Universidad de las Ciencias Informáticas

Facultad 7

Trabajo de diploma para optar por el título de Ingenieros en Ciencias Informáticas

Sistema para la Planificación de Pruebas dentro del Grupo de Calidad en el Centro de Informática Médica (CESIM)

Autoras: Thelma Jefferson Betancourt

Dayami Pompa Hechavarría

Tutoras: Ing. Diana Rosa Alfonso Espinosa

Ing. Darling Darias Pérez

Ciudad de La Habana, Julio del 2010

"Año 52 de la Revolución"

Resumen

El presente trabajo surge como resultado de la necesidad de crear un sistema que facilite el proceso de planificación de pruebas que se lleva a cabo en el Grupo de Calidad del Centro de Informática Médica (por sus siglas CESIM). Este trabajo tiene como objetivo desarrollar una aplicación web que permita elevar la calidad del proceso de planificación de pruebas del Grupo de Calidad del CESIM. Para el logro de este se implementó una herramienta informática que automatice el proceso de Planificación de pruebas, diseñando una aplicación Web que sea capaz de planificar de acuerdo a los valores entrados por el planificador.

Para el desarrollo del mismo, se utiliza el Visual Paradigm es su versión 6.4 como herramienta de modelado y UML 2.0 como lenguaje. Para el diseño de las páginas se emplea el Dreamweaver, el cual soporta los CCS, HTML y JavaScript. Para el desarrollo con Python 2.6 y el Framework Django1.1 se usa el Wing IDE en su versión 3.2 y para la administración de la Base de Datos el PostgreSQL 8.3.

Esta aplicación permitirá interactuar a través de la red con la información almacenada en la aplicación. Permitirá gestionar la información de las planificaciones, así como exportarla en pdf, Para garantizar la portabilidad de la aplicación se desarrollará bajo ambientes multiplataforma, fundamentalmente con herramientas de software libre.

Índice

Índice

Introducción

A nivel mundial en toda organización, industria y empresa de software se realizan actividades que garanticen su desarrollo. La principal responsabilidad de la dirección de una empresa es la consecución de los fines para los que ha sido creada. La función directiva consiste en tomar aquellas decisiones que posibiliten que la misma consiga lograr sus fines y en orientar, igualmente, en la toma de decisiones a los diferentes componentes de la organización. Dentro de este proceso directivo toma un papel fundamental el proceso de planificación.

Este proceso es una técnica para minimizar la incertidumbre y dar más consistencia al desempeño de cualquier empresa desarrolladora de software. Es un proceso de toma de decisiones para alcanzar un futuro deseado, teniendo en cuenta los factores internos y externos que pueden influir en el logro de los objetivos. [1]

Muchos proyectos fracasan porque no se planifican correctamente, se sobreestiman los ingresos o no se tienen en cuenta los gastos que se van a generar. Toda planificación previa a la puesta en marcha de un proyecto es muy positiva; e incluso, como diría cualquier persona con experiencia en el desarrollo de nuevos productos o servicios, resulta del todo necesaria para una buena consecución de los objetivos.

A pesar de las variadas ventajas del proceso de planificación, existen limitaciones durante su ejecución. Entre las cuales se encuentran:

Una planificación irreal: es cuando los usuarios solicitan un sistema que tenga costo cero y los ingenieros no son capaces de enfrentar un plan porque no están entrenados para usar métodos de planificación y frecuentemente, las estimaciones no se basan en datos reales.

Una mala calidad del trabajo de la planificación: las prácticas pobres de ingeniería, la carencia de métricas adecuadas de calidad y las decisiones de los directivos guiadas por una planificación irreal; traen como consecuencia tiempos de pruebas impredecibles, productos con muchos defectos, demoras en la aceptación de los usuarios y una extensa garantía de servicio y reparaciones. Una pobre calidad afecta la planificación y la torna ineficiente.

Un personal inadecuado: en múltiples ocasiones el personal asignado a un proyecto se incorpora tarde, no cubre las necesidades en cuanto a cantidad y calidad y no se incorporan a

tiempo parcial al proyecto. Como consecuencia el trabajo se demora o descuida, es ineficiente y afecta el desempeño del equipo. Independientemente del plan, los proyectos deben comenzar en tiempo y con todo el personal.

Los cambios no controlados: es importante recordar que siempre ocurren cambios en los requerimientos, que los planes del proyecto se basan en el alcance del trabajo conocido, que los cambios siempre requieren más trabajo, sin planes detallados los equipos no pueden estimar el efecto o magnitud de los cambios y que si los equipos no controlan cada cambio, se pierde gradualmente el control del plan del proyecto.

La Universidad de las Ciencias Informáticas (UCI), la cual forma parte del crecimiento de la industria de software cubana, no está exenta de estas dificultades. Al surgir esta, aumenta en un gran por ciento la producción de software a nivel nacional. En medio de este desarrollo se observó que la cantidad de especialistas del departamento de Calidad Central no alcanzaba para revisar los proyectos de las diez facultades. Cuando este departamento comenzó a funcionar estaba compuesto por un grupo pequeño para realizar todas las actividades entre las que se encuentran: las revisiones, auditorías, pruebas, entre otras.

Debido a todo esto se llegó a la conclusión que era necesario la creación de laboratorios de calidad de software, en cada facultad para que se pudiera procesar en tiempo todas las solicitudes del cliente. Dentro de estos laboratorios se llevan a cabo una serie de actividades para asegurar la calidad de un producto, evidenciándose como la más importante el proceso de planificación de pruebas.

El proceso de Planificación de Pruebas realizado actualmente en el Grupo de Calidad del CESIM[1] se realiza manualmente, lo que conlleva a que dicha planificación resulte compleja debido al alto número de solicitudes de pruebas recibidas desde los departamentos productivos que atiende el mismo.

Existen una serie de riesgos que pueden atentar contra una buena planificación, estos pueden ser de varios tipos (Tecnológicos, Personales, Organización y Herramientas) dentro de estos están: los problemas de asistencia y puntualidad, la planificación de varios procesos de pruebas a la vez, falta de preparación en el rol a desempeñar, tareas no distribuidas equitativamente, fallos en la red, entre otros. Debido a esos riesgos esta planificación puede

[1] **CESIM**: **C**entro **de I**nformática **M**édica

tener una variación constante, por lo que requiere un esfuerzo extra y sostenido del planificador de pruebas del Grupo de Calidad del CESIM.

Además, existe la posibilidad de que se introduzcan errores por parte del planificador afectando con ello el cumplimiento de las pruebas que se le realizan a los diferentes proyectos, provocando una serie de dificultades como son: una planificación irreal, que los proyectos no sean liberados en tiempo, una distribución incorrecta de los recursos disponibles tanto tecnológicos como humanos, entre otras.

A partir de la situación problemática definida anteriormente el **problema a resolver** se centra en ¿Cómo mejorar el proceso de Planificación de Pruebas desarrollado en el grupo de Calidad del Centro de Informática Médica?

Se define como **objeto de estudio** el proceso de planificación de las actividades de calidad de software, centrando el **campo de acción** en el proceso de Planificación de Pruebas del grupo de Calidad del CESIM.

Por lo que se plantea como **Objetivo general** de la investigación**:** Desarrollar una aplicación que permita elevar la calidad del proceso de planificación de pruebas del Grupo de Calidad del CESIM.

Las **tareas investigativas** a desarrollar son las siguientes :

- Analizar el estado del arte de las herramientas para la planificación de pruebas para determinar las semejanzas y diferencias funcionales que presentan con respecto a la aplicación que se propone.
- Definir las variables necesarias para realizar una planificación individual a una solicitud de pruebas realizada.
- Describir la metodología, tecnología, herramientas, plataforma y lenguaje a utilizar para el desarrollo del sistema.
- Realizar el modelo del negocio.
- Realizar un levantamiento de requisitos para el sistema.
- Realizar el análisis y el diseño del sistema.
- Implementar las funcionalidades del sistema para darle solución a las necesidades actuales del cliente.
- Probar las funcionalidades del sistema para la planificación de pruebas con casos de Prueba lo más real posible.

Capítulo 1: Fundamentación teórica

En este capítulo se ofrece una breve panorámica de diferentes sistemas que se usan para planificar. Se describen los lenguajes de programación más adecuados, así como la metodología de desarrollo de software propuesta. Se pretende sentar las bases teóricas para un correcto análisis del proceso de planificación de pruebas. Finalizando con la selección de la metodología, tecnología, herramientas, plataforma y lenguaje a utilizar para el desarrollo del sistema.

1.1 Principales conceptos

Un **proceso** es un conjunto de actividades o eventos que se realizan o suceden (alternativa o simultáneamente) con un fin determinado. Este término tiene significados diferentes según la rama de la ciencia o la técnica en que se utilice. [2]

"La **planificación** es un proceso de toma de decisiones para alcanzar un futuro deseado, teniendo en cuenta la situación actual y los factores internos y externos que pueden influir en el logro de los objetivos". [3]

"Es una técnica para minimizar la incertidumbre y dar más consistencia al desempeño de la empresa". [4]

Además, es una forma concreta de la toma de decisiones que aborda el futuro específico que los gerentes quieren para sus organizaciones, Es un proceso continuo que refleja los cambios del ambiente en torno a cada organización y se adapta a ellos. Es el proceso de establecer metas y elegir medios para alcanzar dichas metas. [5]

Existen disímiles definiciones de planificación por diferentes autores, pero en general se evidencia una coordinación de pensamientos, exponiendo reiteradamente que la planificación es el establecimiento de una meta, por lo cual se trazan planes para lograrla. Idealizándose el futuro e intentando llevarlo a la realidad.

1.2 Proceso actual de planificación de pruebas

El proceso de planificación de pruebas que se lleva a cabo en el Grupo de Calidad del CESIM, se realiza de forma manual. El asesor de calidad atiende las solicitudes a medida que van llegando al proyecto, en dependencia de la fecha de llegada y de acuerdo con la prioridad que estas tengan, serán atendidas. Una vez analizada la solicitud, el asesor de calidad pide la

realización de la planificación de pruebas al planificador, este hace una estimación del tiempo que durará las iteraciones de pruebas que se le realizan a los productos. Las pruebas que se le realizan a los productos van a constar de tres iteraciones, las cuales poseen un tiempo de revisión, de respuesta y de verificación. Además, a esta planificación se le agrega un tiempo más, pues se pueden presentar riesgos que pueden atentar contra esta. Luego de haber realizado la tercera iteración, finalmente, el producto es liberado.

1.3 Estado del arte a nivel internacional

La planificación es un proceso muy estudiado en el mundo de la informática, existen diversas investigaciones que proponen soluciones a problemas de esta índole. A continuación se describirán algunas herramientas que se utilizan actualmente para la planificación.

1.3.1 Microsoft Project 2003

Microsoft Project estándar 2003 es una herramienta para la administración de proyectos, que se puede usar muy fácilmente para planificar, programar y controlar el proyecto. Esta herramienta, facilita la administración del proyecto, se puede tener un control en diferentes niveles de acuerdo con las necesidades del cliente. Permite generar varios tipos de diagramas, entre ellos Gantt y Pert, los cuales son muy fáciles de realizar con esta herramienta, mejora la capacidad de organización del trabajo y se obtiene una comunicación eficaz entre los equipos de desarrollo del software.

Posibilita que el líder del proyecto ofrezca una panorámica actual al cliente sobre la situación de cada uno de los equipos de trabajo y del software en general. En cuanto a seguridad, ofrece el uso de contraseñas si el proyecto lo necesita, para evitar que personal ajeno realice cambios en el mismo.

Quizás una de sus principales ventajas sea que en la UCI, el Microsoft Project está contenido dentro del programa de la asignatura Ingeniería de Software, lo que lo convierte en un software popular y con el cual está familiarizada la comunidad universitaria. Entre sus grandes inconvenientes se encuentra que es un software que produce la compañía norteamericana Microsoft, por ende, es un software con el cual no se le es permitido trabajar a los cubanos. Su costo asciende los 100 dólares y es incompatible con varios sistemas operativos, entre ellos el Linux. Puede ser muy engorroso para proyectos muy grandes (de más de 100 tareas) pues resulta casi imposible mostrar en pantalla el desarrollo completo de los mismos. [6]

1.3.2 GForge

GForge es un portal y a la vez una ramificación de la versión 2.61 del software de SourceForge. El enfoque que toma actualmente no es solo dirigido a la comunidad del Software Libre, también se considera un objetivo para las empresas de software y otras entidades que necesiten un entorno colaborativo para sus proyectos.

En realidad no es una herramienta de planificación, sino de Gestión de Proyectos, sin embargo, es posible adaptarla a la planificación, pues es posible especificar las fechas de inicio y finalización de las tareas, el número de horas para cada elemento, asignar el encargado de darle cumplimiento a la tarea de la forma persona-asignación o equipo-asignación. Gestiona dependencias entre tareas y las mismas se encuentran organizadas en sub-proyectos. Una tarea tiene como información asociada; porcentaje de completitud, prioridad, descripción breve y detallada, fecha de inicio y de finalización, asignación, dependencias y una estimación en horas, del tiempo que se empleará en llevar a cabo la misma.[7]

El inconveniente de esta herramienta es que no se adapta a las características que debe poseer el sistema que se quiere, en esta herramienta se le debe introducir una fecha de inicio y de fin de la actividad y lo que se busca es que el sistema luego de entrarle un juego de datos sea capaz de dar el tiempo que se demoraría en ejecutar las actividades.

1.3.3 OpenWorkbench

OpenWorkbench es una herramienta de planificación de proyectos, al estilo de Microsoft Project pero con la diferencia de que es de código abierto. Se ejecuta en entornos Windows y es completamente libre y gratuito. Soporta un amplio abanico de funciones tales como la asignación de tareas hacia delante y hacia atrás, análisis de hitos y tiempo estimado para la terminación de la solución.

Posee una habilidad única de programar tareas de acuerdo con restricciones de recursos y esta es una variedad nueva con respecto a otro software de planificación de proyectos. Esta característica ofrece un plan mucho más real y utiliza menos tiempo, pues calcula la duración de la tarea en esfuerzo por disponibilidad de recursos. Esta herramienta tiene un programa que te permite afinar calendario, tales como las fechas de comienzo y finalización. Permite además un sinfín de tareas, tales como diseño de gráficas de Gantt, asignación de tareas, relaciones, tiempos de actividad, entre otros.

Entre sus principales inconvenientes está que no funciona con Linux, por tanto, es imposible de usar para usuarios amantes de este y de otros Sistemas Operativos. Sin embargo, es una herramienta que al ser Opensource viabiliza el problema de licencias en Cuba que es un tema tan complicado. Adaptable al entorno de trabajo con tan solo modificar cualquiera de sus

componentes. [8] El inconveniente de esta herramienta es que no se adapta a las características que debe poseer el sistema.

1.3.4 Project KickStart 4

Project KickStart4 es una herramienta que posee nuevas características con respecto a sus versiones anteriores, entre ellas una interfaz gráfica mucho más fácil de usar incluso para aquellos usuarios que se inician en el mundo de la planificación de software y que aún no tienen gran experiencia en el uso de estas herramientas.

Entre sus inconvenientes se encuentra que es caro ($ 129.95), para usarse debe usar una computadora con tecnología Pentium, Sistema Operativo (SO) Windows 95, 98, Me, 2000, NT y XP y 32 MB RAM., sea, al remitirse solo a la familia Windows como SO lo deja inutilizable para usuarios que usen Linux, Macintosh u otros. Sus necesidades tecnológicas lo hacen imposible de usar para usuarios del tercer mundo u otros lugares subdesarrollados que no usen dicha tecnología. Además, teniendo en cuenta su precio y el no conocimiento de su código fuente, hace imposible realizarle modificaciones para familiarizarlo con el entorno de trabajo en que se desea emplear. [9]

1.3.5 B-kin Project Monitor.

B-kin Project Monitor es una herramienta básica para obtener visibilidad sobre el avance del proyecto y su impacto sobre costos y recursos. Genera automáticamente informes sobre los recursos, tanto humanos como materiales asignados. Con él se le puede dar seguimiento a los proyectos.

Este software puede utilizarse de forma online (ofrece a través de la Web una visión permanentemente actualizada de la situación de los procesos de negocio más dinámicos.) Una característica interesante de este software es que exporta datos a Microsoft Project y Microsoft Excel para adaptar informes a las necesidades del cliente, o para la integración con otras fuentes de datos. Garantiza la seguridad y confidencialidad de los clientes online.

Desafía uno de los retos más importantes de los líderes de proyectos de software y es la organización de equipos, donde las personas realizan diversas tareas, con distintos sistemas informáticos y a veces en ubicaciones geográficas diversas. Es por ello que se dice que el software online corrige las limitaciones técnicas de las herramientas de gestión de proyectos convencionales. Resuelve el problema de la aplicación corriendo en la estación de trabajo del planificador, evita el problema del mantenimiento, actualizaciones manuales, etc. No consume espacio en disco al ser una aplicación que corre en la Web, suponiendo que el planificador no

desee instalarla en su ordenador. Puede ser consultada desde cualquier estación de trabajo y en cualquier lugar del mundo. [10]

Entre sus inconvenientes se encuentra que es caro (19 EUR/USD). Es una herramienta para la gestión de proyecto en general es decir sus características no se adaptan a las que debe poseer un sistema para planificar las pruebas. Teniendo en cuenta su precio y que muchos usuarios no conocen de su código fuente, esto hace imposible que se le puedan realizar modificaciones para familiarizarlo con el entorno de trabajo en que se desea emplear.

1.3.6 DotProject

Dotproject es una completa herramienta que permite gestionar las distintas fases y tareas que componen un proyecto. A menudo, esta gestión implica un control en recursos humanos, materiales, que hacen que esta labor se torne compleja y prácticamente inabordable sin la ayuda de determinadas herramientas que den soporte a esta tarea de planificación y gestión de proyectos. Dotproject se perfila como una interesante herramienta para trabajar en entornos colaborativos, permitiendo a los integrantes del equipo trabajar compartiendo información relativa a los proyectos.[11]

Su principal inconveniente es que no se adapta a las características del sistema que se quiere realizar, porque esta herramienta no te permite generar una planificación del tiempo que se va a demorar una prueba.

1.3.7 RedMine

Es una herramienta de gestión de proyectos de software con interface web. Es un sistema multi-plataforma y soporta múltiples motores de bases de datos relacionales. Soporta a múltiples proyectos.

Hace seguimiento de los tiempos relacionados. Maneja archivos, documentos y noticias. Genera información en base a calendarios y diagramas Gantt. Permite el auto registro de usuarios. Permite la creación de tareas vía correo electrónico. [12]

1.4 Estado del arte a nivel nacional

A nivel nacional no existe ningún sistema que permita realizar una planificación de prueba, para esto en algunos sectores lo que utilizan es la plantilla Microsoft Excel.

En la Universidad de la Ciencias Informáticas:

En la Facultad 8 el proyecto Cuerpo de Investigaciones Científicas, Penales y Criminalísticas (CICPC), para la planificación de pruebas utilizan la herramienta Microsoft project.

En la Facultad 4 en el proyecto Sistema de Gestión Penitenciaria de Venezuela(SIGEP), para la planificación de pruebas utilizan la herramienta RedMine.

En el Grupo de Calidad del Centro de Informática Médica, para la planificación de pruebas utilizan la plantilla Microsoft Excel.

Microsoft Excel:

Excel es un programa que sirve para hacer hojas de cálculo y de eso se puede sacar mucho provecho, pues permite realizar cálculo,gráficar y expresar mucha información. La cual se puede utilizar para generar un cronograma de una planificación.

1.5 Valoración de estas herramientas

Luego de realizado un análisis sobre todas las herramientas existentes a nivel mundial y nacional se llega a la conclusión de que pese a las ventajas que las herramientas antes mencionadas, poseen, existen algunas limitaciones para su uso. Pues algunos son muy caros, otros no son compatibles con algunos sistemas operativos. Además todas estas aplicaciones antes mencionadas son en su totalidad, software propietarios o sea son herramientas donde la adquisición de sus licencias en porciento monetario se torna muy elevado, así como el soporte y mantenimiento de las mismas sería un proceso más costoso para el centro. También en la mayoría, su función principal es la de gestionar proyectos, equipos de trabajo, asignar tareas, entre otras y lo que se quiere es que se centre solamente en lo que es la planificación de pruebas.

Con estas herramientas no se puede realizar una planificación de pruebas como la que se quiere, porque ninguna de estas aplicaciones, planifican a partir de la entrada de un conjunto de variables. El sistema luego de entrados los valores internamente hará un cálculo y estimará la fecha de culminación de las pruebas. Todos estos aspectos han impulsado al análisis y diseño, así como a la realización de un prototipo funcional del producto.

1.6 Tendencias y tecnologías a utilizar

Teniendo en cuenta que la herramienta a desarrollar es un módulo de la plataforma a desarrollarse por parte del Grupo de Calidad del CESIM y para la cual se definió una arquitectura donde se determinaron las herramientas, metodologías y lenguaje de programación, la realización del módulo de Planificación de pruebas se ajusta a la definición realizada de forma tal que sea compatible en el momento de la integración tanto para la plataforma del proyecto como para la de la facultad las cuales serán creadas bajo los mismos principios de arquitectura, herramientas, metodologías y lenguaje de programación.

Metodología de desarrollo:

Proceso unificado de rational (RUP)

Todo desarrollo de software es riesgoso y difícil de controlar, pero si no se lleva una metodología de por medio, se obtienen clientes insatisfechos con el resultado y desarrolladores aún más insatisfechos.

El proceso unificado es una versión libre y abierta del proceso iterativo e incremental de ingeniería de software propuesto por Jacobson, Booch y Rumbaugh (los "tres amigos") en su libro "El proceso unificado de desarrollo de software", publicado por Addisson-Wesley en 1999. El lenguaje para especificar y modelar en el RUP es Lenguaje Unificado de Modelado (UML), por lo cual puede apoyarse en cualquier herramienta que soporte UML.

RUP proporciona al equipo de proyecto, procedimientos y herramientas que promueven las siguientes prácticas [13]

- Desarrollos iterativos.
- Uso de arquitectura basada en componentes.
- Verificación continua de la calidad del software.
- Gestión de cambios y requisitos.
- Implementa las mejores prácticas de desarrollo de software.

Fundamentación de su elección:

Como metodología de desarrollo se decide utilizar RUP porque es un proceso de ingeniería de software bien definido y estructurado, que provee un marco de proceso adaptable a las necesidades y características de cada proyecto específico. Tiene tres características fundamentales:

- Dirigido por casos de uso.
- Proceso centrado en la arquitectura.
- Iterativo e incremental.

Que sea iterativo e incremental, reduce los riesgos basado en la retroalimentación temprana .Las pruebas continuas e iterativas promueven una mejor evaluación del estado del proyecto. Se pueden administrar mejor los cambios.

1.6.1 Herramienta CASE:

Visual Paradigm

Es una herramienta de desarrollo que se integra al Entorno de Desarrollo Integrado (IDE, por sus siglas en inglés) de Eclipse. Diseñada para desarrollar software con Programación Orientada a Objetos (POO), busca reducir la duración del ciclo de desarrollo. Es un producto distinguido que facilita la organización de los diagramas, su misión es diseñar, integrar y desplegar las aplicaciones de la empresa y sus bases de datos subyacentes. La herramienta ayuda al equipo de desarrollo de software a agilizar el modelado del software, aumentando al máximo y acelerando el trabajo en equipo y las contribuciones individuales. [14]

Ventajas de Visual Paradigm [15]

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

Fundamentación de su elección:

Como herramienta de modelado visual se decide utilizar Visual Paradigm ya que para UML Enterprise representa la herramienta de modelado más poderosa y de mejor valor en el mercado actual. Combina las funcionalidades de todas las ediciones en una amplia plataforma de modelado visual. Está diseñada para brindar apoyo a arquitectos, desarrolladores, diseñadores UML, analistas de procesos de negocio y modeladores de datos con el fin de agilizar todo el proceso de desarrollo de código del modelo para aplicaciones empresariales complejas.

1.6.2 Lenguaje de modelado: Lenguaje Unificado de Modelado (UML) [16]

UML es el lenguaje de modelado más conocido y utilizado en la actualidad; aún cuando todavía no es un estándar oficial, está apoyado en gran manera por el Grupo de Administración de Objetos (OMG). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. Ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como: procesos de negocios, funciones del sistema y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

¿Qué es UML?

Es importante remarcar que UML es un "lenguaje" para especificar y no un método o un proceso, se utiliza para definir un sistema de software, para detallar los artefactos en el sistema, documentar y construir.

1.6.3 Framework :

Django

Es un framework de desarrollo para la web de código abierto escrito en Python, y cumple con el paradigma del Modelo Vista Controlador.

Inicialmente Django fue desarrollado para gestionar aplicaciones web de páginas orientadas a noticias de World Online. Django se centra en automatizar todo lo posible y se adhiere al principio DRY (Don't Repeat Yourself). [17]

Facilidades que ofrece Django [18]

- Mapeador objeto-relacional.
- Define modelos de datos completamente en Python y accede a los datos mediante una potente API de acceso dinámico a la base de datos.
- Interfaz de administración automático.
- Ahorra los tediosos trabajos de crear interfaces para añadir o actualizar contenidos. Django genera un sistema de administración automáticamente listo para ser usado en producción.
- Diseño de URL elegantes.

- Django permite diseñar bellas URL sin limitaciones específicas del framework.
- Sistema de plantillas.
- Posee un potente y extensible lenguaje de plantillas para separar diseño, contenido y código Python de una forma amigable para diseñadores.
- Sistema de cache.
- Utiliza el memcached u otro framework de cache para obtener un súper rendimiento. El cacheo es tan granular como se desee.
- Internacionalización.

Django soporta aplicaciones multilenguaje permitiendo especificar string de traducción y proporcionando herramientas para funcionalidad específica del lenguaje.

Fundamentación de su elección:

Se decide utilizar este Framework porque es el que está definido en la arquitectura de la facultad, es el más recomendable para trabajar con Python.

1.6.5 Lenguajes de programación:

Python [19]

Este lenguaje se puede ejecutar sobre Windows, Linux/Unix, Mac OS X y ha sido portado a Java y .Net máquinas virtuales. Es libre de usar, incluso para productos comerciales, debido a su licencia OSI de código abierto.

Entre las principales bondades que brinda este lenguaje de programación se encuentra el ser un lenguaje multiparadigma lo que se puede interpretar como que el mismo permite optar por varios estilos ya sea programación orientada a objetos, estructurada o funcional.

Características del lenguaje Python:

- Multiplataforma

Hay versiones disponibles de Python en muchos sistemas informáticos distintos. Originalmente se desarrolló para Unix, aunque cualquier sistema es compatible con el lenguaje siempre y cuando exista un intérprete programado para él.

- Interpretado

Quiere decir que no se debe compilar el código antes de su ejecución. En realidad sí que se realiza una compilación, pero esta se realiza de manera transparente para el programador. En ciertos casos, cuando se ejecuta por primera vez un código, se producen unos bytecodes que

se guardan en el sistema y que sirven para acelerar la compilación implícita que realiza el intérprete cada vez que se ejecuta el mismo código.

- Interactivo

Python dispone de un intérprete por línea de comandos en el que se pueden introducir sentencias. Cada sentencia se ejecuta y produce un resultado visible, que puede ayudarnos a entender mejor el lenguaje y probar los resultados de la ejecución de porciones de código rápidamente.

- Orientada a Objetos

La programación orientada a objetos está soportada en Python y ofrece en muchos casos una manera sencilla de crear programas con componentes reutilizables.

- Funciones y Librerías

Dispone de muchas funciones incorporadas en el propio lenguaje, para el tratamiento de string, números, archivos, etc. Además, existen muchas librerías que podemos importar en los programas para tratar temas específicos como la programación de ventanas o sistemas en red o cosas tan interesantes como crear archivos comprimidos.

- Sintaxis clara

Destacar que Python tiene una sintaxis muy visual. En muchos lenguajes, para separar porciones de código, se utilizan elementos como las llaves o las palabras clave begin y end. Para separar las porciones de código en Python se debe tabular hacia dentro, colocando un margen al código que iría dentro de una función o un bucle. Esto ayuda a que todos los programadores adopten las mismas notaciones y que los programas de cualquier persona tengan un aspecto muy similar.

Ventajas de Python

Entre las características que distinguen a este lenguaje de programación podemos encontrar varias de sus principales ventajas entre las que podemos citar:

- Sintaxis simple, clara y sencilla.
- Tipiado dinámico.
- Gran cantidad de librerías disponibles.
- Lenguaje de alta potencia.

Fundamentación de su elección:

Para realizar la implementación del sistema propuesto se utilizará el lenguaje Python porque es el lenguaje que está definido en la arquitectura de la facultad. A pesar de ser un lenguaje bastante complejo posee grandes ventajas que facilitan su uso como es el caso de ser gratuito y posee una enorme eficiencia en la implementación de un software, puede integrarse de forma sencilla a múltiples bases de datos y tiene gran número de funciones predefinidas lo que les facilita el trabajo a los programadores que necesiten hacer uso de él. Este puede ser ejecutado sobre Windows, Linux, Mac Os X.

1.6.6 Entorno de desarrollo. [20]

Wing IDE Professional es un potente entorno de desarrollo integrado (IDE) para el trabajo con el lenguaje de programación Python. Wing IDE se centra en aumentar la productividad y la calidad del código, especialmente en proyectos complejos con cambios constantes en los requerimientos.

Wing IDE permite el desarrollo de plataformas de escritorios y de aplicaciones Web, un manejo fácil en la integración de aplicaciones y un manejo sencillo de las pruebas de software.

1.6.7 Sistema Gestores de base de datos:

PostgreSQL. [21]

PostgreSQL es un potente motor de bases de datos, que tiene prestaciones y funcionalidades equivalentes a muchos gestores de bases de datos comerciales. Está considerado como uno de los gestores de bases de datos de código abierto más avanzados del mundo, con más de 15 años de desarrollo activo y una arquitectura probada que le ha dado una popularidad en la fiabilidad, integridad de los datos y exactitud. Además, corre en plataformas como Linux, Windows, Solaris, Mac OS X entre otras muchas.

Características de PostgreSQL

- DBMS por sus siglas en English (Database Management System) 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 las consultas SQL declarativas, control de concurrencia multi-versión, soporte multi-usuario, transacciones, optimización de consultas, herencia y arreglos.
- Altamente Extensible:
  PostgreSQL soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario.

- Soporte SQL Comprensivo:
  PostgreSQL soporta la especificación SQL99 e incluye características avanzadas, tales como las uniones (joins) SQL92.
- Integridad Referencial:
  PostgreSQL soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos.
- 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, Java/JDBC, Ruby TCL, C/C++ y Pike.
- Control de Concurrencia Multi-Versión:
  Es la tecnología que PostgreSQL usa para evitar bloqueos innecesarios. El PostgreSQL mantiene una ruta a todas las transacciones realizadas por los usuarios de la base de datos y es capaz de manejar los registros sin necesidad de que los usuarios tengan que esperar a que los registros estén disponibles.

Fundamentación de su elección:

Se decide utilizar PostgreSQL como servidor gestor de bases de datos porque es un sistema de gestión de bases de datos objeto-relacional cuyo código fuente es libremente disponible. Es el sistema de gestión de bases de datos de código abierto más potente del mercado además de utilizar un modelo cliente/servidor usando multiprocesos en vez de multihilos para garantizar la estabilidad del sistema.

Conclusiones

En este capítulo se abordaron varios temas que constituyen el fundamento teórico y tecnológico para la realización del módulo para la planificación de pruebas. De manera general el capítulo, se enfoca en descripción de las tendencias y tecnologías que se tendrán en cuenta para la construcción del prototipo funcional. Por lo que se decide utilizar como lenguaje de programación Python en su versión 2.6, el framework sería el Django con la versión 1.1, como entorno de desarrollo el Wing IDE en su versión 3.2, el lenguaje de modelado sería UML, la herramienta CASE Visual Paradigm en su versión 6.4 y como gestor de base de datos PosgreSQL en su versión 8.3.

Capítulo 2: Características del sistema

En el Capítulo se abordan las características del sistema, el objeto de estudio, la situación problemática, objeto de informatización y la propuesta del sistema. Se definen los requisitos funcionales y no funcionales. Se realiza el modelado del sistema y se describen los casos de uso.

2.1. Objeto de estudio

2.1.1. Situación Problemática

El proceso de Planificación de Pruebas realizado actualmente en el Grupo de Calidad del CESIM[2] se realiza manualmente, lo que conlleva a que dicha planificación resulte compleja debido al alto número de solicitudes de pruebas recibidas desde los departamentos productivos que atiende el mismo.

Existen una serie de riesgos que pueden atentar contra una buena planificación, estos pueden ser de varios tipos (Tecnológicos, Personales, Organización y Herramientas) dentro de estos están: los problemas de asistencia y puntualidad, la planificación de varios procesos de pruebas a la vez, falta de preparación en el rol a desempeñar, tareas no distribuidas equitativamente, fallos en la red, entre otros. Debido a esos riesgos esta planificación puede tener una variación constante, por lo que requiere un esfuerzo extra y sostenido del planificador de pruebas del Grupo de Calidad del CESIM.

Existe además la posibilidad de que se introduzcan errores por parte del planificador afectando con ello el cumplimiento de las pruebas que se le realizan a los diferentes proyectos, provocando una serie de dificultades como son: una planificación irreal, que los proyectos no sean liberados en tiempo, una distribución incorrecta de los recursos disponibles tanto tecnológicos como humanos, entre otras.

2.1.2. Objeto de automatización

La automatización del proceso de planificación de pruebas, comienza cuando se pasa a planificar las pruebas, donde se inserta una serie de variables como son (cantidad de casos de uso por complejidad que puede ser (baja, mediana, alta), tipo de pruebas, nombre del proyecto, nombre del módulo, tiempo de respuesta del equipo de desarrollo, cantidad de probadores, cantidad de máquinas, tiempo del profesor, iteración, fecha de inicio, tiempo de verificación).Ya insertadas estas variables, se realiza la planificación donde se generaría el cronograma de la misma dando la fecha de culminación de las pruebas que se le realicen a un módulo

[2] **CESIM**: **C**entro **de I**nformáticas **M**édicas

determinado. Además, se puede realizar una búsqueda de las planificaciones ya insertadas, esta aplicación permite ver toda la información de la planificación seleccionada, eliminarla así como modificarla. La aplicación también te muestra una opción donde puedes ver mediante un diagrama de Gantt la duración de una prueba a un módulo determinado.

Para que el sistema calcule la fecha de culminación de las pruebas se utilizan las siguientes variables:

TPP-Tiempo que se necesita para probar en minutos

CCA-Cantidad de casos de Uso de Complejidad Alta

CCM-Cantidad de casos de Uso de Complejidad Media

CCB-Cantidad de casos de Uso de Complejidad Baja

CP-Cantidad de Probadores

TP-Tiempo del Profesor

TR-Tiempo de respuesta

TV- Tiempo de verificación

TPH-Tiempo de Pruebas en Horas

TPST-Tiempo de Pruebas pos Secciones de Trabajo

TFP- Fecha Final de Pruebas

ST- Secciones de trabajo (2)

Fórmulas para realizar el cálculo.

TPP= (CCA*180min+CCM*120min+CCB*90min/CP)

TPH=TPP/60min

TPST=TPH/4h (Se divide entre 4 horas por que es el tiempo de una sección de trabajo)

TFP=TPST/2+TP+TR+TV +1(Se divide entre 2 por que se realizan al día dos secciones de trabajo)

2.1.3. Propuesta del sistema

La herramienta que se propone desarrollar es un módulo que automatizará el proceso de planificación de pruebas, el cual se lleva a cabo en el Grupo de Calidad del CESIM.

El módulo de planificación:

El principal objetivo de este módulo es la planificación, para poder realizar una planificación de pruebas se debe insertar los datos de un proyecto determinado. Insertar la complejidad de los casos de uso, que pueden ser (simple, mediana, alta), en qué iteración se encuentran, la fecha en que se le van a realizar las pruebas, entre otras variables. Este módulo, además de realizar las planificaciones permite gestionar las mismas, además de brindar la opción de exportar la planificación en un documento en formato PDF. El módulo brinda la posibilidad de poder consultar las planificaciones realizadas mediante un diagrama de Gantt.

2.2. Modelo del negocio

2.2.1. Descripción del negocio

El modelo de negocio que se describe responde a un conjunto de actividades realizadas dentro del proceso de planificación de pruebas que se desarrolla en el Grupo de Calidad del CESIM en la UCI. Inicialmente el solicitante (Líder de proyecto, Jefe de módulo o a una persona responsable de realizar la solicitud, realiza una solicitud de revisión de sus productos al Grupo de Calidad. Posteriormente se revisa la solicitud para ver si cumple con los requisitos que exige el Grupo de Calidad, o de tener planificado pruebas a otro producto, se le informa al proyecto que no se puede comenzar la planificación de las pruebas. En caso contrario se decide aceptar el proceso de pruebas realizando previamente la planificación de ellas.

2.2.2. Actores y trabajadores del negocio

Actores del negocio	Justificación
Solicitante	Jefe de proyecto (jefe de módulo o el encargado de realizar la solicitud) realiza la solicitud de pruebas de sus productos de software al Grupo de Calidad.

Tabla 2.1 Actores del negocio.

Trabajadores del negocio	Justificación
Asesor de calidad	Persona que lleva el control de todo el proceso de pruebas y liberación de un producto de software. También lleva el control de las solicitudes de pruebas que se le realizan al Grupo de Calidad del CESIM.
Planificador	Persona que realiza la planificación de pruebas.

Tabla 2.2 Trabajadores del negocio.

2.2.3 Diagramas del modelo del negocio

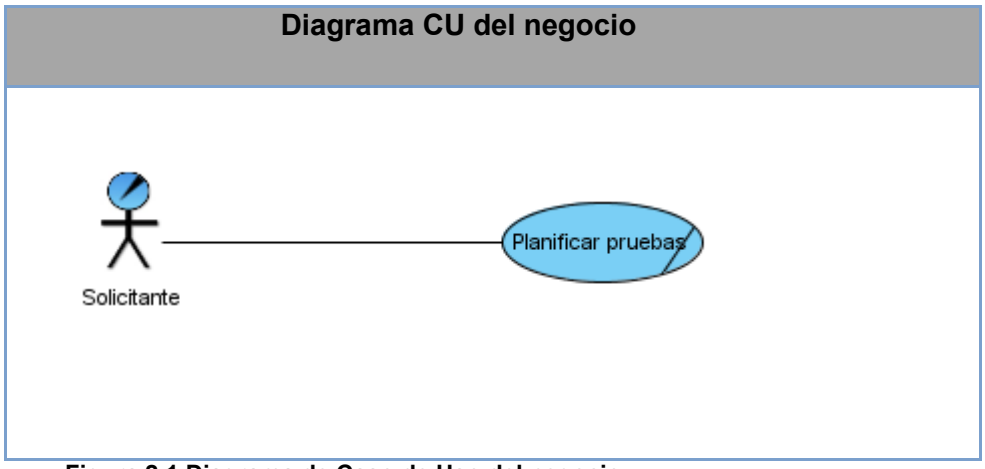


Figura 2.1 Diagrama de Caso de Uso del negocio.

2.2.4. Descripción textual

Nombre del Caso de Uso	**Planificar pruebas**
Actores	Solicitante (inicia).
Propósito	El objetivo de este caso de uso es obtener una planificación de pruebas que corresponda a las necesidades existentes y que esta sea lo más exacta posible.
Resumen	El caso de uso se inicia cuando el solicitante hace una solicitud de pruebas de liberación de sus productos de software. El asesor de calidad aprueba o no la solicitud, de ser aprobada le pide al planificador que realice la planificación de pruebas.
Curso Normal de los eventos	
Acciones del Actor	**Respuesta del proceso de negocio**
1. El solicitante realiza la solicitud de pruebas	1.1 El asesor de calidad revisa la solicitud

para sus productos de software.	de pruebas. 1.2 El asesor de calidad decide el proceso de pruebas si cumple con las condiciones de la solicitud de pruebas.(ver flujo alterno 1.1) 1.3 El asesor de calidad recepciona la solicitud de pruebas. 1.4 El asesor de calidad pide al planificador realizar planificación de las pruebas al producto de software.
Curso alternativo de los eventos.	
Acción 1.1	El asesor de calidad no acepta la solicitud y notifica al líder que no cumple con los requisitos.
Prioridad	Alta

Tabla 2.3 Descripción del caso de uso del negocio "Planificar pruebas"

2.2.5. Diagrama de actividades

Diagrama de actividades del CU "Planificar pruebas"

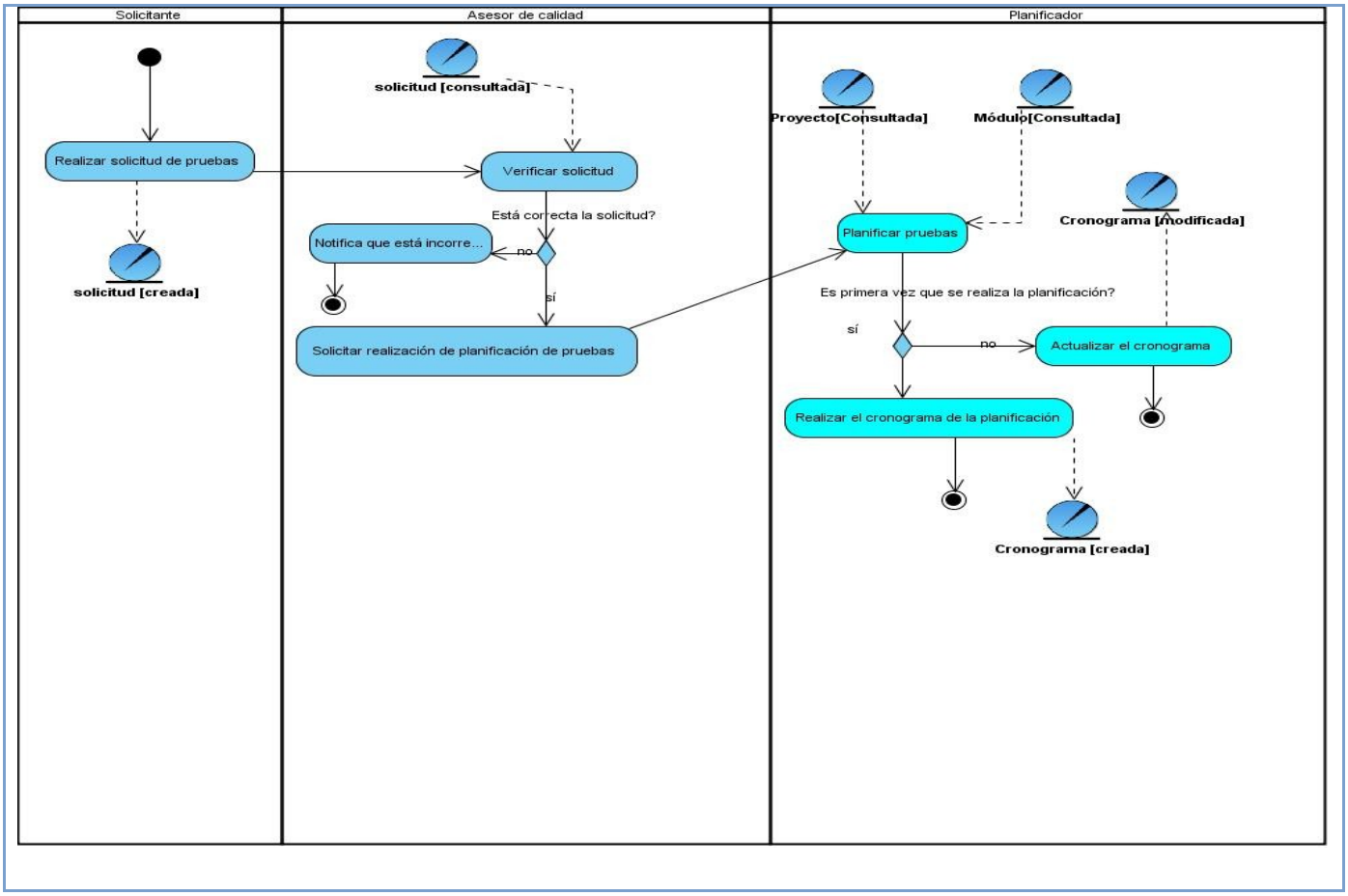


Figura 2.2 Diagrama de actividades."Planificar pruebas"

2.2.6. Modelo de objeto

Modelo de objeto

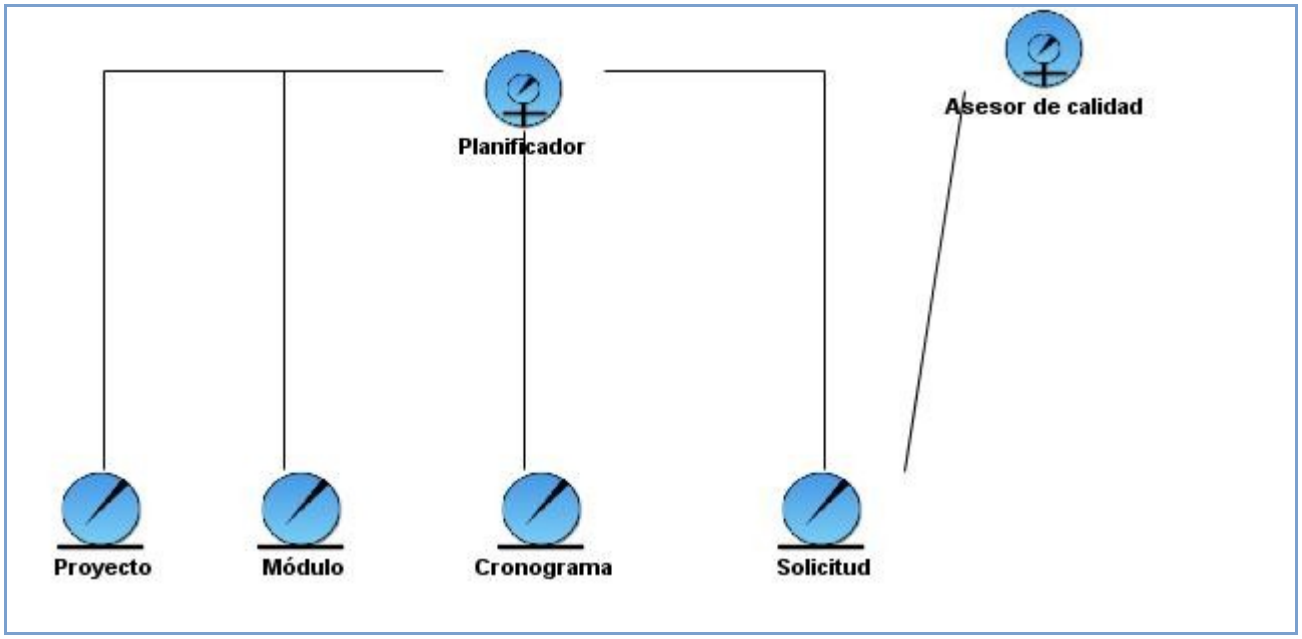


Figura 2.3 Modelo de objeto.

2.2.7 Reglas del negocio

La regla a tener en cuenta es:

1. Para realizar una planificación de pruebas se debe de aceptar la solicitud de pruebas.

2.3 Especificación de los requisitos de software

Este flujo de trabajo es uno de los más importantes, porque en él se establece qué debe hacer exactamente el sistema que se construya. En esta línea los requisitos son el contrato que se debe cumplir y firmar, de modo que los usuarios finales tienen que comprender y aceptar los requisitos que se especifiquen.

Los requisitos pueden clasificarse, por lo general se dividen en dos grupos: requisitos funcionales (capacidades o condiciones que el sistema debe cumplir) y requisitos no funcionales (son propiedades o cualidades que el producto debe tener).

2.3.1 Requisitos funcionales

El sistema será capaz de:

RF- 1 Realizar Planificación de pruebas.

RF-2 Gestionar planificación.

- RF-2.1 Buscar planificación.

- RF-2.2 Ver planificación.
- RF-2.3 Modificar planificación.
- RF-2.4 Eliminar planificación.
- RF-2.5 Exportar planificación.
- RF-2.6 Próxima iteración

RF-3 Mostrar planificación.

RF-4 Registrar iteración.

2.3.2 Requisitos no funcionales

Requerimiento de apariencia o interfaz externa

El sistema debe tener una interfaz sencilla con una apariencia amigable con facilidad de navegación para el usuario.

Requerimiento de usabilidad

El sistema debe ser diseñado de forma tal, que los usuarios que harán uso del mismo obtengan los conocimientos necesarios en el menor tiempo posible para la explotación de sus funcionalidades.

Requerimiento de rendimiento

Debido a que se trata de una aplicación cliente/servidor debe ser eficiente, con capacidad adecuada de procesamiento y cálculo, así como requiere de un tiempo de respuesta relativamente pequeño.

Requerimiento de soporte

El sistema tendrá un diseño que permita la realización de una planificación de pruebas, permitiendo la explotación de la aplicación de forma eficiente. Se incluye la posibilidad de permitir la solución de problemas en tiempo real de ejecución que garantice la solución de fallas que pueda presentar. Debe ser garantizado su compatibilidad y adaptabilidad con los distintos Sistemas Operativos.

Requerimiento de portabilidad

El sistema podrá ser usado bajo los sistemas operativos Windows y Linux.

Requerimiento de seguridad

El sistema debe comunicarse usando un protocolo seguro. Chequear si el usuario que está accediendo al sistema está autenticado y brindarle servicio de autenticación. Mostrar las operaciones de acuerdo con el rol del usuario y no más. Permitir que cuando se borre cualquier información pueda existir una acción de advertencia antes de realizar la acción.

Requerimientos de software

Para el desarrollo de la herramienta se utilizará una computadora Pentium 4 con 3.0 GHz de CPU y 504 MB de RAM, como sistema operativo Windows XP con Service Pack 3 y el sistema operativo Linux.

Requerimientos de hardware

Para el desarrollo y ejecución de esta aplicación se necesita conexión a la red local, por lo que requiere de tarjeta de red.

2.4 Modelo del sistema

El modelo del sistema representa la funcionalidad completa de un sistema mostrando su interacción con los agentes externos. Esta representación se hace a través de las relaciones entre los actores (agentes externos) y los casos de uso (acciones) dentro del sistema. Los diagramas de casos de uso definen conjuntos de funcionalidades afines que el sistema debe cumplir para satisfacer todos los requisitos que tiene a su cargo. Esos conjuntos de funcionalidades están representadas por los diferentes diagramas del sistema.

Descripción de los actores del sistema

Actores del sistema	Justificación
Asesor de calidad	Persona que lleva el control de todo el proceso de pruebas y liberación de un producto de software .También lleva el control de las solicitudes de pruebas que se le realizan al Grupo de Calidad del CESIM.
Planificador	Persona que realiza la planificación de pruebas.

Tabla 2.4 Descripción de los actores del sistema.

2.4.1 Diagrama de caso de uso del sistema

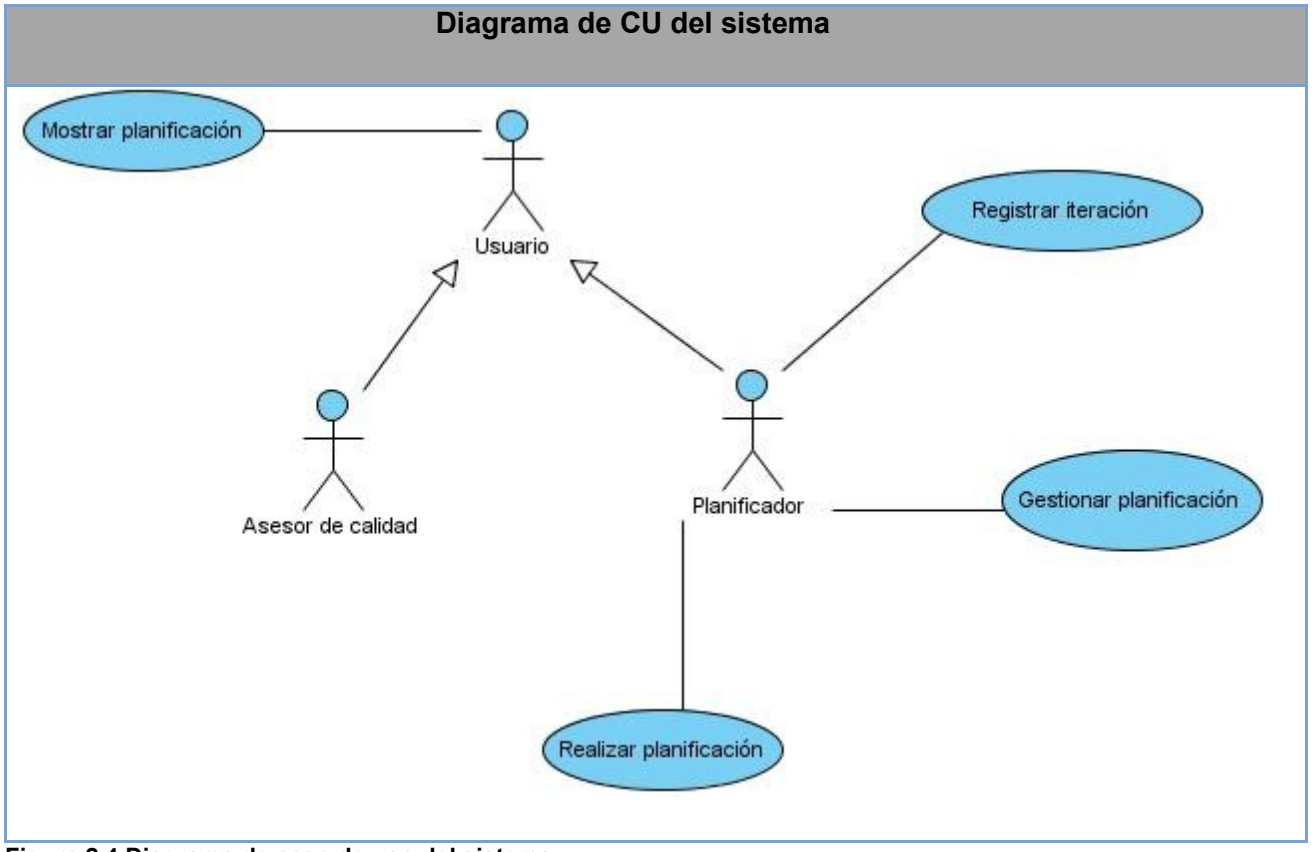


Figura 2.4 Diagrama de caso de uso del sistema.

2.5 Descripción de casos de uso

Caso de uso 1: Realizar planificación de pruebas.	
Propósito	Realizar una planificación de prueba y estimar el tiempo de duración.
Actor: Planificador.	
Resumen: El planificador realiza la planificación teniendo en cuenta los recursos disponibles y la complejidad de los casos de pruebas del módulo, una vez terminada la planificación se genera automáticamente en otro lugar un cronograma de la misma.	
Precondiciones	
Que se encuentren insertados los proyectos y módulos a planificar.	
Acción del actor	**Respuesta del sistema**
1.El planificador selecciona la opción	2 El sistema muestra la interfaz con los siguientes

"Realizar planificación"	datos a introducir : • Departamento • Proyecto • Módulo • Nombre de prueba • Descripción • Cantidad de casos de uso simple. • Cantidad de casos de uso medio. • Cantidad de casos de uso alto. • Cantidad de máquinas. • Cantidad de personas. • Fecha de inicio. • Iteración. • Tiempo de respuesta. • Tiempo de profesor. • Tiempo de verificación.
3. El planificador introduce los datos en el sistema y presiona el botón Registrar.	4. El sistema verifica que todos los campos obligatorios estén llenos. 5. El sistema brinda la posibilidad de introducir un departamento, un proyecto, un módulo y una iteración desde esta vista. 6. En caso de quedar campos vacios ver flujo alterno 6.1 7. El sistema almacena los datos de la planificación y

	muestra un mensaje de confirmación. “ Se ha registrado satisfactoriamente”
Flujo alterno Sección ”Realizar planificación”	
	6.1 En caso de dejar campos vacíos el sistema muestra un mensaje : “Hay errores en el formulario”
7. El planificador llena los campos.	8. El sistema almacena los datos. Muestra un mensaje “Se ha registrado satisfactoriamente”

Tabla 2.5 Descripción del caso de uso “Realizar planificación de pruebas”

Caso de uso 2: Gestionar planificación.	
Propósito	Permitir realizar un conjunto de operaciones sobre una planificación, tales como: buscarla, verla, eliminarla y modificarla.
Actores: Planificador.	
Resumen: Una vez realizada la planificación se puede buscar, permitiendo seleccionarla para modificarla, verla o eliminarla. Este da la opción de exportar todas las planificaciones hechas hasta el momento.	
Precondiciones	
Que existan planificaciones. Se gestionaron las planificaciones	
Referencias	RF-2.1, RF-2.2, RF-2.3, RF-2.4, RF-2.5, RF-2.6
Postcondiciones	Quedan actualizadas las planificaciones.

Sección “Buscar Planificación”	
Acción del actor	**Respuesta del sistema**
1. El planificador selecciona la opción “Buscar planificación”.	2. El sistema muestra para realizar la búsqueda por los siguientes criterio:

3. El planificador introduce los criterios de búsqueda.	• Nombre del módulo. • Nombre del proyecto. 4. El sistema muestra el resultado de la búsqueda. En caso de no existir planificaciones con esos criterios ver flujo alterno 4.1.
5. Selecciona una planificación para realizar operaciones sobre ella. • Modificar planificación • Eliminar planificación • Ver planificación • Exportar en formato pdf • Próxima iteración	6. El sistema muestra una interfaz en dependencia de la opción que seleccione el planificador. De seleccionar "Modificar Planificación "ver esta sección. De seleccionar "Eliminar Planificación "ver esta sección. De seleccionar "Ver Planificación "ver esta sección. De seleccionar "Exportar en formato pdf' ver esta sección. De seleccionar "Próxima iteración' ver esta sección.
Flujo alterno Sección "Buscar planificación"	
	4.1 El sistema muestra un mensaje diciendo "No existe la planificación solicitada"
Sección "Modificar Planificación"	
Acción del actor	**Respuesta del sistema**

1. El planificador escoge la opción modificar planificación.	2.El sistema muestra una interfaz para modificar la planificación donde aparecen los siguientes criterios • Departamento • Proyecto • Módulo • Nombre de prueba • Descripción • Cantidad de casos de uso simple. • Cantidad de casos de uso medio. • Cantidad de casos de uso alto. • Cantidad de máquinas. • Cantidad de personas. • Fecha de inicio. • Iteración. • Tiempo de respuesta. • Tiempo de profesor. • Tiempo de verificación. • Dependencia
4. El planificador introduce los nuevos datos.	5. El sistema modifica la planificación. Este muestra un mensaje diciendo “Se ha modificado satisfactoriamente”.
Sección “Eliminar Planificación”	

1. El planificador selecciona la opción “Eliminar planificación”.	2. El sistema muestra un mensaje de confirmación. ¿Está seguro de eliminar el elemento?
3. Si desea eliminar la planificación presiona el botón “Sí”. En caso de no querer eliminar el elemento ver flujo alterno 3.1.	4. El sistema elimina la planificación.
Flujo alterno Sección ”Eliminar planificación”	
3.1 Si el planificador no desea eliminar la planificación presiona el botón “No”.	3.2. El sistema se queda en la interfaz de listar planificaciones.
Sección “Ver Planificación”	
1.El planificador selecciona la opción “Ver planificación” 3. El planificador , para volver a la vista anterior, presiona el botón “Volver”	2. El sistema muestra la planificación con todos los datos de la misma. 4. El sistema muestra la vista anterior.
Sección “Exportar en formato pdf”	
1. El planificador selecciona la opción “Exportar en formato pdf”	2. El sistema muestra un documento en formato pdf con todas las planificaciones realizadas hasta el momento. Este permite guardarlo en cualquier parte de la computadora.
Sección “Próxima iteración”	

1. El planificador selecciona la opción “Próxima iteración”	2.El sistema muestra una interfaz para realizar la próxima iteración de la planificación seleccionada donde aparecen los siguientes criterios • Departamento • Proyecto • Módulo • Nombre de prueba • Descripción • Cantidad de casos de uso simple. • Cantidad de casos de uso medio. • Cantidad de casos de uso alto. • Cantidad de máquinas. • Cantidad de personas. • Fecha de inicio. • Iteración. • Tiempo de respuesta. • Tiempo de profesor. • Tiempo de verificación. • Dependencia
3. El planificador pondría en el campo de la iteración, la iteración que van a realizar y si cambiaron algunos de los recursos disponibles, los cambiarían por los actuales que tienen en ese momento.	4. El sistema almacena los datos.

Tabla 2.6 Descripción del caso de uso “Gestionar planificación”

Caso de uso 3: Mostrar planificación	
Propósito	Los Usuario (planificador o el asesor de calidad) podrán ver las planificaciones hechas hasta el momento.
Actores: Usuario	
Resumen: Los Usuarios podrán ver las planificaciones de pruebas hechas a los diferentes proyectos.	
Precondiciones	
Que se encuentren Planificaciones de Pruebas.	
Acción del actor	**Respuesta del sistema**
1. El planificador selecciona la opción “Mostrar planificación”.	2. El sistema muestra un diagrama de Gantt con todas las planificaciones hechas hasta el momento ordenadas por la fecha según hallan sido realizadas. En caso de no existir planificaciones, ver flujo alterno 2.1.
Flujo alterno Sección ”Mostrar planificación”	
	2.1 El sistema muestra un mensaje de información “No existen planificaciones, debe registrarla” y este te lleva a la interfaz de “Realizar planificación”.

Tabla 2.7 Descripción del caso de uso “Mostrar planificación de prueba”

Conclusiones.

El desarrollo de este capítulo propició un mayor entendimiento de la propuesta realizada, teniendo en cuenta que se describieron las características del sistema en términos de requerimientos funcionales y no funcionales. Se logró identificar los actores y los casos de uso del sistema, los cuales fueron modelados gráficamente mediante el Diagrama de Casos de Uso del Sistema, así como la descripción de los mismos. Este capítulo sentó las bases para comenzar el flujo de Análisis y Diseño del módulo a desarrollar.

Capítulo 3: Análisis y Diseño del sistema

En este capítulo se analizan los casos de usos del sistema para diseñar las clases que se implementarán, se representan los diagramas de clases del diseño así como las descripciones de las clases.

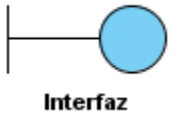
3.1 Análisis

Durante el análisis, se analizan los requisitos que se describieron en el levantamiento de requerimientos realizado anteriormente para refinarlos y estructurarlos. Con el objetivo de conseguir una compresión más precisa de ellos y una descripción que sea fácil de mantener y que ayude a estructurar el sistema entero, incluyendo su arquitectura. Es importante hacer notar que el análisis hace abstracciones, evita resolver algunos problemas y trata algunos requisitos que serán pospuestos al diseño y la implementación. [22]

3.1.1 Modelo de análisis

El modelo de análisis estructura los requisitos de un modo que facilita su comprensión, su preparación, su modificación y en general su mantenimiento, además se puede considerar como la primera aproximación al modelo de diseño. [23]

Para realizar el modelo de análisis se tienen en cuenta tres clases:

Clases	Descripción	Representación
Interfaz	Las clases de interfaz se utilizan para modelar la interacción entre el sistema y sus actores.	 Interfaz
Entidad	Las clases de entidad se utilizan para modelar información que poseen larga vida y que es a menudo persistente.	 Entidad
Control	Las clases controladoras representan coordinación, secuencia, transacciones y control de objetos y son utilizadas para encapsular el control de un caso de uso.	 Control

3.1.2 Diagrama de clases de análisis

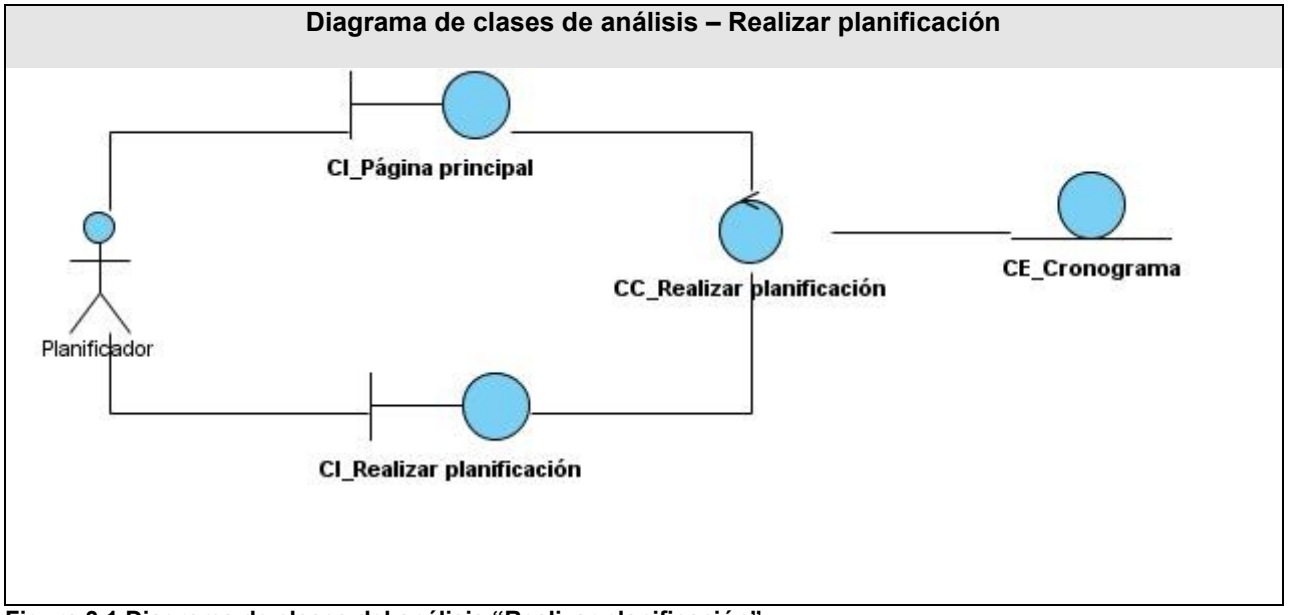


Figura 3.1 Diagrama de clases del análisis "Realizar planificación".

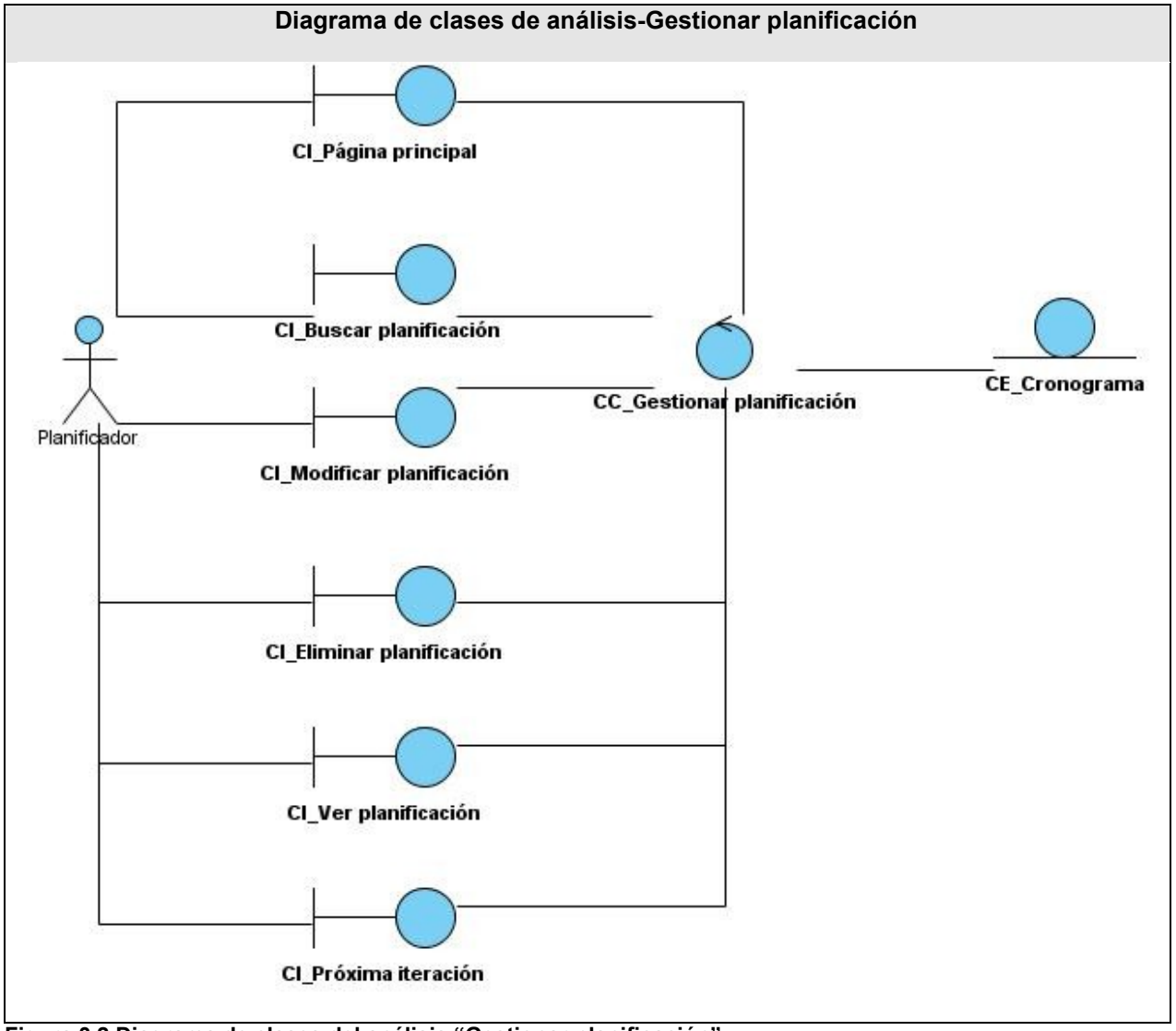


Figura 3.2 Diagrama de clases del análisis "Gestionar planificación".

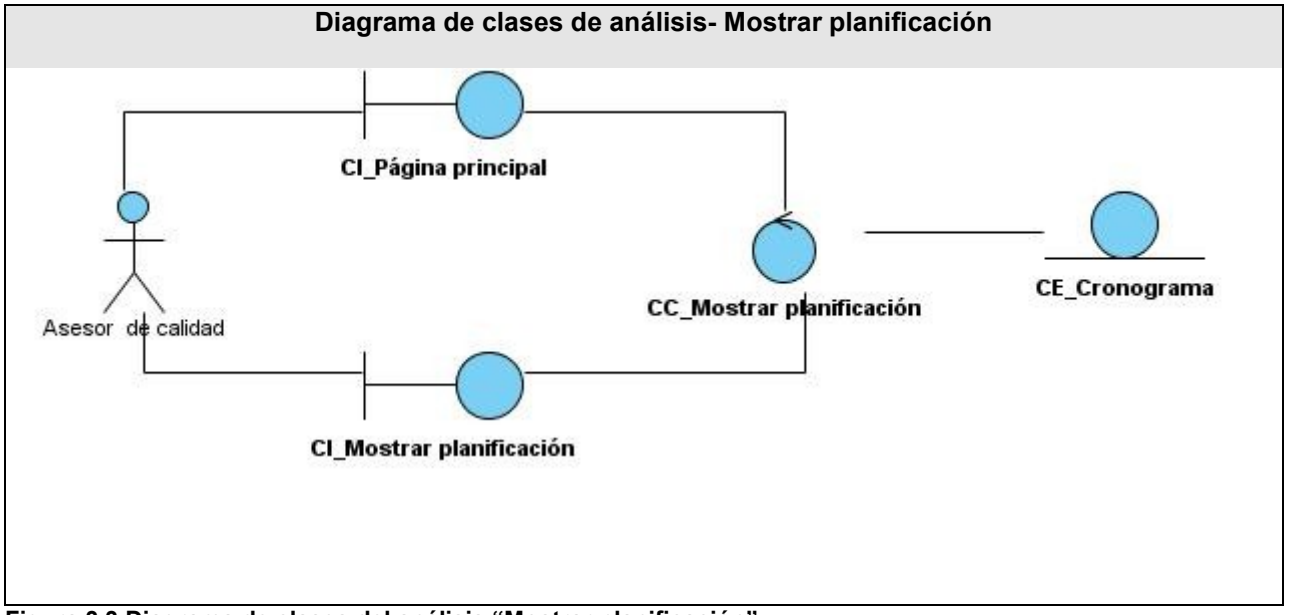


Figura 3.3 Diagrama de clases del análisis "Mostrar planificación".

3.2 Diseño

En el diseño se modela el sistema y se encuentra su forma para que soporte todos los requisitos, incluyendo los requisitos no funcionales y otras restricciones. Una entrada esencial en el diseño es el resultado del análisis. El propósito principal es adquirir una comprensión con profundidad de los aspectos relacionados con los requisitos funcionales, no funcionales y restricciones asociadas a la programación, sistemas operativos y tecnologías de interfaz de usuario. [24]

UML posee una extensión para el modelado de aplicaciones web, esa extensión es usada para el diseño de las clases. Los estereotipos que usa esta extensión son:

<<Client Page>> Una instancia de Página Cliente es una página web, con formato HTML. Cada página cliente solo puede ser construida por una página servidor.

<<Form>> Grupo de elementos de entrada que son parte de una página cliente. Sus atributos son los elementos de entrada del formulario.

<<Server page>> Representa la página web que contiene código que se ejecuta en el servidor.

3.2.1 Descripción de la arquitectura a utilizar. [25]

Las aplicaciones Web pueden desarrollarse utilizando cualquier arquitectura. La arquitectura del patrón MVC es un paradigma de programación bien conocido para el desarrollo de aplicaciones con interfaz gráfica.

El patrón MVC es un patrón de arquitectura de software en el cual todo el proceso está dividido en tres capas, típicamente estas capas son el Modelo, la Vista y el Controlador. aqui

El Modelo incorpora la capa del dominio y persistencia, es el encargado de guardar los datos en un medio persistente (ya sea una base de datos, un archivo de texto, XML[3], registro, etc.). Se refiere a la lógica del negocio o servicio y los datos asociados con la aplicación.

La Vista se encarga de presentar la interfaz al usuario, en sistemas Web, esto es típicamente HTML, aunque pueden existir otros tipos de vistas. En la vista, sólo se deben realizar operaciones simples. Es la encargada de la presentación de los datos.

El Controlador es el que escucha los cambios en la vista y los envía al modelo, este último retorna los datos a la vista. Es el que atiende las peticiones y componentes para la toma de decisiones de la aplicación.

El propósito del MVC es aislar los cambios. Es una arquitectura preparada para los cambios, que separa los datos y lógica de negocio de la lógica de presentación. A continuación se muestra un esquema de este modelo:

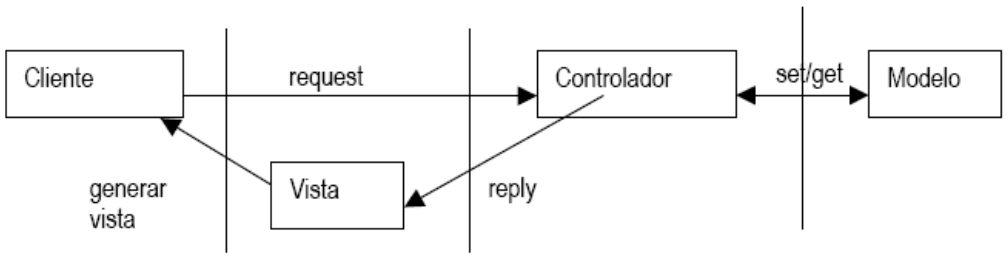


Figura 3.4 Esquema del patrón de arquitectura MVC.

[3] Es un Lenguaje de Etiquetado Extensible muy simple, pero estricto que juega un papel fundamental en el intercambio de una gran variedad de datos. Es un lenguaje muy similar a HTML, pero su función principal es describir datos y no mostrarlos como es el caso de HTML. XML es un formato que permite la lectura de datos a través de diferentes aplicaciones, Así como estructurar, almacenar e intercambiar información.

Django se rige por una arquitectura MVC (Modelo-Vista-Controlador), aunque se distingue en la vista de MVC los datos que se presentan (lo que ellos llaman vista) y la manera de mostrarlo, la plantilla. Por ello, ellos denominan a su arquitectura MTV (Model-Template-View, Modelo-Plantilla-Vista). El controlador en este caso sería el propio framework.[26]

3.2.2 Patrones

Los patrones de diseño no son más que la descripción de un problema y la solución del mismo, de forma que se pueda utilizar en diferentes contextos dando respuesta a interrogantes comunes. No es más que la solución efectiva que se le dio a un problema en un momento dado y puede ser reusable aplicándose en diferentes problemas de diseño en distintas circunstancias.

Los patrones GRASP (General Responsibility Assignment Software Patterns) son usados para la asignación de responsabilidades. Estos patrones son considerados como buenas prácticas recomendables en el diseño del software.

- **Experto:** La responsabilidad siempre recae sobre la clase que conoce toda la información necesaria para poder crear un objeto.
- **Creador:** Ayuda a identificar quien debe ser el responsable de la creación o instanciación de nuevos objetos o clases.
- **Controlador:** Sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, por lo que recibe las peticiones de los usuarios y se encarga de enviarlos a las distintas clases según el método llamado.
- **Alta cohesión:** La información que sea almacenada en la clase debe ser coherente y debe poseer alta relación con la misma.
- **Bajo acoplamiento:** La idea fundamental es tener las clases lo menos relacionadas posibles, lo que permite que la modificación de una de las clases repercuta lo menos posible en cada una de las restantes.

3.2.3 Diagramas de clases del diseño

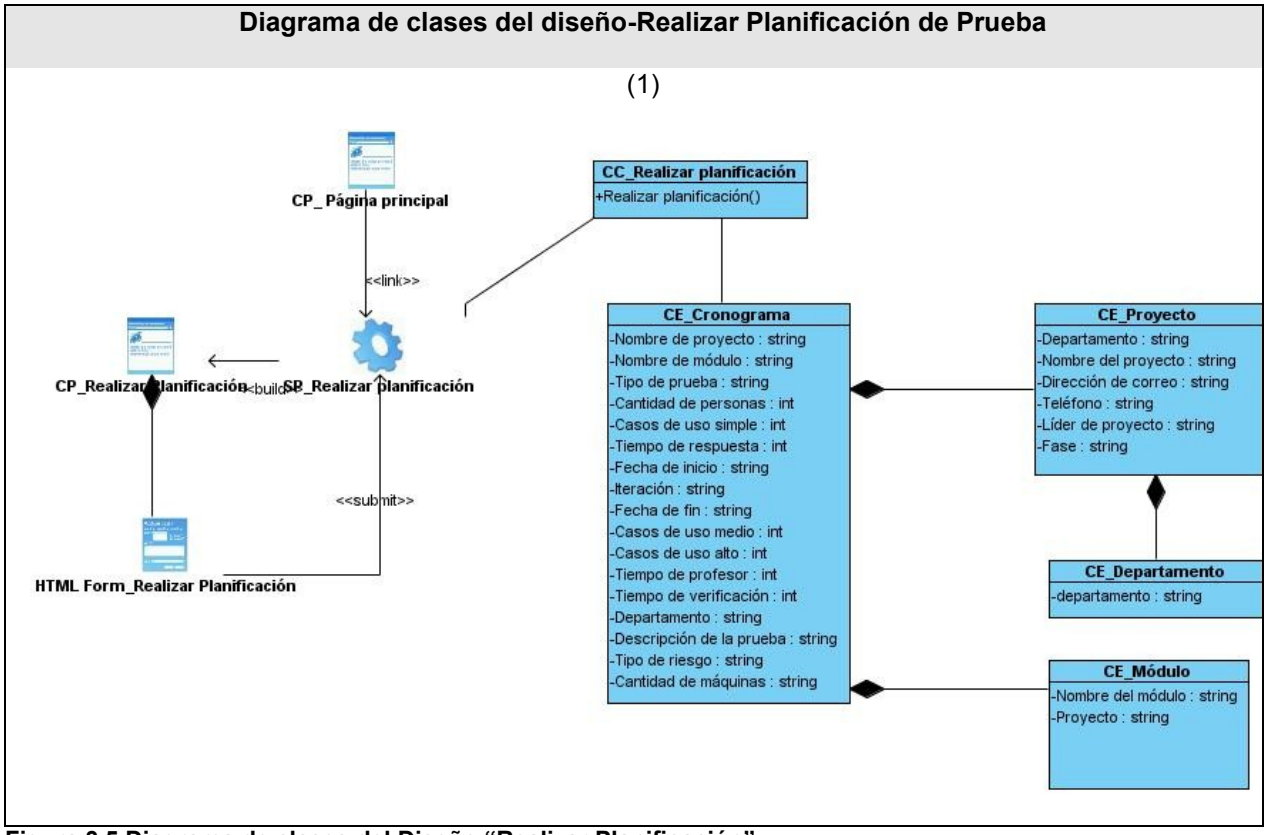


Figura 3.5 Diagrama de clases del Diseño "Realizar Planificación".

Diagrama de clases del diseño-Gestionar Planificación

(3)

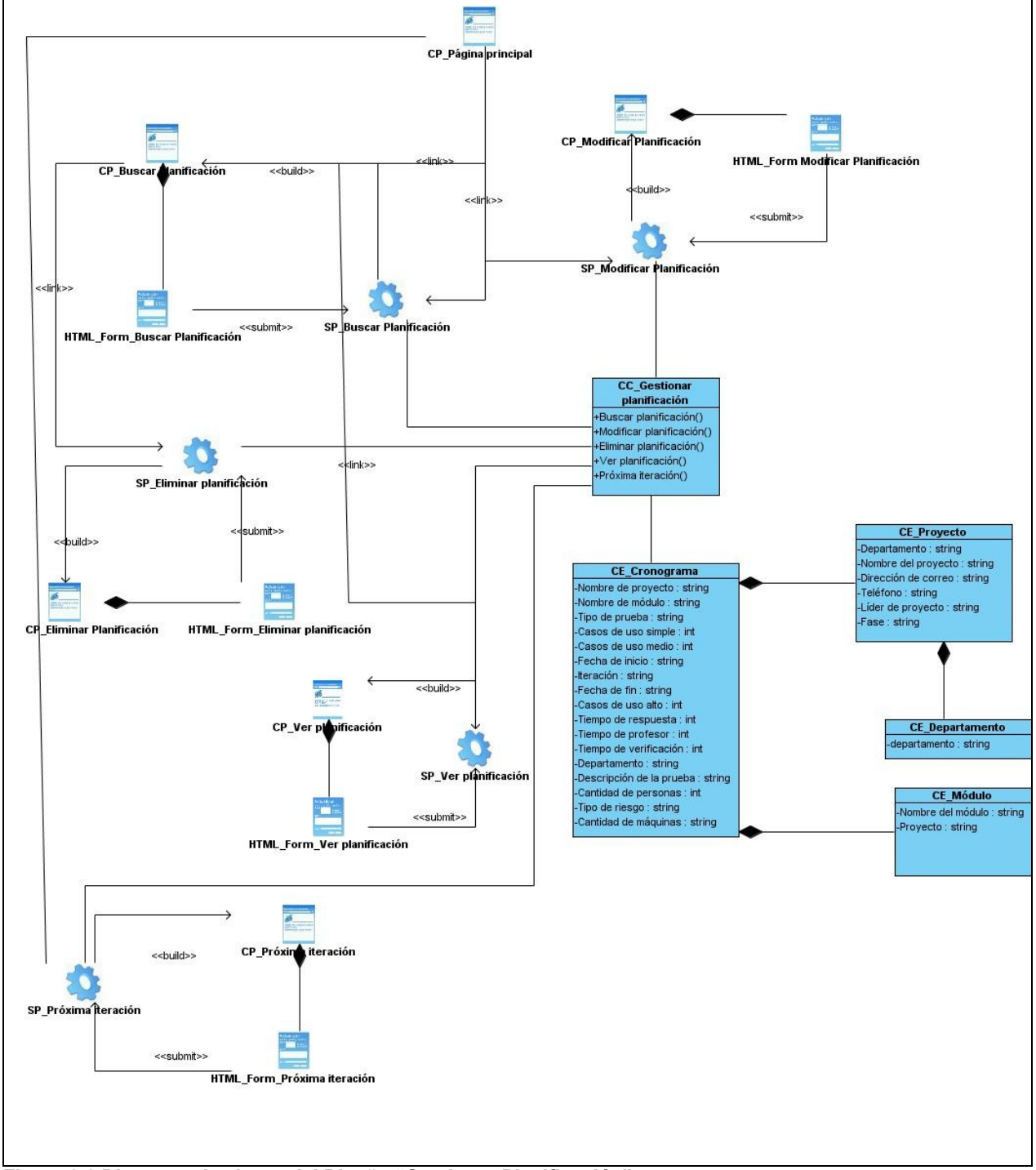


Figura 3.6 Diagrama de clases del Diseño "Gestionar Planificación".

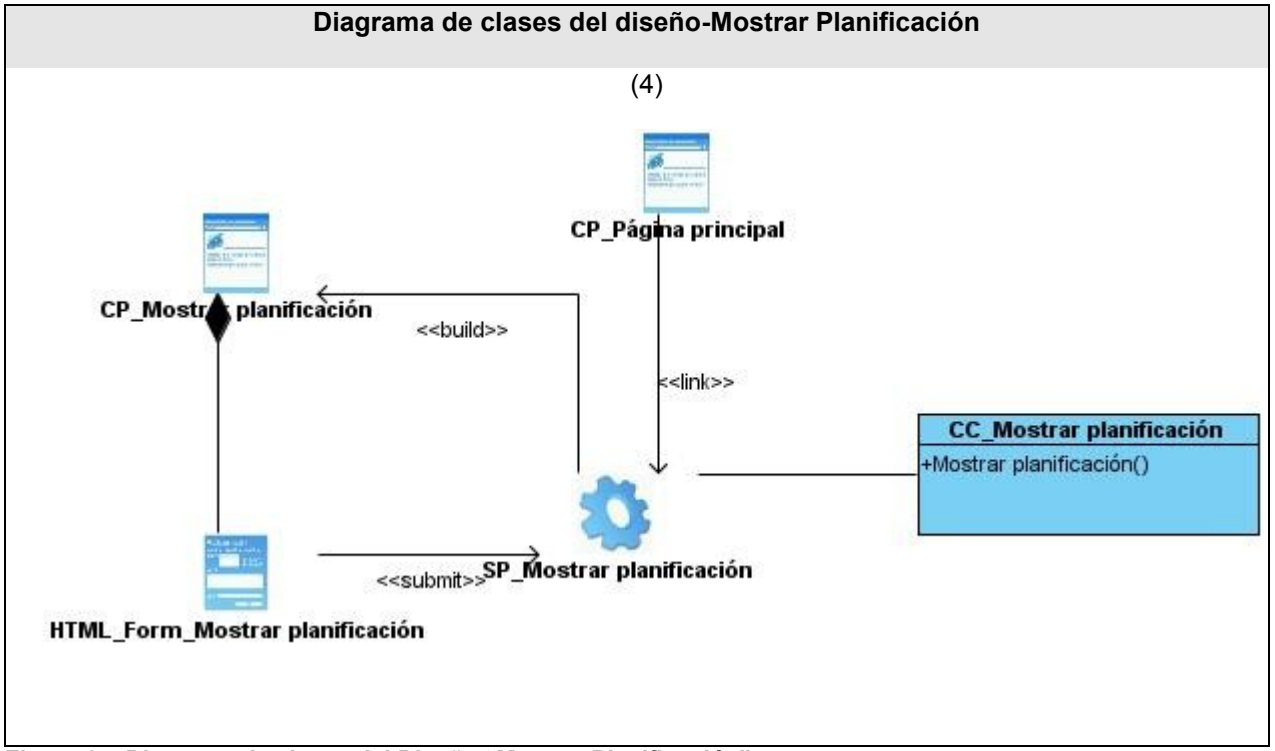


Figura 3.7 Diagrama de clases del Diseño "Mostrar Planificación".

3.2.4 Diagramas de secuencia

Es el diagrama que muestra las interacciones entre los objetos organizados en una secuencia temporal. En particular muestra los objetos participantes en la interacción y la secuencia de mensajes intercambiados.

Dentro del conjunto de mensajes representados dispuestos en una secuencia temporal, cada rol en la secuencia se muestra como una línea de vida, es decir, una línea vertical que representa el rol durante cierto plazo de tiempo, con la interacción completa. Los mensajes se muestran como flechas entre líneas de vida. Un diagrama de secuencia puede mostrar un escenario, es decir, una historia individual de transacción. Un uso de un diagrama de secuencia es mostrar la secuencia del comportamiento de un caso de uso.

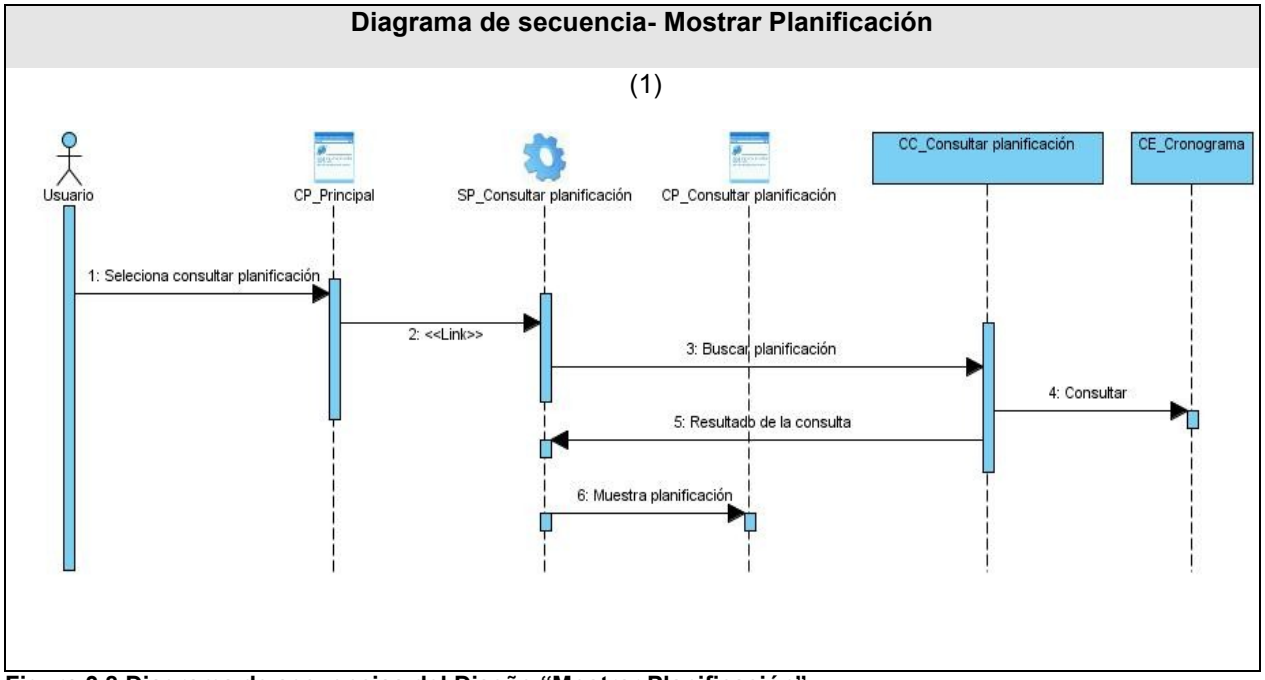


Figura 3.8 Diagrama de secuencias del Diseño "Mostrar Planificación".

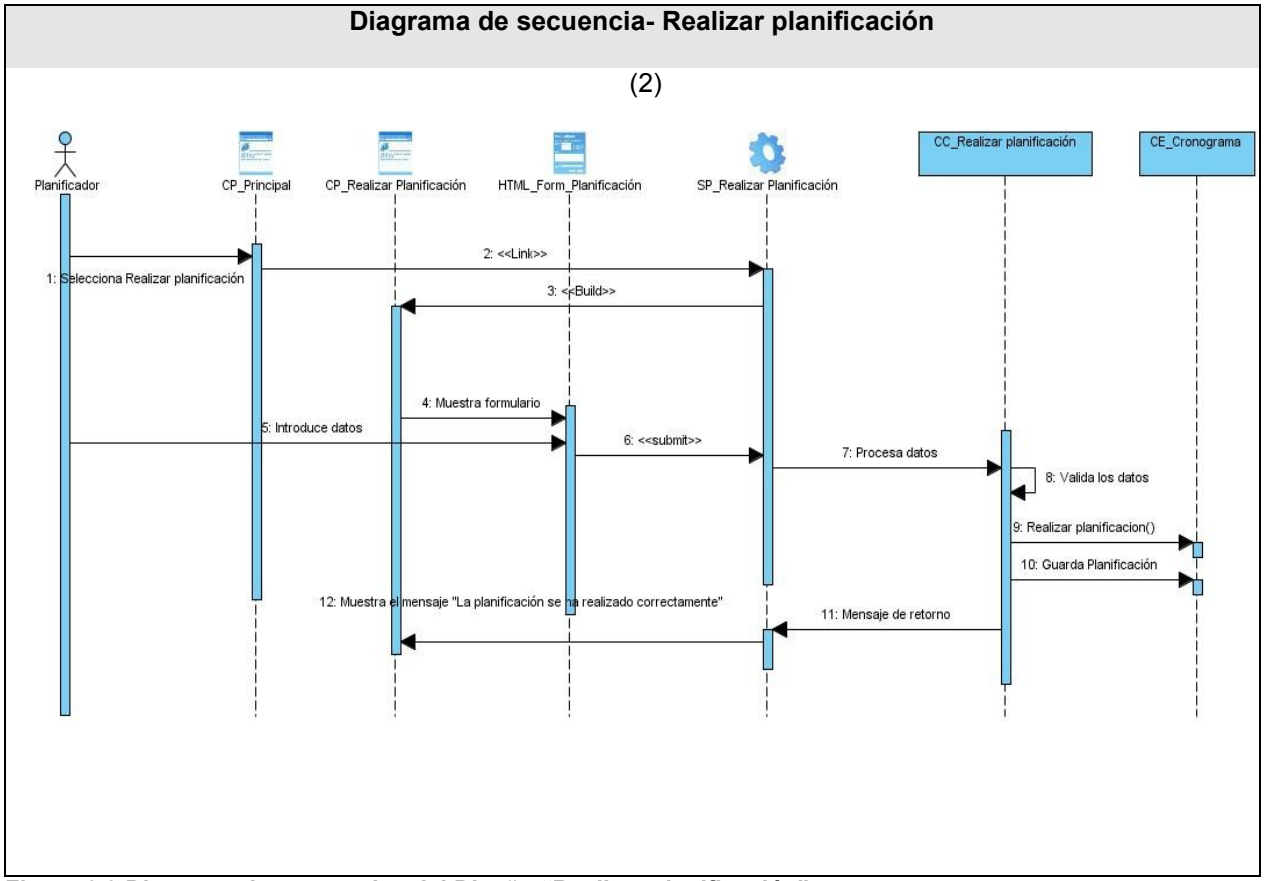


Figura 3.9 Diagrama de secuencias del Diseño "Realizar planificación".

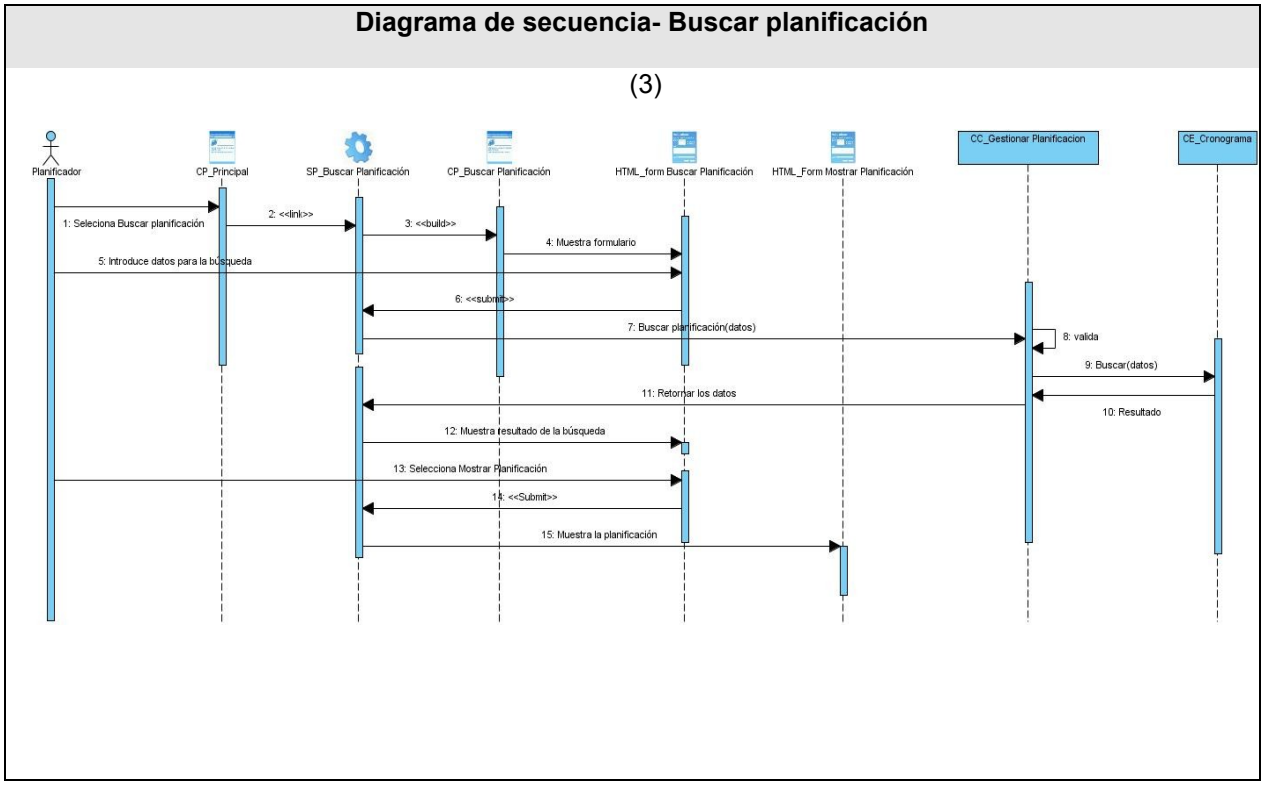


Figura 3.10 Diagrama de secuencias del Diseño "Buscar planificación".

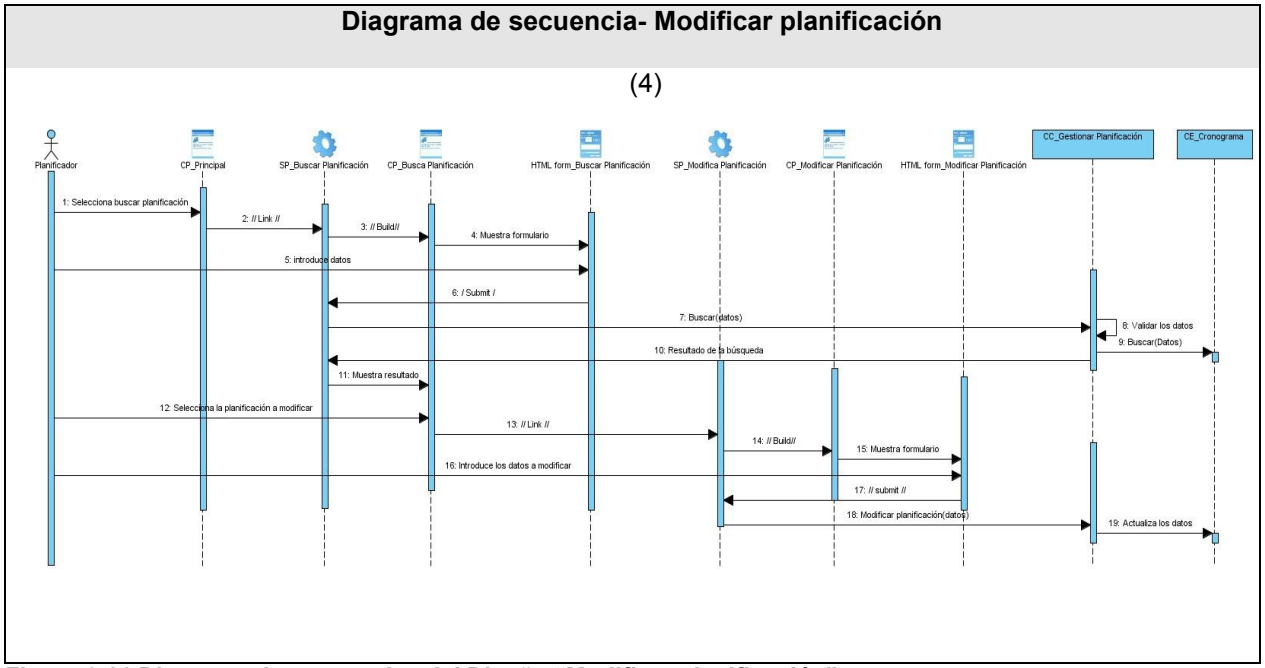


Figura 3.11 Diagrama de secuencias del Diseño "Modificar planificación".

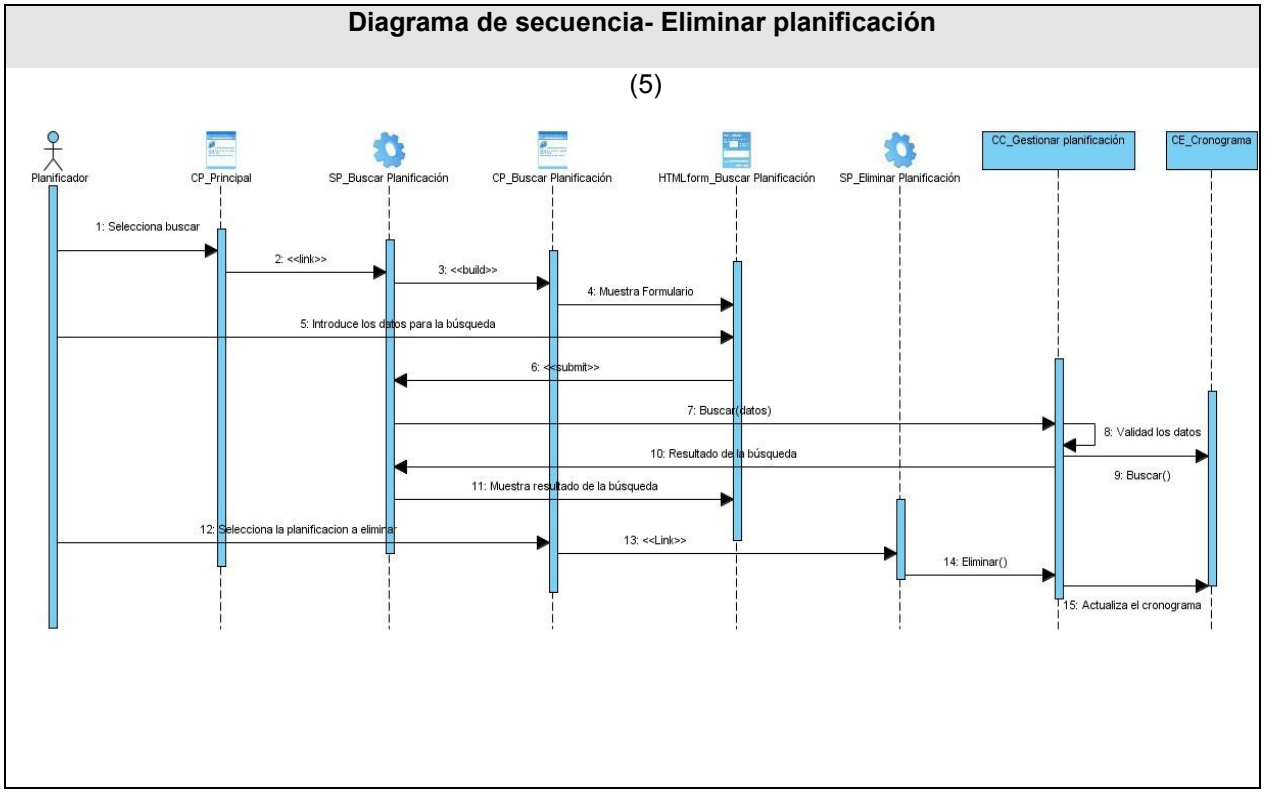


Figura 3.12 Diagrama de secuencias del Diseño "Eliminar planificación".

Para consultar los demás diagramas **Ver Anexo 1**

3.2.5 Descripción de las clases.

Nombre: Realizar planificación	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Realizar planificación
Descripción:	Permite realizar una planificación, luego de introducir un conjunto de variables.

Nombre: Gestionar planificación	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Buscar planificación
Descripción:	Permite realizar una búsqueda de las planificaciones insertadas en la aplicación.

Nombre:	Modificar planificación.
Descripción:	Permite modificar los datos de una planificación.
Nombre:	Ver planificación.
Descripción:	Permite ver los datos de una planificación.
Nombre:	Eliminar planificación.
Descripción:	Permite eliminar una planificación
Nombre:	Exportar planificación
Descripción:	Permite guardar el documento en cualquier parte de la máquina.

Nombre: Mostrar planificación	
Tipo de clase: Controladora	
Para cada responsabilidad:	
Nombre:	Mostrar planificación.
Descripción:	Permite ver las planificaciones realizadas hasta el momento en un diagrama de Gantt

Nombre: Planificación	
Tipo de clase: Entidad	
Atributo	**Tipo**
Nombre del proyecto	string
Nombre del módulo	string
Tipo de prueba	string
Cantidad de máquina	int
Cantidad de personas	int
Casos de uso simple	int
Casos de uso medio	int
Casos de uso alto	int
Tiempo de respuesta	int

Iteración	int
Fecha de inicio	string
Fecha de fin	string
Tiempo de profesor	int
Tiempo de verificación	int
Dependencia	string

Nombre: Realizar planificación	
Tipo de clase: Interfaz	
Atributo	**Tipo**
Departamento	ComboBox
Nombre del proyecto	ComboBox
Nombre del módulo	ComboBox
Tipo de prueba	TextBox
Cantidad de máquina	TextBox
Cantidad de personas	TextBox
Casos de uso simple	TextBox
Casos de uso medio	TextBox
Casos de uso alto	TextBox
Tiempo de verificación	TextBox
Tiempo de respuesta	TextBox
Iteración	TextBox
Tiempo de profesor	TextBox
Fecha de inicio	Calendario
Registrar	Booton
Cancelar	Booton

Para cada responsabilidad:	
Nombre:	Registrar
Descripción:	Permite guardar la planificación realizada.
Nombre:	Cancelar
Descripción:	Permite cancelar la opción.

Para ver más información **Ver Anexo 2**

Conclusiones

Durante el presente capítulo se realizó la modelación de los diagramas de clases de análisis y el diseño del sistema donde dejamos todo listo para el próximo flujo, implementación. También realizamos las descripciones de las clases, así como la arquitectura y patrones a utilizar. Se diseñó una base de datos donde se garantiza el cumplimiento de todos los requisitos funcionales.

Capítulo 4 Implementación y Prueba

En este capítulo se realiza el diseño de la base de datos. Además se reflejan los diferentes diagramas correspondientes a este flujo de trabajo, donde se implementará la propuesta realizada en el análisis y diseño, se representa el diagrama de despliegue, los de componentes y el modelo de prueba correspondiente a la aplicación desarrollada.

4.1Diseño de la Base de Datos

4.1.1Diagrama Entidad Relación

Diagrama de Entidad Relación.

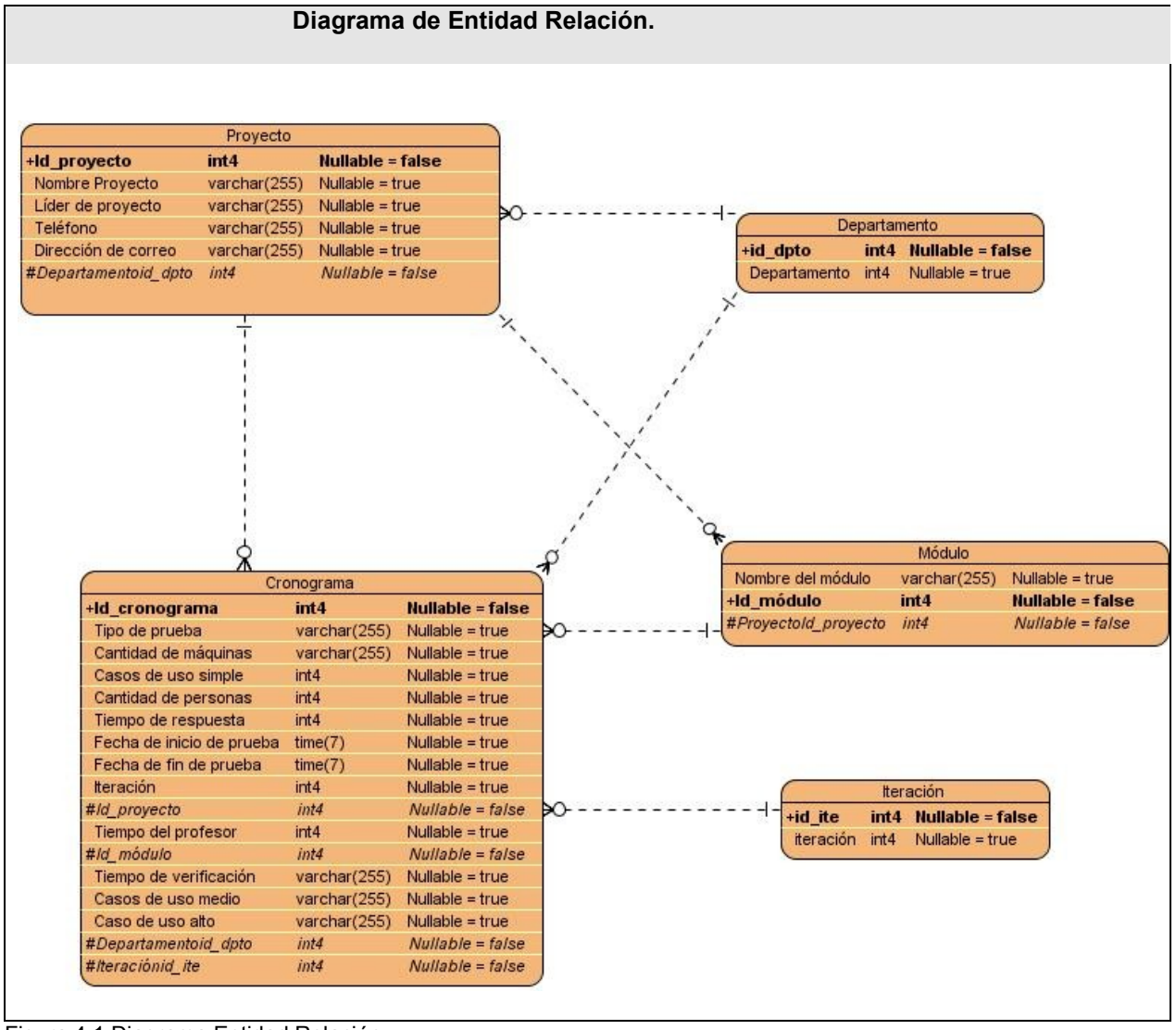


Figura 4.1 Diagrama Entidad Relación.

1.1.2 Descripción de las tablas de la Base de Datos

Nombre: Planificación.		
Descripción:		
Atributo	**Tipo**	**Descripción**
Departamento	varchar	Nombre del departamento al que pertenece ese proyecto.
Id cronograma	integer	Identificador del cronograma.
Nombre del proyecto	varchar	Nombre del proyecto al que le va ha realizar la planificación.
Nombre del módulo	varchar	Nombre del módulo al que se le va ha realizar la planificación de prueba.
Tipo de prueba	varchar	Nombre de la prueba que se le va a realizar al producto.
Cantidad de personas	integer	Cantidad de personas que le van a realizar las pruebas al producto.
Casos de uso simple	integer	Cantidad de casos de uso simple a probar.
Casos de uso medio	integer	Cantidad de casos de uso medio a probar.
Casos de uso alto	integer	Cantidad de casos de uso alto a probar.
Cantidad de máquinas	integer	Cantidad de máquinas disponibles para realizar las pruebas.
Tiempo de respuesta	integer	Tiempo que se va a demorar el equipo de desarrollo en darle respuesta a las no conformidades.
Tiempo de profesor	integer	Tiempo en que el profesor va a revisar las no conformidades que han encontrado los estudiantes.
Fecha de inicio de prueba	date	Fecha en que se comienza a probar el producto.
Iteración	integer	La iteración en que se esta probando.
Tiempo de verificación	integer	Tiempo de verificar si las no conformidades fueron resueltas.
Id proyecto	integer	Identificador del proyecto.
Id prueba	integer	Identificador de prueba.

4.2 Implementación

En la implementación se empieza con el resultado del análisis y diseño, y se implementa el sistema en términos de componentes, es decir: ficheros de código fuente, scripts, ficheros de códigos binarios y ejecutables. Dentro de los principales objetivos se encuentran: Distribuir el sistema asignando componentes ejecutables a nodos en el diagrama de despliegue e Implementar las clases y subsistemas encontrados en el diseño, entre otros. [27]

4.2.1 Diagrama de despliegue

Los Diagramas de Despliegue muestran la disposición física de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Un nodo es un recurso de ejecución tal como un computador, un dispositivo o memoria. Los estereotipos permiten precisar la naturaleza del equipo. [28]

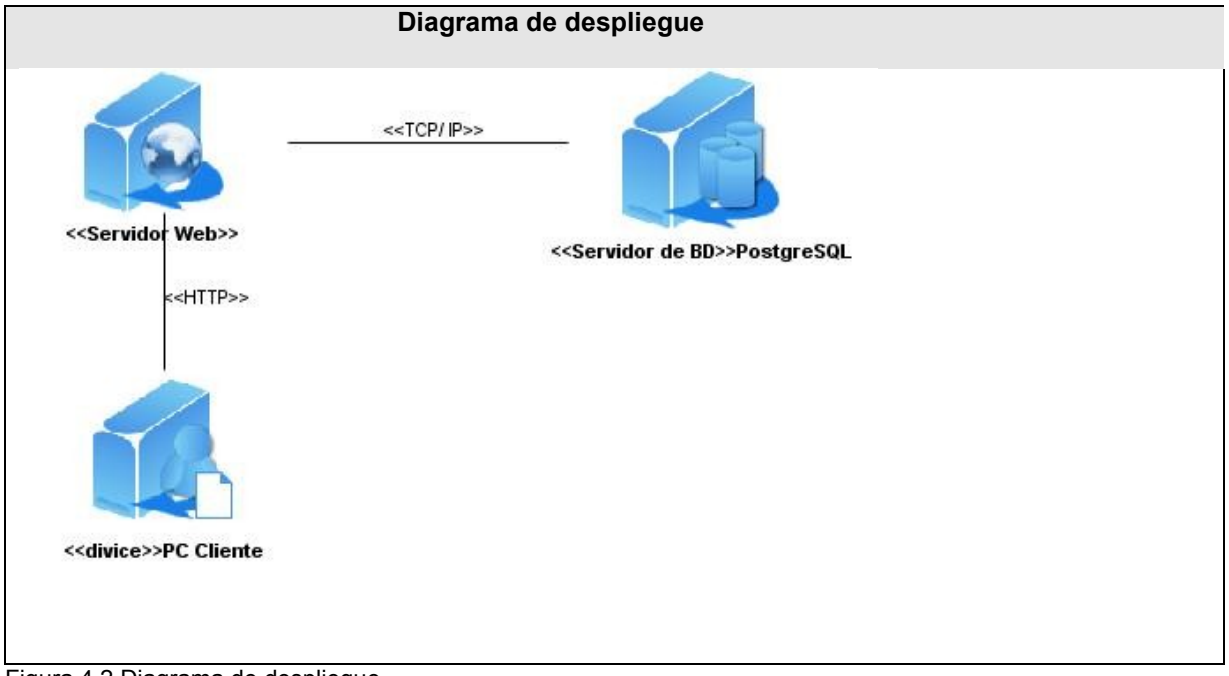


Figura 4.2 Diagrama de despliegue

4.2.2 Diagrama de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Los componentes representan todos los tipos de elementos software que entran en la fabricación de aplicaciones informáticas. Las relaciones de dependencia se utilizan en los

diagramas de componentes para indicar que un componente utiliza los servicios ofrecidos por otro componente.

Un diagrama de componentes representa las dependencias entre componentes software, incluyendo componentes de código fuente, componentes del código binario, y componentes ejecutables; hace parte de la vista física de un sistema, la cual modela la estructura de implementación de la aplicación por sí misma. Esta vista proporciona la oportunidad de establecer correspondencias entre las clases y los componentes de implementación. [29]

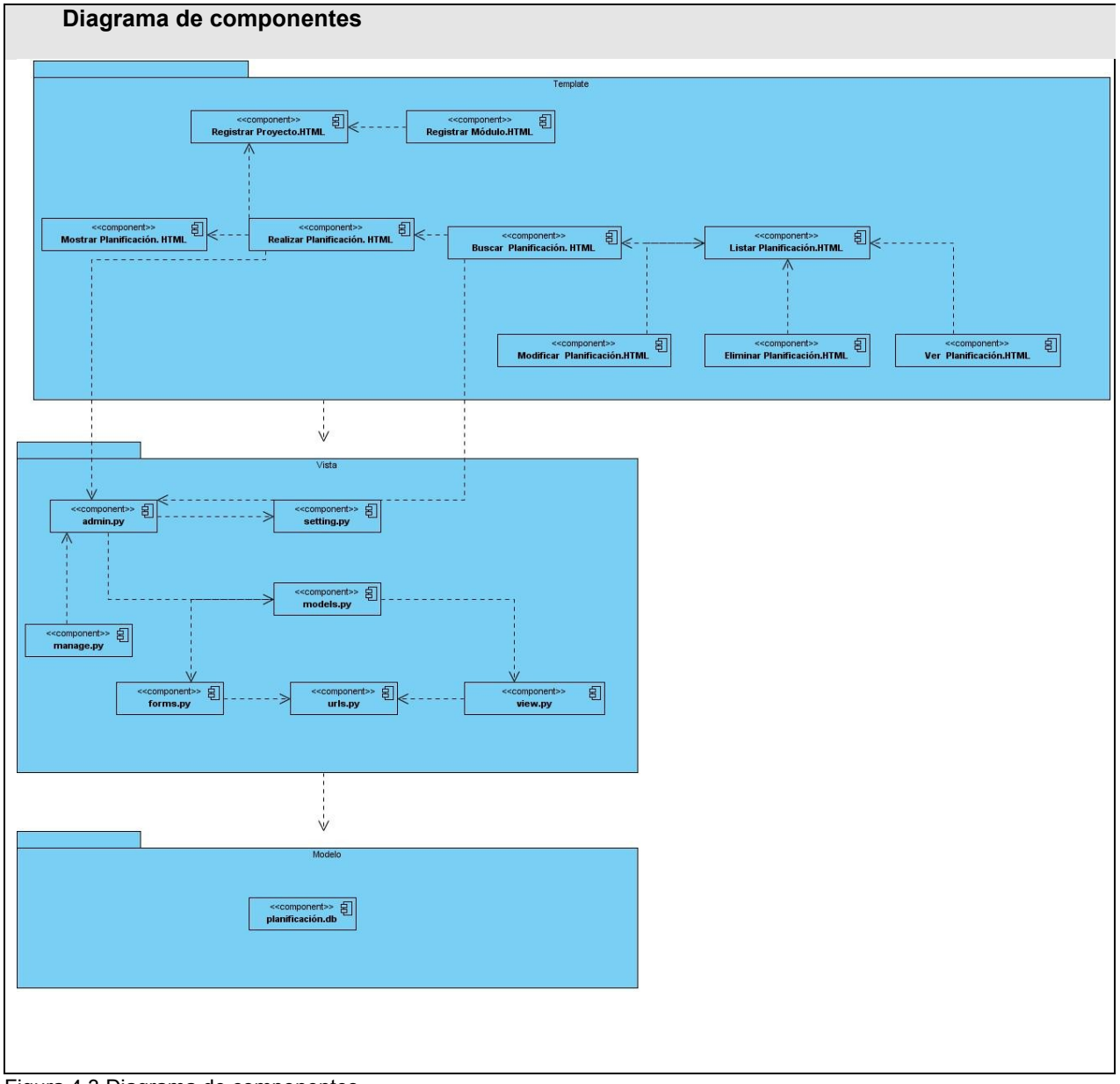


Figura 4.3 Diagrama de componentes.

4.3 Tratamiento de errores

Se contará con un sistema de tratamiento de errores para disminuir la posibilidad de cometerlos. Para esto se contará con la validación de la información introducida en el sistema. Mediante la interfaz Web se evitará que el usuario asuma un papel activo en la introducción de la información, para esto se contará con cuadros de opción, menú de selección lo cual facilitará la entrada de datos. La información que requiera ser adicionada por el usuario se validará mediante funciones que garanticen que sea válida y que el cuadro de texto no esté vacío si es obligatorio llenarlo. Si hay un error en la información le saldrá al usuario un mensaje en pantalla indicándole el error, al oprimir Aceptar el mensaje desaparece y el usuario podrá seguir introduciendo los datos en el formulario.

También se validarán las opciones correspondientes a la modificación de datos del servidor de base de datos. Si se desea eliminar algún elemento de la Base de Datos se preguntará al usuario si está seguro de realizar dicha acción. Así se logra que se realicen las operaciones que se desean y que se rectifiquen al cometer un error.

4.4 Seguridad

La seguridad en el sitio está implementada a través del servidor de base de datos SQlite y el uso de variables de sesión para restringir el acceso a los usuarios a determinadas páginas.

4.5 Prueba

Un instrumento adecuado para determinar el estado de la calidad de un producto de software es el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad, con el objetivo de medir el grado en que el software cumple con los requerimientos.

4.5.1 Modelo de Caso de Prueba

Caso de Uso Gestionar planificación

Sección: Buscar Planificación

Id del escenario	EC 1	EC 2
Escenario	Buscar Planificación	Flujo alterno “ Buscar planificación’
Variable 1 (Proyecto)	SACCEM	SACCEM

Variable 2 (Módulo)	Radiofísica	Radiofísica
Botón 1 (Buscar)	NA	NA
Respuesta del Sistema	Muestra: Listado de planificaciones que cumplen con los criterios de búsqueda, mostrando los siguientes atributos: • Módulo. • Proyecto. • Iteración • Fecha de inicio • Prueba • Y los íconos para Ver, Modificar, Eliminar, Próxima iteración	Cuando no encuentra la información muestra el mensaje de información: “No existe la planificación solicitada.”.
Resultado de la Prueba	Satisfactorio	Satisfactorio

Sección: Modificar Planificación

Id del escenario	EC 1	EC	EC 3	EC 4
Escenario	Modificar planificación	Cancelar operación	Existen datos incompletos	Existen datos incorrectos
Variable 1 (Departamento) Este campo es de selección	SAS		SAS	SAS
Variable 2 (Proyecto) Este campo es de selección	SACCEM		SACCEM	SACCEM

Variable 3 (Módulo) Este campo es de selección	Radiofísica		Radiofísica	Radiofísica
Variable 4 (Nombre de prueba)	funcional			funcional
Variable 5 (Descripción)	Se le hará pruebas a este módulo, para ver su eficacia.			Se le hará pruebas a este módulo, para ver su eficacia.
Variable 6 (Cantidad de casos de uso simple)	10			sdfsd
Variable 7 (Cantidad de casos de uso medio)	10			rfgdf
Variable 8 (Cantidad de casos de uso alto)	10		dffd	rgff
Variable 9 (Cantidad de máquinas)	10		sds	dsfsdf
Variable 10 (Cantidad de personas)	12			dsdfd
Variable 11 (Fecha de inicio)	15/6/2010			15/6/2010

Variable 13 (Iteración)	1era			1era
Variable 14 (Tiempo de respuesta)	1		dsfsdsd	gbf
Variable 15 (Tiempo de profesor)	1			sdsd
Variable 16 (Tiempo de verificación)	2			dsd
Variable 1 (Dependencia)	NA			dd
(Modificar)	NA		NA	NA
(Cancelar)		NA		
Respuesta del Sistema	Modifica la planificación. Muestra un mensaje informativo.”Se ha modificado satisfactoriamente”	Regresa a la vista anterior.	Muestra un mensaje al lado de los campos incompletos. “Este campo es obligatorio”	Muestra un mensaje al lado de los campos incorrectos. “Introduzca números enteros”
Resultado de la Prueba	**Satisfactoria**	**Satisfactoria**	**Satisfactoria**	**Satisfactoria**

Caso de uso Mostrar planificación

Sección: Mostrar Planificación

Id del escenario	EC 5	EC 6
Escenario	Mostrar planificación	Flujo alterno “Mostrar planificación”
Respuesta del Sistema	Si existen planificaciones registradas, el sistema muestra el diagrama de Gantt con todas las planificaciones realizadas hasta el momento.	Si no existen planificaciones el sistema muestra un mensaje “No existen planificaciones, debe registrarla” y regresa a la vista de “Realizar planificación”
Resultado de la Prueba	Satisfactorio	Satisfactorio

Caso de Uso Realizar planificación

Realizar planificación

Id del escenario	EC 7	EC 8	EC 9
Escenario	Registrar Planificación	Cancelar operación	Existen datos incompletos
Variable 1 Departamento	SAS		SAS
Variable 2 Proyecto	SACCEM		SACCEM
Variable 3 Módulo	Radiofísica		Radiofísica
Variable 4 Nombre de prueba	funcional		
Variable 5 Descripción	Se le hará pruebas a este módulo, para ver su eficacia.		
Variable 6 Cantidad de casos de uso	10		fd

simple			
Variable 7 Cantidad de casos de uso medio	10		fd
Variable 8 Cantidad de casos de uso alto	10		df
Variable 9 Cantidad de máquinas	10		s
Variable 10 Cantidad de personas	10		f
Variable 11 Fecha de inicio	15/6/2010		15/6/2010
Variable 13 Iteración	1era		
Variable 14 Tiempo de respuesta	1		
Variable 15 Tiempo de profesor	1		
Variable 16 Tiempo de verificación	1		
Botón 2 (Registrar)	NA		NA

Botón 3 (Cancelar)		NA	
Respuesta del Sistema	Se registra la planificación. Y muestra el mensaje de información “Se ha registrado satisfactoriamente”.	Cancela la acción de registrar. Vuelve a la página de inicio.	Muestra el mensaje de información “Hay errores en el formulario”. Y un indicador al lado de los campos incorrectos o incompletos
Resultado de la Prueba	**Satisfactorio**	**Satisfactorio**	**Satisfactorio**

Para ver más información **Ver Anexo 3**

Conclusiones.

El desarrollo de este capítulo propició un mayor entendimiento de la propuesta realizada, teniendo en cuenta que se describieron las tablas de la base de datos. Se realizó la implementación del sistema y se mostraron losl diagramas de despliegue y de componentes donde se evidencia como va estar dividido el sistema. Además, se diseñaron casos de pruebas para comprobar la eficacia del módulo desarrollado.

Conclusiones

Una vez concluida la investigación y la herramienta que informatiza el proceso de planificación de pruebas, se ha dado cumplimiento al objetivo planteado y se obtuvieron los resultados que a continuación se mencionan:

- A partir del estudio de las deficiencias del proceso de planificación de pruebas y las nuevas exigencias del cliente, se derivaron los requerimientos funcionales del sistema a informatizar.
- El análisis del estado del arte reveló que no existen aplicaciones informáticas a nivel nacional e internacional, que se ajusten a la forma en que se llevan a cabo los procesos del negocio.
- Como resultado del modelamiento de los flujos de trabajos, que propone la metodología RUP mencionados en las tareas de la investigación, se obtuvieron varios artefactos necesarios para obtener una visión detallada del sistema a informatizar.
- Se implementó la herramienta informática propuesta.
- Se probó el sistema para valorar la eficacia del mismo.

Recomendaciones

Por la experiencia adquirida durante la realización del presente trabajo de diploma las autoras recomiendan:

- Especificar que en la solicitud de los proyectos se incluyan la complejidad de los artefactos a revisar.
- Gestionar las solicitudes de prueba para una próxima versión del sistema.
- Agregar la opción de realizar la planificación dando los días y que este dé la cantidad de personas que necesite para revisar un módulo determinado.
- Realizar un histórico de las planificaciones para próximas versiones.

Referencia Bibliográfica

1. El concepto de planificación. [Online] [Cited: noviembre 19, 2009.] http://www.inta.gov.ar/bariloche/desarrollo/gesrural/trabajos/planificacion/Archivos/documento20

2. Proceso. [Online] [Cited: noviembre 19, 2009.] http://es.wikipedia.org/wiki/Proceso

3. **Costa, Sussana.** La planificación. [Online] [Cited: noviembre 19, 2009.] http://www.monografias.com/trabajos35/la-planificacion/la-planificacion.shtml.

4. **Fernández, Eduardo.** Planificación. [Online] [Cited: noviembre 19, 2009.] http://www.monografias.com/trabajos34/planificacion/planificacion.shtml.

5. **Silva, Rosanna.** Planificación. [Online] http://www.slideshare.net/rosilfer/planificacin-1965280

6. **Aguilera, Dinella.** Sistema Integrado de Gestión Estadísticas(SIGE)

7. Ídem 6.

8. Ídem 6.

9. Ídem 6.

10. Ídem 6.

11. Gestión de proyectos con dotProject.[Online][Cited: diciembre 1, 2009] http://www.abartiateam.com/dotproject

12. Redmine.[Online][Cited: diciembre 1, 2009]http://www.redmine.org/

13. **Jacobson, Ivar, Booch, Grady y Runbaugh, James**. El proceso Unificado de Desarrollo de Software. 1999. p. 173. Vol. I.

14. **Hernandis, José Alberto.** Visual Paradigm for UML. [Online] [Cited: diciembre 2, 2009.] http://www.versioncero.com/noticia/210/visual-paradigm-for-uml.

15. **Caballero, Ismael.** Una herramienta CASE: Visual Paradigm. [Online] [Cited: diciembre 4, 2009.] http://alarcos.inf-cr.uclm.es/per/fgarcia/isoftware/doc/LabTr1_VP.pdf.

16. **Gracia, Joaquin.** UML: Diagramas UML. ¿Qué es UML? [Online] [Cited: diciembre 6, 2009.] http://www.ingenierosoftware.com/analisisydiseno/uml.php.

17. Django TARINGA! *El libro de Django.* [Online] [Cited: enero 12, 2010.] http://www.taringa.net/posts/ebooks-tutoriales/1609719/Django-Framework-Espa%C3%B1ol.html.

18. Django. [Online] [Cited: enero 18, 2010.] http://www.djangoproject.com.

19. Python, web oficial. [Online] [Cited: enero 16, 2010.] http://www.python.org.

20. Comunidad linperial de desarrolladores. [Online] [Cited: enero 19, 2010.] http://www.linperial.com/communities/forums/developers/?q=node/62.

21. eAprende. *Gestor de Base de Datos.* [Online] [Cited: enero 20, 2010.] http://www.eaprende.com/gestor-de-basededatos-mysql-postresql-sqlite.html.

22. __.El proceso Unificado de Desarrollo de Software. 1999. pp. 165-166. Vol. I.

23. __.El Proceso Unificado de Desarrollo de Software. 1999. p. 205. Vol. I.

24. . Ídem 22.

25. **Helman, Dean.** Model-View-Controller. [Online] [Cited: enero 21, 2010.] http://ootips.org/mvc-pattern.html.

26. Django http://es.wikipedia.org/wiki/Django. [Online] [Cited: enero 21, 2010.] http://es.wikipedia.org/wiki/Django.

27. El proceso unificado de desarrollo de software. [Online] [Cited: febrero 20, 2010.] http://bibliodoc.uci.cu/pdf/reg00060.pdf .

28. **Wesley, Addison, et al.** Consultoría de seguridad. [Online] [Cited: marzo 10, 2010.] http://www.creangel.com/uml/despliegue.php

29. **Wesley, Addison, et al.** Consultoría de seguridad. [Online] [Cited: marzo 10, 2010.] http://www.creangel.com/uml/componente.php

Bibliografía

1. **Aguilera, Dinella.** Sistema Integrado de Gestión Estadísticas(SIGE)

2. **Costa, Sussana.** La planificación. [Online] [Cited: noviembre 19, 2009.] http://www.monografias.com/trabajos35/la-planificacion/la-planificacion.shtml.

3. **Caballero, Ismael.** Una herramienta CASE: Visual Paradigm. [Online] [Cited: diciembre 4, 2009.] http://alarcos.inf-cr.uclm.es/per/fgarcia/isoftware/doc/LabTr1_VP.pdf.

4. Conferencia 5. Flujo de trabajo de Prueba. [Online] [Cited: marzo 30, 2010.]

5. Django TARINGA! *El libro de Django.* [Online] [Cited: enero 12, 2010.] http://www.taringa.net/posts/ebooks-tutoriales/1609719/Django-Framework-Espa%C3%B1ol.html.

6. Django. [Online] [Cited: enero 18, 2010.] http://www.djangoproject.com

7. Django http://es.wikipedia.org/wiki/Django. [Online] [Cited: enero 21, 2010.]

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

8. El concepto de planificación. [Online] [Cited: noviembre 19, 2009.]

http://www.inta.gov.ar/bariloche/desarrollo/gesrural/trabajos/planificacion/Archivos/documento

9. eAprende. *Gestor de Base de Datos.* [Online] [Cited: enero 20, 2010.]

http://www.eaprende.com/gestor-de-basededatos-mysql-postresql-sqlite.html.

10. El proceso unificado de desarrollo de software. [Online] [Cited: febrero 20, 2010.]

http://bibliodoc.uci.cu/pdf/reg00060.pdf .

11. **Fernández, Eduardo.** Planificación. [Online] [Cited: noviembre 19, 2009.] http://www.monografias.com/trabajos34/planificacion/planificacion.shtml.

12. **Gracia, Joaquin.** UML: Diagramas UML. ¿Qué es UML? [Online] [Cited: diciembre 6, 2009.] http://www.ingenierosoftware.com/analisisydiseno/uml.php.

13. Python, web oficial. [Online] [Cited: enero 16, 2010.] http://www.python.org

14.Proceso. [Online] [Cited: noviembre 19, 2009.] http://es.wikipedia.org/wiki/Proceso

15. **Grady Booch, James Rumbaugh Ivar Jacobson. s.l. : Addison Wesley Longman.** *El proceso Unificado de Desarrollo de Software.* 1999. pp. 165-166. Vol. I.

16. **Grady Booch, James Rumbauch. Ivar Jacobson. s.l. : Addison Wesley Longman.** *El Proceso Unificado de Desarrollo de Software.* 1999. p. 205. Vol. I.

17. —. *El proceso Unificado de Desarrollo de Software.* 1999. p. 173. Vol. I.

18. Gestión de proyectos con dotProject.[Online][Cited: diciembre 1, 2009] http://www.abartiateam.com/dotproject

19. **Hernandis, José Alberto.** Visual Paradigm for UML. [Online] [Cited: diciembre 2, 2009.] http://www.versioncero.com/noticia/210/visual-paradigm-for-uml.

20. **Helman, Dean.** Model-View-Controller. [Online] [Cited: enero 21, 2010.] http://ootips.org/mvc-pattern.html

21. http://office.microsoft.com/es-hn/project/HA010731223082.aspx

22. http://www.abartiateam.com/dotproject

23. http://www.desarrolloweb.com/scripts/dotproyect-gestion-proyectos-php.html

24.http://www.webmasterlibre.com/2006/06/05/dotproject-203/

25. http://www.vtc.com/products/Microsoft-Project-2003-Espanol-tutorials.htm

26.http://www.scribd.com/doc/2539650/manual-microsoft-project-2003

27. http://www.liquidplanner.com/

28. http://www.b-kin.com/

29. http://www.entrebits.com/software/project-kickstart/3_0

30. http://www.todoprogramas.com/negocios/proyectos/index.asp?orden=valoracion

. http://www.wingide.com.

31. http://www.compute-rs.com/es/consejos-1571565.htm

32. **Informáticas, Universidad de las Ciencias.** Confernecia 7 Ingeniería de Software. Ciudad de la Habana : s.n., 2010.

33. **Informáticas, Universidad de las Ciencias.** *Conferencia 6 Ingeniería de Software.* Ciudad de la Habana : s.n.

34. Proceso. [Online] [Cited: noviembre 19, 2009.] http://es.wikipedia.org/wiki/Proceso

35.**Reyes Prieto, Dariel Fernando; Seguí Garcia, Arian.**Herramienta para la gestión de la información del Expediente de Proyecto

36. Redmine.[Online][Cited: diciembre 1, 2009]http://www.redmine.org/

37. **Silva, Rosanna.** Planificación. [Online] http://www.slideshare.net/rosilfer/planificacin-1965280.

38. **Wesley, Addison, et al.** Consultoría de seguridad. [Online] [Cited: marzo 10, 2010.] http://www.creangel.com/uml/despliegue.php

39. **Wesley, Addison, et al.** Consultoría de seguridad. [Online] [Cited: marzo 10, 2010.] http://www.creangel.com/uml/componente.php

API: Application Programming Interface.

DBMS: Database Management System

Herramienta Case: Ingeniería de Software Asistida por Ordenador

IDE: Entorno Integrado de Desarrollo

Linux: Sistema Operativo. paradigmas más prominentes del software libre y del desarrollo del código abierto.

MacOSX: es el actual sistema operativo de la familia de ordenadores Macintosh.

OSI: Licencias de código abierto.

OMG: Grupo de Administración de Objetos.

OpenSource: Código abierto

POO: Programación orientada a objetos.

SOAP: Simple Object Access Protocol

UML: Lenguaje Unificado de Modelado

Anexo 1

Caso de uso 4: Registrar iteración	
Propósito	El Usuario (planificador) podrá registrar las n iteraciones que necesite un módulo determinado.
Actores: Usuario	
Resumen: El Usuario podrá registrar las iteraciones.	
Acción del actor	**Respuesta del sistema**
1. El planificador selecciona la opción “Registrar iteración”.	2. El sistema muestra una interfaz para registrar la iteración en caso de que no esté. En caso de existir, ver flujo alterno 2.1.
Flujo alterno Sección ”Mostrar planificación”	
	2.1 El sistema permite seleccionarla, para poder realizar la planificación.

Diagramas de secuencia del diseño.

Diagrama de secuencia- Exportar planificación

(6)

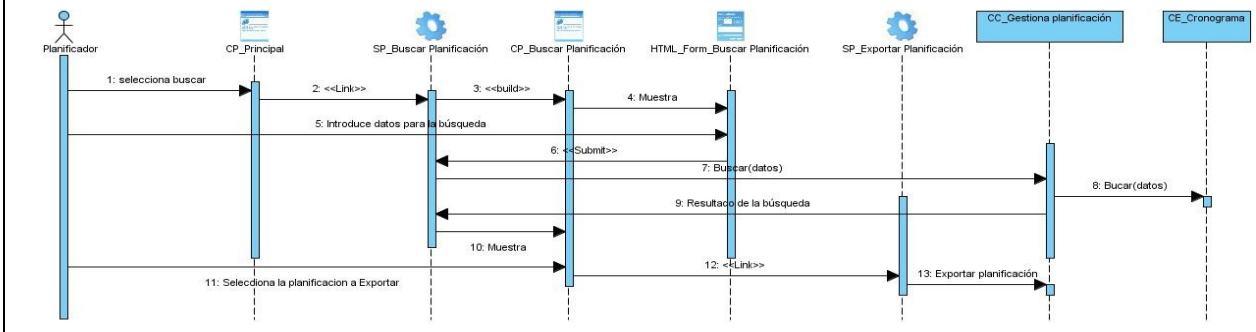


Diagrama de Secuencia, CU Gestionar planificación, Sección Exportar planificación.

Diagrama de secuencia- Próxima iteración

(7)

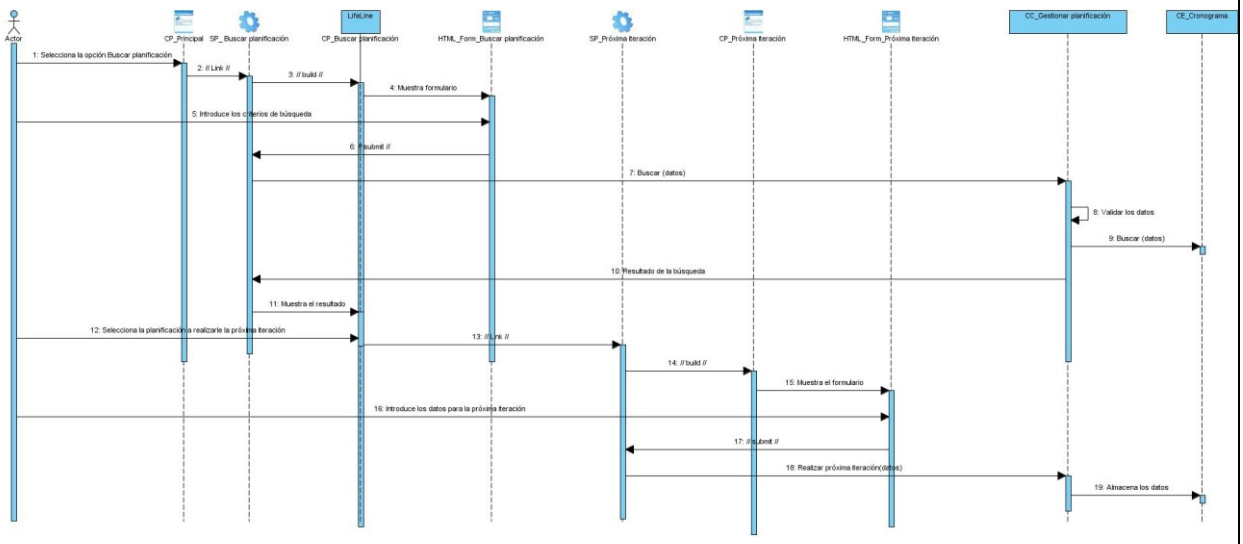


Diagrama de Secuencia, CU Gestionar planificación, Sección Próxima iteración.

Diagrama de Análisis- Registrar iteración

(8)

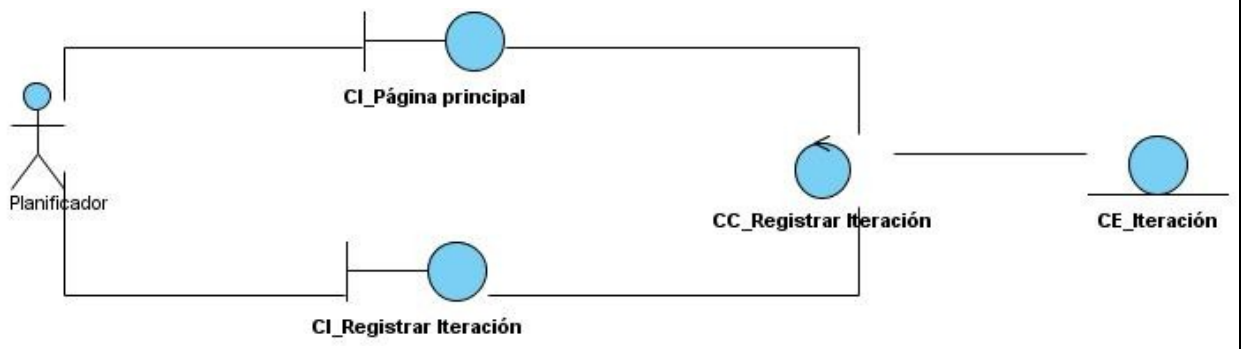


Diagrama de Análisis, CU Registrar iteración

Diagrama de Diseño- Registrar iteración

(9)

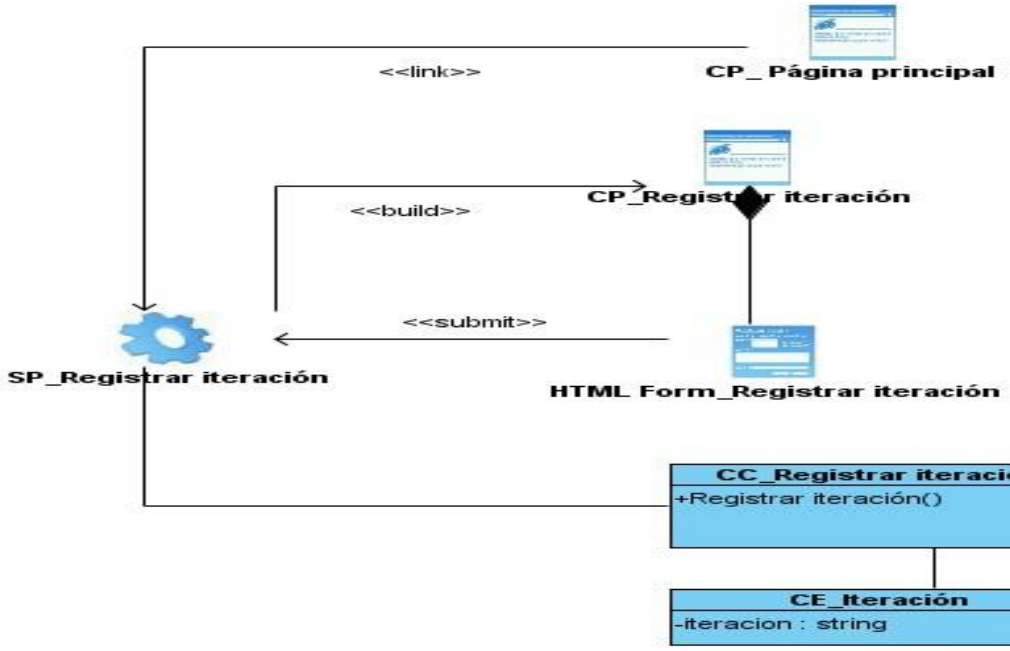
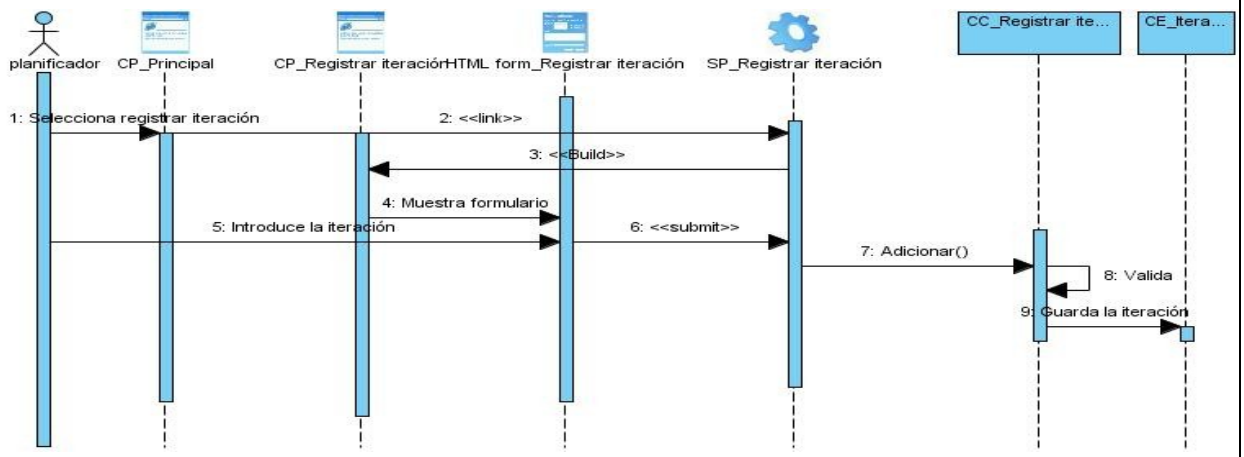


Diagrama de Diseño, CU Registrar iteración

Diagrama de secuencia- Registrar iteración

(8)



Anexo 2

Descripción de las clases del diseño

Nombre: Mostrar planificación	
Tipo de clase: Interfaz	
Para cada responsabilidad:	
Nombre:	Mostrar planificación
Descripción:	Permite mostrar las planificaciones

Nombre: Registrar iteración
Tipo de clase: Interfaz
Para cada responsabilidad:

Nombre:	Registrar iteración
Descripción:	Permite registrar las iteraciones.

Nombre: Buscar planificación	
Tipo de clase: Interfaz	
Atributo	**Tipo**
Nombre del proyecto	ComboBox
Nombre del módulo	ComboBox
Iteración	ComboBox
Tipo de prueba	TextBox
Buscar	Booton
Cancelar	Booton
Para cada responsabilidad:	
Nombre:	Buscar planificación
Descripción:	Permite realizar una búsqueda con los criterios deseados.
Nombre:	Cancelar
Descripción:	Permite cancelar la opción.

Nombre: Modificar planificación	
Tipo de clase: Interfaz	
Atributo	**Tipo**
Departamento	ComboBox
Nombre del proyecto	ComboBox
Nombre del módulo	ComboBox
Tipo de prueba	CharField
Cantidad de personas	IntegerField

Casos de uso simple	IntegerField
Casos de uso medio	IntegerField
Casos de uso alto	IntegerField
Cantidad de máquinas	IntegerField
Tiempo de respuesta	IntegerField
Tiempo de profesor	IntegerField
Fecha de inicio de prueba	DateField
Iteración	ComboBox
Tiempo de verificación	IntegerField
Modificar	Booton
Cancelar	Booton
Para cada responsabilidad:	
Nombre:	Modificar
Descripción:	Permite realizarle cambios a una planificación.
Nombre:	Cancelar
Descripción:	Permite cancelar la opción.

Nombre: Eliminar planificación	
Tipo de clase: Interfaz	
Atributo	**Tipo**
Si	Booton
No	Booton
Para cada responsabilidad:	
Nombre:	Si
Descripción:	Permite eliminar una planificación.
Nombre:	No

Descripción:	Permite cancelar la opción.

Nombre: Ver planificación	
Tipo de clase: Interfaz	
Atributo	**Tipo**
Departamento	ComboBox
Proyecto	ComboBox
Módulo	ComboBox
Tipo de prueba	CharField
Iteración	Textbox
Fecha de inicio	DateField
Cantidad de máquina	Textbox
Cantidad de personas	Textbox
Casos de uso simple	Textbox
Casos de uso medio	Textbox
Casos de uso alto	Textbox
Tiempo de profesor	Textbox
Tiempo de respuesta	Textbox
Tiempo de verificación	Textbox
Fecha de fin	DateField
Volver	Booton
Para cada responsabilidad:	
Nombre:	Volver
Descripción:	Permite volver a la opción anterior.

Anexo 3

Caso de Uso Gestionar Planificación

Sección: Ver Planificación

Id del escenario	EC 10
Escenario	Ver planificación
Botón 4 (Volver)	NA
Respuesta del Sistema	Muestra los datos de la planificación y regresa a la vista de "Listado de planificaciones".
Resultado de la Prueba	Satisfactorio

Sección: Eliminar Planificación

Id del escenario	EC 11	EC 12
Escenario	Eliminar planificación	Flujo alterno "Eliminar planificación"
Botón 5 (Si)	NA	
Botón 6 (No)		NA
Respuesta del Sistema	El sistema elimina la planificación.	Regresa a la vista de "Listar planificación".
Resultado de la Prueba	Satisfactorio	Satisfactorio

Sección: Próxima iteración

Id del escenario	EC 1	EC 2	EC 3

Escenario	Próxima iteración	Cancelar operación	Existen datos incompletos
Variable 1 Departamento	SAS		SAS
Variable 2 Proyecto	SACCEM		SACCEM
Variable 3 Módulo	Radiofísica		Radiofísica
Variable 4 Nombre de prueba	funcional		
Variable 5 Descripción	Se le hará pruebas a este módulo, para ver su eficacia.		
Variable 6 Cantidad de casos de uso simple	10		fd
Variable 7 Cantidad de casos de uso medio	10		fd
Variable 8 Cantidad de casos de uso alto	10		df
Variable 9 Cantidad de máquinas	10		s
Variable 10 Cantidad de	10		f

personas			
Variable 11 Fecha de inicio	15/6/2010		15/6/2010
Variable 13 Iteración	2da		
Variable 14 Tiempo de respuesta	1		
Variable 15 Tiempo de profesor	1		
Variable 16 Tiempo de verificación	1		
Botón 7 (Registrar)	NA		NA
Botón 8 (Cancelar)		NA	
Respuesta del Sistema	Se registra la planificación. Y muestra el mensaje de información “Se ha registrado satisfactoriamente”.	Cancela la acción de registrar. Vuelve a la página “Listado de planificación”.	Muestra el mensaje de información “Hay errores en el formulario”. Y un indicador al lado de los campos incorrectos o incompletos
Result ado de la Prueba	**Satisfactorio**	**Satisfactorio**	**Satisfactorio**