

Universidad de las Ciencias Informáticas

FACULTAD 6



TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Tema: Componente de Gestión de Planificación Genérica

Autores: Darinca Sueiro Rosales
Karel Ricardo Armán

Tutores: Ing. Yoandrys S. Pacheco Jerez
Ing. Guillermo Luzua Farías

La Habana, Cuba
Junio de 2013
“Año 55 de la Revolución”

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos al centro GEYSED de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Darinca Sueiro Rosales

Firma del Autor

Karel Ricardo Armán

Firma del Autor

Ing. Yoandrys S. Pacheco Jerez

Firma del Tutor

Ing. Guillermo Luzua Farías

Firma del Tutor

DATOS DE CONTACTO

Tutor: Ing. Yoandrys S. Pacheco Jeréz (ypacheco@uci.cu)

Profesor instructor, graduado de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI) curso 2007. Miembro del Departamento de Ciencias Básicas e Ingeniería de Software; imparte la asignatura de Práctica Profesional II y Física. Se desempeñó como arquitecto del grupo de Desarrollo de componentes y tecnologías durante 2 años y actualmente es desarrollador de la plataforma SIAV, ambos del centro de desarrollo GEYSED de la Facultad 6.

Tutor: Ing. Guillermo Luzua Farías (gluzua@uci.cu)

Graduado de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI) curso 2012, alcanzó el Nivel Superior en el año 2012 en la Universidad de las Ciencias Informáticas, La Habana, Cuba. Cuenta con 2 años de experiencia como programador en la línea de las transmisiones de archivos multimedia del Centro Geoinformática y Señales Digitales (GEYSED) de la Universidad de las Ciencias Informáticas. Actualmente se desempeña como programador del módulo de planificación genérica del departamento.

Agradecimientos

*A nuestros padres por apoyarnos y querernos siempre,
Por darnos su amor y cariño incondicional
Y por estar acompañandonos en nuestras debilidades.
A nuestras hermanas por simplemente estar tan cerca de nosotros.*

*A nuestros abuelos,
Aquéllos que se encuentran con nosotros y a los que no,
Por su amor .
A todos los que se han mantenido a nuestro lado
Durante este tiempo.*

*A nuestros tutores,
Por ayudarnos a realizar nuestros sueños.
Y a los amigos
Que ha sabido estar cerca de nosotros
Para levantarnos bien alto.*

*A todas aquellas personas que nos apoyaron,
Que confiaron en nosotros,
Que sin pedir nada a cambio,
Fueron capaces de estendernos su mano.*

Dedicatoria

A nuestros padres.

*Por habernos dado todo lo que somos como personas, por habernos
inculcando los principios, los valores, la perseverancia y
el empeño, acompañándonos siempre de una gran dosis de amor,
Sin pedir nunca nada a cambio.*

A nuestras hermanas.

Por su cariño incondicional.

A nuestros abuelos.

*Por tanta comprensión que siempre nos han dado
Y algunos que no se encuentren físicamente.*

A nuestras familias.

Por ayudarnos siempre a alcanzar nuestras metas.

A nuestros amigos.

Por apoyarnos aún en los momentos más difíciles.

RESUMEN

El proceso de planificación dentro del departamento Señales Digitales de la facultad 6 se realiza para cada producto existente en el departamento de forma individual. La implementación de esta funcionalidad por separado implica la utilización de recurso humano para cada uno de los proyectos con los que cuenta el departamento. Actualmente no existe un sistema que sea capaz de interrelacionar este proceso de planificación de manera que centralice y estandarice la ejecución de las planificaciones. En la presente investigación se muestra como objetivo principal el desarrollo de un componente genérico capaz de centralizar y estandarizar la planificación en el centro. El Componente de Gestión de Planificación Genérica está compuesto por dos módulos, uno llamado módulo de Planificación, encargado de la creación de las planificaciones y acciones, y otro llamado módulo de Ejecución, que es el encargado de la ejecución de estas planificaciones. Todo el proceso de desarrollo fue guiado por la metodología de Proceso Unificado de Desarrollo. Incluye un estudio de las tecnologías y herramientas existentes. La creación de este componente permitirá agilizar el proceso de planificación y ejecución de las planificaciones para el Departamento de Señales Digitales. El mismo se realizó con tecnologías libres.

PALABRAS CLAVE: Componente, planificación, ejecución, tarea.

Introducción	1
CAPÍTULO 1: Fundamentación teórica.	5
1.1 Conceptos asociados al dominio del problema.	5
1.1.2 Planificación.	5
1.1.3 Control.	5
1.1.4 Gestión.	6
1.2 Descripción del objeto de estudio.	6
1.3 Situación problemática.	7
1.4 Análisis de las soluciones existentes.	8
1.4.1 GESPRO.	8
1.4.2 Sistemas de Planificación de Recursos Empresariales.	9
1.4.3 Sistema de Planificación y Control de la Producción.	10
1.5 Metodología, lenguajes y herramientas.	12
1.5.1 Metodología de desarrollo de software.	12
1.5.2 Lenguaje de modelado.	13
1.5.3 Herramienta CASE.	14
Lenguajes de programación del módulo Planificación.	15
JavaScript 1.5	15
HTML 5	15
PHP 5.3	16
CSS 3	16
Framework del módulo Planificación.	16
Symfony 2.1	16
Dojo 1.8	17
Entorno de desarrollo integrado del módulo Planificador.	17
Netbeans 7.1	18
Lenguaje de programación del módulo Ejecutor.	18
C++	18
Framework del módulo Ejecutor.	18
Qt 4.8.1	18
Entorno de desarrollo integrado del módulo Ejecutor.	18
Qt-Creator 2.4.1	18
Servidores de aplicaciones.	19

Sistema gestor de base de datos.....	19
Servidor Web.	19
Servicios Web (SW).....	20
1.6 Conclusiones parciales.....	21
CAPÍTULO 2: Descripción de la solución propuesta.....	22
2.1. Modelo de dominio.	22
2.1.1 Glosario de términos del dominio.....	23
2.2 Modelo del sistema.....	24
2.2.1 Requisitos.....	24
Requisitos funcionales.	24
Requisitos no funcionales.	26
2.2. Diagrama de casos de uso del sistema.	29
2.3.1 Descripción de actores del sistema.	31
2.3.2 Descripción de los casos de usos del sistema.....	31
2.4 Conclusiones parciales.	37
CAPITULO 3: Diseño e implementación de la solución propuesta.....	38
3.1 Patrones.	38
3.1.1 Patrones de arquitectura.....	38
3.1.2 Patrones de diseño.	39
3.2 Diseño de interfaz.....	40
3.3 Modelo de diseño.	41
3.4 Modelo de implementación.	43
3.5 Modelo de despliegue.....	44
3.5.1 Descripción de los nodos.	45
3.6 Modelo de datos.	45
3.6.1 Diagrama Entidad-Relación.	45
3.6.2 Descripción de las tablas.	46
4.1 Conclusiones parciales.....	46
CAPITULO 4: Validación y pruebas.....	47
4.2 Modelo de pruebas.	47
4.3 Pruebas de caja negra.....	47
4.4 Pruebas de caja blanca.	52
4.5 Conclusiones parciales.....	55
Recomendaciones.....	57

Referencias bibliográficas.....	58
Bibliografía	¡Error! Marcador no definido.
Acrónimos	61
Anexos	62

Índice de figuras

Figura 1. Comparación de algunas soluciones existentes.....	12
Figura 2. Diagrama de dominio.....	23
Figura 3. Diagrama de casos de uso del sistema para el módulo de Planificación.....	30
Figura 4. Diagrama de casos de uso del sistema para el módulo ejecución.....	30
Figura 5. Diagrama de clases del diseño para el CU Gestionar objetos de planificación.	41
Figura 6. Diagrama de clases del diseño para el CU Gestionar acciones.	42
Figura 7. Diagrama de clases del diseño para el módulo de Ejecución.....	42
Figura 8. Diagrama de implementación del módulo de Ejecución.	43
Figura 9. Diagrama de implementación del módulo de Planificación.....	44
Figura 10. Diagrama de despliegue.	44
Figura 11. Diagrama entidad relación.....	45
Figura 12. Representación del código del método vigilante.	53
Figura 13. Grafo del flujo asociado al método vigilante.	53

Índice de tablas

Tabla 1. Descripción de los actores del sistema.....	31
Tabla 2. Descripción del caso de uso Gestionar objetos de planificación.....	31
Tabla 3. Descripción del CU Chequear planificación.....	36
Tabla 4. Escenarios del CU Gestionar objeto de planificación.....	48
Tabla 5. Descripción de las variables del CU Gestionar objeto de planificación.	49
Tabla 6. Matriz de datos del CU Gestionar objeto de planificación.	50
Tabla 7. Caminos del grafo asociado del método vigilante.	54
Tabla 8. Caso de prueba para el camino No 1.	54

Introducción

La planificación ha sido un punto importante para cualquier empresa. La misma es un método que minimiza el riesgo en cualquier actividad. Planificar, implica estudiar todo el proceso en el cual se estará trabajando, es proponer metas y objetivos que lleven al éxito en el resultado final.

La planificación de tareas permite elaborar de forma ordenada las acciones que posteriormente serán ejecutadas. Es por eso que a principios, las empresas o instituciones necesitaban contar con una persona que fuera capaz de manejar toda la información, y que a la vez pudiera planificar determinadas acciones o tareas. Con el desarrollo de la tecnología y el aumento de volumen de información que se maneja, surge la necesidad de automatizar este proceso de planificación. Es por ello que un sistema informático que facilite este trabajo sería un gran apoyo para el control de la información.

La planificación de tareas en los proyectos constituye un punto importante para la construcción de productos de software, es por eso que la misma tiene como objetivo obtener una distribución de las actividades en el tiempo y una utilización de los recursos que minimice el coste del proyecto cumpliendo sus expectativas. La misma es una programación de actividades y una gestión de recursos para obtener un objetivo de coste por el cliente. La ejecución de estas tareas planificadas es vital para algunas empresas o instituciones y es uno de los campos más activos en la informática. Esto evita la necesidad de contar con una persona encargada de mandar a ejecutar las tareas, cuando se cuenta con un sistema informático que pueda gestionarlo.

Los centros de desarrollo se crean para formar especialistas y profesionales que apoyen e impulsen la actualización y modernización tecnológica, dentro del campo de las ciencias informáticas. Cuba cuenta con la UCI, donde se desarrollan una serie de proyectos, cada uno de ellos tiene funciones específicas, objetivos y metas que cumplir. La misma posee un conjunto de herramientas que posibilitan toda la gestión de los proyectos, además de permitir la planificación de desarrollo de un producto de software. En la UCI actualmente existen diversas soluciones que se caracterizan por la automatización de la información y un adecuado manejo de las planificaciones de los recursos con los que se cuenta.

En la actualidad los productos que se implementan en el Departamento de Señales Digitales del centro Geoinformática y Señales Digitales (GEYSED), incorporan dentro de sus soluciones elementos que tributan a la creación y desarrollo de planificaciones con diferentes objetivos, específicos para cada solución. Este departamento está compuesto

por varios productos: la Plataforma de Televisión Informativa PRIMICIA, el Sistema de Video Vigilancia SURIA (SURIA) y el Sistema de Transmisión y Monitorización de Canales Virtuales (STCV). Cada uno de ellos implementa por independiente un módulo de planificación.

La plataforma PRIMICIA realiza la planificación de dos formas, por bloques y por noticias. La planificación por bloques cuenta con noticias, los mismos tienen parámetros como son: fecha de inicio, fecha de fin, hora de inicio y hora de fin. Cada bloque funciona como un recipiente, que dentro le cabrán varias noticias y para un día puede existir más de un bloque. La otra planificación es la que se realiza por noticias. Una noticia tiene como parámetros: la fecha de inicio, fecha de fin, hora de inicio, hora de fin y la duración. Este sistema utiliza como recurso la noticia y la acción principal es publicar noticia, dígame imágenes, audio y videos y los bloques tienen asociadas varias noticias.

En el caso del Sistema de Video Vigilancia SURIA, realiza el proceso de planificación por reglas, cada regla tiene asociados varios parámetros como son: fecha de inicio, fecha de fin, hora de inicio y hora de fin, y otros parámetros como son si la planificación se efectuará el día entero, si se repetiría la semana, el mes o en una fecha específica. Los recursos identificados en este sistema son las cámaras y la acción principal es grabar; Esta grabación tiene diferentes iniciadores entre los que se pueden mencionar grabar por un determinado rango de tiempo o grabar aleatoriamente.

El Sistema de Transmisión y Monitorización de Canales Virtuales realiza la planificación de materiales audiovisuales. En el entorno de este sistema una planificación tiene asociado un espacio y este tiene medias, aunque la planificación pudiera tener medias independientes al igual que un espacio. Una media tiene parámetros tales como: fecha de inicio, fecha de fin, hora de inicio, hora de fin y duración. Un canal tiene una programación o parrilla de transmisión como también se le puede llamar, que sería la planificación de un canal. Este producto tiene como recursos los canales y las medias asociadas a estos canales. Como acciones principales que se identifican están la de transmitir y monitorizar. De manera unificadora e identificando las principales características de la planificación dentro del departamento se pueden identificar dos tipos de planificaciones; el primero no se ha descrito porque es el común para todos los proyectos. Es la planificación de los recursos humanos, los tiempos de máquinas y tareas nativas de cada uno de los proyectos. Estas tareas están acorde con la metodología de desarrollo que se emplea para desarrollar cada producto y quedan gestionadas de manera independiente por proyecto. La otra planificación es la que se refiere a las acciones dentro de los productos

que se realizan en los proyectos y que fueron descritas anteriormente. Esta planificación si se puede centralizar e identificar procesos comunes en sus características. La misma se puede describir en: una planificación de veinte cuatro horas puede contener bloques o espacios; y recursos de manera independiente. Así mismo los bloques son contenedores de recursos. Estos recursos y bloques tienen características comunes tales como: nombre, hora de inicio, hora de fin, fecha de inicio, fecha de fin y periodicidad.

Para desarrollar todos estos procesos de planificación comunes en los diferentes productos se utiliza un personal diferente por cada proyecto, lo que es un inconveniente desde el punto de la gestión y distribución óptima del personal con que cuentan los proyectos para el desarrollo. Por lo que se hace necesario la centralización y optimización de los recursos para lograr reducir el tiempo de desarrollo de los productos, además de estandarizar el proceso de planificación para los mismos y lograr que la ejecución de las planificaciones sea de manera automática.

Atendiendo a la situación antes descrita, se arriba al **problema de investigación**: ¿Cómo centralizar el proceso de planificación en los productos que se implementan en el Departamento de Señales Digitales?

Se define como **objeto de estudio**: Los procesos de planificación dentro del desarrollo de software.

Teniendo como **objetivo general**: Desarrollar un componente para la gestión de las planificaciones genéricas para los productos del Departamento de Señales Digitales.

Aplicado al **campo de acción**: El proceso de planificación en el desarrollo de software en el Departamento de Señales Digitales.

Se presenta como **idea a defender** que: Con la implementación de un componente de gestión de planificación genérica, se centralizará la planificación de los productos que se desarrollan en el departamento de Señales Digitales.

Para dar cumplimiento al objetivo general se definen las siguientes **Tareas de la investigación**:

- Revisión bibliográfica de los enfoques teóricos relacionados con la planificación de recursos en una entidad.
- Descripción de las soluciones existentes que gestionan planificaciones.
- Selección de las herramientas y tecnologías a utilizar en el desarrollo del Componente de Gestión de Planificación Genérica.
- Confección de los artefactos y la documentación según la metodología de desarrollo seleccionada.

- Implementación del Componente de Gestión de Planificación Genérica.
- Validación del Componente de Gestión de Planificación Genérica.

Para obtener los conocimientos necesarios que hicieron posible el cumplimiento del objetivo trazado en la investigación, se pusieron en práctica diferentes **métodos científicos** tanto teóricos como empíricos (**Álvarez, 2000**):

Métodos teóricos:

Histórico-lógico: se empleó para realizar un estudio analítico de la trayectoria histórica de los procesos de planificación como parte del desarrollo de una aplicación de software, y de cómo este ha influido históricamente en el proceso de gestión de la planificación en la universidad. Además se investigó sobre la información referente a las soluciones existentes a lo largo del desarrollo de la planificación.

Analítico-sintético: analiza las teorías planteadas y documentos, para luego extraer los elementos más importantes que se relacionan con los procesos de planificación como parte de una aplicación de software y adaptarlo a los productos del Departamento de Señales Digitales.

Modelación: en el desarrollo de la investigación es necesario el modelado de la arquitectura que se propone, es por eso que se utiliza este método para crear modelos con vistas que favorezcan la comprensión de la misma.

Métodos empíricos:

Análisis documental: en la revisión de la literatura especializada para consultar la información necesaria en el proceso de investigación.

Método observador: el mismo permitió identificar las variables de entradas que contiene cualquier planificación.

CAPÍTULO 1: Fundamentación teórica.

En el presente capítulo se estarán abordando los aspectos teóricos relacionados con las planificaciones genéricas. Se especifican una serie de conceptos asociados al dominio del problema, los cuales tienen como objetivo facilitar un mejor entendimiento de la situación problemática, creando así una base de conocimientos del tema. Además cuenta con un análisis de las principales herramientas y tecnologías con las cuales se trabajaron.

1.1 Conceptos asociados al dominio del problema.

Para un mejor entendimiento de las características de la gestión de planificación, su funcionamiento y relación con un componente genérico es ineludible enunciar varios conceptos dentro de los que resaltan: planificación, control y gestión.

1.1.2 Planificación.

La planificación, es el proceso metódico diseñado para obtener un objetivo determinado. Según **(Robalino, 2012)** en el sentido más universal, implica tener uno o varios objetivos a realizar junto con las acciones requeridas para concluirse exitosamente.

La planificación es algo esencial en el trabajo de un gestor porque le permitirá conseguir sus objetivos. Planificar supone analizar y estudiar los objetivos propuestos así como la forma en la que se van a conseguir. La misma es una herramienta de acción para decidir qué se va hacer y por qué, dando lugar a la creación de un plan que guía el trabajo.

1.1.3 Control.

La **(RAE, 2013)** define que control es:

- Comprobación, inspección, fiscalización, intervención.
- Dominio, mando, preponderancia.

Dando como conclusión que el control no es más que la comprobación de algo, ya sea aplicado a una empresa o institución. Es la forma de mando a chequear que las actividades o tareas sean cumplidas correctamente, o simplemente velar por la organización. En este caso, en la presente investigación se provee el control específicamente sobre las planificaciones, cualquiera sea el negocio en que se implante este componente.

1.1.4 Gestión.

Según la **(RAE, 2013)** gestión es:

- Acción y efecto de gestionar.
- Acción y efecto de administrar.

La gestión no es más que la forma de realizar un conjunto de trámites que se van a llevar a cabo para resolver un asunto o concretar un proyecto. Es importante destacar que la gestión tiene como objetivo primordial aumentar los resultados óptimos de una industria o empresa.

1.2 Descripción del objeto de estudio.

En la actualidad los procesos de planificación dentro del desarrollo de software se han convertido en un importante punto para el avance de los mismos, siendo así de mucha utilidad para lograr un producto de calidad y satisfacción para el cliente.

La planificación incluye la estimación los recursos, negociar los objetivos, crear una agenda de proyecto e identificar y analizar los riesgos. **(Navarro, 2006)** plantea que los componentes principales que debe tener un plan de desarrollo para un producto de software son los siguientes:

- Estimación de costos: se deben identificar los costos de cada uno de los componentes.
- Programación del tiempo: se indica cuando empieza y cuando termina cada fase.
- Planificación del personal: se identifican cuantas personas se necesitan para el desarrollo del mismo.
- Estructuración del equipo de trabajo: se establece la composición de cada grupo de trabajo.
- Verificación y control de la calidad: verifica constantemente si los componentes del proyecto están cumpliendo con los requisitos establecidos.
- Gerencia de configuración: el plan debe indicar de forma específica los mecanismos que se utilizarán para atender la necesidad y solicitudes de cambio en el proyecto.
- Monitoreo del proyecto: indica cómo la gerencia monitoreará las actividades del proyecto y se encargará de que se cumpla el plan de trabajo establecido.
- Manejo o gestión de riesgos: el plan debe establecer qué se hará en casos de

retraso o que ocurrirá si se pierde uno o varios miembros del personal.

El objetivo de la planificación de proyecto es proporcionar un marco de trabajo que permita hacer estimaciones de recursos, coste y planificación temporal. La planificación es un proceso de toma de decisiones para alcanzar un objetivo trazado.

Planificar favorece el desarrollo de software, es por ello que para la obtención de un producto de calidad, se hace necesario hacer una correcta planificación que posibilite obtener un buen desarrollo. La planificación se hace cada vez más importante, influye en todo el proceso del desarrollo del software.

1.3 Situación problemática.

En la actualidad el proceso de planificar es muy empleado en diferentes áreas, dentro de las que se incluyen los centros educacionales, instituciones y empresas. Este proceso trae como consecuencia ahorro en tiempo, debido a que permite de manera ordenada cumplir con el tiempo establecido los parámetros planificados. Con el surgimiento de la informática y el constante avance tecnológico que existe en la actualidad, se han desarrollado sistemas informáticos capaces de llevar el control de diferentes recursos en una entidad determinada. La posibilidad de realizar este proceso de forma genérica ocupa un punto importante a tener en cuenta, debido a que permite la realización de actividades planificadas sin importar el tipo de recurso y se ajusta a las necesidades de cada entidad.

El Departamento de Señales Digitales del centro GEYSED de la Facultad 6 cuenta con soluciones de software que se engloban en los diferentes proyectos existentes en el área; cada proyecto presenta soluciones diferentes, pero presentan en común la necesidad de realizar la planificación de recursos tales como: canales virtuales, archivos multimedia y cámaras de video vigilancia. Actualmente el uso de recursos humanos no es optimizado debido a la necesidad de utilizar esta fuerza de trabajo en la implementación de módulos que gestionen la planificación de los recursos antes mencionados. Esto trae como consecuencia la implementación, en muchos casos de un mismo proceso en los diferentes proyectos con el recurso humano de cada equipo de trabajo.

El desarrollo de un componente genérico es una solución propuesta por el Departamento. El mismo fue concebido para el empleo en los proyectos y como una alternativa más fiable de los sistemas que existen. Un componente genérico se refiere a una aplicación que maneje todas las planificaciones del centro, que sea capaz de ser adaptable a cualquier negocio. Para la implementación de este componente se tendrán en cuenta las funcionalidades de los módulos de planificación de cada uno de los proyectos, teniendo

como base las soluciones que ya poseen los productos para realizar la planificación y logrando que el proceso de planificación sea genérico y adaptable a cada negocio que lo necesite dentro de los estudiados u otros.

Como se evidenció en la explicación anterior, existen varias soluciones en el departamento que implementan sus planificaciones por separadas. Las mismas no son factibles, ya que el departamento quisiera que el proceso estuviese ordenado y centralizado, pero que a su vez posibilite la realización de las operaciones pertinentes a cada proyecto. Por ende es de gran importancia una herramienta que pueda llevar el control de todas las planificaciones en los proyectos.

Después de haber efectuado el análisis de la situación problemática se decide realizar un componente que permita la gestión de las planificaciones de diferentes recursos de forma diferente; esto permitirá establecer un componente para la realización de las diferentes planificaciones en el Departamento de Señales Digitales, y que pueda ser utilizado por cualquier otra entidad. Para lograr dar solución a todas las necesidades existentes en cada proyecto, se realizó un estudio sobre las necesidades que tiene cada uno, lo cual permitirá dar solución a la problemática existente y cuyo resultado será una propuesta que integre a los proyectos asociados al Departamento.

1.4 Análisis de las soluciones existentes.

La gestión de la planificación ha constituido para el mundo un punto muy importante, porque muchas empresas o instituciones necesitan de un software que facilite el trabajo de una serie de personas. Es por ello que a nivel mundial se han desarrollado un conjunto de sistemas capaces de solucionar este problema. En Cuba, entre los que se pueden mencionar y destacar se encuentra el módulo de gestión de proyectos (GESPRO), y a nivel mundial el Sistema de Planificación de Recursos Empresariales (ERP) y el Sistema de Planificación y Control de la Producción (SPCP).

1.4.1 GESPRO.

Es un software que permite la gestión de proyectos, es bastante amplio ya que abarca todo lo relacionado con la gestión, planificación y control. En él se puede ver la planificación, ejecución, gestión, logística, entre otras opciones de la gestión de un proyecto.

Se trata de una solución de código abierto de uso extendido, tanto en el ámbito público como privado, y destinada a facilitar el control de proyectos, canalizando la comunicación

entre los distintos actores participantes y ofreciendo además un amplio abanico de posibilidades de personalización. La misma incluye otras herramientas tales como: calendario de actividades, foro, visor del repositorio de control de versiones y control de flujo de trabajo basado en roles e integración con correo electrónico entre otras.

La herramienta está escrita usando el *Framework Ruby on Rails*. La misma es de software libre y de código abierto. Esta tiene un módulo de gestión de planificaciones. El mismo se ajusta al entorno, pero no presenta la ejecución de tareas planificadas, siendo estas características a tener en cuenta para el módulo que se necesita en el departamento. Por lo que se continúa con el estudio y análisis de otras soluciones existentes tales como los Sistemas de Planificación de Recursos Empresariales (ERP por sus siglas en inglés).

1.4.2 Sistemas de Planificación de Recursos Empresariales

Los ERP son sistemas de información gerenciales que integran y manejan muchos de los negocios asociados con las operaciones de producción y de los aspectos de distribución de una compañía comprometida en la producción de bienes o servicios.

Según **(Turn, 2009)** la Planificación de Recursos Empresariales es un término derivado de la Planificación de Recursos de Manufactura (MRPII por sus siglas en inglés) y seguido de la Planificación de Requerimientos de Material (MRP por sus siglas en inglés). Los sistemas ERP típicamente manejan la producción, logística, distribución, inventario, envíos, facturas y la contabilidad para la compañía que lo utilice. Los ERP pueden intervenir en el control de muchas actividades de negocios como ventas, entregas, pagos, producción, administración de inventarios, calidad de administración y la administración de recursos humanos.

En **(Rivera, 2008)** se expone que los ERP están funcionando ampliamente en todo tipo de empresas modernas. Todos los departamentos funcionales que están involucrados en la operación o producción están integrados en un solo sistema. Además de la manufactura o producción, almacenamiento, logística e información tecnológica, incluyen además la contabilidad, y suelen incluir recursos humanos, las herramientas de mercadotecnia y administración estratégica.

Algunas de las principales características de los ERP son:

- Optimización de los procesos empresariales.
- Acceso a toda la información de forma confiable, precisa y oportuna (integridad de datos).

- La posibilidad de compartir información entre todos los componentes de la organización.
- Eliminación de datos y operaciones innecesarias de reingeniería.

El propósito fundamental de un ERP es otorgar apoyo a los clientes del negocio, tiempos rápidos de respuesta a sus problemas, así como un eficiente manejo de información que permita la toma oportuna de decisiones y disminución de los costos totales de operación.

Realizado el análisis de los sistemas ERP se puede destacar que son sistemas buenos en cuanto al manejo de la producción, logística y distribución, maneja la gestión en tiempo real, pero tiene la desventaja que para muchas empresas es casi imposible pagar el costo de las licencias, ya que lograr su mantenimiento, es difícil, y más cuando se trata de un sistema que crece rápidamente. Además que no representa una solución genérica.

Estos sistemas presentan gestión de tareas, pero no se ajustan al entorno y tampoco presenta ejecución de tareas, por eso no presenta una alternativa genérica. En este caso podemos emplear algunos elementos del mismo relacionado con la gestión de tareas.

Además de los ERP también se pueden mencionar a los Sistema de Planificación y Control de la Producción (SPCP), los cuales igualmente gestionan las planificaciones pero con otro enfoque.

1.4.3 Sistema de Planificación y Control de la Producción.

Los SPCP, están formados por un conjunto de niveles estructurados (jerárquicamente) de planificación que contemplan tanto los Planes Agregados, los Planes Maestros, la Gestión de Materiales, así como, los niveles de Ejecución o Gestión de Taller.

En la actualidad existen diferentes alternativas de SPCP, acorde a las características propias del proceso productivo (variedad, volumen de producción, complejidad del producto, nivel técnico y tecnológico), cuyo objetivo es controlar el proceso de producción dentro del sistema empresarial.

Cuando se habla de planificación y control de la producción, se suele hacer referencia a métodos y técnicas que se pueden subdividir en aquellas dirigidas a planificar y controlar "operaciones de procesos" y "operaciones de proyecto."

De los antes mencionados se centró la atención en el primero mencionado. **(Escobar, 2004)** plantea que la implantación de un sistema MRP debe tener en cuenta la educación de los empleados, la formación y el convencimiento para subsanar los posibles errores. La introducción del MRP provoca mejoras en la programación y gestión de inventarios,

Capítulo 1: Fundamentación teórica

pero no provee muchas cosas que algunos esperarían. Consecuentemente, los fabricantes comenzaron una aproximación más amplia: el MRP II. El mismo no reemplaza al MRP y tampoco es una mejora de él, sino que representa un esfuerzo por ampliar el espectro de la planificación de recursos para la fabricación e involucrar a otras áreas funcionales de la empresa en el proceso de planificación.

Algunas de las principales diferencias entre los MRP y los MRP II dadas por **(Escobar, 2004)** son:

MRP

- Planifica las necesidades de aprovisionarse de materia prima (programar inventarios y producción), basado en el plan maestro de producción, como principal elemento.
- Sólo abarca la producción.
- Surge de la práctica y la experiencia de la empresa (no es un método sofisticado).

MRP II

- Planifica la capacidad de recursos de la empresa y control de otros departamentos de la empresa, basado como principal punto de apoyo en la demanda, y estudios de mercado.
- Abarca más departamentos, no sólo producción si no también el de compras, calidad, financiero y otros.
- Surge del estudio del comportamiento de las empresas (método sofisticado).
- Sistema de bucle cerrado (permite la mejora continua en cuanto a la calidad de los productos) para, en caso de error replanificar la producción.
- Mejor adaptación a la demanda del mercado.
- Mayor productividad.

Realizado el estudio de las soluciones existentes se puede llegar a la conclusión que estas aplicaciones solucionan parcialmente algunos elementos del problema presente en el trabajo que se lleva a cabo. También que alguna de estas soluciones, son dependientes de sistemas operativos propietarios y que hay que pagar licencias para su uso. Además de no ser soluciones en la que se puedan gestionar las planificaciones de manera genérica para cualquiera de los productos del Departamento de Señales Digitales e iniciar la ejecución automática de estas planificaciones. Tomando en consideración las desventajas expuestas que inciden en el uso de las soluciones anteriormente analizadas, para la planificación de los productos del departamento, se puede valorar la importancia de contar con una solución personal que se ajuste a los requisitos de los productos antes

mencionados.

Así mismo, esta solución debe tener en cuenta los procesos que intervienen en la gestión de las tareas. Se le agrega al anteriormente mencionado la ejecución de dichas tareas como otro de los procesos esenciales para el entorno donde se va a utilizar la solución. Garantizando que todos los negocios que la utilicen puedan gestionar sus planificaciones de manera centralizada y ordenando los procesos que componen la misma.

A continuación la Figura 1 describe de manera general algunas de las soluciones existentes.

	Gestión de planificaciones	Ajuste al entorno	Ejecución de tareas
HRM	Sí	No	No
GESPRO	Sí	Sí	No
SPCP	Sí	Sí	No

Figura 1. Comparación de algunas soluciones existentes.

1.5 Metodología, lenguajes y herramientas.

Para obtener la solución de la problemática existente se realizó una selección de un conjunto de herramientas y tecnologías, las que se utilizaron para confeccionar dicha solución. A continuación, en el siguiente epígrafe se fundamentará la elección de las herramientas que se utilizan en la solución final. Así como la argumentación de las decisiones tomadas en cuanto a la estructura de la misma y los lenguajes de modelado y programación que se utilizaron para modelar y escribir la aplicación respectivamente. De igual manera se precisa la metodología de software que guía todo los procesos que se generaron en el desarrollo de la solución.

1.5.1 Metodología de desarrollo de software.

Una metodología es una guía que define reglas o métodos a seguir para todos los procesos que intervienen en la construcción de un software. La misma, según **(Pérez, 2008)** surge con la necesidad de utilizar procedimientos, técnicas, herramientas y soporte documental para el desarrollo de software. Esta brinda documentación importante referente a las políticas y procedimientos que se van a seguir en el desarrollo de un software, llevando el control de todos los procesos que se realizan.

En la actualidad las metodologías de desarrollo se clasifican en pesadas o ligeras. Esta clasificación se establece por la rigurosidad de cada una en sus procesos y la generación de artefactos. Como metodología pesada se puede mencionar el Proceso Unificado Desarrollo (RUP por sus siglas en inglés).

RUP es una metodología tradicional que hace énfasis en la planificación y control del proyecto, se adapta fácilmente a las condiciones del proyecto mediante su configuración previa a aplicarse. Es un proceso de software genérico que puede ser utilizado para gran cantidad de tipos de sistemas de software, y que además define un conjunto de actividades que guían los esfuerzos de las personas implicadas en el proyecto, asignando tareas y responsabilidades para asegurar la producción de software de muy alta calidad, que satisfaga las necesidades de los usuarios finales, dentro de un calendario y presupuesto predecible.

El colectivo de autores **(Jacobson, y otros, 2000)** definen que en RUP se han agrupado las actividades en grupos lógicos definiéndose nueve flujos de trabajo principales. Los seis primeros son conocidos como flujos de ingeniería y los tres últimos como flujos de apoyo.

Además, RUP posee entre sus características principales que es guiado por Casos de Uso (CU), centrado en la arquitectura e iterativo e incremental.

Se escoge RUP como metodología; la misma es ideal para realizar la documentación de todos los procesos que intervienen en la construcción del Componente de Gestión de Planificación Genérica, por ser adaptable a las necesidades del desarrollo. Agregado a esto se puede mencionar que RUP es la metodología empleada en el Departamento de Señales Digitales y se desea seguir un estándar en todos los artefactos generados. Además se integra excelentemente con el Lenguaje Unificado de Modelado (UML).

1.5.2 Lenguaje de modelado.

El desarrollo de software es un proceso que cuenta con diferentes actividades, entre las que se pueden destacar: el modelado del negocio, el diseño arquitectónico, el diseño detallado de diagramas, la programación y las pruebas de la aplicación. UML es un lenguaje muy utilizado para analizar, diseñar y definir la solución de implementación de un software. Este es un lenguaje gráfico para visualizar, construir y documentar un sistema.

Un lenguaje de modelado proporciona un vocabulario y reglas para permitir una comunicación entre todas las partes que intervienen en el desarrollo de un software. En este caso, UML se centra en la representación gráfica de cualquier sistema que se desee

implementar. Algunos de los objetivos de este lenguaje planteados por **(Orallo, 2003)** son: visualizar, especificar, construir y documentar. Aunque también está pensado para modelar sistemas complejos.

En el presente trabajo se utiliza como lenguaje de modelado UML 8.0, mediante el cual se visualizaron y especificaron las principales actividades del Componente de Gestión de Planificación Genérica, y en cada una de ellas las tareas correspondientes que se desarrollan en dichas actividades. Para elaborar las tareas con este lenguaje se empleó una herramienta de Ingeniería de Software Asistida por Ordenador (CASE¹ por sus siglas en inglés).

1.5.3 Herramienta CASE.

Las herramientas CASE son una variedad de aplicaciones informáticas que están destinadas a aumentar la productividad en el desarrollo de software, reduciendo los costos en términos de tiempo y de dinero. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como: el proceso de realizar un diseño del proyecto, cálculo de costos, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras. **(Landeros, 2004)**

En este trabajo se empleó la herramienta CASE Visual Paradigm. La misma da soporte al modelado visual con UML, así como la notación y modelado de procesos de negocio. Además que es una herramienta fácil de interactuar por parte de los usuarios. Los beneficios que trae son varios, por un lado el uso de los lenguajes visuales facilitan su asimilación y entendimiento para todos los integrantes del equipo de desarrollo del componente genérico y por otro lado la documentación del proyecto se realiza de forma organizada y guiada por los CU, siguiendo los estándares de la documentación del proyecto STCV, donde se está implementando la solución. También se puede decir que tiene una ventaja muy importante y es que destaca la efectividad y productividad que se consigue en labores de diseño arquitectónico y mantenimiento. Estas son características que apoyan la elección de esta herramienta para realizar el modelado de la solución.

1.5.4 Lenguajes de programación, *Frameworks* y entornos de desarrollo.

La confección de la solución se estructuró bajo dos tipos de aplicaciones informáticas, las aplicaciones web y las de escritorio, dando un híbrido entre estas. Esto se logró

¹ Acrónimos del inglés *Computer Aided Software Engineering*.

separando en dos módulos las responsabilidades del componente. Para la confección de los mismos se utilizaron diferentes lenguajes de programación, *Frameworks* y entornos de desarrollo, los cuales se caracterizan en los siguientes epígrafes.

Lenguajes de programación del módulo Planificación.

Según **(Rodríguez, 2011)** un lenguaje de programación es un idioma artificial que puede ser interpretado o ejecutado por máquinas. Pueden usarse para crear programas que controlen el comportamiento físico o lógico de las mismas y para expresar algoritmos con precisión. Está formado por un conjunto de símbolos, reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones.

Para la implementación del módulo Planificador se utilizaron los siguientes lenguajes de programación:

JavaScript 1.5

- Permitió a los desarrolladores crear acciones en el módulo web.
- No requiere de compilación ya que el lenguaje funciona del lado del cliente y los navegadores son los que se encargan de interpretar este código.
- Es independiente de plataforma para su interpretación, cualidad esencial en el desarrollo del módulo.
- Está basado en objetos, lo que permitió a los desarrolladores escribir de una manera sencilla y estructurada todo el código escrito en este lenguaje.
- Es un lenguaje basado en objetos.
- Es un lenguaje interpretado (script), no compilado.
- Su código se integra a las páginas escritas en HTML² lo que aumentó su portabilidad en las páginas escritas en el Planificador. **(Gonzalez, 2013)**

HTML 5

- Es el lenguaje con el que se definen las páginas web en el módulo Planificador.
- Básicamente se trata de un conjunto de etiquetas que sirven para definir el texto y otros elementos que componen una página web dentro del sistema.
- Es compatible con la mayoría de los navegadores web. **(Alvarez, 2001)**

² Acrónimos del inglés *Hypertext Transfer Protocol*.

PHP³ 5.3

- Es un lenguaje de programación del lado del servidor gratuito e independiente de plataforma.
- Posee una gran biblioteca de funciones y abundante documentación.
- Es completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en Base de Datos (DB). **(Alvarez, 2001)**

CSS⁴ 3

- Es una tecnología que nos permite crear páginas web de una manera más exacta.
- Es la mejor forma de separar los contenidos y su presentación.
- Permite incluir márgenes, tipos de letra, fondos y colores. **(Lazaro, 2001)**

Estas características mencionadas anteriormente son las principales que se tuvieron en cuenta para la utilización de estos lenguajes de programación en la implementación del módulo Planificación del componente.

Framework del módulo Planificación.

Según **(Alfonso, 2010)** expresa que un *Framework* es una estructura conceptual y tecnológica de soporte definida, normalmente, con artefactos o módulos de software concretos, con base en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio.

En la implementación del módulo Planificador se emplearon los *Frameworks* que se caracterizan a continuación según las ventajas que se identificaron para su utilización en la implementación del módulo.

Symfony 2.1

- Es un conjunto de herramientas y utilidades que simplifican el desarrollo de las aplicaciones web.

³ Acrónimos del inglés *Hypertext Preprocessor*.

⁴ Acrónimos del inglés *Cascading Style Sheets*.

- Integra las ventajas de otros *Framework* y mejorado sus antecesores para su utilización en el desarrollo de aplicaciones web utilizando PHP.
- Es un *Framework* estable que posee una amplia y verás documentación.
- Es fácil de instalar y configurar en la mayoría de plataformas.
- Es independiente del sistema gestor de base de datos que se desee utilizar.
- Utiliza el ORM⁵ Doctrine, lo que aumenta la seguridad y facilita el trabajo con los datos almacenados. **(Eguiluz, 2012)**

Dojo 1.8

- Es un *framework* escrito en el lenguaje de programación JavaScript.
- Permite a los desarrolladores realizar funcionalidades y efectos dinámicos en páginas web.
- Es fácil de integrar con varios *Frameworks* entre los que se incluye Symfony.
- Resuelve asuntos de usabilidad como pueden ser la navegación y detección del navegador.
- Posee un módulo interactivo de calendario. **(DOJO, 2012)**

Estos son los dos *Frameworks* que se utilizan en la implementación del módulo de Planificación. Las características que se expusieron son algunas de las que se tuvieron en cuenta a la hora de escoger el marco de trabajo para el desarrollo del módulo.

Entorno de desarrollo integrado del módulo Planificador.

Un Entorno de Desarrollo Integrado (IDE)⁶, según **(González, 2011)** es un programa que cuenta con las potencialidades de interpretar, compilar, depurar y diseñar interfaces. Con su utilización los desarrolladores agilizan la creación de programas. Un IDE puede soportar varios lenguajes de programación o estar específicamente diseñado para solo uno y puede estar integrado con alguna plataforma o no.

También se puede agregar que es una herramienta orientada al uso de los desarrolladores para la creación o edición de aplicaciones de software. En dependencia de las características del IDE usado serán los beneficios que se obtendrán en el desarrollo de las aplicaciones. Además los IDEs son una interfaz de abstracción de alto nivel que permite al desarrollador invocar un conjunto de instrucciones de bajo nivel en

⁵ Acrónimos del inglés *Object relational model*.

⁶ Acrónimos del inglés *Integrated Development Environment*.

los Sistemas Operativos que se ejecuten.

A continuación se detallan las principales características del IDE utilizado en la elaboración del módulo Planificador.

Netbeans 7.1

- Es un IDE libre y gratuito que está disponible para varios Sistemas Operativos.
- Permite la creación rápida de páginas web utilizando PHP.
- Permite que las aplicaciones sean desarrolladas a partir de módulos, lo cual posibilita que se pueda extender la aplicación, agregándole módulos nuevos.
- Se integra fácilmente con los *Frameworks* seleccionados.
- Tiene una comunidad que le ofrece soporte y documentación a nivel mundial.

Lenguaje de programación del módulo Ejecutor.

C++

- Es un lenguaje de programación muy potente y robusto, lo que permite trabajar tanto a bajo como alto nivel en lo que se refiere a creación de sistemas complejos.
- Tiene gran soporte para la programación orientada a objetos.
- Permite la reutilización de código de una manera productiva lo que facilitará el diseño de nuevos componentes. **(Bustamante y otros, 2004)**

Framework del módulo Ejecutor.

Qt 4.8.1

- Es un *Framework* para el desarrollo de aplicaciones multiplataforma.
- Este *Framework* utiliza C++ de manera nativa lo cual representa muchas ventajas para el desarrollo porque se acopla con el lenguaje seleccionado para la implementación del módulo.
- Posee un tratamiento de *Signals* y *Slot* que son una ventaja considerable sobre otros *Framework* ya que permiten la implementación de eventos de manera sencilla.
- Tiene un tratamiento de memoria que permite que el tiempo de ejecución de este sea menor que los programas escritos en Java o C#. **(Muñoz, 2011)**

Entorno de desarrollo integrado del módulo Ejecutor.

Qt-Creator 2.4.1

- Está diseñado para que el desarrollo con C++ sea rápido y posee un potente editor de código para este lenguaje.
- Es un *Framework* para el desarrollo de aplicaciones multiplataforma.
- Tiene una abundante documentación incluida.
- Posee un soporte para refactorización de código.
- Cuenta con un resaltado y auto-completado de código.

Servidores de aplicaciones.

La aplicación hace uso de un Sistema Gestor de Base de Datos y un Servidor Web, los que se detallan a continuación en el siguiente epígrafe.

Sistema gestor de base de datos

(González, 2000) expresa que un Sistema Gestor de Base de Datos (SGBD) es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones. Se compone de un lenguaje de definición de datos, un lenguaje de manipulación de datos y un lenguaje de consulta. Un SGBD permite definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de los mismos.

Como SGBD se seleccionó PostgreSQL el mismo posee un conjunto de ventajas que se tuvieron en cuenta a la hora de confeccionar la solución y se detallan en el siguiente sub-epígrafe.

PostgreSQL

- Suministro de almacenamiento persistente de los datos.
- Posee un sistema de copias de seguridad.
- Logra la restricción de accesos no autorizados.
- Logra estabilidad y robustez ante grandes peticiones.
- Tiene un tratamiento óptimo para peticiones concurrentes. (González, 2000)

Servidor Web.

Un Servidor Web es un programa que está diseñado para transferir hipertextos, páginas web o paginas HTML. La principal característica de los Servidores Web es almacenar los archivos de un sitio y ser publicados en una red para que pueda ser visitado por los usuarios. (Bautista, 2010)

Capítulo 1: Fundamentación teórica

Básicamente un servidor web es una computadora para guardar y publicar datos que luego serán consultados. Cuando un usuario entra en una página de Internet lo que sucede es que su navegador se comunica con el servidor, enviando y recibiendo datos que determinan qué es lo que ve en la pantalla.

En la solución se utilizó en Servidor Web Apache, el que tiene un conjunto de características que fueron de vital importancia para establecer la solución.

Apache 2.2

- Se puede utilizar en múltiples Sistemas Operativos.
- Es una tecnología gratuita de código abierto.
- Permite la implementación de los últimos y más nuevos protocolos.
- El servidor puede ser manipulado vía línea comandos, lo que hace la administración remota muy conveniente.
- Es personalizable. **(Bautista, 2010)**

Servicios Web (SW).

Los SW permiten establecer una comunicación entre aplicaciones distribuidas, que no son más que una aplicación de software identificada por una URL⁷, cuyas interfaces y vinculaciones son capaces de ser definidas, descritas y descubiertas como artefactos XML⁸. Estos soportan la interacción con otros agentes de software mediante el intercambio de mensajes basados en XML a través de protocolos basados en Internet.

(Alonso; y otros, 2004)

Los mismos son una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar información. Entre los estándares que se pueden utilizar está: XML que es el formato para los datos que se vayan a intercambiar y SOAP⁹. Los servicios web tienen utilidades y ellas se denotan en la forma de comunicar diferentes aplicaciones conectadas a una red de datos mediante el protocolo HTTP. Es importante destacar que estos servicios pueden utilizar cualquier protocolo, pero el más común es TCP¹⁰.

XML RPC

Llamada a Procedimiento Remoto (RPC, del inglés Remote Procedure Call) es un protocolo que implementa la comunicación entre dos aplicaciones. Fue el primer

⁷ Acrónimos del inglés *Uniform Resource Locator*.

⁸ Acrónimos del inglés *Extensible Markup Language*.

⁹ Acrónimos del inglés *Simple Object Access Protocol*.

¹⁰ Acrónimos del inglés *Protocolo de Control de Transmisión*.

mecanismo que surgió para invocar procedimientos remotos vía XML, ofrece una manera muy sencilla de invocar operaciones en sistemas heterogéneos a través de una estructura simple. **(XML-RPC.Com, 2012)**

XML RPC ofrece ventajas viables para establecer la comunicación entre varias aplicaciones distribuidas. Consiste en el envío de mensajes mediante XML que se establecen entre las aplicaciones que se quieren comunicar, una realiza la función de cliente y otra de servidor, lo que presenta a XML

RPC como una tecnología de servicios web cliente-servidor con una estructura sencilla que permite procesar y devolver información. **(XML-RPC.Com, 2012)**

Algunas de las funcionalidades de XML RPC son:

- Permite establecer una comunicación entre aplicaciones distribuidas sin importar el Sistema Operativo en que estén corriendo las mismas o el lenguaje que se haya utilizado para escribirla **(XML-RPC.Com, 2012)**.
- Soporta lenguajes como: Perl, Python, Java, C/C++, PHP y otros. **(Dr. Gómez Roberto, 2009)**
- Posee las estructuras de datos tales como: enteros, boolean, string, array y otros. **(Dr. Gómez Roberto, 2009)**
- Usa HTTP como transporte de los datos y XML como codificación. **(XML-RPC.Com, 2012)**
- Se pueden establecer aplicaciones cliente-servidor.

1.6 Conclusiones parciales.

En la elaboración de este capítulo se abordaron conceptos generales que sirvieron de guía para lograr un correcto desarrollo de esta investigación. Realizando una caracterización de los sistemas de planificación en cuanto a su utilización a nivel internacional y nacional, así como las ventajas y desventajas que estos poseen, se logró demostrar la importancia de su utilización para la gestión y ejecución de tareas. Se obtuvieron los elementos, como tecnologías y herramientas para el desarrollo del componente.

CAPÍTULO 2: Descripción de la solución propuesta

En el presente capítulo se hace referencia a la descripción de la propuesta de sistema siguiendo la metodología RUP. Se describen los principales artefactos que comprenden la modelación del dominio y el levantamiento de requisitos. Específicamente se muestra el modelo de dominio y los principales conceptos asociados a él. Se detallan los requisitos funcionales y los no funcionales que debe cumplir el sistema, así como los actores del mismo, el diagrama de CU y su descripción.

2.1. Modelo de dominio.

Cuando no es posible identificar claramente los procesos del negocio, la metodología RUP propone realizar un modelo de dominio, que es un subconjunto del modelo de negocio. (Loja, 2008). El modelo de dominio permite de manera visual mostrar al usuario los principales conceptos que se manejan en el dominio del problema. Este modelo de dominio es una representación de las clases conceptuales del mundo real, no como tal de componentes de software. Se utiliza principalmente cuando los flujos de información son pocos entendibles o difusos.

El modelo desarrollado no se trata de un conjunto de diagramas que describen clases de software, sino que puede considerarse como un diccionario visual que ayuda para el buen entendimiento.

A través de clases relacionadas se representan todos los objetos del dominio. Esto se hará con el lenguaje UML, lo que permitirá tener una mejor comprensión de la estructura y la dinámica del entorno, así como los problemas que existen dentro del mismo y la identificación de las mejoras potenciales. A continuación se muestra la Figura 2, donde se representa el Diagrama del dominio.

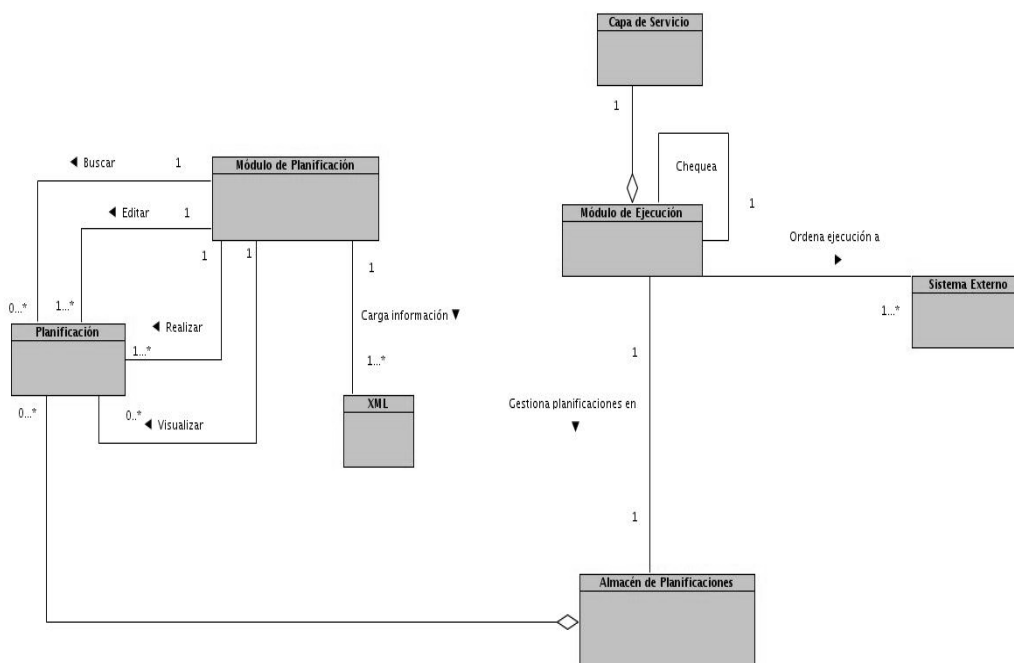


Figura 2. Diagrama de dominio.

2.1.1 Glosario de términos del dominio.

Módulo de Planificación: es un calendario de planificaciones donde se muestran los recursos, las planificaciones y en las cuales se pueden realizar una serie de funcionalidades como son: adicionar, buscar y editar planificaciones. Además se manejan los parámetros de cada una como son: nombre de la planificación, hora de inicio, hora de fin, fecha de inicio y fecha de fin.

Planificación: es una estructura donde se relacionan los parámetros asociados a una planificación. Los parámetros relacionados son de las planificaciones que se han realizado. En el calendario se pueden buscar, editar y realizar planificaciones.

Almacén de planificaciones: espacio lógico donde se guardan las planificaciones. Cada una de ellas con sus características correspondientes. Se guarda además el nombre del negocio, la acción y sus parámetros. Entre estos se pueden mencionar la fecha de inicio y de fin, hora de inicio y de fin.

Módulo de Ejecución: es el encargado de consultar el almacén de planificaciones para obtener una lista de planificaciones diarias. Las cuales según su hora de inicio es ordenada su ejecución.

XML: tiene una estructura definida, debe ser completado de forma manual según el negocio al que se relacione el sistema. Es aquí donde se definen los recursos que se van

Capítulo 2: Descripción de la solución propuesta

a planificar de acuerdo al negocio. Para ello necesita parámetros como: dirección IP¹¹ (máquina donde se encuentra), usuario, contraseña, puerto, entre otros.

Sistema externo: son aplicaciones externas encargadas de ejecutar la planificación, varían según el entorno del negocio. Son además agentes externos que utilizarán el sistema cuando el ejecutor mande a ejecutar la acción.

Capa de servicio: brinda una serie de servicios y funcionalidades referentes a los requisitos funcionales, que posibilitan el funcionamiento del componente en caso de no contar con el módulo de planificación.

2.2 Modelo del sistema.

Al terminar de establecer la relación entre las clases que conforman el dominio del sistema del Componente de Gestión de Planificación Genérica y describir los principales conceptos a través del modelo de dominio, se presentan los principales artefactos del flujo de trabajo de requisitos con la definición de los requisitos funcionales y los no funcionales para el componente, así como sus actores y CU.

2.2.1 Requisitos.

Los requisitos funcionales y no funcionales muestran las capacidades o condiciones que el sistema debe cumplir y las propiedades o cualidades que el producto debe tener, los cuales en la fase de construcción deben ser posibles de probar o verificar. Los requerimientos de software son las necesidades de los clientes, los servicios que los usuarios desean que proporcione el sistema de desarrollo y las restricciones en las que debe operar. **(Pressman, 2000)**

Requisitos funcionales.

Los requisitos funcionales no son más que la determinación clara y concisa de qué debe ser capaz de hacer el sistema, éstas se corresponden con operaciones realizadas de forma oculta o condiciones extremas a determinar por el sistema. **(Pressman, 2005)**

Los mismos muestran de forma específica que debe realizar el sistema, logrando con ellos una mayor efectividad en la organización y progreso de la aplicación. La solución que se propone es el Componente de Gestión de Planificación Genérica, el mismo está compuesto por dos módulos: uno de planificación y otro de ejecución. A continuación se especifican por módulos los requisitos funcionales de cada uno.

¹¹ Acrónimo de IP: Protocolo de Internet.

Para el módulo: Planificación

- **RF1.** Insertar información en la BD.
- **RF2.** Editar información en la BD.
- **RF3.** Eliminar información en la BD.
- **RF4.** Buscar información en la BD.
- **RF5.** Crear fichero de configuración XML.
- **RF6.** Insertar objeto de planificación.
- **RF7.** Editar objeto de planificación.
- **RF8.** Eliminar objeto de planificación.
- **RF9.** Visualizar planificaciones.
- **RF10.** Mostrar información de la planificación.
- **RF11.** Filtrar planificaciones por día.
- **RF12.** Filtrar planificaciones por semana.
- **RF13.** Filtrar planificaciones por mes.
- **RF14.** Filtrar planificaciones por recursos.
- **RF15.** Adicionar acciones.
- **RF16.** Modificar acciones.
- **RF17.** Eliminar acciones.
- **RF18.** Visualizar acciones.
- **RF19.** Autenticar negocio.
- **RF20.** Crear negocio.
- **RF21.** Editar negocio.
- **RF22.** Eliminar negocio.
- **RF23.** Mostrar negocios registrados.
- **RF24.** Editar contraseña del negocio.

Para el módulo: Ejecución

- **RF25.** Chequear planificaciones.
- **RF26.** Ordenar ejecución de planificación de grabación.
- **RF27.** Ordenar ejecución de planificación al gestor de procesos de medias.
- **RF28.** Publicar servicio web para adicionar acciones.
- **RF29.** Publicar servicio web para modificar acciones.
- **RF30.** Publicar servicio web para eliminar acciones.
- **RF31.** Publicar servicio web para listar acciones.

Capítulo 2: Descripción de la solución propuesta

- **RF32.** Publicar servicio web para insertar objetos de planificación.
- **RF33.** Publicar servicio web para editar objetos de planificación.
- **RF34.** Publicar servicio web para eliminar objetos de planificación.
- **RF35.** Publicar servicio web listar objetos de planificación.
- **RF36.** Registrar negocio.

Requisitos no funcionales.

Los requisitos no funcionales (RNF) son propiedades que debe cumplir el sistema. Los mismos se encargan de velar por temas como la seguridad y fiabilidad. Definen que el sistema tenga una interfaz amigable y de fácil manejo para al usuario.

➤ **Requisitos de Usabilidad (RNF 1).**

Los usuarios finales de este sistema deben ser otros sistemas de software, no obstante la aplicación debe ser configurada y administrada por un usuario administrador que debe tener conocimientos medios en informática, de ser posible especialista en soporte de software y dominar los datos de acceso a la BD del sistema con el que se desea integrar el componente.

➤ **Tipo de aplicación informática (RNF 2).**

El Componente de Gestión de Planificación Genérica tendrá dos módulos, uno web (módulo Planificador) y uno de escritorio (módulo Ejecutor).

➤ **Requisitos de Finalidad (RNF 3).**

El objetivo que persigue el sistema es, proveer una herramienta de planificación adaptable a cualquier entorno, que permita planificar acciones asociadas a recursos.

➤ **Requisitos de Hardware (RNF 4).**

Para el Servidor de Base de Datos:

RNF 4.1: 2 GB de memoria RAM, además de un CPU de dos núcleos a 2.4GHz.

Para el Servidor Web (Apache2.2):

RNF4.2: es recomendado 1GB de memoria RAM, además de un CPU de dos núcleos a 2.4GHz.

Para el Ejecutor:

RNF 4.3: 512Mb de memoria RAM como mínimo, pero es recomendado 1GB de memoria

Capítulo 2: Descripción de la solución propuesta

RAM; además de un CPU de dos núcleos a 2.4GHz.

➤ **Requisitos de Software (RNF5).**

Para el Servidor de BD:

RNF 5.1: Sistema Operativo: Windows o Linux. PostgreSQL 9.1

Para el Servidor Web (Apache2.2):

RNF 5.2: Sistema Operativo: Windows o Linux.

Para el Ejecutor:

RNF 5.3: Ubuntu 12.4. Biblioteca *Libqxmlrpc*

Para PC cliente:

RNF 5.4: Sistema Operativo: Windows o Linux. Mozilla Firefox 16.0 o superior.

➤ **Requisitos de Confiabilidad (RNF 6).**

RNF 6.1 Si se interrumpe la energía o la red de datos en la estación del servidor de base de datos, en las estaciones cliente no se podrá realizar ninguna acción sobre el componente. Una vez reanudados estos servicios el usuario podrá acceder nuevamente a las funcionalidades del sistema refrescando el componente.

RNF 6.2 Si ocurren cambios de hora en el servidor web, se afectarán la ejecución de las tareas planificadas, pues depende de las configuraciones realizadas en esta estación. Se recomienda sincronizar el horario en ambas estaciones de trabajo para evitar incongruencias.

RNF 6.3 El sistema se mantendrá activo el tiempo que lo requiera el usuario.

RNF 6.4 Tras una excepción el sistema es capaz de continuar su funcionamiento independientemente de la funcionalidad afectada.

➤ **Requisitos de Eficiencia (RNF 7).**

Tiempo de respuesta

RNF 7.1 El tiempo de respuesta promedio por transacción debe ser como mínimo de 300 mili-segundo. El tiempo de respuesta máximo debe ser de 5 segundos.

Capacidad

RNF 7.2 El sistema puede soportar máximo 200 conexiones simultáneas, este número

Capítulo 2: Descripción de la solución propuesta

puede variar según las características del servidor donde este desplegada la aplicación; el número que se propone está en correspondencia con el ambiente especificado en el RNF 4.

➤ **Requisitos de Soporte (RNF 8).**

RNF 8.1 Para la implantación del sistema se debe contar en las PCs con un navegador preferentemente Mozilla Firefox 16.0 o superior.

RNF 8.2 El sistema permitirá responder a una amplia gama de problemáticas siendo capaz de integrarse a varios negocios o aplicaciones permitiendo la realización de planificaciones con diferentes fines.

RNF 8.3 El sistema posee una capa de servicios que permitirá si así lo requiere, el negocio que utilice el Componente de Gestión de Planificación Genérica para acceder a sus funcionalidades prescindiendo del módulo web de planificación.

RNF 8.4 La instalación y el mantenimiento del sistema deben ser entendibles por los usuarios finales. Los instaladores incluirán todas las especificaciones que necesite el sistema para funcionar correctamente.

➤ **Restricciones de diseño (RNF 9).**

RNF 9.1 El lenguaje de programación que se utiliza en el Componente de Gestión de Planificación Genérica para el servidor web Apache 2.2. serán: PHP5, JavaScript y HTML5.

RNF 9.2 Se emplean para el desarrollo del Componente de Gestión de Planificación Genérica en el servidor web los *Frameworks* Symfony 2.1.2 y DOJO 1.8.

RNF 9.3 El lenguaje de programación que se utiliza en el Ejecutor es: C++ y como *Framework* Qt 4.8.1 con el IDE Qt-Creator 2.4.1.

RNF 9.4 Ambos módulos emplearán como Sistema Gestor de Base de Datos PostgreSQL 9.1.

RNF 9.5 Para la implementación del Componente de Gestión de Planificación Genérica se emplea la biblioteca *Libqxmllrpc* para la publicación de servicios web.

➤ **Manual de usuarios (RNF 10).**

RNF10.1 El sistema deberá brindar a sus clientes finales, manuales de usuarios en los que se especificarán todas las funciones que pueden realizar y cómo hacerlo.

Capítulo 2: Descripción de la solución propuesta

➤ Interfaces de usuario (RNF 11).

RNF11.1 Las interfaces gráficas implementadas por el sistema deben concebirse con un ambiente sencillo y de navegación fácil para el usuario. Los colores serán convenientemente utilizados dada la funcionalidad y objetivo del sistema, siendo claros en la mayor parte de la aplicación. Logrando una vista agradable a los usuarios y resaltando con otras tonalidades los mensajes de interacción de los que dependen las funcionalidades críticas.

RNF11.2 Las planificaciones deben mostrarse en una vista de calendario.

RNF11.3 Los recursos deben mostrarse en la región lateral agrupados por categorías y de forma jerárquica.

➤ Interfaces de comunicación (RNF 12).

RNF 12.1 El Componente de Gestión de Planificación Genérica para ordenar la planificación gestionada debe comunicarse con sistemas externos, para ello implementa *plugins* de ejecución que varían según los parámetros establecidos por estos sistemas.

➤ Requisitos legales y de derecho de autor (RNF 13).

RNF 13.1 Todas las herramientas de desarrollo usadas son libres, *Framework Qt* e IDE *Qt-Creator* con licencia LGPL¹², v 2.1, *Visual Paradigm* con licencia GNU/Linux y *Libqxmllrpc* bajo licencia LGPL v2.1.

2.2. Diagrama de casos de uso del sistema.

Un diagrama de CU representa la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios u otros sistemas. A partir de la identificación del actor que interactúa con la aplicación, así como la recopilación del conjunto de funcionalidades escritas en forma de requerimientos, que a su vez han sido agrupados en CU según sus peculiaridades, se conforma el Modelo de CU que se representa a continuación. (Sommerville, 2005)

El Componente de Gestión de Planificación Genérica cuenta con una serie de CU que representan todos los procesos que este debe realizar. Dicho componente está compuesto por dos módulos. A continuación se muestran en las siguientes Figuras 3 y 4 los diagramas de CU para cada módulo.

¹² Acrónimo de LGPL: Licencia Pública General Reducida.

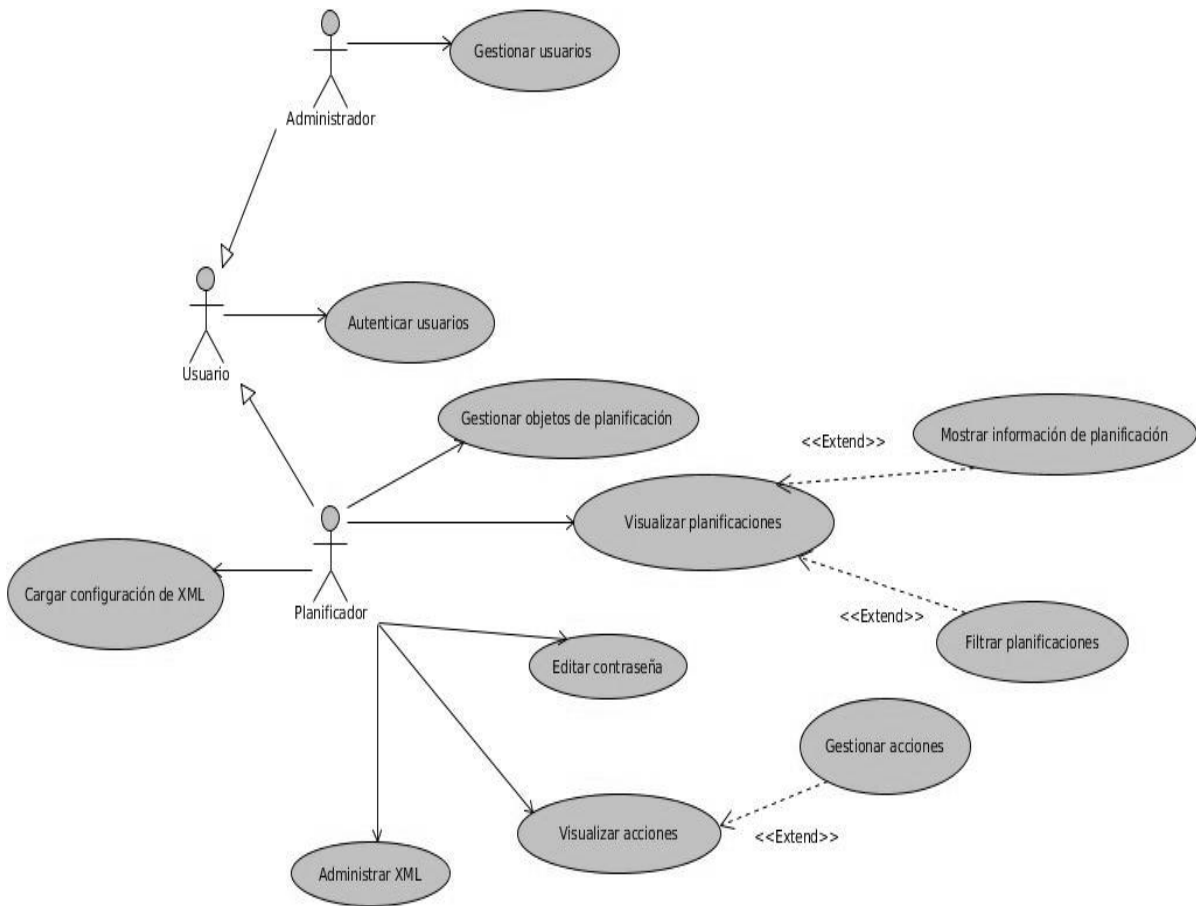


Figura 3. Diagrama de casos de uso del sistema para el módulo de Planificación.

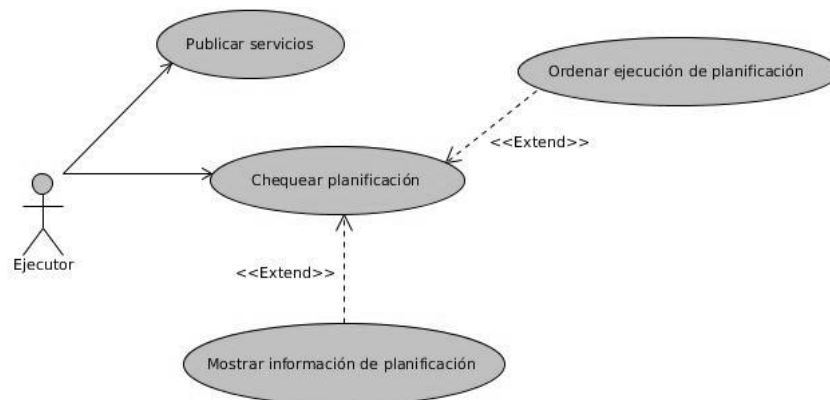


Figura 4. Diagrama de casos de uso del sistema para el módulo ejecución.

Capítulo 2: Descripción de la solución propuesta

2.3.1 Descripción de actores del sistema.

Un actor del sistema representa un conjunto coherente de roles que juegan los usuarios. Además, él no es parte del sistema, es el encargado de inicializar los CUS, y es beneficiado con el resultado de los mismos. Es una entidad externa al sistema que se modela y que puede interactuar con él. Los actores representan a los seres humanos, a un software, hardware externo u otros sujetos que interactúen con el sistema que se esté especificando. (Sommerville, 2005)

Durante el desarrollo de la aplicación se definieron una serie de actores, los mismos se describen a continuación en la Tabla 1:

Tabla 1. Descripción de los actores del sistema.

Actor	Objetivo
Planificador	Es el usuario encargado gestionar las planificaciones, las acciones, autenticar el negocio y gestionar la configuración del XML correspondiente al negocio.
Administrador	Es el usuario encargado de gestionar los negocios.
Usuario	Persona con acceso al sistema que puede autenticarse.

2.3.2 Descripción de los casos de usos del sistema.

Cada uno de los CU identificados presentan características particulares que para su mayor entendimiento se hace necesario describirlos textualmente. A través del modelado de los CU se pueden observar las funciones del futuro sistema. No es realmente una aproximación a la orientación a objetos; es una forma de modelar procesos y de ayudar al cliente con las perspectivas del sistema. A continuación se brinda un resumen de los CU identificados para el sistema en la Tabla 2.

Tabla 2. Descripción del caso de uso Gestionar objetos de planificación.

Objetivo	El objetivo que persigue el actor es poder crear, modificar, eliminar y buscar planificaciones.
Actores	Planificador.
Resumen	El caso de uso se inicia cuando el planificador adiciona, modifica, elimina o busca una planificación. Termina cuando se haya realizado alguna de estas operaciones.

Capítulo 2: Descripción de la solución propuesta

Complejidad	Media	
Prioridad	Crítico	
Precondiciones	El usuario existe en la BD. Existen planificaciones disponibles en la BD.	
Poscondiciones	Se gestionaron las planificaciones.	
Flujo de eventos		
Flujo básico “Gestionar objetos de planificación”.		
	Actor	Sistema
1.	Selecciona una de las siguientes acciones: - Adicionar planificación. - Modificar planificación (Ir a la Sección 1 Modificar planificación). - Eliminar planificación (Ir a la Sección 2 Eliminar planificación). - Buscar planificación (Ir a la Sección 3 Buscar planificación)	
2.	Selecciona el botón crear.	
3.		Se comunica con el Componente de Gestión de Planificación Genérica. Muestra la interfaz de crear planificación.
4.	Inserta los datos para crear planificación: Nombre. Recurso. Fecha inicio. Fecha fin. Hora inicio. Hora fin. Descripción. Todo el día. Pulsa en el botón Guardar. Ver prototipo de interfaz 1.	

Capítulo 2: Descripción de la solución propuesta

5.		El Componente de Gestión de Planificación Genérica valida los datos y envía notificación de la inserción de la planificación realizada. Valida que no existan campos vacíos, que la hora de fin no sea menor que la de inicio y que la fecha de fin no sea menor que la de inicio. Inserta los datos en la Base de Datos.
6.		Muestra en el calendario la planificación realizada. Termina el caso de uso.

Prototipo de Interfaz 1.

Sección 1: “Editar Planificación”

Flujo básico Editar Planificación.

	Actor	Sistema
	Selecciona el botón editar.	
		Se comunica con el Componente de Gestión de Planificación Genérica. Muestra la interfaz de planificación.
	Inserta los datos para editar la planificación: Nombre. Recurso. Fecha inicio. Fecha fin. Hora inicio. Hora fin.	

Capítulo 2: Descripción de la solución propuesta

	Descripción. Pulsa en el botón Guardar. Ver el prototipo de interfaz 2.	
		El Componente de Gestión de Planificación Genérica valida los datos y envía notificación de la inserción de la planificación realizada. Valida que no existan campos vacíos, que la hora de fin no sea menor que la de inicio, que la fecha de fin no sea menor que la de inicio, que existan planificaciones. Modifica los datos en la Base de Datos.
		Muestra en el calendario la planificación que fue modificada. Termina el caso de uso.

Prototipo de Interfaz 2.

Sección 2: “Eliminar Planificación”

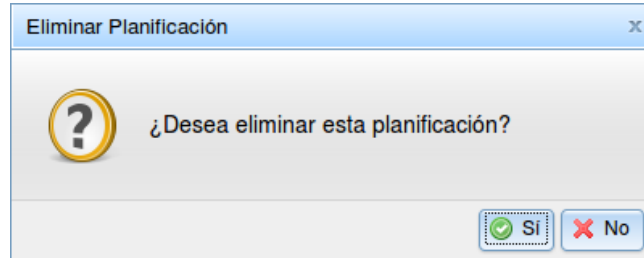
Flujo básico Eliminar Planificación.

	Actor	Sistema
1.	Selecciona una planificación y selecciona el botón eliminar.	
2.		Muestra un mensaje de confirmación para realizar la acción.
3.	Selecciona la opción sí. Ver prototipo de interfaz 3.	
4.		Comunicándose con el Componente de Gestión de Planificación Genérica elimina la o las planificaciones seleccionadas y actualiza la Base

Capítulo 2: Descripción de la solución propuesta

		de Datos luego de eliminada la planificación.
5.		Actualiza el calendario de planificaciones. Termina el caso de uso.

Prototipo de Interfaz 3.

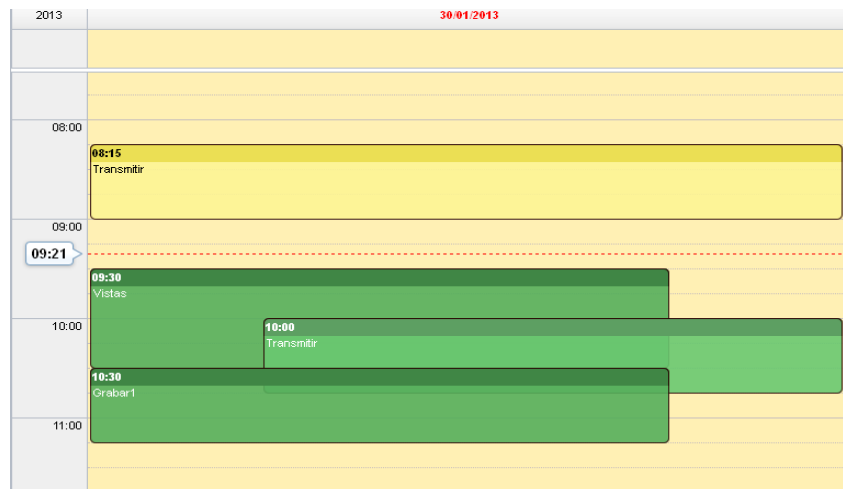


Sección 3: “Buscar Planificación”

Flujo básico Buscar Planificación.

	Actor	Sistema
1.	Selecciona la siguiente opción de filtrado: Día. Ver prototipo de interfaz 4.	
2.		Se muestran las planificaciones existentes en el componente para ese día. Termina el caso de uso.

Prototipo de Interfaz 4.



Flujos alternos

3a. Evento: Selecciona la opción no en el mensaje de notificación.

	Actor	Sistema
3a.	Pulsa la opción no.	

Capítulo 2: Descripción de la solución propuesta

1.		Cancela la operación y termina el caso de uso.
Relaciones	CU Incluidos	
	CU Extendidos	
Requisitos no funcionales	RnF 7.1 El tiempo de respuesta promedio por transacción debe ser de 3 segundos. El tiempo de respuesta máximo debe ser de 5 segundos. RnF 7.2 El sistema puede soportar máximo 200 conexiones simultáneas, este número puede variar según las características de las estaciones de trabajo; el número que se propone está en correspondencia con el ambiente especificado en el RnF 4 Ambiente.	
Asuntos pendientes	No existen.	

Tabla 3. Descripción del CU Chequear planificación.

Objetivo	El objetivo que persigue el actor es chequear las planificaciones que existen en la cola de planificaciones.		
Actores	Ejecutor.		
Resumen	El caso de uso se inicia cuando el ejecutor desea chequear planificaciones. Termina cuando se haya efectuado la operación anterior.		
Complejidad	Media		
Prioridad	Crítico		
Precondiciones	La BD debe tener planificaciones.		
Poscondiciones	Se chequearon las planificaciones.		
Flujo de eventos			
Flujo básico “Chequear planificación”.			
	Actor	Sistema	
1.	Ejecuta el archivo para obtener en una cola las planificaciones ordenadas por hora.		
2.		Chequea cada 5 segundos si es la hora de que la primera planificación que esta en la cola se ejecute. Termina el caso de uso.	
Relaciones		CU Incluidos	Ordenar ejecución de planificación.

Capítulo 2: Descripción de la solución propuesta

		Mostrar información de planificación.
	CU Extendidos	
Requisitos no funcionales	RnF 6.1 El tiempo de respuesta promedio por transacción debe ser de 3 segundos. El tiempo de respuesta máximo debe ser de 5 segundos. RnF 6.2 El sistema puede soportar máximo 200 conexiones simultaneas, esto número puede variar según la características del servidor web; el número que se propone está en correspondencia con el ambiente especificado en el RnF 4.	
Asuntos pendientes	No existen.	

2.4 Conclusiones parciales.

En el presente capítulo se presentó la propuesta del sistema a desarrollar. Se estructuró el modelo de dominio, describiendo los conceptos fundamentales que en él se exponen, pues no existen procesos de negocio bien definidos. Se listan los requerimientos funcionales y no funcionales que debe cumplir el software para satisfacer las necesidades existentes. Además, se definieron los actores y casos de uso del sistema, así como la descripción textual de cada uno de ellos, lo que constituye un paso de gran importancia para las próximas etapas de desarrollo.

Capítulo 3: Diseño e implementación de la solución propuesta

CAPITULO 3: Diseño e implementación de la solución propuesta

En el siguiente capítulo se describen los elementos relacionados con el diseño e implementación del sistema propuesto. Dentro de los artefactos desarrollados en el capítulo se encuentran: el diagrama de clases del diseño, el de componentes y el modelo de despliegue del Componente de Gestión de Planificación Genérica. Además se abordan los patrones de diseño y arquitectura que se emplean en el componente.

3.1 Patrones.

Un patrón es una descripción de un problema y la solución, a la que se da un nombre, y que se puede aplicar a nuevos contextos; idealmente, proporciona consejos sobre el modo de aplicarlo en varias circunstancias, y considera los puntos fuertes y compromisos.

(González Jiménez, 2011)

Los patrones son utilizados en el desarrollo del software para establecer una forma de organizar y estructurar el desarrollo. Estos son una guía para realizar algunas acciones dentro del proceso de desarrollo, especificando un conjunto de subsistemas predefinidos y las funcionalidades de cada uno. Además facilitan la comprensión de la situación en que se encuentre algún problema que se quiera solucionar.

3.1.1 Patrones de arquitectura.

El Componente de Gestión de Planificación Genérica cuenta con dos módulos, un módulo escritorio y otro web, y cada uno de ellos tiene por independiente asociado un patrón de arquitectura. El módulo de escritorio presenta el patrón de arquitectura N capas y el módulo web el patrón Modelo Vista Controlador (MVC). **(González Jiménez, 2011)**

Se seleccionó para la aplicación de escritorio el patrón en N capas por las facilidades que brinda a la hora de desarrollar el sistema. Este patrón permite dividir sistemas de software complicados, para así hacerlo más entendibles y fáciles de implementar. En este esquema, si el flujo de información se establece de manera ascendente cada capa solamente interactúa con su inmediata superior, o con su inmediata inferior en el caso de que el flujo sea de manera descendente. Algunas ventajas de este patrón son:

- Se puede entender una capa como un todo, sin considerar las otras.
- Las capas se pueden sustituir con implementaciones alternativas de los mismos servicios básicos.

Capítulo 3: Diseño e implementación de la solución propuesta

- Se minimizan dependencias entre capas.
- Las capas posibilitan la estandarización de servicios.
- Luego de tener una capa construida, puede ser utilizada por muchos servicios de mayor nivel.

Se seleccionó para la aplicación web el patrón de arquitectura Modelo Vista Controlador pues se ve frecuentemente utilizado en aplicaciones web. Además este tiene una característica muy importante y es que separa la lógica del negocio de la interfaz, lo que facilita la evolución por separado de ambos aspectos, incrementa la reutilización y flexibilidad. La utilización de este patrón tiene varias ventajas, entre ellas:

- Es posible utilizar diferentes vistas para un mismo modelo, como por ejemplo la representación de un conjunto de datos o como un diagrama de barras.
- Es posible construir nuevas vistas sin necesidad de modificar el modelo original.
- Proporciona un mecanismo de configuración a componentes complejos muchos más tratables que el puramente basado en eventos.

3.1.2 Patrones de diseño.

Los patrones de diseño son una guía para obtener soluciones a diferentes problemas presentados en el desarrollo de un software. Los mismos permiten la aplicación de diseños ya probados a diferentes escenarios problemáticos. Para la implementación del Componente de Gestión de Planificación Genérica se tuvieron en cuenta los patrones GRASP¹³ y GoF¹⁴ los cuales se describen a continuación. **(González Jiménez, 2011)**

Patrones GRASP

Los patrones GRASP, más que patrones propiamente dichos, son una serie de “buenas prácticas” de aplicación recomendable en el diseño de software. Entre ellos, se pueden mencionar los patrones Experto, Creador, Bajo acoplamiento y Alta cohesión, porque guardan directa relación con la creación y asignación de responsabilidades a los objetos. Todo patrón tiene un nombre, plantea un problema y aporta una solución. **(Tabares, 2010)**

Creador: Está diseñado para asignar la responsabilidad de crear una instancia de una clase determinada a otra. Se utiliza este patrón debido a la característica del sistema de ser orientado a objetos. Se evidencia en las clases consumidoras de los componentes.

¹³ Acrónimo de GRASP: Patrones para Asignar Responsabilidades.

¹⁴ Acrónimo de GoF: *Gang Of Four*.

Capítulo 3: Diseño e implementación de la solución propuesta

Mediante los objetos que poseen estas clases se establece la comunicación entre los componentes.

Alta cohesión: es una medida que determina cuán relacionadas y adecuadas están las responsabilidades de una clase, de manera que no realice un trabajo colosal; una clase con baja cohesión realiza un trabajo excesivo, haciéndola difícil de comprender, reutilizar y conservar. Este patrón se utiliza pues se tiene una única clase controladora para centrar todas las funcionalidades. Se evidencia con la utilización de una única clase controladora.

Experto: posibilita una adecuada asignación de responsabilidades facilitando la comprensión del sistema, su mantenimiento y adaptación a los cambios con reutilización de componentes. Se utiliza porque se hace uso de clases entidades. Este patrón se evidencia específicamente en las clases entidades de la aplicación web.

Patrones GoF

Observador: Permite a los objetos captar dinámicamente las dependencias entre objetos, de tal forma que un objeto notificará a los objetos dependientes de él cuando cambia su estado, siendo actualizados automáticamente. Se utiliza en las clases de la capa de presentación que dependen de las clases que se encuentran en la capa controladora y es necesario que estas notifiquen en caso de algún cambio. Donde más se destaca la presencia de este patrón en el método vigilante en la parte de la aplicación de escritorio.

3.2 Diseño de interfaz.

La metodología RUP propone la realización del análisis y el diseño para lograr un acercamiento de los desarrolladores con los procesos a implementar. Según **(Pressman, 2002)** el análisis tiene como misión el estudio de los requisitos, así como el estructurarlos y darle el refinamiento pertinente a los mismos, con el objetivo de conseguir una comprensión más precisa de ellos. Por su parte, el diseño es una representación abstracta de lo que se va a construir, contribuye a formar una arquitectura sólida y ayuda a crear un plano para la implementación. Mediante él se modela el sistema para que soporte todos los requisitos incluyendo los no funcionales.

Uno de los principios de RUP a los cuales **(IBM, 2007)** hace referencia, precisa que la metodología debe de adaptarse al tamaño del proceso y a las necesidades del proyecto. **(Jacobson y Rumbaugh, 2000)** expresa que el análisis es una etapa de la metodología RUP que debe de evidenciarse en la construcción del software. De esta se puede prescindir de acuerdo a la manera en que se haya decidido implementar el sistema,

Capítulo 3: Diseño e implementación de la solución propuesta

pasando a analizar los requisitos como parte del diseño o como parte integrada de la captura de los requisitos, lo que deja en evidencia que la manera de emplear el análisis es consecuente con la construcción del sistema.

En consideración a lo anterior expuesto y bajo el principio de que la metodología es altamente configurable, pudiendo obviarse artefactos durante el desarrollo, se decide pasar del flujo de requisitos directamente al diseño, sin la implementación del análisis. Esta decisión es tomada luego de realizar un estudio y constatar que para el desarrollo del Componente de Gestión de Planificación Genérica, no se contempló el análisis como parte de la implementación de las soluciones. Además, los requisitos a implementar en el Componente de Gestión de Planificación Genérica son bien conocidos y se cuenta con una comprensión de los mismos. En general, las tecnologías sobre las que se desarrolla el sistema son conocidas, lo que permite prescindir del refinamiento de los requisitos e ir directo al diseño.

3.3 Modelo de diseño.

El modelo de diseño es un modelo de objetos que describe la realización física de los CU, centrándose en como los requisitos funcionales y no funcionales tienen impacto en el sistema. Sirve como abstracción de la implementación del sistema.

A continuación se muestran dos diagramas de clases del diseño en las Figuras 5 y 6. En los anexos se encuentran íntegros los diagramas.

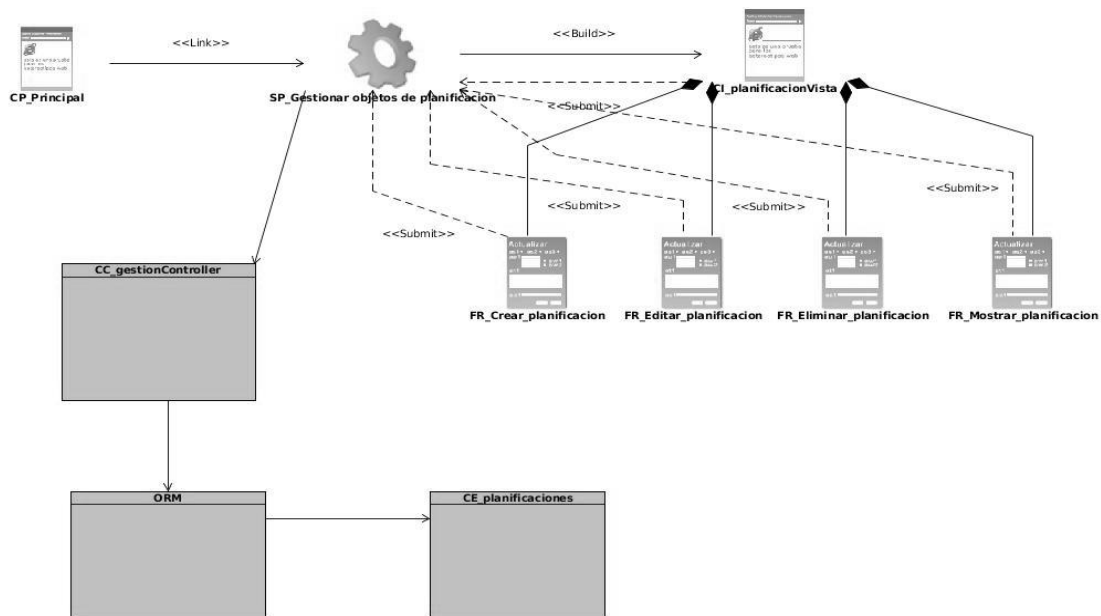


Figura 5. Diagrama de clases del diseño para el CU Gestionar objetos de planificación.

Capítulo 3: Diseño e implementación de la solución propuesta

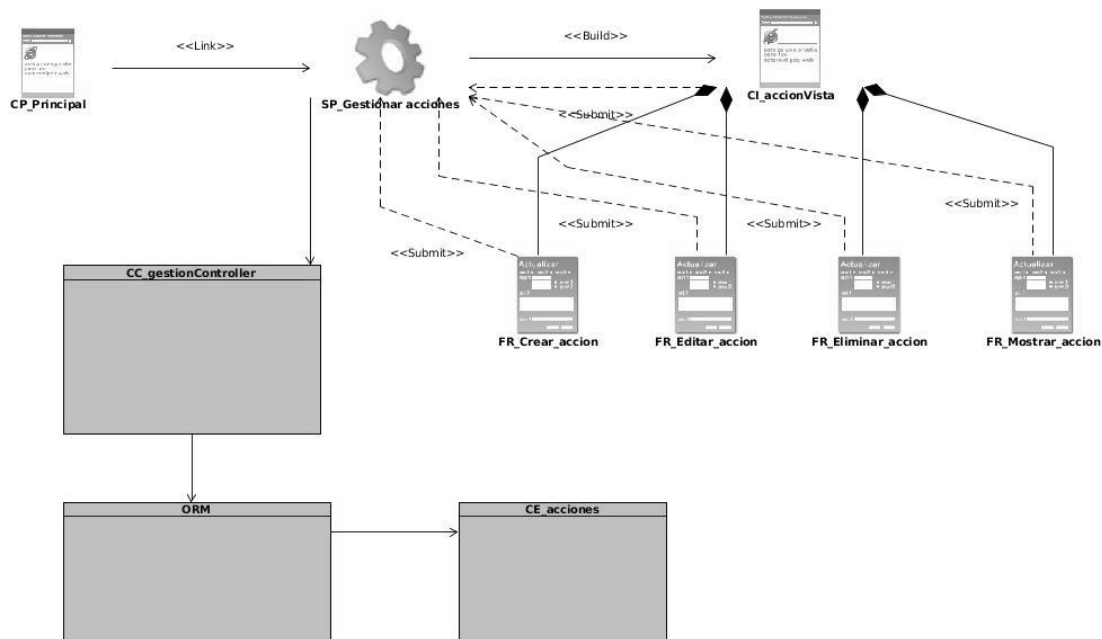


Figura 6. Diagrama de clases del diseño para el CU Gestionar acciones.

A continuación se muestra el diagrama de clases del diseño para el módulo de Ejecución en la Figura 7. En los anexos se encuentra el diagrama íntegro.

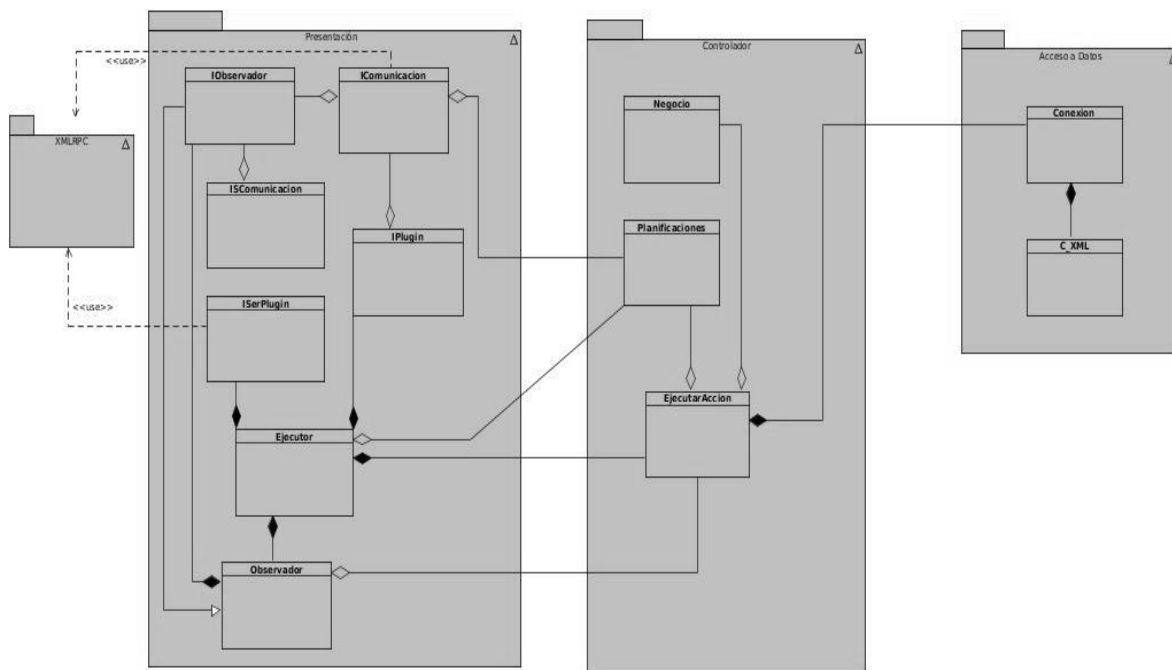


Figura 7. Diagrama de clases del diseño para el módulo de Ejecución.

3.4 Modelo de implementación.

Los diagramas de componentes son usados para estructurar el modelo de implementación en términos de módulos y mostrar las relaciones entre los elementos de implementación. Un diagrama de componentes tiene un nivel más alto de abstracción que un diagrama de clase, usualmente un componente se implementa por una o más clases (u objetos) en tiempo de ejecución. Un componente puede comprender una gran porción de un sistema.

El modelo de implementación del módulo de Ejecución está conformado por varias clases y la biblioteca *Libqxmlrpc* en forma de componentes y la relación entre ellas. En este se encuentran las clases consumidoras de los componentes, las publicadoras de los mismos y las clases que componen la interfaz. A continuación se muestra el diagrama de implementación. En los anexos se encuentra íntegro.

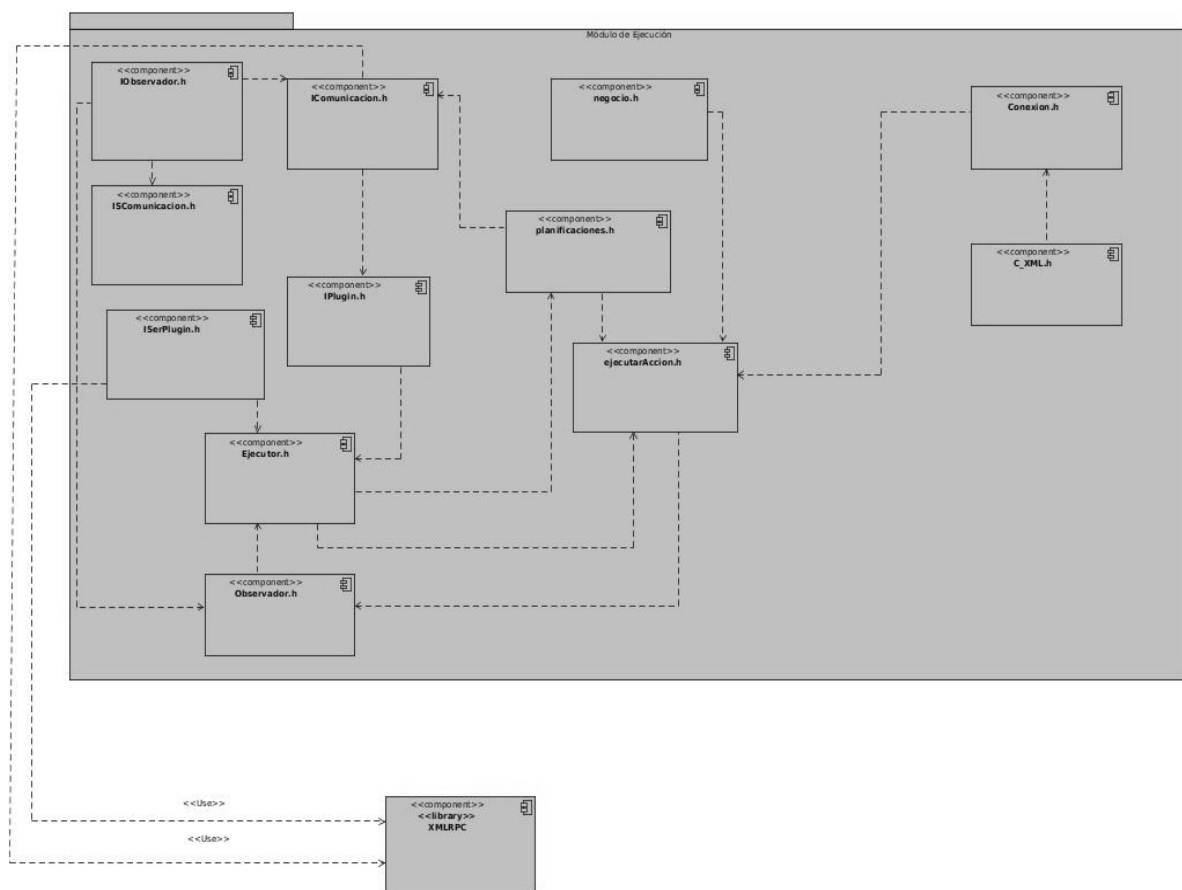


Figura 8. Diagrama de implementación del módulo de Ejecución.

Capítulo 3: Diseño e implementación de la solución propuesta

El modelo de implementación del módulo de Planificación está compuesto por varias clases en forma de componentes y la relación entre ellas. En este se encuentran las clases entidades de los componentes y la clase controladora. A continuación se muestra el diagrama de implementación.

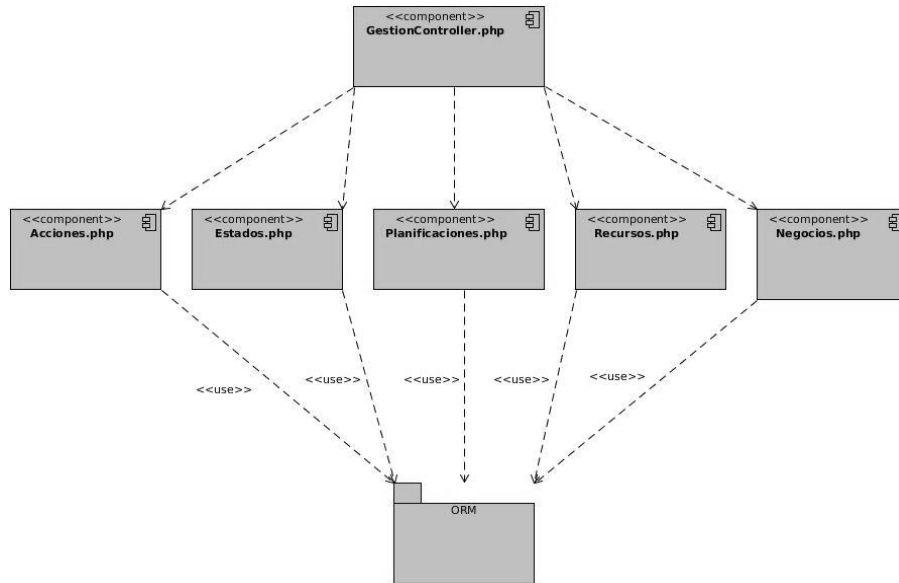


Figura 9. Diagrama de implementación del módulo de Planificación.

3.5 Modelo de despliegue.

La vista de despliegue contiene los nodos que forman la topología hardware sobre la que se ejecuta el sistema, la distribución, entrega e instalación de las partes que constituyen el sistema físico (Márquez, 2003). Esta vista suministra una base para la comprensión de la distribución física de un sistema a través de nodos. Es decir, describe los nodos físicos necesarios para la configuración de la plataforma donde se ejecutará el sistema.

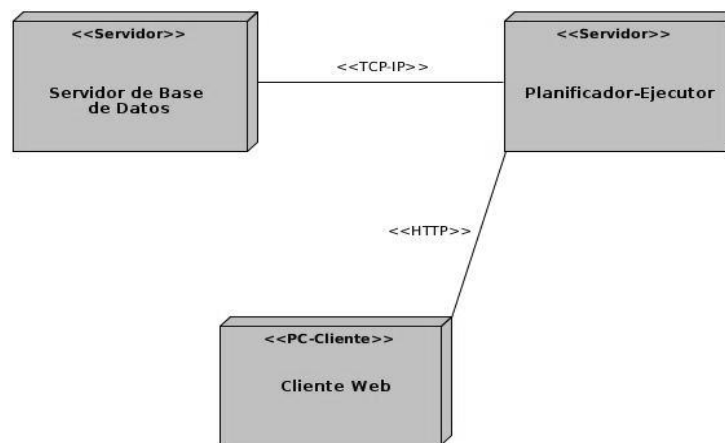


Figura 10. Diagrama de despliegue.

Capítulo 3: Diseño e implementación de la solución propuesta

3.5.1 Descripción de los nodos.

A continuación se describen los nodos físicos representados en el diagrama de despliegue.

El nodo Cliente Web representa una PC donde se encuentra implantado el negocio, la parte del módulo de Planificación. El nodo Planificador-Ejecutor representa una PC donde se encuentra la aplicación del Planificador y la del Ejecutor, el Ejecutor es el encargado de listar las planificaciones por hora, mandarlas a ejecutar llegado su tiempo de ejecución y publicar servicios web. La aplicación Planificador se encarga de gestionar las planificaciones y acciones. Además se encuentra instalado el módulo de planificación en el servidor web. El nodo Servidor de Base de Datos representa una PC donde se encuentra el servidor de base de datos que va a proporcionar los datos de las planificaciones, acciones y usuarios que se encuentran en su servidor.

3.6 Modelo de datos.

Las bases de datos desempeñan un papel crucial en casi todas las áreas de desarrollo de software. Su diseño es una de las tareas más importantes en la construcción de un sistema que requiera manejar la persistencia de la información. El almacenamiento de los datos de una aplicación es uno de los elementos más importantes, para ello es necesario diseñar un modelo de datos que soporte y almacene los datos necesarios para un correcto funcionamiento del sistema, en la siguiente gráfica se muestra la estructura seleccionada para el almacenamiento en PostgreSQL.

3.6.1 Diagrama Entidad-Relación.

A continuación se muestra el modelo lógico de la base de datos en el diagrama Entidad-Relación.

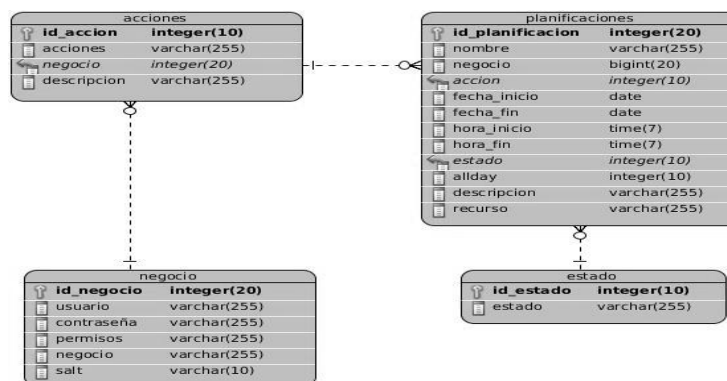


Figura 11. Diagrama entidad relación.

Capítulo 3: Diseño e implementación de la solución propuesta

3.6.2 Descripción de las tablas.

Tabla: Acciones.

Esta tabla almacena la información de las acciones asociadas a un negocio, junto a su respectiva descripción.

Tabla: Negocio.

La tabla negocio es la encargada de guardar lo referente al negocio, como son: nombre del negocio, la contraseña, el usuario y los permisos del mismo.

Tabla: Estado.

En esta tabla se almacena el estado de las planificaciones, los mismos son: a la espera, en ejecución, ya ejecutada y fallida.

Tabla Planificaciones.

En esta tabla se almacena la información asociada a las planificaciones, así como: nombre, negocio, acción, fecha de inicio, fecha fin, hora de inicio, hora fin y otros parámetros relacionados a las mismas.

4.1 Conclusiones parciales

Este capítulo abordó la realización de los diagramas del sistema, lo cual constituye una importante entrada para el próximo flujo de trabajo: la implementación. Se realizó además una descripción de los patrones de diseño usados y de la arquitectura seleccionada, que solidifican la implementación del componente. En el mismo se confeccionó un modelo de diseño el cual representa la estructura que posee el componente. También se generó el modelo de componentes, este permitió describir los elementos representados en el diseño en forma de componentes de software. Por su parte la confección del modelo de despliegue, proporciona una guía a seguir para la instalación del sistema.

CAPITULO 4: Validación y pruebas.

Para validar la solución propuesta y la correcta integración del Componente de Gestión de Planificación Genérica se hace necesario someter al software a un conjunto de pruebas que garantizarán la calidad y la detección de posibles errores que afecten el correcto funcionamiento de la aplicación. Estas pruebas garantizaron la correcta cohesión entre lo que se diseñó y lo que se encuentra implementado, velando siempre porque se cumplan las necesidades que dieron inicio a la construcción del software; requisito indispensable para el correcto funcionamiento del componente.

4.2 Modelo de pruebas.

El modelo de pruebas es una guía para conocer el correcto funcionamiento de la aplicación, su estado actual y los posibles errores que pueden detectarse. Es por ello que al someter al componente a una serie de pruebas, eleva la calidad y capacidad del mismo. Además que estas pruebas garantizaron la integración del diseño y lo implementado. Para la validación de la aplicación se seleccionaron dos técnicas, la técnica de las pruebas de caja negra y la técnica de las pruebas de caja blanca.

4.3 Pruebas de caja negra.

Las pruebas de caja negra son las que se aplican a la interfaz del software. Una prueba de este tipo examina algún aspecto funcional del sistema que tiene poca relación con la estructura lógica interna del software. **(Pressman, 2005)**

En el caso de la técnica de las pruebas de caja negra se le realizaron al módulo de Planificación. Esta técnica se enmarca más en el entendimiento de que es lo que hace sin darle importancia a como lo hace, por lo que deben estar bien definidas sus entradas y salidas, es decir su interfaz, es por ello que no se precisa definir ni conocer los detalles internos de su funcionamiento.

Algunos de los métodos empleados para estas pruebas son los siguientes:

- Métodos de pruebas basados en grafos.
- Análisis de valores equivalentes.
- Adivinando el error.
- Partición equivalente.

Para realizar las pruebas al sistema se utilizó la técnica de caja negra empleando el

Capítulo 4: Validación y pruebas

método de partición equivalente, el cual presenta las siguientes características:

(Pressman, 2002) plantea que la partición equivalente se presenta como un método de prueba de caja negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. Un caso de prueba ideal descubre de forma inmediata una clase de errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico. La partición equivalente se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar. **(Pressman, 2002)**

El objetivo de partición equivalente es reducir el posible conjunto de casos de prueba en uno más pequeño, un conjunto manejable que evalúe bien el software. Se toma un riesgo porque se escoge no probar todo. Así que se necesita tener mucho cuidado al escoger las clases. **(Patton, 2005)**

A continuación en la siguiente tabla se muestran las pruebas del CU Gestionar objetos de planificación; exactamente a la sección 1. Para ellas se aplicó el método de partición equivalente.

Pre-condiciones: el usuario tiene que estar autenticado y poseer los permisos para realizar esta operación. Además, debe de seleccionar la acción que desea planificar.

Tabla 4. Escenarios del CU Gestionar objeto de planificación.

Sección	Escenarios para esta sección	Descripción del escenario
SC: 1 Adicionar planificación.	EC: 1.1 Adicionar planificación correctamente.	Llena todos los campos que conforman la planificación y se muestra el mensaje: <i>“La planificación se ha guardado correctamente”</i> . Muestra la tarea en el calendario de color verde (en espera). Almacena la tarea en la base de datos.
	EC: 1.2 Adicionar planificación incorrectamente.	Cuando se quiere guardar una planificación sin que estén todos los campos obligatorios no se habilita el botón guardar. Además, el sistema resalta con color rojo los campos

Capítulo 4: Validación y pruebas

		obligatorios que deben de ser llenados.
	EC: 1.3 Adicionar planificación. Cancelar.	Cancela la operación y cierra la ventana para insertar nueva tarea de planificación.
	EC: 1.4 Adicionar planificación. Cerrar.	Cancela la operación y cierra la ventana para insertar nueva tarea de planificación.

Tabla 5. Descripción de las variables del CU Gestionar objeto de planificación.

No.	Nombre del campo	Clasificación	Valores nulos	Descripción
1	Nombre	Campo de texto	No	Permite una cadena de caracteres alfa-numéricos.
2	Recursos	Lista desplegable	No	Permite seleccionar el recurso.
3	Fecha de inicio	Caja de fecha	No	Permite seleccionar la fecha.
4	Fecha de fin	Caja de fecha	No	Permite seleccionar la fecha.
5	Hora de inicio	Caja de tiempo	No	Permite seleccionar o introducir la hora.
6	Hora de inicio	Caja de tiempo	No	Permite seleccionar o introducir la hora.
7	Todo el día	Chequeable	Si	Permite seleccionar el tiempo de la planificación.
8	Descripción	Área de texto	Si	Permite introducir una descripción de la planificación.
9	Guardar	Botón	No	Permite guardar la planificación.
10	Atrás	Botón	No	Permite ir atrás para seleccionar una acción.

Capítulo 4: Validación y pruebas

11	Cancelar	Botón	No	Permite cancelar la planificación.
----	----------	-------	----	------------------------------------

Tabla 6. Matriz de datos del CU Gestionar objeto de planificación.

Escenario	V 1	V 2	V 3	V 4	V 5	V 6	V 7	V 8	Respuesta	Resultado	Flujo central
EC: 1.1 Adicionar planificación correctamente.	(Plan)	(PC) V	(5/0 4/20 13) V	(5/ 04/ 20 13) V	(8: 30 am) V	(9:5 0 am) V			El sistema muestra el mensaje: “La planificación se ha guardado correctamente”.	Satisfactorio.	1. El usuario accede al sistema para crear una planificación. 1.1. El sistema muestra una interfaz para introducir los datos de la planificación. 2. El usuario inserta todos los datos de la planificación.

Capítulo 4: Validación y pruebas

											ión. 2.1. Se guarda la planificación en la base de satos.
EC: 1.2 Adicionar planificación incorrectamente.	(vacío)	(PC) V	(5/04/2013) V	(5/04/2013) V	(8:30 am) V	(9:50 am) V			El sistema muestra en rojo que el campo es obligatorio.	Satisfactorio.	
EC: 1.3 Adicionar planificación. Cancelar.	(Plan) V	(PC) V	(5/04/2013) V	(5/04/2013) V	(8:30 am) V	(9:50 am) V			Cancela la operación y cierra la ventana para insertar nueva tarea de planificación.	Satisfactorio.	
EC: 1.4 Adicionar planificación. Cerrar.	(vacío)	(PC) V	(5/04/2013) V	(5/04/2013) V	(8:30 am) V	(9:50 am) V			Cancela la operación y cierra la ventana para insertar nueva tarea de planificación.	Satisfactorio.	

4.4 Pruebas de caja blanca.

Las pruebas de caja blanca, en ocasiones llamadas pruebas de cristal, es un método de diseño que usa la estructura de control descrita como parte del diseño al nivel de componentes para derivar los casos de pruebas. La prueba de caja blanca del software se basa en un examen cercano al detalle procedimental. Se prueban las vistas lógicas del software y la colaboración entre componentes, al proporcionar casos de pruebas que ejercen conjuntos específicos de condiciones, bucles o ambos. **(Pressman, 2005)**

En el caso de la técnica de las pruebas de caja blanca se le realizaron al módulo de Ejecución. Estas pruebas están dirigidas a las funcionalidades internas del sistema. Se ejecutan probando caminos lógicos por los cuales deberá transitar el sistema una vez puesto en funcionamiento y se observan los resultados arrojados en diferentes puntos; lo que permitió constatar que los resultados son los esperados. Estas pruebas se desarrollan de forma que se asegure que la operación interna se ajusta a las especificaciones, y que todos los componentes internos se han probado de forma adecuada.

Dentro de las pruebas de caja blanca se encuentra la prueba del camino básico. Este método permite que el diseñador de casos de pruebas obtenga una medida de complejidad de un diseño procedimental y que use esta medida como guía para definir un conjunto básico de rutas de ejecución. Los casos de pruebas derivados para ejercitar el conjunto básico deben garantizar que se ejecuta cada instrucción del programa por lo menos durante la prueba. **(Pressman, 2005)**

Esta prueba se basa en la obtención de la complejidad lógica de un diseño. El método consiste en la representación mediante un grafo de flujo asociado de todos los posibles caminos que se puedan iniciar en una determinada porción de código y aplicarles a cada uno de estos caminos un caso de prueba. **(Pressman, 2005)**

A la representación realizada por el grafo se le calculará la complejidad ciclomática. Los pasos a seguir para la aplicación del método son los siguientes:

- Dada una selección del código fuente se representa un grafo asociado.
- Se calcula la complejidad ciclomática del grafo.
- Se determina un conjunto básico de caminos independientes.
- Se elaboran los casos de pruebas que obligan a la ejecución de cada camino del conjunto básico.

Cada uno de los casos de pruebas que se elaboren y apliquen garantizarán la ejecución al menos una vez de cada uno de los caminos del grafo. A continuación se muestra el

Capítulo 4: Validación y pruebas

código del método vigilante que es el encargado de preguntar cada cierto tiempo a la cola de prioridades si la primera planificación que está en cola le toca ejecutarse en ese tiempo.

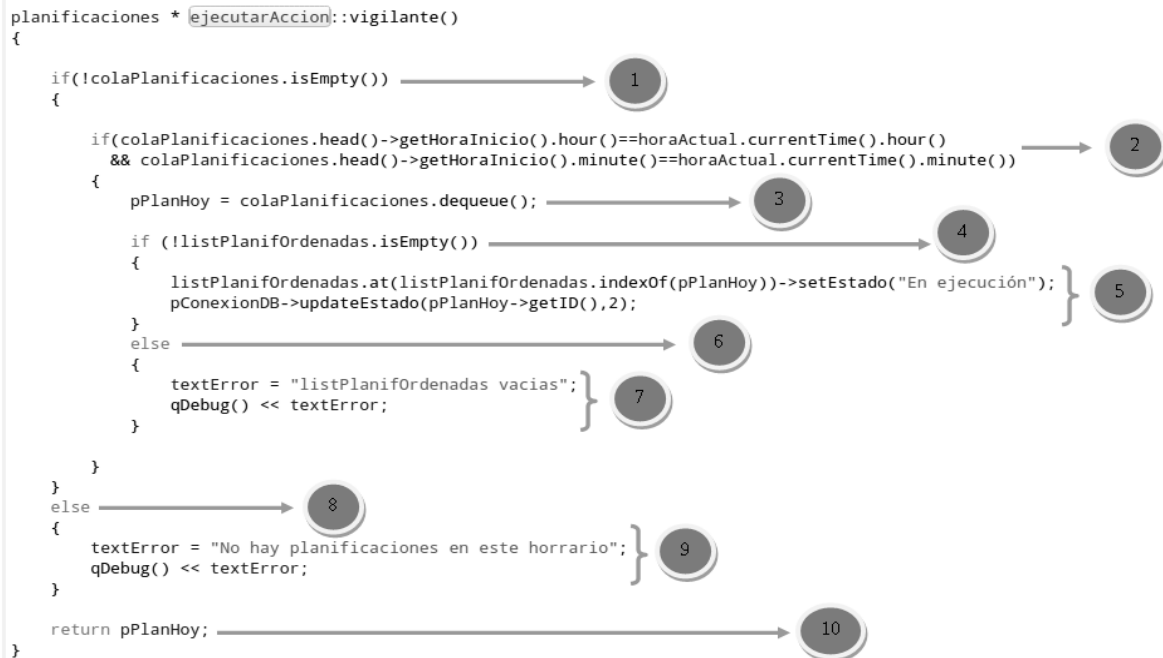


Figura 12. Representación del código del método vigilante.

A continuación se muestra el grafo de flujo asociado al código del método antes plasmado.

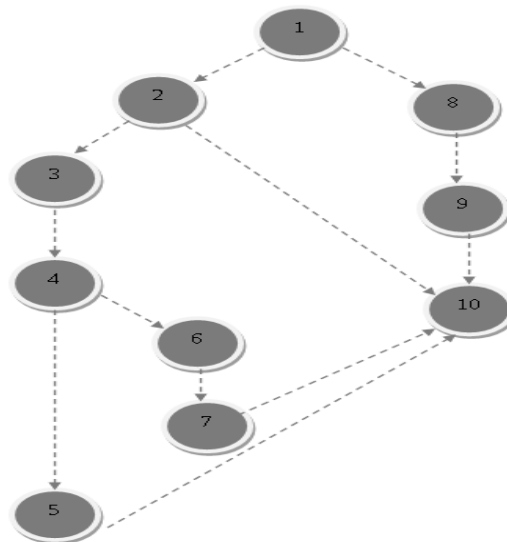


Figura 13. Grafo del flujo asociado al método vigilante.

Cálculo de la complejidad ciclomática:

La complejidad ciclomática es una métrica de software extremadamente útil pues proporciona una medición cuantitativa de la complejidad lógica de un programa. El valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa. Además, proporciona un límite superior para el número de pruebas que se deben realizar para asegurar que se ejecute cada sentencia al menos una vez.

Complejidad ciclomática $V(G) = \text{Cantidad de Aristas } [A] - \text{Cantidad de nodos } [N] + 2$.

- $V(G) = A - N + 2$
- $V(G) = 12 - 10 + 2$
- $V(G) = 4$

El valor obtenido para la complejidad ciclomática es 4, lo que deja claro que existen 4 posibles caminos a recorrer para el flujo; de igual forma es la cantidad mínima de pruebas que se le realizó al código.

Tabla 7. Caminos del grafo asociado del método vigilante.

No.	Camino
1	1-2-10
2	1-8-9-10
3	1-2-3-4-5-10
4	1-2-3-6-7-10

Dando continuidad al método y contando con el grafo y los caminos identificados previamente se pasa a la creación de casos de pruebas para cada uno de estos caminos. La tabla muestra el caso de prueba para el camino 1-2-10, los restantes casos de pruebas se encuentran en los anexos.

Tabla 8. Caso de prueba para el camino No 1.

Caso de prueba para el camino básico No. 1	
Camino	1-2-10
Descripción	Verifica que la cola no esté vacía. Seguidamente chequea la hora de inicio y la hora de fin. En caso que no cumpla con los parámetros retorna la planificación.

Entrada	Chequea cada cierto tiempo la cola de planificaciones.
Resultado esperado	Se manda a ejecutar la planificación.

Resultados de las pruebas.

Luego de terminada la fase de pruebas se identificaron un conjunto de no conformidades en los CU de la aplicación. Por lo que fue necesario realizarle una segunda iteración de pruebas al sistema. En esta segunda iteración se obtuvieron todos los resultados satisfactorios. Para finalmente poder comprobar que la aplicación cumple con los requisitos planteados al inicio del desarrollo y que responde exitosamente las peticiones de los usuarios.

4.5 Conclusiones parciales.

En este capítulo se desarrollaron las pruebas pertinentes a la aplicación, teniendo en cuenta la técnica que se le aplicaría a cada módulo, ya que ellos tienen características diferentes y se hacía necesario emplear técnicas heterogéneas. Mediante las pruebas de caja negra y caja blanca se pudo validar el funcionamiento del componente de acuerdo con los requisitos, asegurando de esta forma que cumpliera lo que se diseñó con lo que se implementó.

Conclusiones

Con la culminación de la presente investigación se logra finalizar el Componente de Gestión de Planificación Genérica para los productos del Departamento de Señales Digitales, centrando el proceso de planificación en un único módulo. De esta manera se da cumplimiento al objetivo general de la investigación.

Luego de la puesta en marcha del componente se establece una comunicación entre todos los productos del departamento, demostrando así que el mismo funciona correctamente. A través del mismo se puede realizar la planificación para cualquiera de los productos existentes en el departamento. Pero además cuenta con la flexibilidad de poder ser incluido a sistemas externos.

Con el desarrollo de este componente se logra reducir el tiempo de implementación de los productos, al contar con un sistema flexible que puede ser utilizado para planificar las tareas que necesiten realizar estos productos. La solución provee al país de una herramienta propia, bajo estándares y tecnologías libres. Finalmente, se logró documentar todo el proceso de desarrollo, aspecto muy importante para los desarrolladores que se dedican a futuras ampliaciones o actualizaciones del sistema.

Recomendaciones

Luego de culminar la aplicación y para un mayor alcance de la misma, se proponen las siguientes recomendaciones.

- Se recomienda continuar con el desarrollo de otros *plugin* de comunicación en diferentes tecnologías, lo que facilitará la asimilación del componente como parte de soluciones de software ya existentes.
- Integrar el módulo de planificación con el componente de seguridad del departamento.

Referencias bibliográficas

- Alfonso, Alex. 2010.** [Online] octubre 6, 2010. [Cited: enero 11, 2013.] <http://www.diigo.net/profile/sesser>.
- Carter, Naye. 2009.** <http://ticnaye.blogspot.com>. *http://ticnaye.blogspot.com*. [Online] junio 13, 2009. [Cited: enero 11, 2013.] <http://ticnaye.blogspot.com/2009/06/erp.html>.
- 2012.** *ecured.cu*. *ecured.cu*. [Online] diciembre 29, 2012. [Cited: enero 11, 2013.] http://www.ecured.cu/index.php/Qt_Creator.
- Escobar, Victor. 2004.** *Sistema de Planificación y Control de la Producción*. 2004.
- 2004.** *Sistemas de Producción*. 2004.
- 2010.** *gestiopolis.com*. *gestiopolis.com*. [Online] febrero 10, 2010. [Cited: enero 10, 2013.] <http://www.gestiopolis.com>.
- González, Francisco Ruiz. 2000.** *Sistema Gestor de Bases de Datos*. CASTILLA LA MANCHA : s.n., 2000.
- Jarquín, Miguel Ángel Nango. 2011.** [Online] junio 6, 2011. [Cited: enero 11, 2013.] <http://es.scribd.com/doc/91607845/Logistica-y-Cadenas-de-Suministro-Unidades-1-6>.
- Nápoli, Lic. Cecilia.** [Online] [Cited: enero 11, 2013.] <http://www.chubut.gov.ar/capacitacion/archives/040323.php>.
- Pérez, Isaías Carrillo. 2008.** *Metodologías de desarrollo*. [book auth.] Rodrigo Pérez González y Aureliano David Rodríguez Martín Isaías Carrillo Pérez. *Metodologías de desarrollo de software*. Universidad Politécnica de Valencia : s.n., 2008.
- Rivera, Ernesto. 2008.** *asesoriainformatica.com*. *asesoriainformatica.com*. [Online] 2008. [Cited: enero 11, 2013.] http://www.asesoriainformatica.com/erp_01.htm.
- Robalino, Ruben Dario. 2012.** *Auditoría Informática*. *es.scribd.com*. [Online] abril 13, 2012. [Cited: enero 11, 2013.] <http://es.scribd.com>.
- Rodriguez, Daily. 2011.** *s-operativosais.blogspot.com*. *s-operativosais.blogspot.com*. [Online] 2011. [Cited: enero 11, 2013.] <http://s-operativosais.blogspot.com/p/lenguaje-de-programacion.html>.
- Orallo, Enrique Hernández. 2003.** *El Lenguaje Unificado de Modelado (UML)*. *El Lenguaje Unificado de Modelado (UML)*. 2003.
- Herramientas Case*. **Gómez, Ruth Priscila Landeros. 2004.** Veracruzana : s.n., 2004.

Aponce, Luis Fonseca. 2010. Los Servidores Web . *Servidores Web* . Buenos Aires : s.n., 2010.

Herramientas Case. **Gómez, Ruth Priscila Landeros. 2004.** Veracruzana : s.n., 2004.

Patrones GRASP [Book Section] / auth. Tabares Ricardo Botero // Patrones Grasp y Anti-Patrones: un enfoque orientado a objetos desde logica de programacion / book auth. Tabares Ricardo Botero. - Antioquia : [s.n.], 2010.

ACHOUR, M y Otros. *Manual de PHP*. [En línea] Manual de PHP, 2012. [Consultado el 29 de Septiembre de 2012]. Disponible en: [<http://php.net/manual/es/index.php>].

ACHOUR, M y Otros. *Servicios Web*. [En línea] Manual de PHP, 2012. [Consultado el 5 de Noviembre de 2012]. Disponible en: [<http://php.net/manual/es/book.soap.php>].

APACHE. *Versión 2.2 del servidor HTTP Apache*. [En línea] Apache HTTP Server Project, 2012. [Consultado el 5 de Septiembre de 2012]. Disponible en: [<http://httpd.apache.org/docs/2.2/>].

DOJO. *The Dojo Toolkit API*. [En línea] Dojo Toolkit 1.8, 2012. [Consultado el 5 de Septiembre de 2012]. Disponible en: [<http://dojotoolkit.org/api/1.8/>].

GONZÁLEZ, Y. *Arquitectura del servidor de streaming distribuido ALLFRYS*. Tesis de ingeniería, Universidad de las Ciencias Informáticas, La Habana, 2011.

HTMLCINCO. *Las 15 APIs más populares de HTML5*. [En línea] Manual de HTML5 en español, 2012. [Consultado el 5 de Octubre de 2012]. Disponible en: [<http://www.htmlcinco.com/15-apis-de-html5/>].

ITU. *Evaluar el potencial de las TIC en el mundo*. [En línea] Aplicando su índice de desarrollo de las TIC la UIT compara a 154 países, 2009. [Consultado el 11 de Marzo de 2012]. Disponible en: [http://www.itu.int/ITU-D/ict/dai/material/DAI_ITUNews_s.pdf].

NAVARRETE, T. *El lenguaje JavaScript*. [En línea], 2007. [Consultado el 15 de septiembre de 2012]. Disponible en: [<http://www.dtic.upf.edu/~tnavarrete/fcsig/javascript.pdf>].

NOKIA. *Modular Class Library*. [En línea], 2008. [Consultado el 15 de febrero de 2011]. Disponible en: [<http://qt.nokia.com/products/library>].

POTENCIER, F. *Symfony 2.1.2 Released*. [En línea] Symfony, 2012. [Consultado el 15 de Octubre de 2012]. Disponible en: [<http://symfony.com/blog/symfony-2-1-2-released>].

CARRILLO, R. *Acerca de la planificación*. [En línea], 2008. [Consultado el 11 de Septiembre de 2012]. Disponible en: [<http://www.slideshare.net/jcfdezmxestra/que-es-planificar>].

(2000)

Desarrolloweb.com [Online] / auth. Lazaro Juliana Monteiro // desarrolloweb.com. - enero 1, 2001. - mayo 10, 2013. - <http://www.desarrolloweb.com>.

Documentación y pruebas. pruebas [Book Section] / auth. Fontela Pablo Suarez y Carlos // Documentación y pruebas. Antes del paradigma de objetos. / book auth. Fontela Pablo Suarez y Carlos. - 2003.

Gestión de las Pruebas Funcionales [Book Section] / auth. Lamancha Beatriz Pérez // Gestión de las Pruebas Funcionales / book auth. Lamancha Beatriz Pérez. - Montevideo, Uruguay : [s.n.], 2007.

<http://www.desarrolloweb.com> [Online] / auth. Alvarez Miguel Angel // <http://www.desarrolloweb.com>. - mayo 9, 2001. - mayo 10, 2013.

Ingeniería del Software: Un Enfoque Práctico [Book] / auth. Pressman Roger. - 2005.

Patrones GRASP [Book Section] / auth. Tabares Ricardo Botero // Patrones Grasp y Anti-Patrones: un enfoque orientado a objetos desde logica de programacion / book auth. Tabares Ricardo Botero. - Antioquia : [s.n.], 2010.

PLATAFORMA DE GESTIÓN DE PROYECTOS-REDMINE FUNCIONALIDADES [Book Section] // PLATAFORMA DE GESTIÓN DE PROYECTOS-REDMINE. - Santiago de Chile : [s.n.], 2000.

Acrónimos

DCUS: Diagrama de Casos de Uso del Sistema.

UCI: Universidad de las Ciencias Informáticas.

MVC: Modelo Vista Controlador.

CU: Caso de Uso.

SGBD: Sistema Gestor de Bases de Datos.

CPLAG: Componente de Gestión de Planificación Genérica.

CASE: Ingeniería de Software Asistida por Computadora. Del inglés Computer Aided Software Engineering.

GEYDED: Geoinformática y Señales Digitales.

HTTP: Protocolo de Transferencia de Hipertexto. Del inglés Hypertext Transfer Protocol.

IP: Protocolo de Internet. Del inglés Internet protocol.

RAM: Memoria de acceso aleatorio. Del inglés Random Access Memory.

RUP: Proceso Unificado Desarrollo. Del inglés Rational Unified Process.

UML: Lenguaje Unificado de Modelado. Del inglés Unified Modeling Language.

URL: Localizador Uniforme de Recursos. Del inglés Uniform Resource Locator.

XML: Lenguaje de marcas extensible. Del inglés de Extensible Markup Language.

PHP: Siglas de PHP Hypertext Pre-processor.

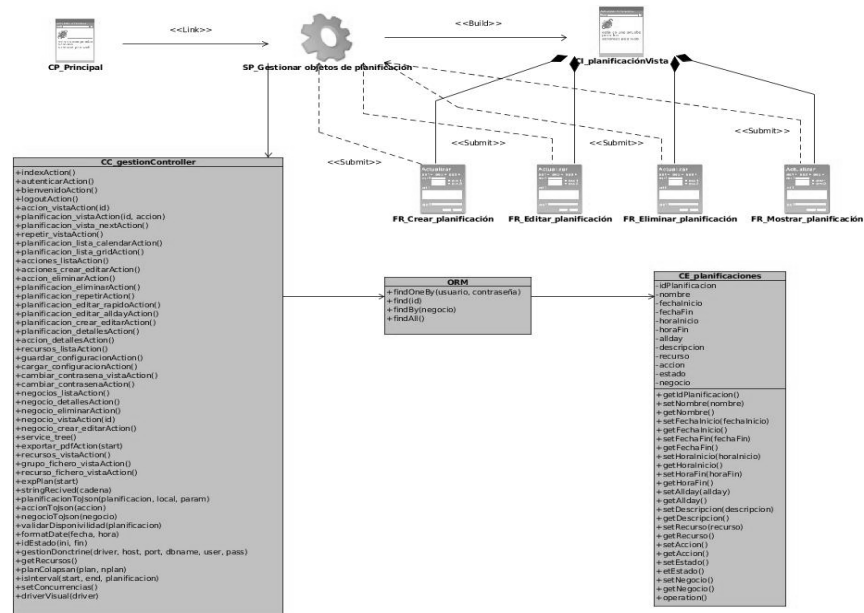
IDE: Entornos de Desarrollo Integrado. Del inglés Integrated Development Environment.

HTTP: Protocolo de transferencia de hipertexto. Del inglés Hypertext Transfer Protocol.

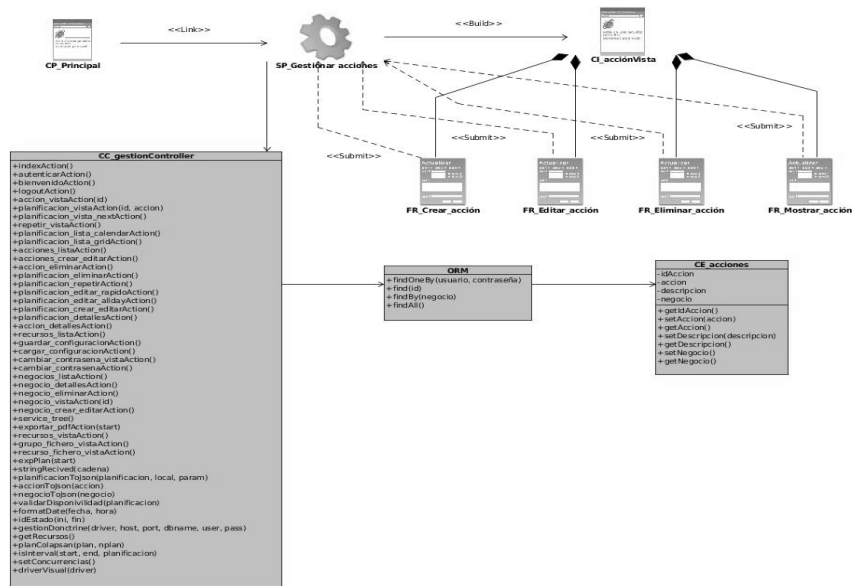
TCP: Protocolo de Control de Transmisión. Del inglés Transmission Control Protocol.

Anexos

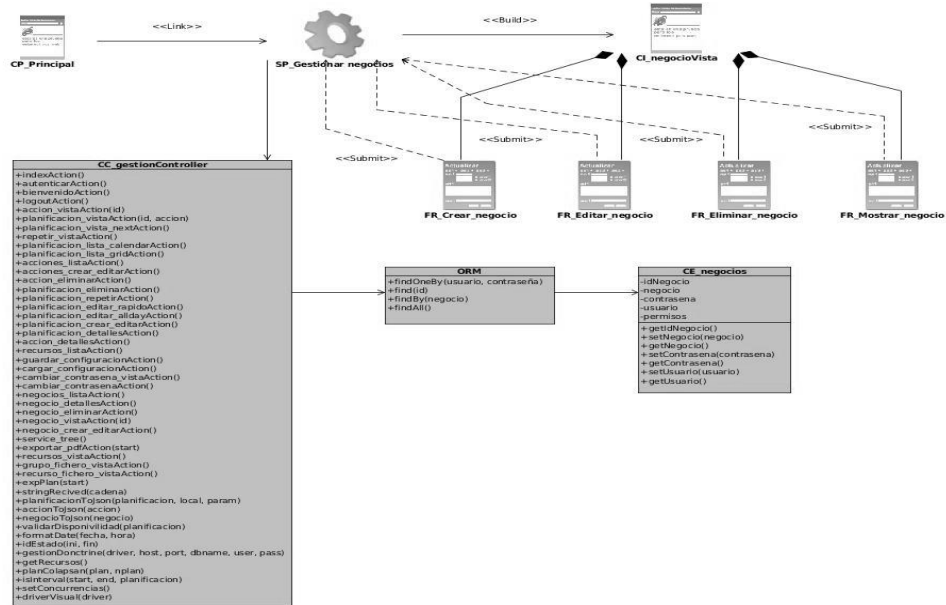
1. Diagrama de clases del diseño del CU Gestionar objetos de planificación.



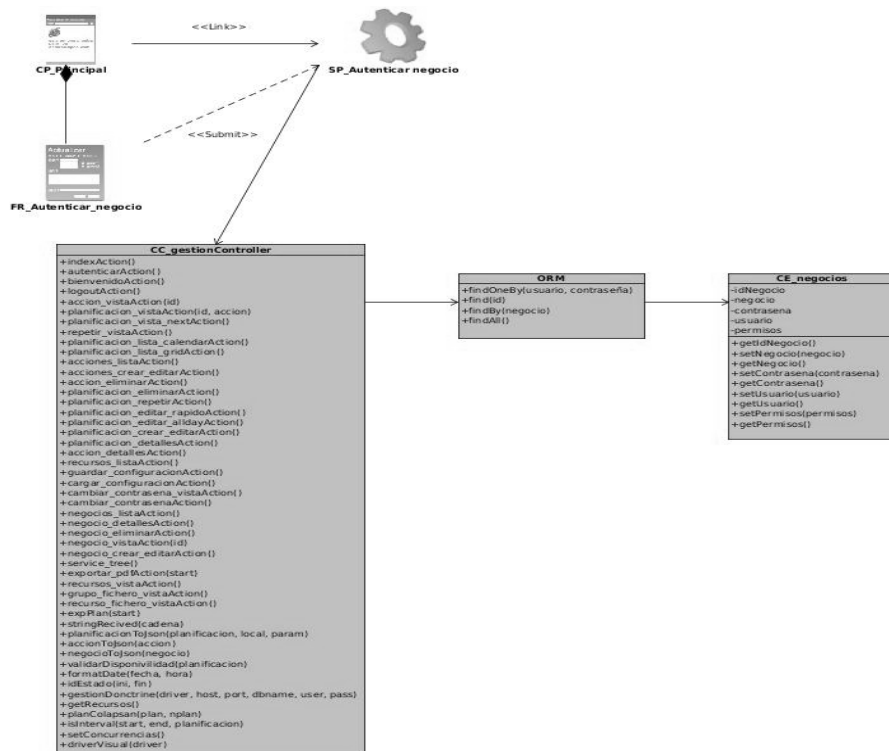
2. Diagrama de clases del diseño para el CU Gestionar acciones.



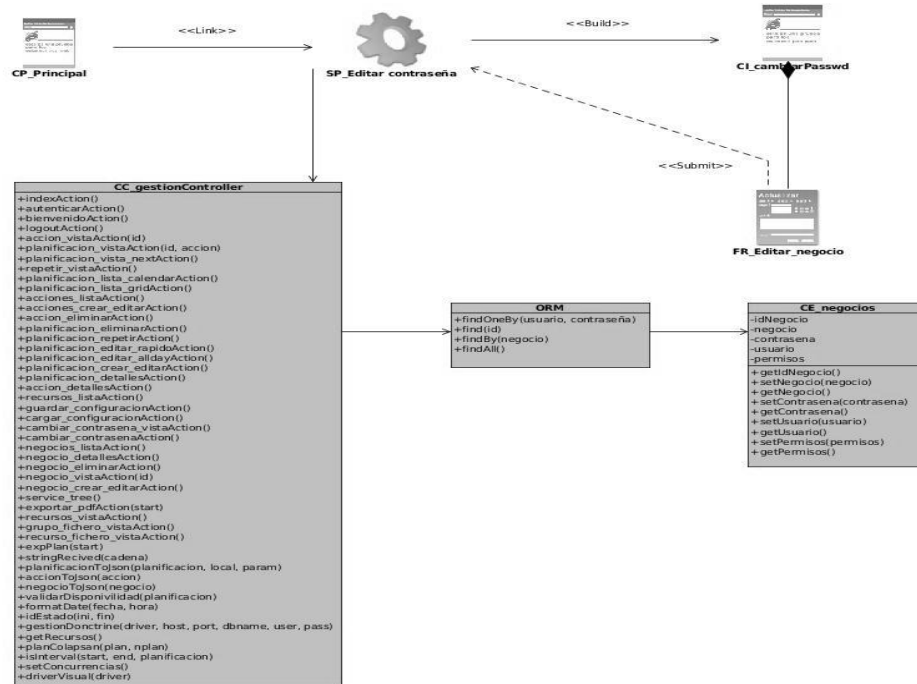
3. Diagrama de clases del diseño para el CU Gestionar negocios.



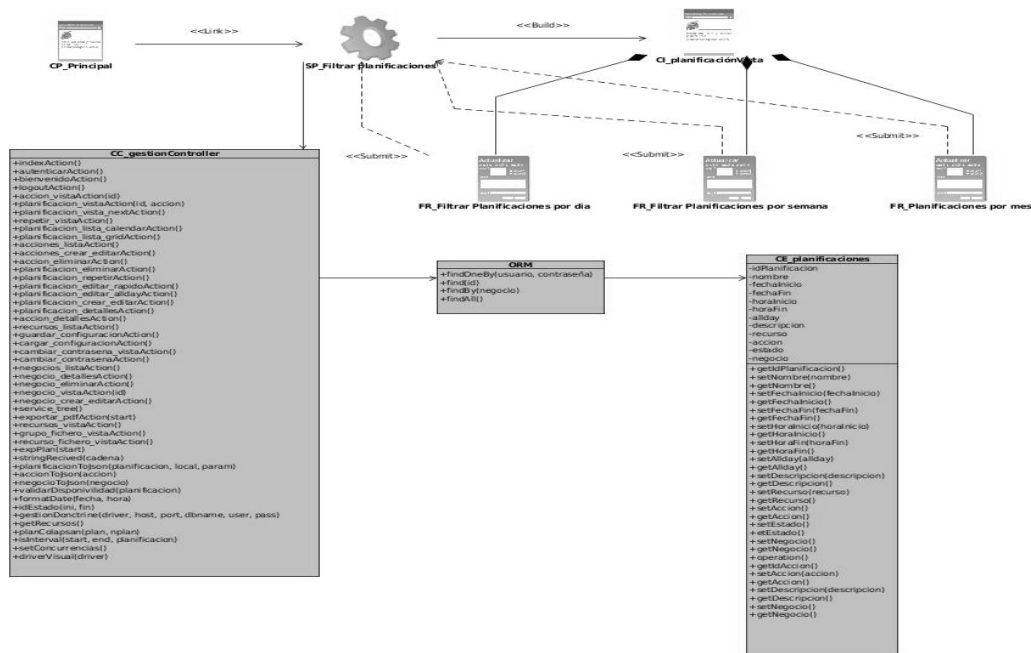
4. Diagrama de clases del diseño para el CU Autenticar negocio.



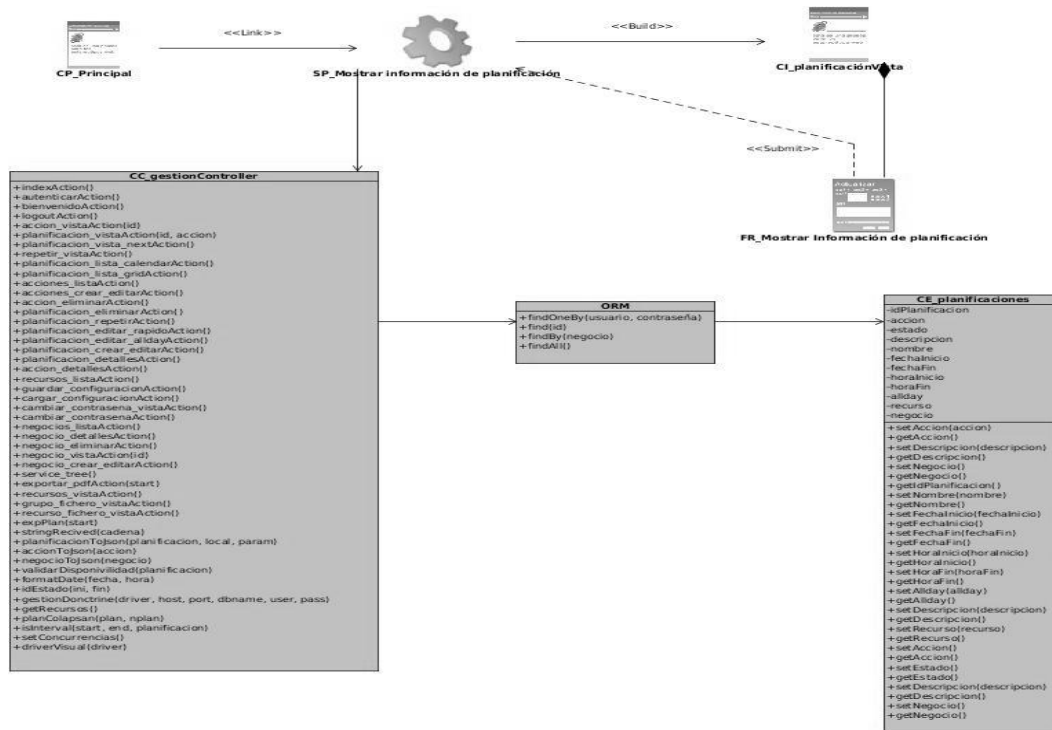
5. Diagrama de clases del diseño para el CU Editar contraseña.



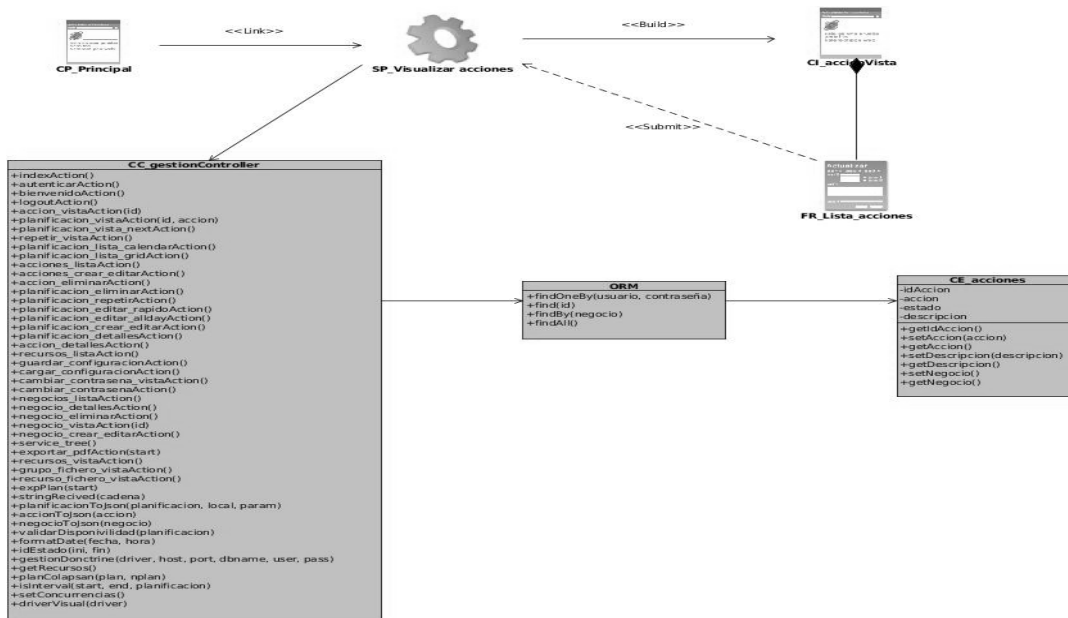
6. Diagrama de clases del diseño para el CU Filtrar planificaciones.



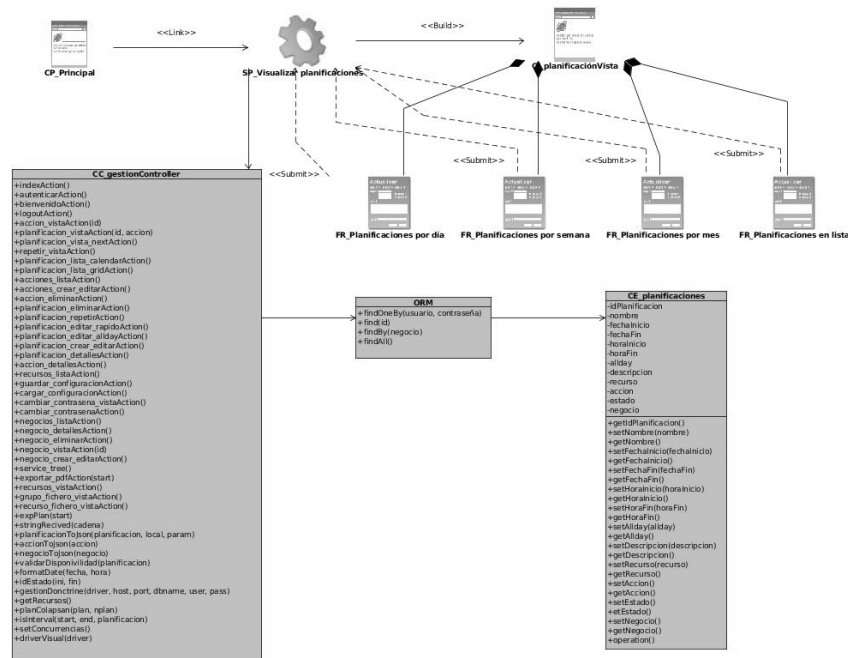
7. Diagrama de clases del diseño para el CU Mostrar información de planificación.



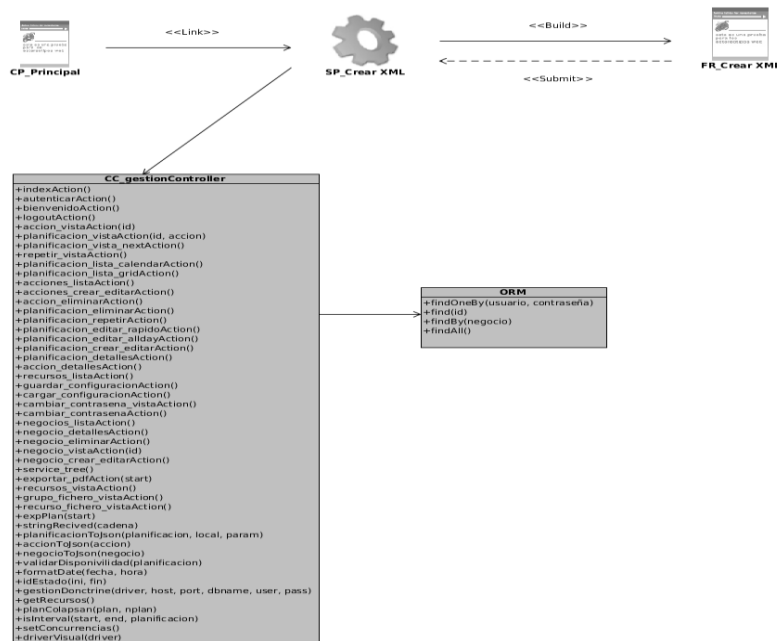
8. Diagrama de clases del diseño para el CU Visualizar acciones.



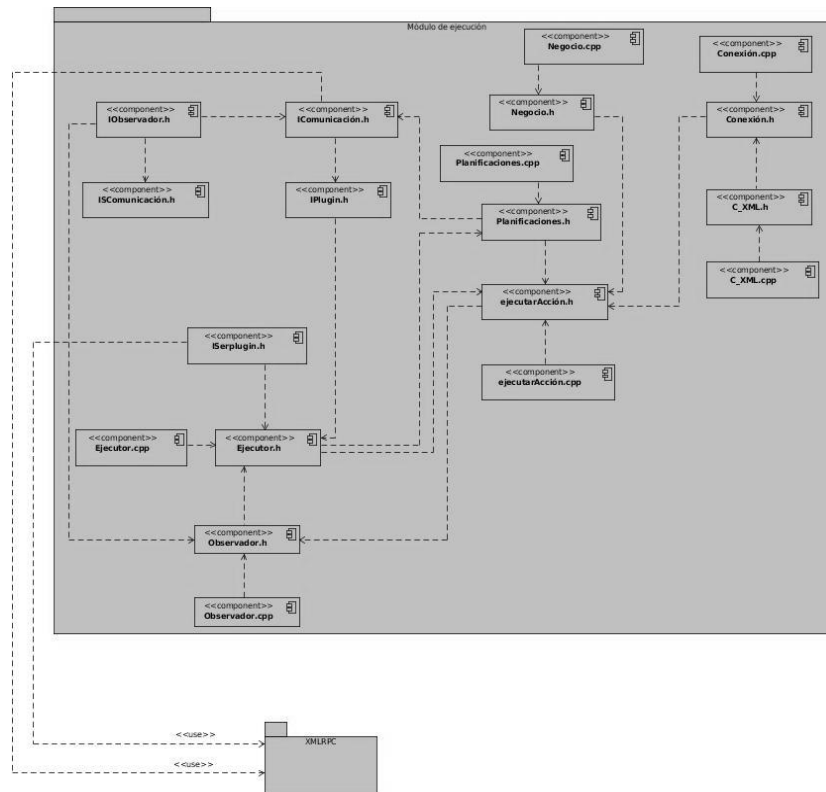
9. Diagrama de clases del diseño para el CU Visualizar planificaciones.



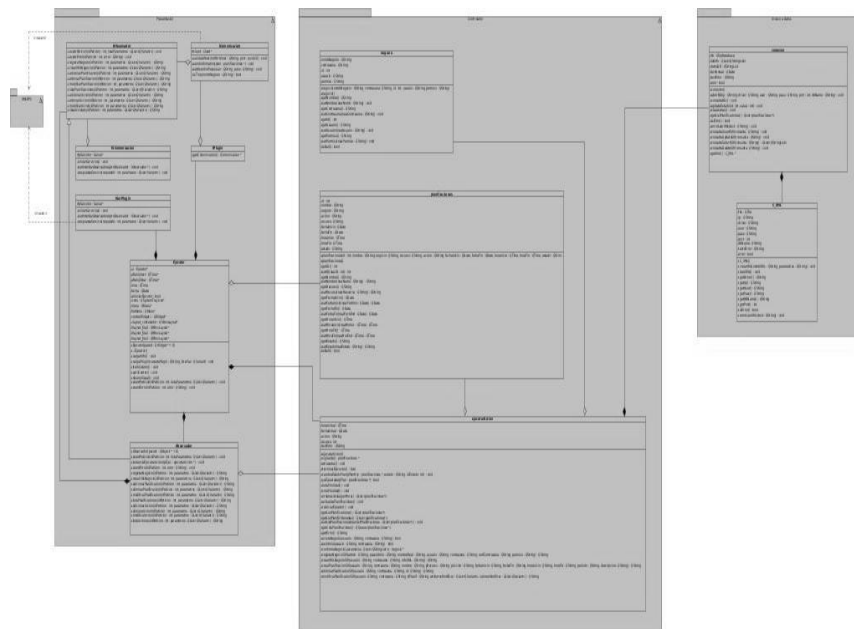
10. Diagrama de clases del diseño para el CU Crear XML.



11. Diagrama de implementación.



12. Diagrama de clases para el módulo de Ejecución.



13. Caso de prueba para el camino No 2.

Caso de prueba para el camino básico No. 2

Camino	1-8-9-10
Descripción	La cola está vacía, por lo tanto muestra un error que no puede ejecutar ninguna planificación porque la cola se encuentra vacía.
Entrada	Chequea cada cierto tiempo la cola de planificaciones.
Resultado esperado	Se muestra un mensaje de error diciendo que no hay planificaciones disponibles para mandar a ejecutar.

14. Caso de prueba para el camino No 3

Caso de prueba para el camino básico No. 3

Camino	1-2-3-4-5-10
Descripción	Verifica que la cola no esté vacía. Comprueba que las horas de inicio y de fin estén correctas. Toma la primera planificación y verifica que la lista no esté vacía y pone esa planificación en el estado en ejecución. Finalmente muestra la planificación.
Entrada	Chequea cada cierto tiempo la cola de planificaciones.
Resultado esperado	Mostrar la planificación.

15. Caso de prueba para el camino No 4.

Caso de prueba para el camino básico No. 4

Camino	1-2-3-4-6-7-10
Descripción	Verifica que la cola no esté vacía.

	Comprueba que las horas de inicio y de fin estén correctas. Toma la primera planificación y verifica que la lista no este vacía en caso de que este vacía muestra un error diciendo que la lista de planificaciones esta vacía. Muestra la planificación.
Entrada	Chequea cada cierto tiempo la cola de planificaciones.
Resultado esperado	Mostrar la planificación.