.

Componente: Que forma parte de alguna cosa o de su composición.

CP: Caso de Prueba Un caso de prueba será un conjunto de entradas con datos de prueba, unas condiciones de ejecución, y unos resultados esperados.

CU: Caso de Uso.

E: Esfuerzo estimado en horas-hombre.

EF: Environment Factor / Factor de Ambiente.

EP: Escenario de prueba.

Framework: Es una estructura de soporte definida, mediante la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

GB: Gigabyte.

HTML: Hypertext Markup Language. Lenguaje de marcado de hipertexto, usado para escribir documentos para servidores World Wide Web.

HTTP: Hypertext Transfer Protocol / Protocolo de Transferencia de Hipertexto. Modo de comunicación para solicitar páginas Web.

IDE: Integrated Development Environment / Entorno de Desarrollo Integrado. Entorno de programación que ha sido empaquetado como un programa de aplicación, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI.
IEEE: Corresponde a las siglas de The Institute of Electrical and Electronics Engineers, el Instituto de Ingenieros Eléctricos y Electrónicos, una asociación técnico-profesional mundial dedicada a la estandarización, entre otras cosas.

IIS: Internet Information Server. Servidor Web recomendable para plataforma Windows 2000, dado por la integración que presenta con su Servicio de Directorios que permite el desarrollo de aplicaciones basadas en la Web fiables y escalables.

Informix: Es una familia de productos RDBMS de IBM, adquirida en 2001 a una compañía (también llamada Informix o Informix Software) cuyos orígenes se remontan a 1980.

Interbase: Es un Sistema de Administración de Base de Datos Relacionales (RDBMS) desarrollada y comercializada por la compañía Borland Software Corporation y actualmente desarrollada por su filial CodeGear.

ISO: Organización Internacional de Estandarización.

LDAP: Lightweight Directory Access Protocol. Es un protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red. Habitualmente, almacena la información de login (usuario y contraseña) y es utilizado para autenticarse aunque es posible almacenar otra información.

Linux: Sistema operativo similar a Unix que utiliza como base las herramientas de sistema de GNU y el núcleo Linux. Su desarrollo es uno de los ejemplos más prominentes de software libre; todo el código fuente puede ser utilizado, modificado y redistribuido libremente por cualquiera.

MB: Megabyte.

Métrica: Una forma de medir y una escala, definidas para realizar mediciones de uno o varios atributos software.

Microsoft: Corporation Microcomputer Software. Empresa multinacional estadounidense dedicada al sector de la informática.

MVC: Model/View/Controller / Modelo Vista Controlador.

Módulo: Encapsula un conjunto de funciones que debe realizar el sistema, las cuales son agrupadas por tener características muy similares y se definen en la etapa de diseño.

MSF: Microsoft Solution Framework. Metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos.

MVCC: Control de Concurrencia Multi-Versión (Multi-Version Concurrency Control).

MVC: Patrón de Arquitectura (Modelo Vista Controlador).

MySQL: Es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones

Novell Netware: Es un Sistema operativo de red. Es una de las plataformas de servicio más fiable para ofrecer acceso seguro y continuado a la red y los recursos de información, sobre todo en cuanto a servidores de archivos. Unix: (registrado oficialmente como UNIX®) es un sistema operativo portable, multitarea y multiusuario.

Oracle: Es un sistema de gestión de base de datos relacional (o RDBMS por el acrónimo en inglés de Relational Data Base Management System), desarrollado por Oracle Corporation.

ORM: El mapeo objeto-relacional (más conocido por su nombre en inglés, Object-Relational mapping, o sus siglas O/RM, ORM, y O/R mapping) es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional.

PDF: Portable Document Format / Formato de Documento Portátil.

PHP: Hypertext Preprocessor. Lenguaje de programación del lado del servidor que permite crear y ejecutar aplicaciones Web dinámicas e interactivas.

Plugins: Pequeños archivos que añaden mayores funcionalidades y características a un programa.

Proyecto: Proceso único consistente en un conjunto de actividades coordinadas y controladas con fechas de inicio y de finalización, llevadas a cabo para lograr un objetivo conforme con requisitos específicos.

RDBMS: es un Sistema Administrador de Bases de Datos Relacionales. Viene del acrónimo en inglés *Relational Data Base Management System*.

RoI: Define el comportamiento y responsabilidad de un individuo o grupo de individuos.

RUP: Rational Unified Process / Proceso Unificado de Desarrollo. Metodología para el desarrollo de software.

SEI: (Software Engineering Institute) es un centro de investigación y desarrollo sostenido por el Departamento de Defensa.

Smalltalk: el lenguaje de programación normalizado desde 1998.

TCF: Technical Complexity Factor / Factor de complejidad técnica.

UAW: Factor de Peso de los Actores sin Ajustar.

UCI: Universidad de las Ciencias Informáticas.
UCP: Puntos de Casos de Uso ajustados.

UML: Unified Modeling Language / Lenguaje Unificado de Modelado.
UUCP: Puntos de Casos de Uso sin ajustar.

UUCP: Puntos de Casos de Uso sin ajustar.

UUCW: Factor de Peso de los Casos de Uso sin ajustar.

UCP: Puntos de Casos de Uso ajustados.

WAL: Write Ahead Logging.

Windows: Familia de sistemas operativos desarrollados y comercializados por Microsoft. Existen versiones para hogares, empresas, servidores y dispositivos móviles, como computadores de bolsillo y teléfonos inteligentes SSL.

WWW: World Wide Web.

XP: Extreme Programing. Metodología ágil para el desarrollo de software.

Anexos

Diagrama de Clase CU: Mostrar Información

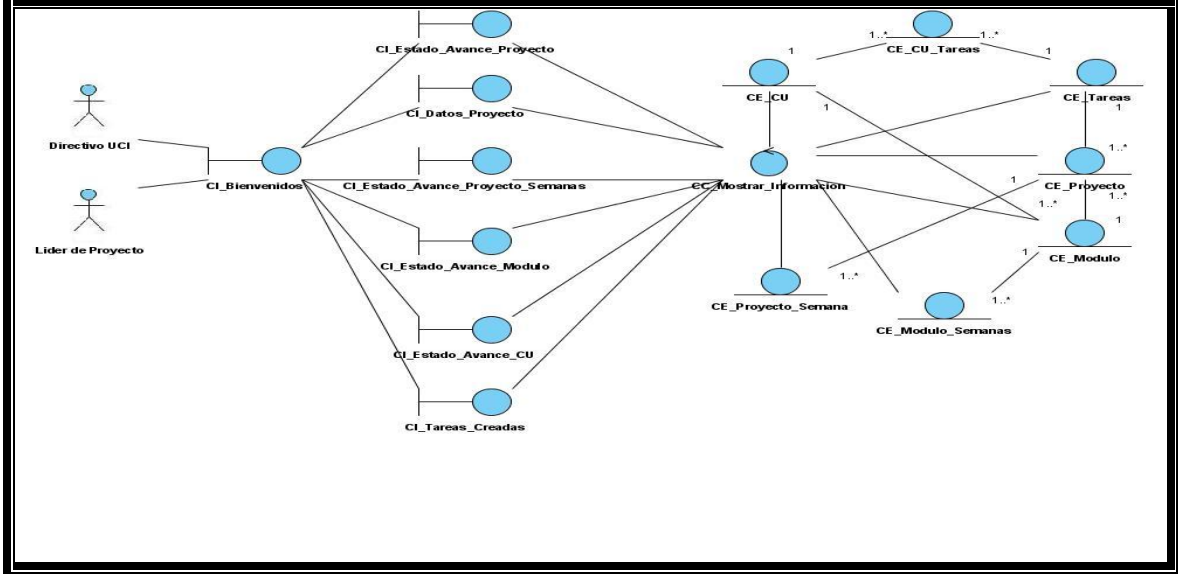

Figura # 61: Diagrama de Clase CU: Mostrar Información.

Diagrama de Clase CU: Graficar

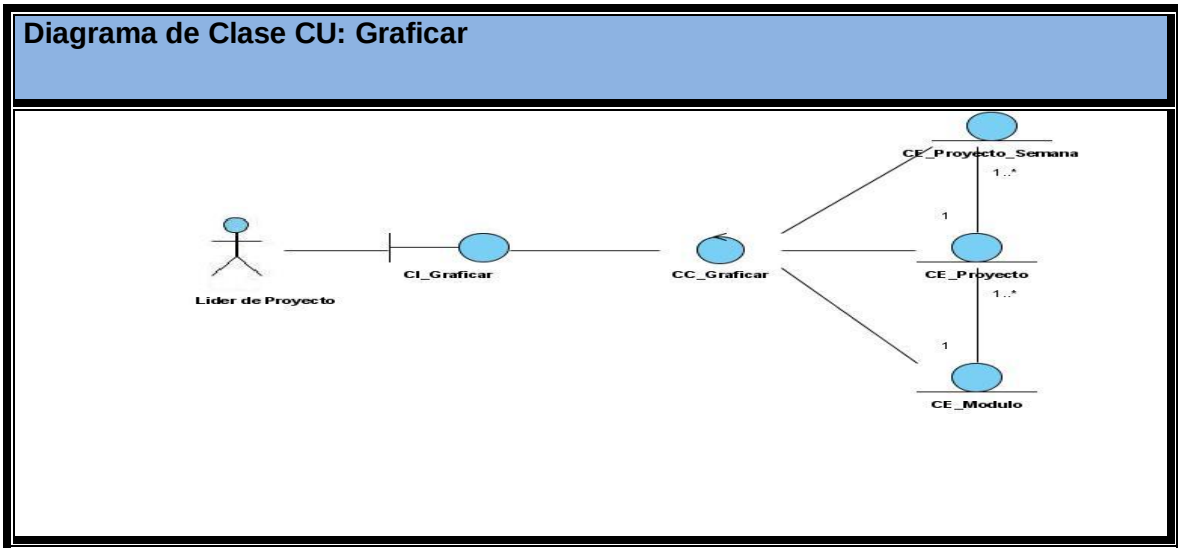

Figura # 62: Diagrama de Clase CU: Graficar.

Diagrama de Clases del diseño CU: Mostrar Información

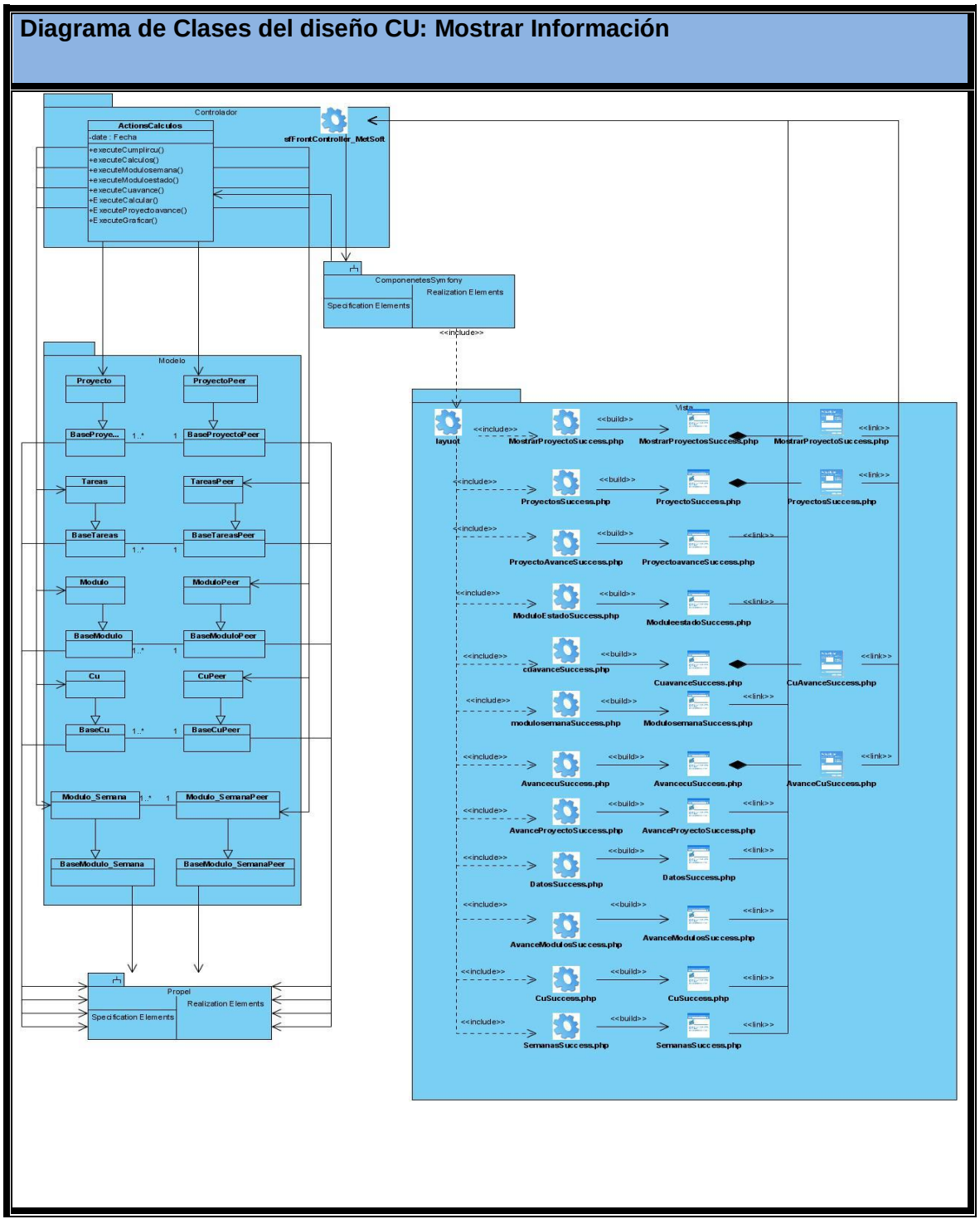

Figura # 63: Diagrama de Clases del diseño CU: Mostrar Información.

Diagrama de Clases del diseño CU: Graficar

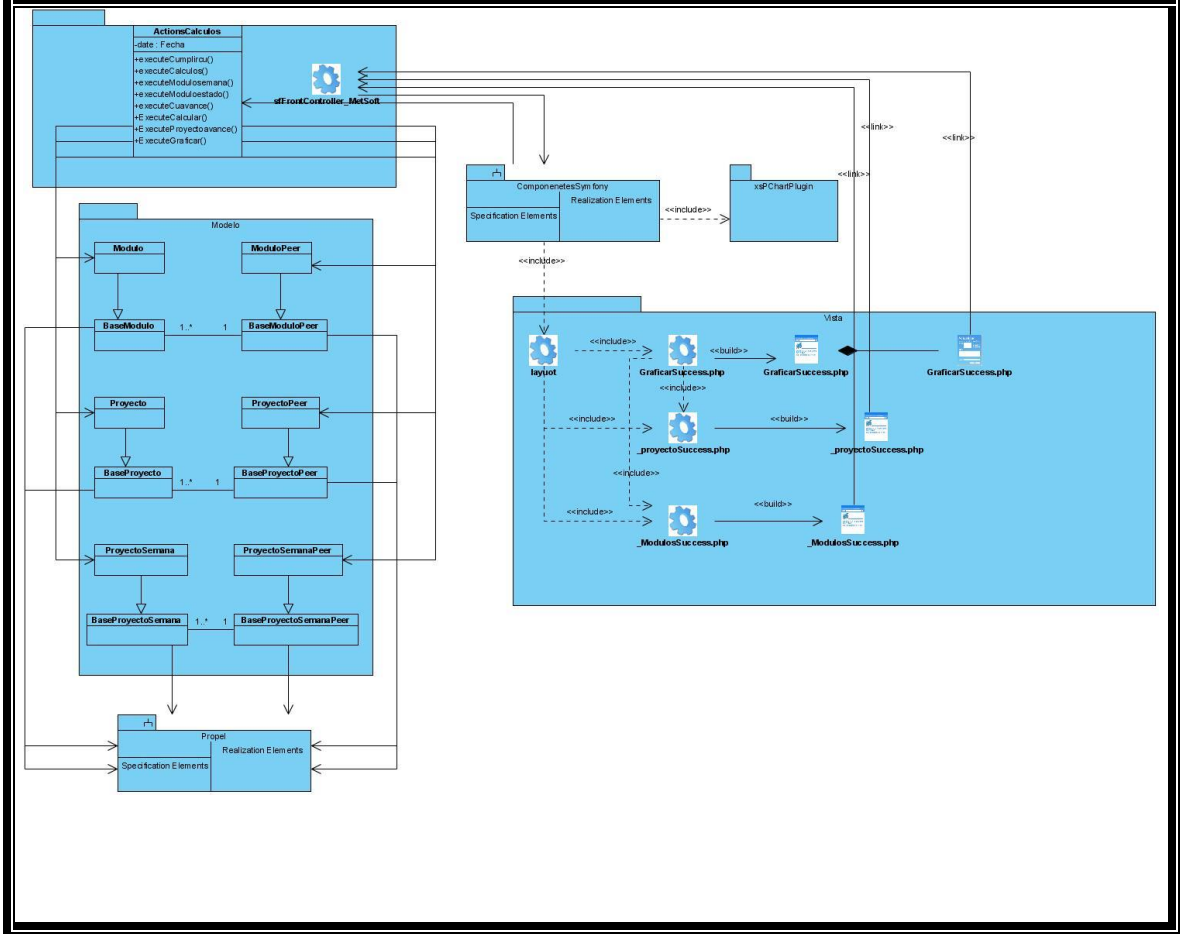

Figura # 64: Diagrama de Clases del diseño CU: Graficar.

Diagrama de Componente CU: Mostrar Información

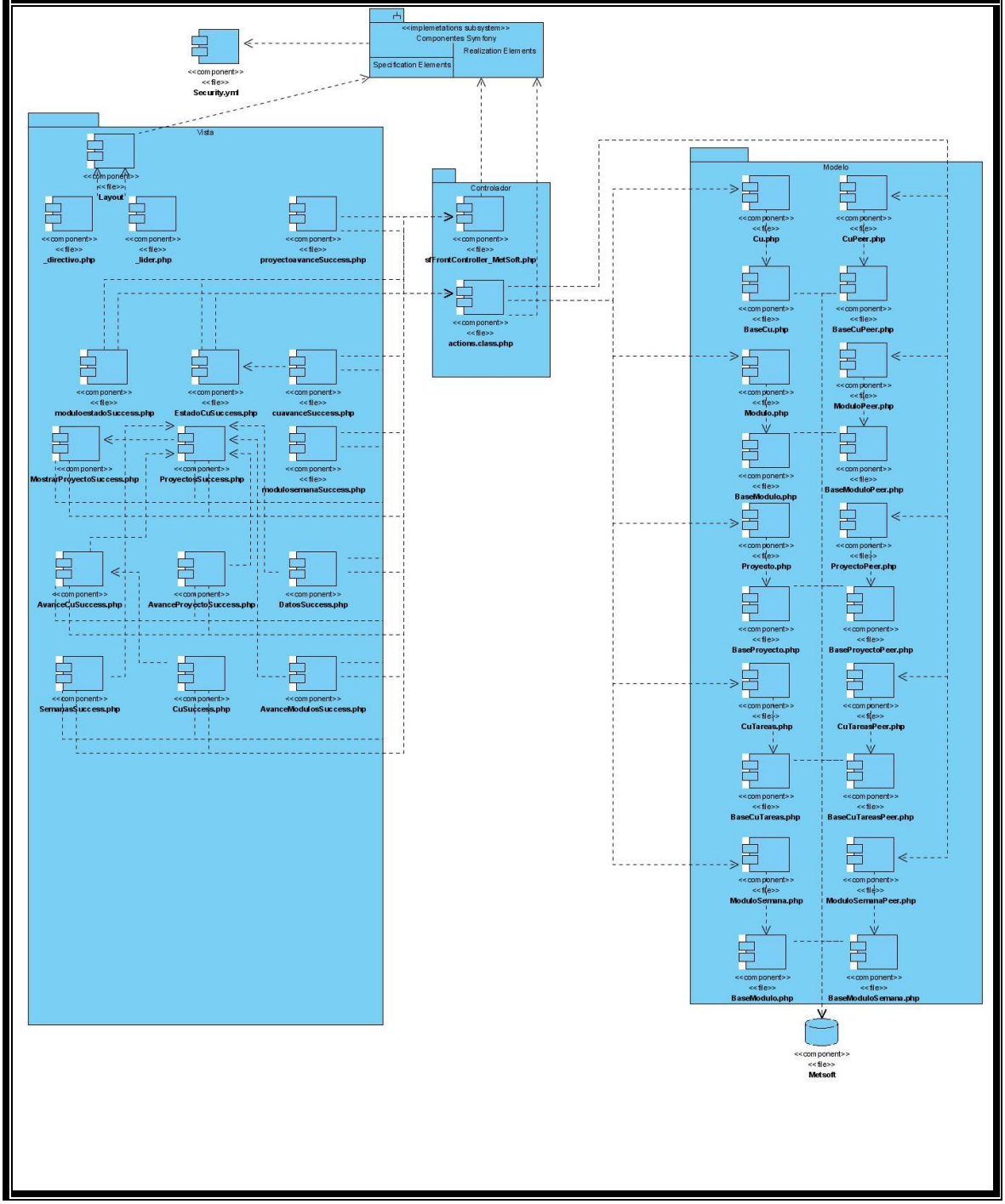

Figura # 65: Diagrama de Componente CU: Mostrar Información.

Diagrama de Componente CU: Graficar

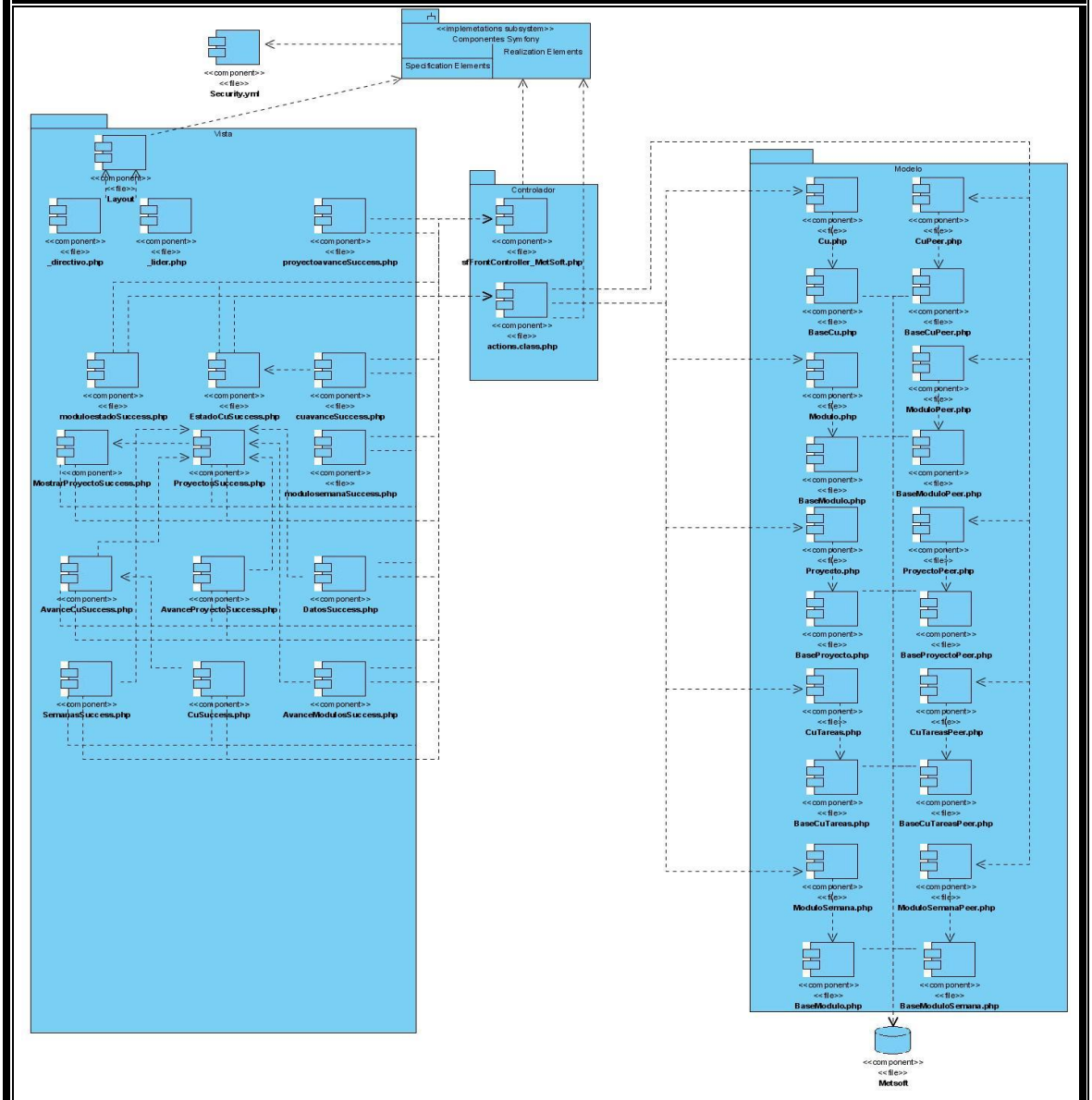

Figura # 66: Diagrama de Componente CU: Graficar.

Diagrama de Secuencia CU: Autentificarse (Sección Administrador)

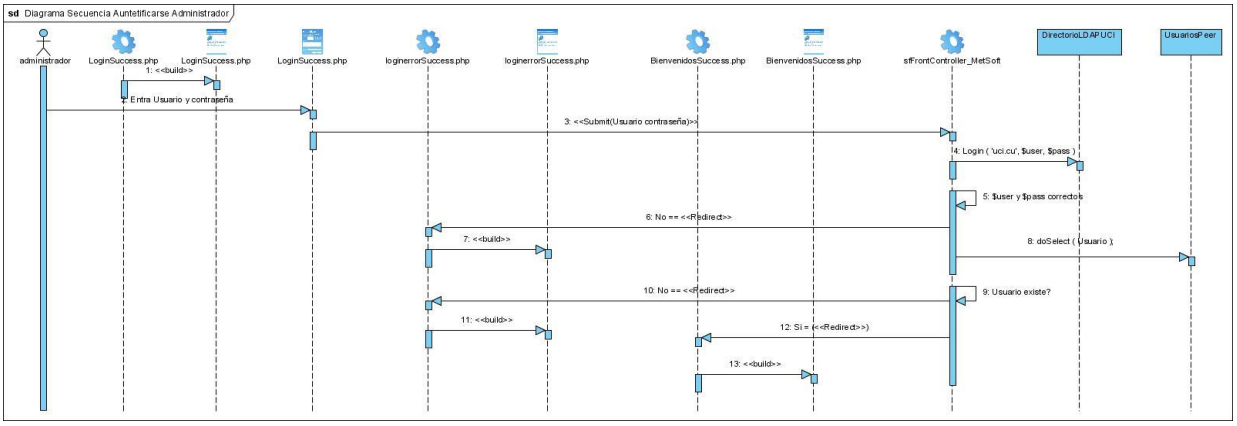

Figura # 67: Diagrama de Secuencia CU: Autentificarse (Sección Administrador).

Diagrama de Secuencia CU: Autentificarse (Sección Directivo)

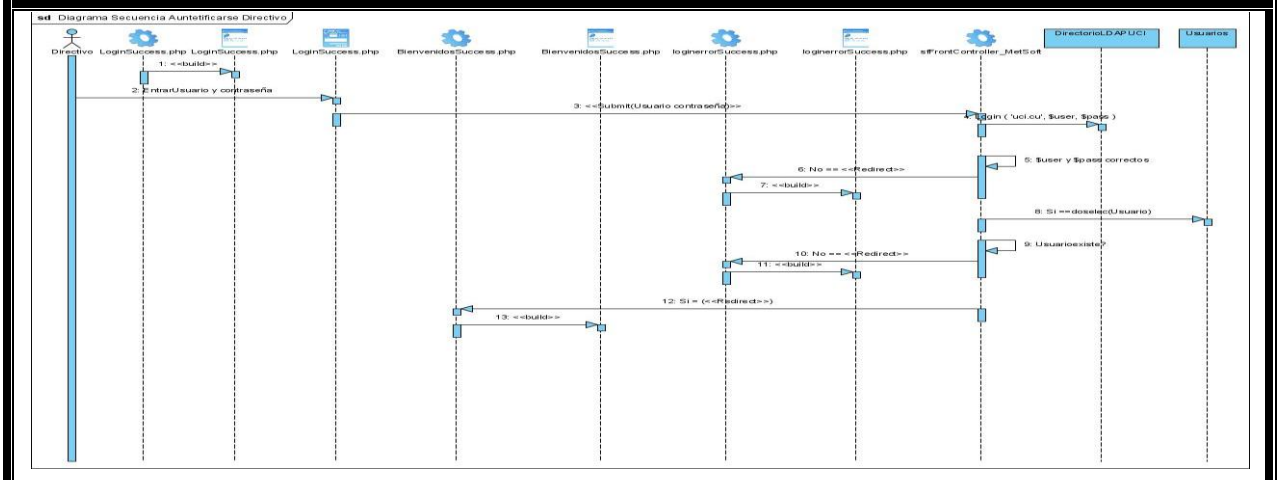

Figura # 68: Diagrama de Secuencia CU: Autentificarse (Sección Directivo).

Diagrama de Secuencia CU: Autentificarse (Sección Implementador)

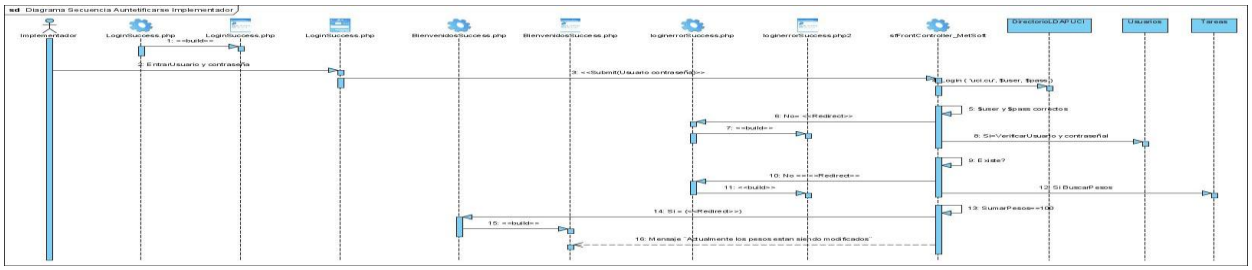

Figura # 69: Diagrama de Secuencia CU: Autentificarse (Sección Implementador).

Diagrama de Secuencia CU: Autentificarse (Sección Líder)

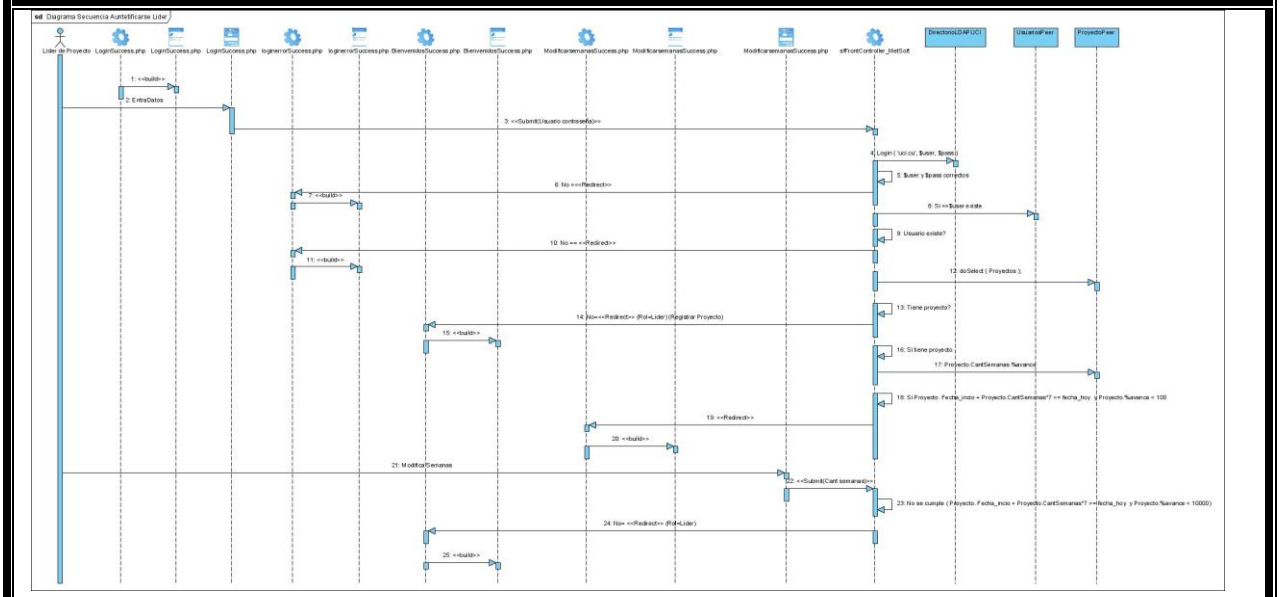

Figura # 70: Diagrama de Secuencia CU: Autentificarse (Sección Líder).

Diagrama de Secuencia CU: Gestionar Proyecto (Sección Registrar Proyecto)

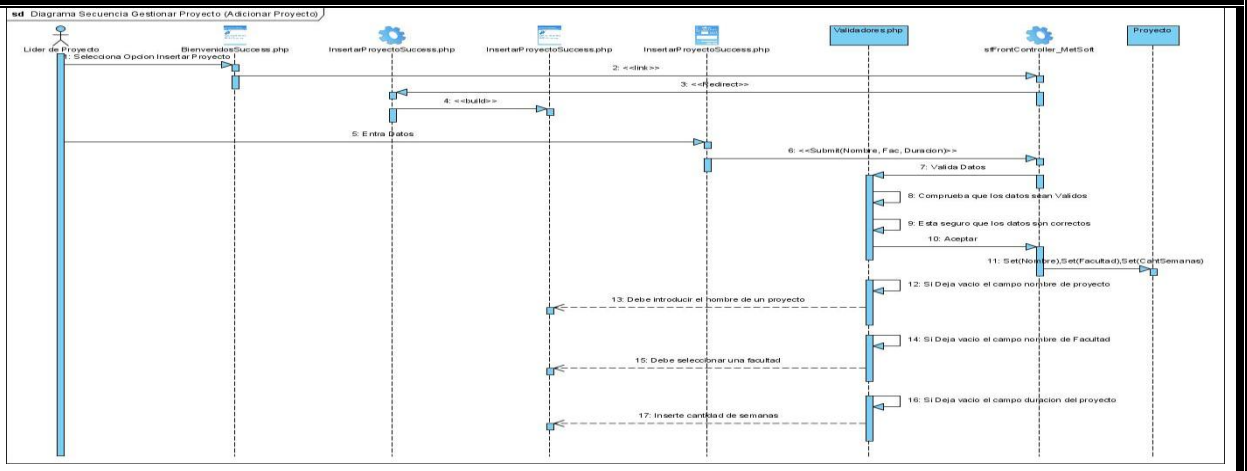

Figura # 71: Diagrama de Secuencia CU: Gestionar Proyecto (Sección Registrar Proyecto).

Diagrama de Secuencia CU: Gestionar Proyecto (Sección Modificar Proyecto)

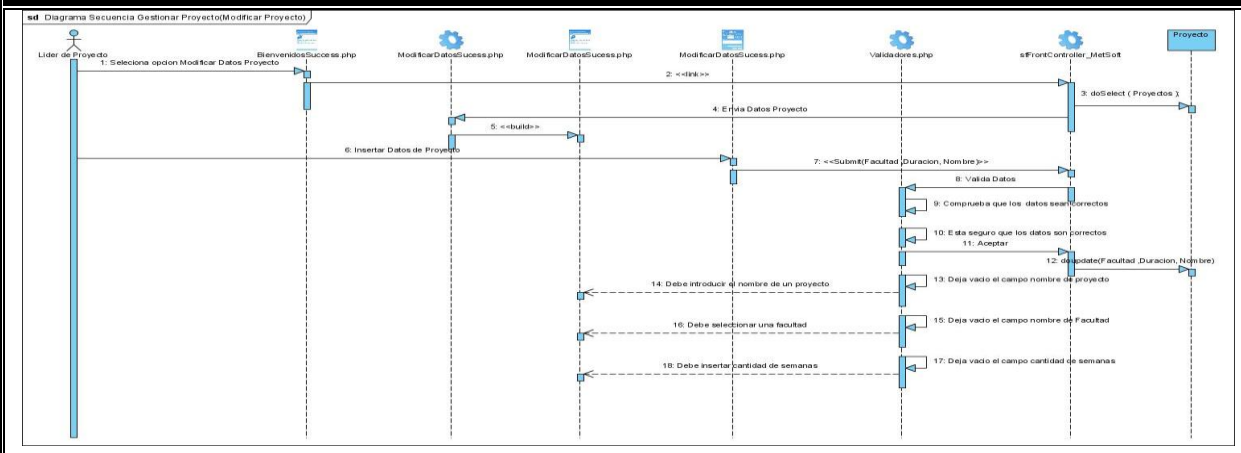

Figura # 72: Diagrama de Secuencia CU: Gestionar Proyecto (Sección Modificar Proyecto).

Diagrama de Secuencia CU: Gestionar Tareas (Sección Crear Tareas)

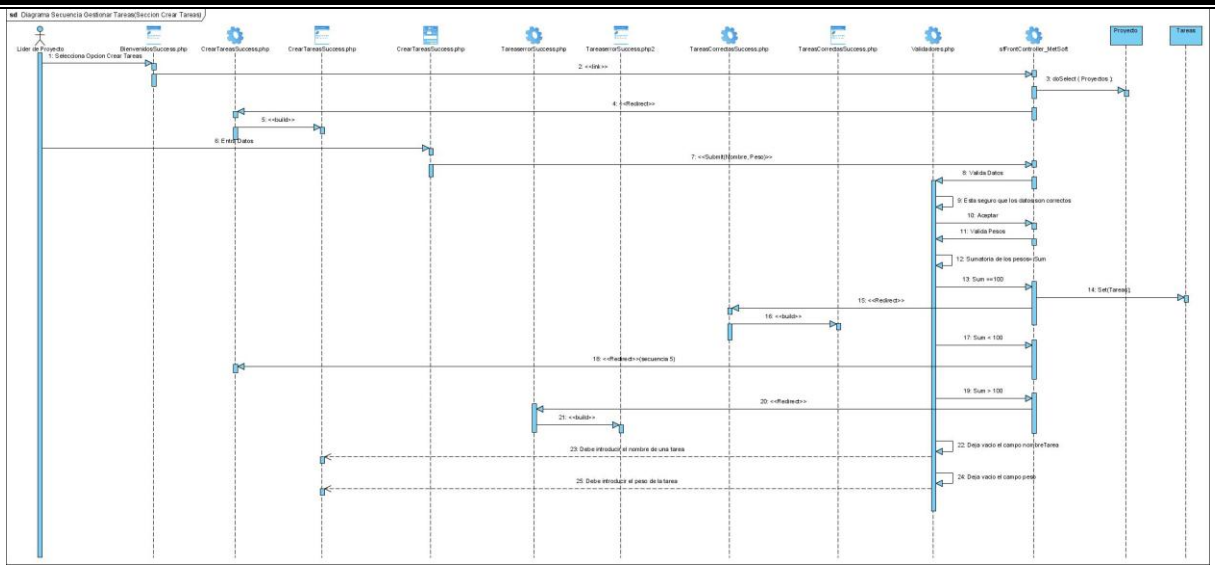

Figura # 73: Diagrama de Secuencia CU: Gestionar Tareas (Sección Crear Tareas).

Diagrama de Secuencia CU: Gestionar Tareas (Sección Eliminar Tareas)

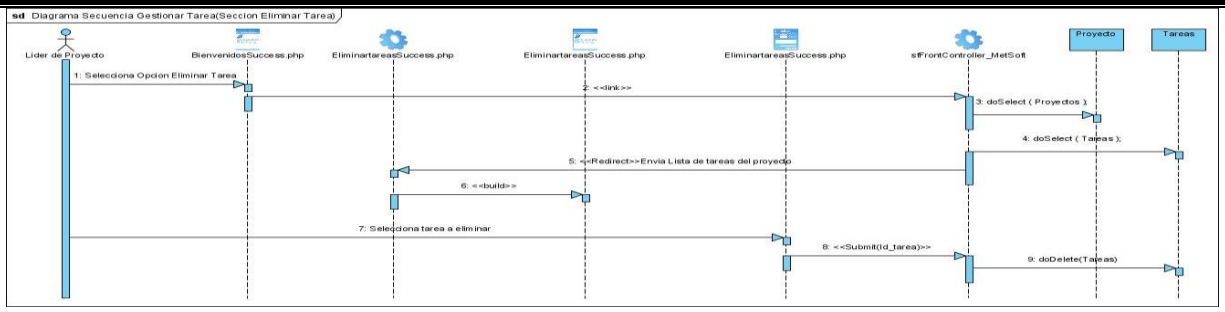

Figura # 74: Diagrama de Secuencia CU: Gestionar Tareas (Sección Eliminar Tareas).

Diagrama de Secuencia CU: Gestionar Tareas (Sección Modificar Pesos)

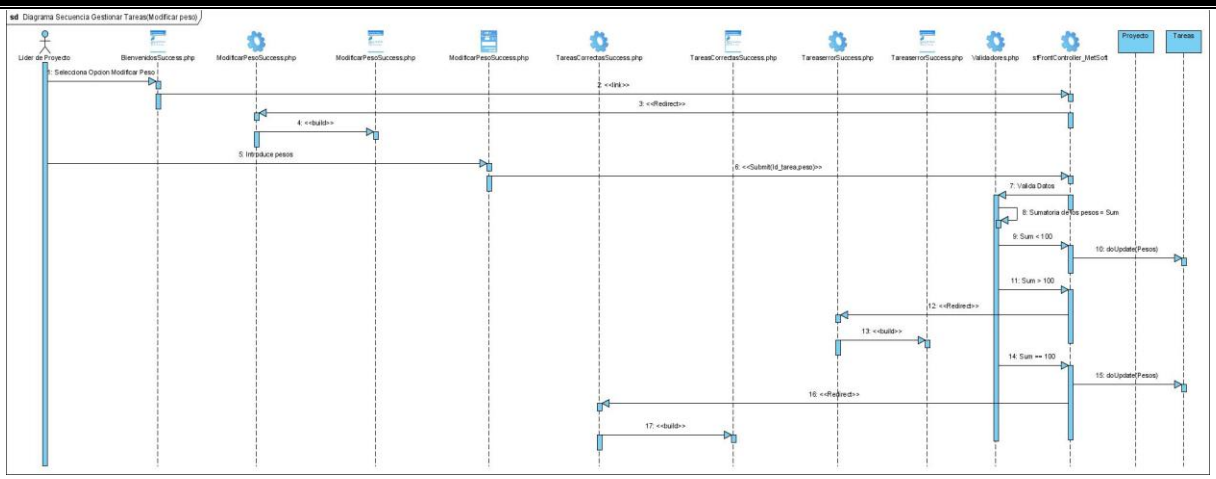

Figura # 75: Diagrama de Secuencia CU: Gestionar Tareas (Sección Modificar Pesos).

Diagrama de Secuencia CU: Crear Módulos

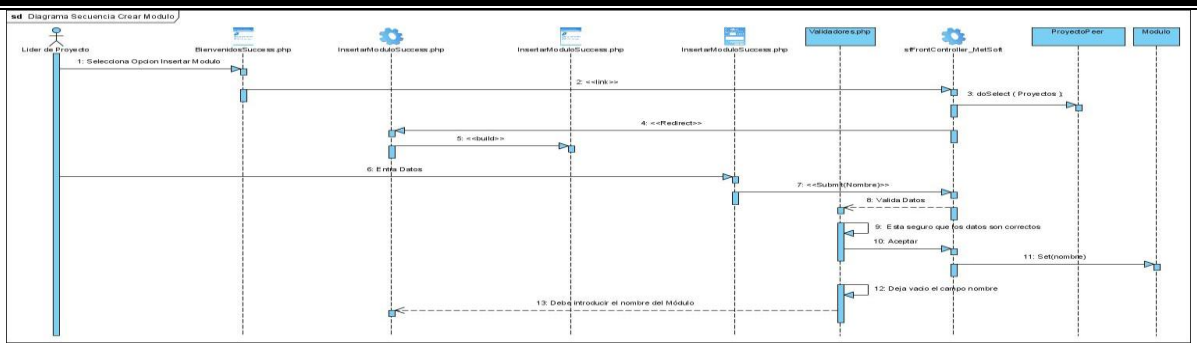

Figura # 76: Diagrama de Secuencia CU: Crear Módulos.

Diagrama de Secuencia CU: Crear CU

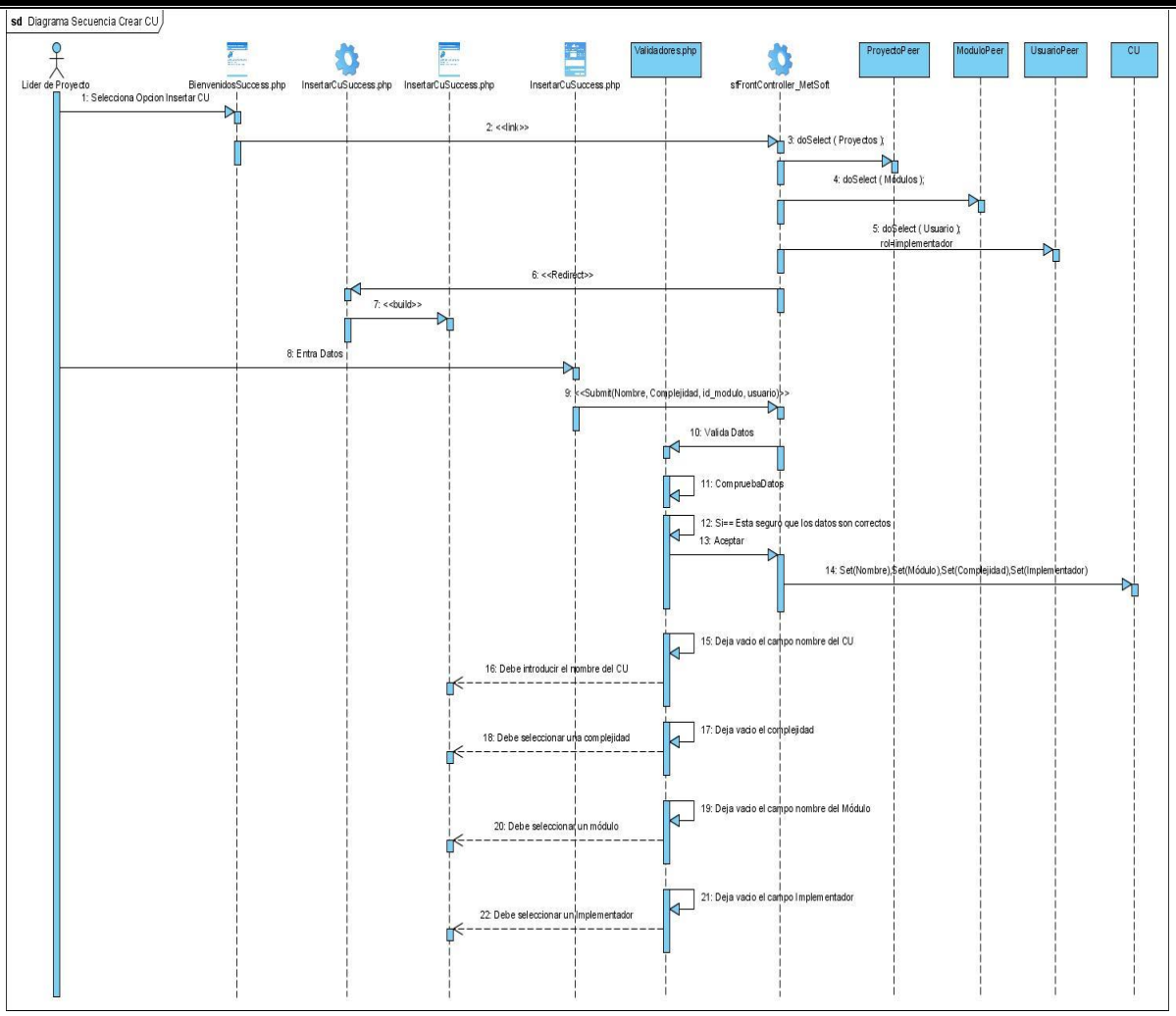

Figura # 77: Diagrama de Secuencia CU: Crear CU.

Diagrama de Secuencia CU: Gestionar Usuario (Sección Insertar Desarrollador)

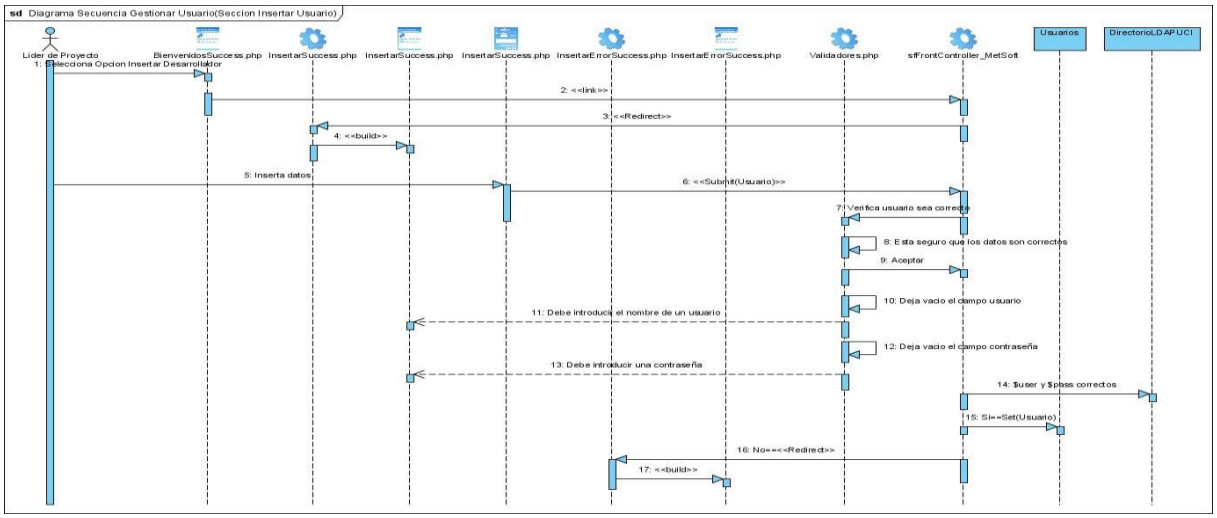

Figura # 78: Diagrama de Secuencia CU: Gestionar Usuario (Sección Insertar Desarrollador).

Diagrama de Secuencia CU: Gestionar Usuario (Sección Eliminar Usuario)

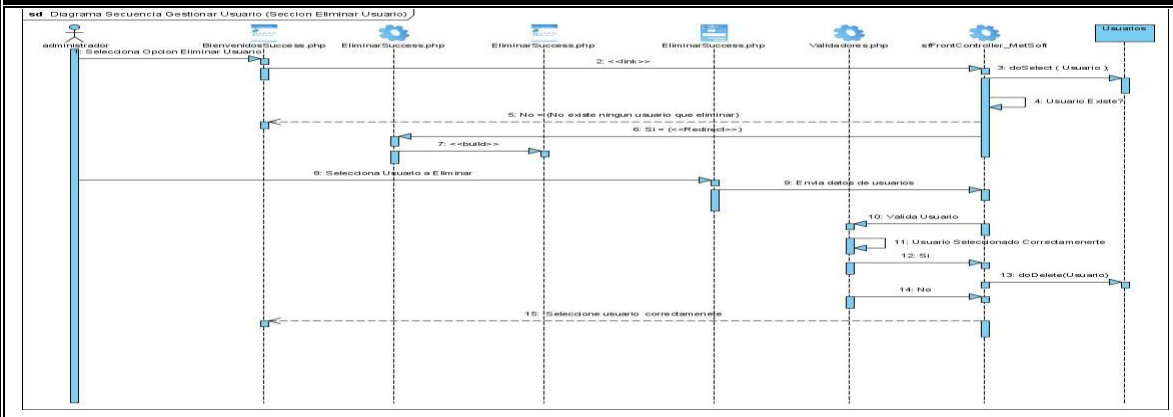

Figura # 79: Diagrama de Secuencia CU: Gestionar Usuario (Sección Eliminar Usuario).

Diagrama de Secuencia CU: Gestionar Usuario (Sección Insertar Usuario)

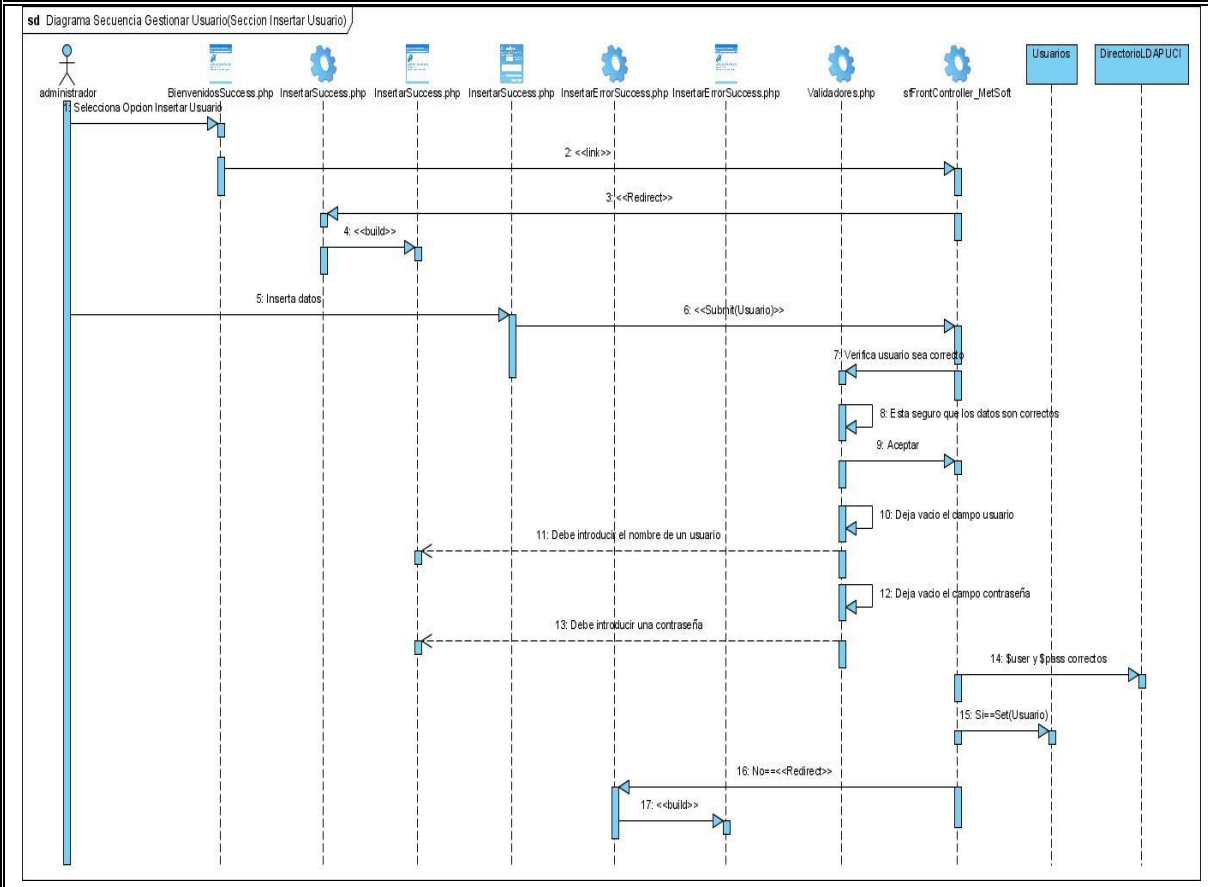

Figura # 80: Diagrama de Secuencia CU: Gestionar Usuario (Sección Insertar Usuario).

Diagrama de Secuencia CU: Gestionar Usuario (Sección Modificar Líder)

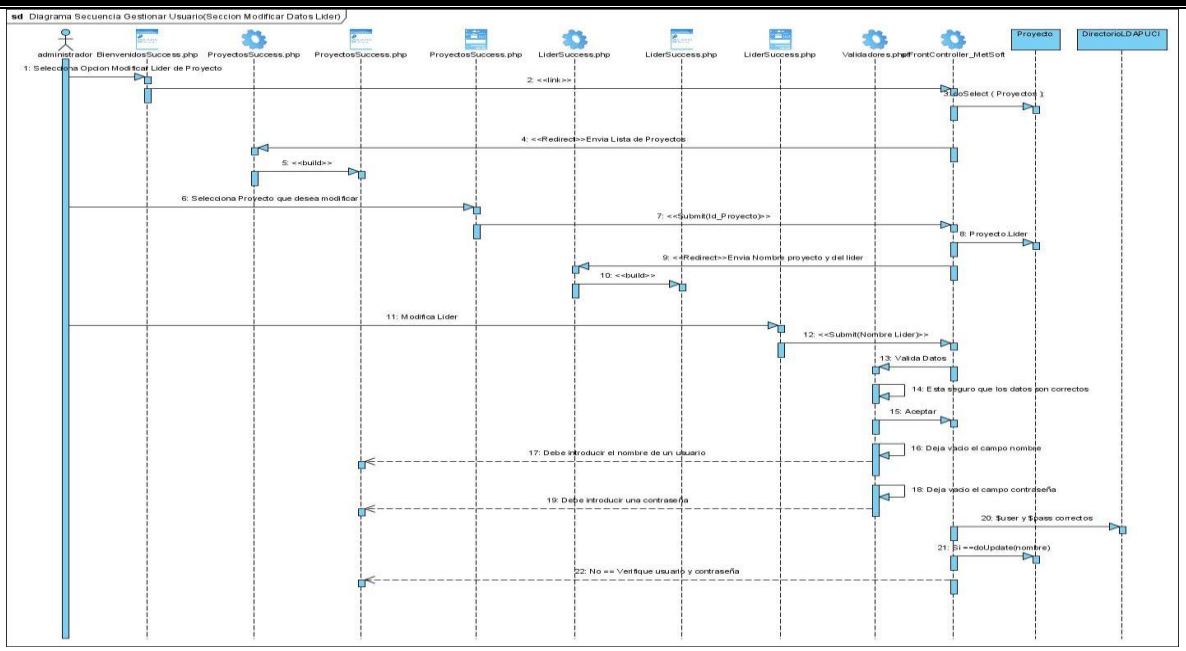

Figura # 81: Diagrama de Secuencia CU: Gestionar Usuario (Sección Modificar Líder).

Diagrama de Secuencia CU: Completar CU

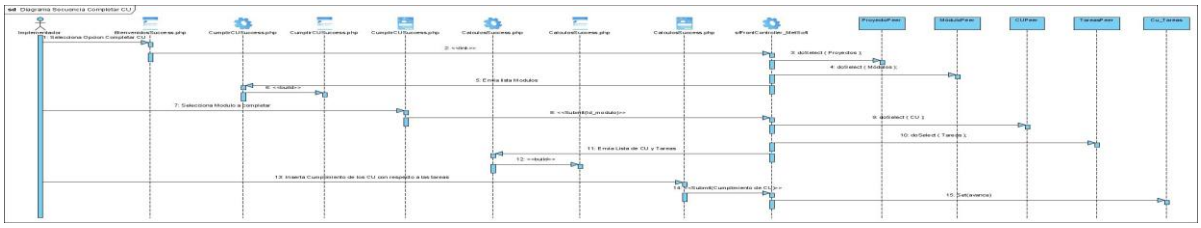

Figura # 82: Diagrama de Secuencia CU: Completar CU.

Diagrama de Secuencia CU: Mostrar Información Líder (Sección Avance CU)

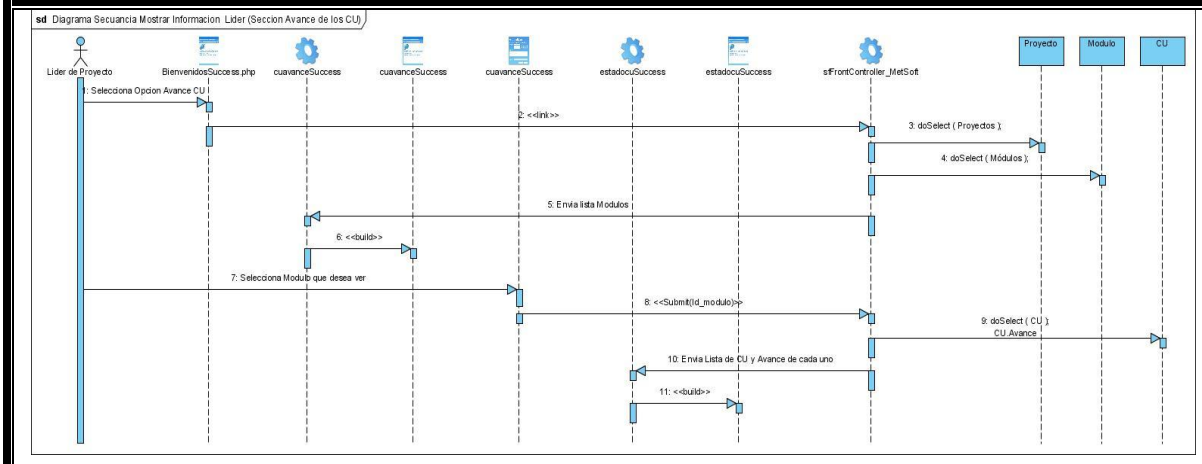

Figura # 83: Diagrama de Secuencia CU: Mostrar Información Líder (Sección Avance CU).

Diagrama de Secuencia CU: Mostrar Información Líder (Sección Avance Proyecto)

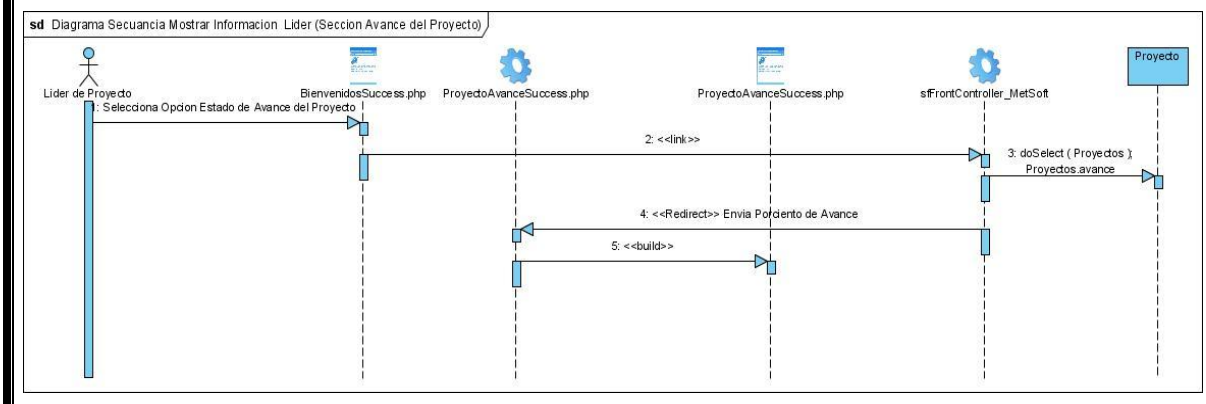

Figura # 84: Diagrama de Secuencia CU: Mostrar Información Líder (Sección Avance Proyecto).

Diagrama de Secuencia CU: Mostrar Información Líder (Sección Avance de los Módulos por semana)

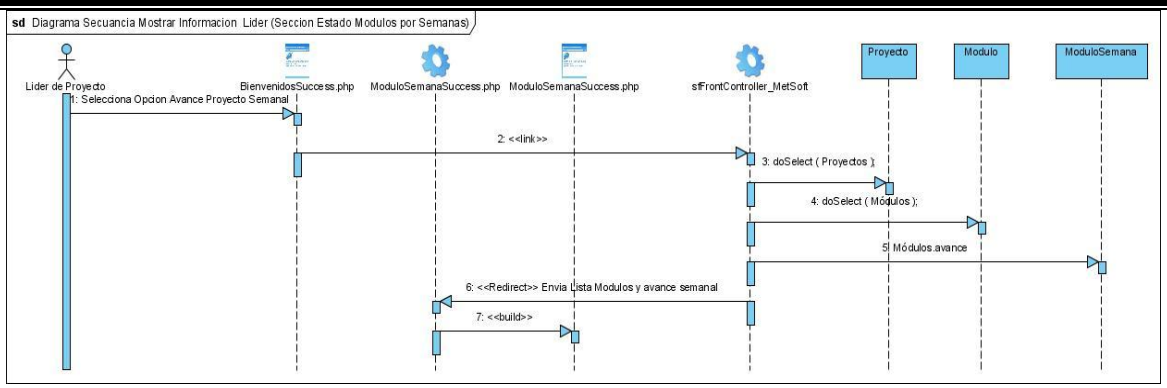

Figura # 85: Diagrama de Secuencia CU: Mostrar Información Líder (Sección Avance de los Módulos por semana).

Diagrama de Secuencia CU: Mostrar Información Líder (Sección Avance de los Módulos)

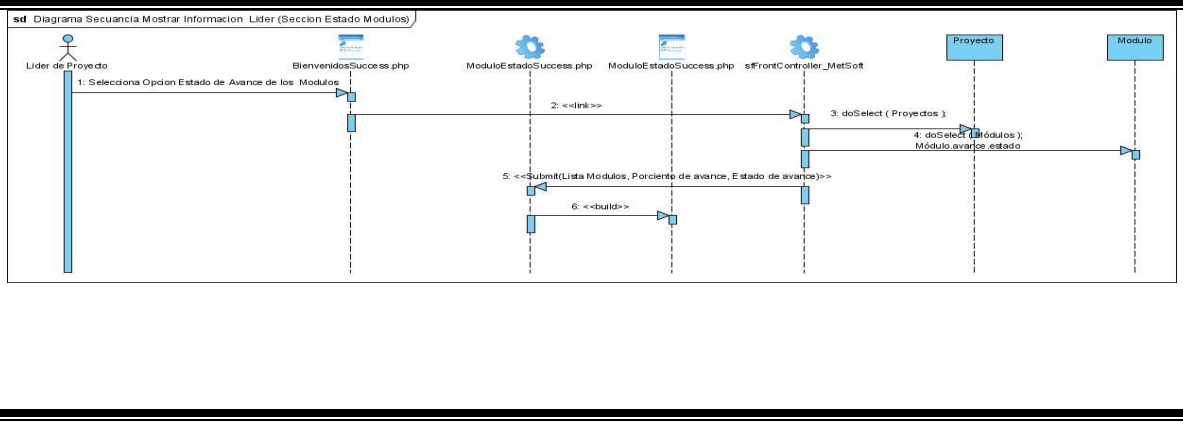

Figura # 86: Diagrama de Secuencia CU: Mostrar Información Líder (Sección Avance de los Módulos).

Diagrama de Secuencia CU: Mostrar Información Directivo (Sección Avance CU)

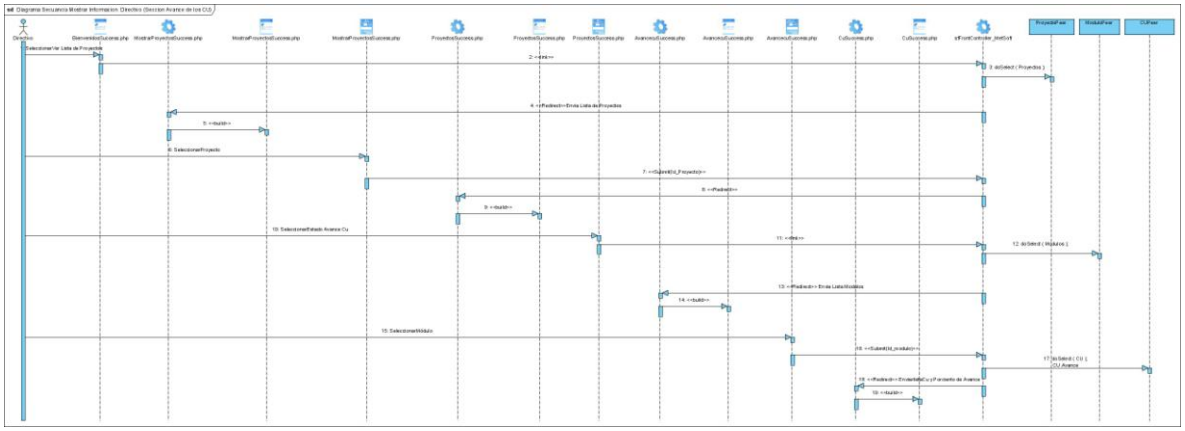

Figura # 87: Diagrama de Secuencia CU: Mostrar Información Directivo (Sección Avance CU).

Diagrama de Secuencia CU: Mostrar Información Directivo (Sección Avance Proyecto)

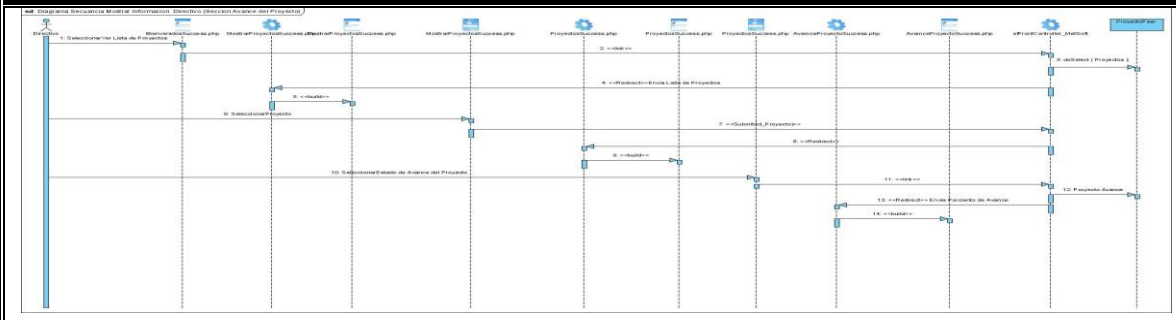

Figura # 88: Diagrama de Secuencia CU: Mostrar Información Directivo (Sección Avance Proyecto).

Diagrama de Secuencia CU: Mostrar Información Directivo (Sección Avance Módulos por semanas)

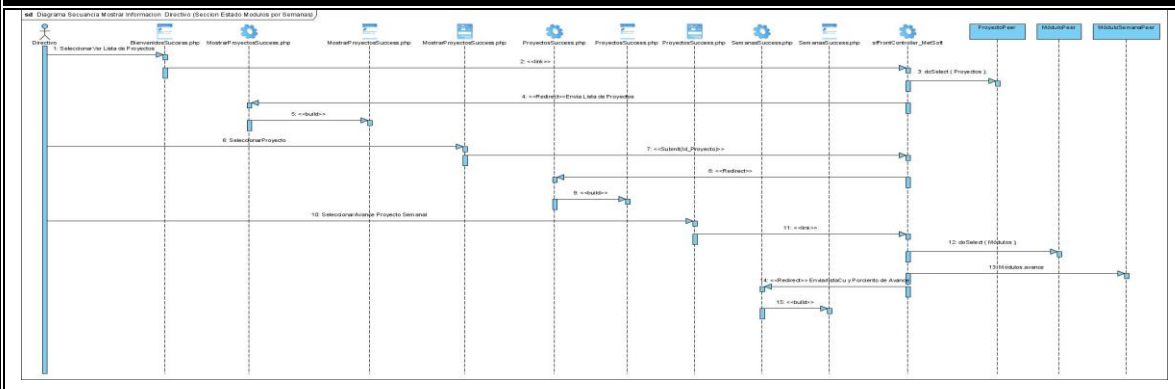

Figura # 89: Diagrama de Secuencia CU: Mostrar Información Directivo (Sección Avance Módulos por semanas).

Diagrama de Secuencia CU: Mostrar Información Directivo (Sección Avance Módulos)

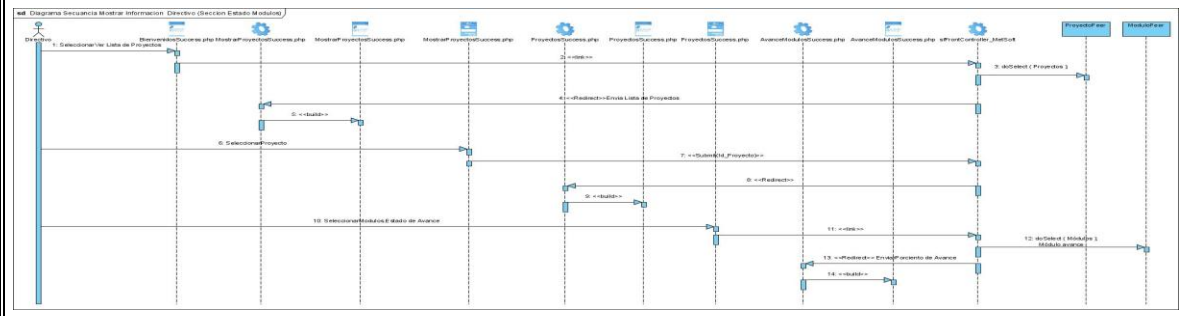

Figura # 90: Diagrama de Secuencia CU: Mostrar Información Directivo (Sección Avance Módulos).

Diagrama de Secuencia CU: Graficar Directivo

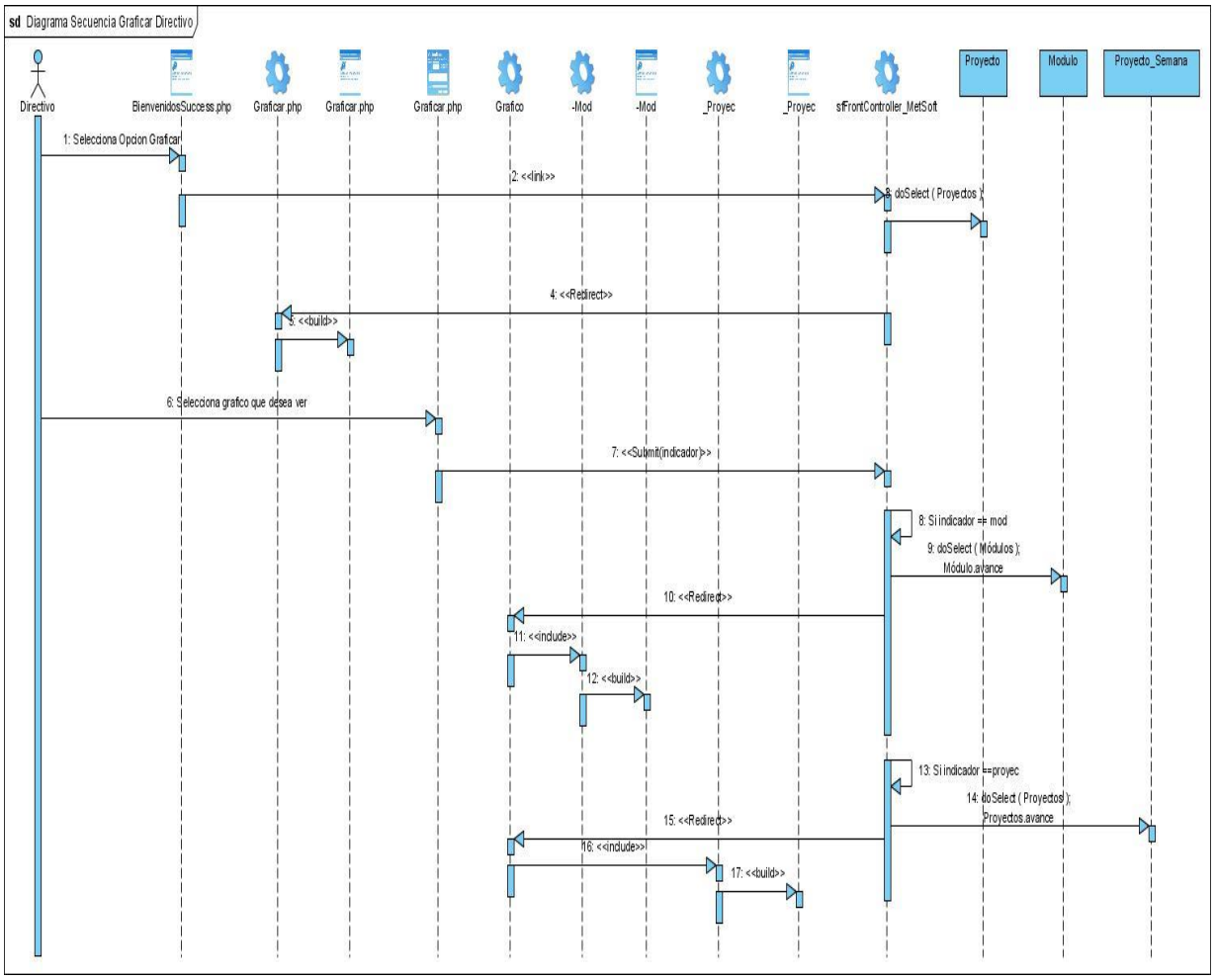

Figura # 91: Diagrama de Secuencia CU: Graficar Directivo.

Diagrama de Secuencia CU: Graficar Líder

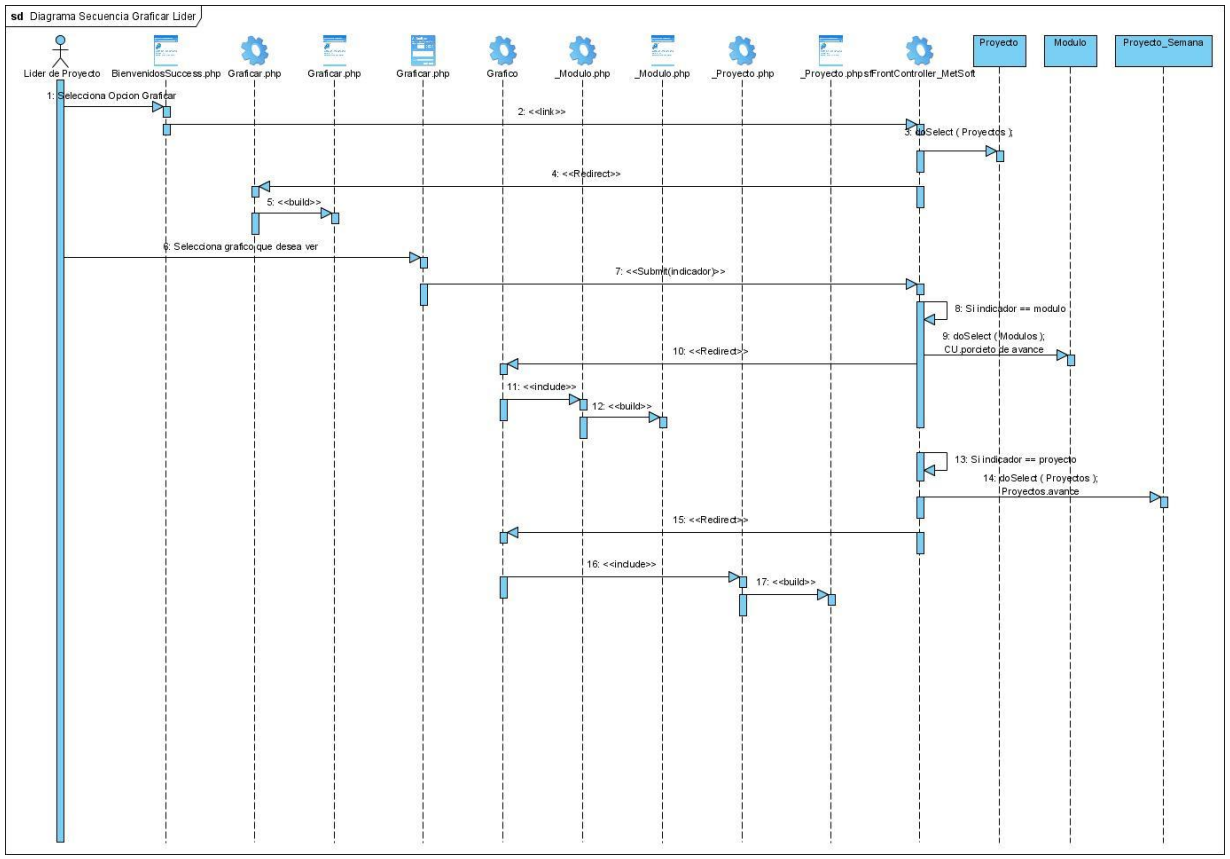

Figura # 92: Diagrama de Secuencia CU: Graficar Líder.