"Análisis y Diseño del Sistema para la Gestión de la Calidad: Módulo Planificación de los Recursos Involucrados en las Actividades de Gestión de la Calidad."

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Autor: Liniet Alfonso González

Tutores: Ing. Maria de los Angeles Rubalcaba Betancourt

Ing. Lidisvey Herrero González

Ciudad de La Habana, 2009.

DECLARACIÓN DE AUTORÍA

Declaro que Liniet Alfonso González es la única autora de este trabajo y autoriza a la Universidad de las Ciencias Informáticas (UCI) y a la facultad 8 para que hagan el uso que estimen pertinente con este trabajo.

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

_____

Firma del Autor

Liniet Alfonso González

_____ _____

Firma del Tutor Firma del Tutor

Ing. Maria de los Angeles Rubalcaba Betancourt Ing. Lidisvey Herrero González

"La inteligencia consiste no sólo en el conocimiento, sino también en la destreza de aplicar los conocimientos en la práctica."

Aristóteles

DEDICATORIA

A mis abuelitos lindos Fela, Remberto y Emilia por ser lo más preciado que me ha dado la vida, porque han sido mi mejor enseñanza, por su cariño, su amor, ternura y porque siempre, estén donde estén, estarán en mi corazón.

A mi mamá Esther y a mi papá Leonel por haberme dado la vida, por sus consejos, su dedicación, su amor, por estar ahí cada vez que los he necesitado, por confiar totalmente en mí y porque siempre han vivido y vivirán para mí.

A mi tatico lindo Pedro Julio por guiarme y ayudarme estos cinco años de mi carrera dedicándome todo su amor, comprensión, paciencia y porque lo quiero desde el día en que lo vi.

A mis primos que han sido mis hermanos Lidy, Maye e Isrra y a mi hermano Leonel los cuales siempre han estado para mí.

A mis tías y tíos por confiar en que sí podría Maritza, Airan, Mariela, Humberto e Ivón.

A mi suegra Ivón por su constante preocupación.

A todos aquellos que confiaron en mí y a los que me vieron crecer.

AGRADECIMIENTOS

A mis abuelitos Fela, Remberto, Emilia por quererme siempre como yo los quiero a ellos y por estar en este día.

A mi mamita Esther por educarme como lo ha hecho y a mi papá por confiar en la educación que me dio mi mamá y por yo quererlos tanto les regalo lo que soy hoy.

A mi familia, hermano, primos, tías, tíos, Mery por ayudarme en todo lo que han podido.

A mi suegra Ivón por traer al mundo al esposo más lindo que existe y por ser tan buena conmigo.

A mi esposo Pedro Julio (Pedri) porque no ha permitido que me rinda ni un minuto.

A mi otra familia Regla, Lucia, Adolfito, Lázaro, Yoania, Rosario, Estevan, Pedro, Iris por estar siempre atentos.

A mi tutora y amiga María que ha sido tan exigente, comprensiva, y hacer que hoy saliera lo mejor posible este trabajo.

A todos los profesores que estuvieron pendientes porque este trabajo saliera por sus criterios que me ayudaron a la perfección del trabajo y a profundizar en los conocimientos.

A mis amigos Lisandra Hernández, Adriana, Dayana, Yalida, Lisandra Díaz, Lily, Reinier, Alexander, Jorge, Yaneisy, María del Toro que permanecieron para cualquier cosa que hiciera falta en todo momento.

A mis compañeros de aula fundamentalmente a las hembras Grisel, Yeny, Lidivel, Yuni, Martha, Yadira y a todos mis amigos que me ayudaron en todos estos años a pasar momentos buenos y malos.

A los compañeros de mi esposo por ayudar en todo lo que pudieron.

RESUMEN

En el presente trabajo se realiza el análisis y diseño del Sistema para la Gestión de la Calidad de la facultad 8, módulo planificación de los recursos involucrados en las actividades de la Gestión de la Calidad; con el objetivo de lograr una mejor organización y eficiencia de los procesos de formación del personal, control de revisiones, auditorías y pruebas de software. Para la consecución de los objetivos propuestos se siguen los estándares de la metodología de desarrollo del Proceso Unificado del Software. Para ello se realiza un estudio de las herramientas que existen internacionalmente para la Gestión de la calidad, de las tecnologías, lenguajes y patrones arquitectónicos a utilizar. Definiéndose del lado del cliente CSS, HTML, Javascript y del lado del servidor PHP, para el almacenamiento de datos se propone PostgreSQL, y como patrón arquitectónico se define el Modelo- Vista Controlador. Se realiza el modelamiento del negocio, levantamiento de requisitos, análisis y diseño del Sistema, cumpliendo con los requisitos pedidos por el cliente y velando porque los artefactos que se generen de estos flujos de trabajo cumplan con las normas de calidad requeridas. Con lo cual quedarán sentadas las bases para la implementación del módulo de planificación.

Palabras claves: Planificación, Sistema de Gestión, Calidad.

Índice

Índice de Figuras

Índice de Tablas

Introducción

El auge en los procesos de desarrollo de software y por consiguiente en la producción de productos software, ha conllevado a que muchas empresas tengan que asumir diversas pérdidas debido a la ineficiente calidad de los mismos. Estas pérdidas son generalmente ocasionadas por varias causas, dentro de ellas se pueden destacar las innumerables deficiencias y defectos detectados en el producto, así como la realización de planificaciones que no se basan en estimaciones previas y que ocasionan incumplimientos en los cronogramas.

Es por esto que la calidad es un factor muy importante a tener en cuenta durante el desarrollo de un producto software, y dentro de ello la planificación de las diferentes actividades enfocadas a las revisiones, auditorías y liberaciones de los diferentes artefactos que se desarrollan en un proyecto productivo.

En la Universidad de las Ciencias Informáticas (UCI) ha crecido en los últimos años el desarrollo de los proyectos productivos de exportación nacional e internacional. Los mismos se encuentran distribuidos en las diferentes facultades de la universidad y distribuidos para su organización en polos productivos, por lo que en cada una de ellas existen grupos de calidad que funcionan paralelos a todos los proyectos productivos, con el fin de garantizar la Calidad en el proceso de desarrollo de software y del producto final. En la Facultad 8 existen dos polos productivos los cuales son de Multimedia, y Deporte; además se desarrolla un proyecto productivo de informatización de una entidad venezolana, el cual es de suma importancia para el gobierno venezolano, este proyecto es denominado CICPC. En cada uno de ellos debe existir un asesor de calidad, el cual debe velar que desde el inicio del proyecto exista la estructura correcta del Equipo de Calidad; y que estén definidas las actividades que se deben desarrollar durante el proceso de desarrollo, con el propósito de obtener un producto eficiente y que cuente con la calidad requerida.

De forma general la facultad consta de un grupo de calidad el cual realiza la función de asesoría y apoyo a los proyectos productivos de desarrollo. Al mismo tiempo está estructurado por diferentes áreas de trabajo que van a facilitar el progreso de los proyectos. Dentro de estas áreas se encuentran la de Formación, Investigación, Pruebas, Aseguramiento de la Calidad y Revisiones. Cada área de trabajo para su cumplimiento tiene que realizar diferentes aspectos a cumplir los cuales en su mayoría no se pueden realizar con eficiencia por los siguientes inconvenientes: no existe manera alguna de tener centralizada las

actividades que se desarrollan en el grupo de calidad, ni se tiene el control acerca de la formación de los estudiantes vinculados al grupo de calidad, entiéndase cantidad de cursos optativos acreditados, así como sus evaluaciones; y la plantilla de profesores. Los recursos tanto humanos como tecnológicos se planifican de forma manual, no hay forma de registrar la cantidad real disponible de recursos humanos y tecnológicos con que se cuenta. El proceso para realizar la planificación de auditorías, pruebas, así como el control de las revisiones de forma general a los diferentes proyectos de la facultad para mejora de la calidad del producto y funcionamiento de los mismos se realiza de forma manual o en herramientas como el Microsoft Word y Excel, los cuales no son eficientes por lo que hace que resulte tedioso este proceso, imposibilitando la entrega en tiempo de los productos.

Por lo anteriormente expuesto se plantea el ***problema científico***: ¿Cómo lograr una mejor organización en las actividades de planificación en el Grupo de Calidad en la facultad 8?

Por lo que la ***idea a defender*** es que si se automatiza la planificación de las actividades de formación del personal, control de revisiones, auditorías y pruebas de software en la facultad 8 se garantizará una mayor eficiencia y control de las tareas.

Para dar solución al problema descrito se propone como ***objetivo general*** de la investigación: elaborar el análisis y diseño de un módulo que contribuya a la planificación de las actividades del Sistema para la Gestión de la Calidad en la Facultad 8, el cual permitirá la automatización para obtener la eficiencia de los procesos de formación del personal, control de revisiones, auditorías y pruebas de software.

De estos se derivan los siguientes ***objetivos específicos***:

- ✓ Definir el proceso de planificación para las actividades que se desarrollan en el proyecto de Calidad de la facultad 8.
- ✓ Desarrollar el flujo de trabajo de Análisis y Diseño para el módulo de planificación.

Para poder dar solución al problema científico el ***objeto de estudio*** se enmarca en el proceso de desarrollo para la gestión de la calidad en la Universidad de las Ciencias Informáticas, específicamente como ***campo de acción*** el análisis y diseño de aplicaciones web dedicadas a la planificación de las actividades de calidad en la Facultad 8.

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

1. Caracterizar las tendencias actuales relacionadas con la planificación de las actividades para la gestión de la calidad en el mundo.

2. Realizar entrevistas a especialistas para identificar acciones de planificación en los procesos a partir de la experiencia acumulada.
3. Investigar las metodologías de desarrollo de software existentes para sistemas de gestión, seleccionando la más adecuada.
4. Analizar las herramientas CASE existentes y escoger la que se adapta a esta aplicación.
5. Desarrollar el modelado del negocio para el proceso de planificación de la calidad en la Facultad 8.
6. Definir los requisitos funcionales del sistema a modelar.
7. Realizar el análisis y diseño del Módulo Planificación del sistema para la Gestión de la Calidad en la Facultad 8.

Capítulo I. Fundamentación Teórica

1.1 Introducción

En el presente capítulo se abordarán algunos conceptos referentes a la planificación; además de recoger las tendencias y tecnologías actuales a través de un estudio profundo del estado del arte en las cuales se encuentra emergida la planeación, así como conceptos de sistemas de gestión para la calidad, principalmente en lo referente a la planificación.

1.2 Tendencias actuales de la planificación

Actualmente en el mundo existen diferentes herramientas para los sistemas de gestión de la calidad (SGC) entre los cuales se encuentran KMKey Quality, ISODOC® y ISOLUCIÓN v.3, estos en su estructura presentan diferentes módulos que les ayuda a su organización y funcionamiento. La herramienta KMKey Quality incorpora un Gestor Documental donde almacenan las distintas revisiones de los documentos propios de un sistema de calidad, tales como el Manual de Calidad, la Política y los Objetivos de Calidad, el Organigrama, el Mapa de Procesos, los Procedimientos, los Formatos o las Instrucciones de Trabajo, permitiendo siempre un acceso directo a la revisión vigente. Este permite que desde el gestor de expedientes cualquier usuario autorizado pueda introducir No Conformidades, así como planificar las correspondientes Acciones Correctivas o Preventivas, repartiendo las tareas a cada uno de los responsables, y controlando plazos y acciones realizadas, Planes de Formación, Planes de Auditorías o Planes de Mejora Continua, el Seguimiento y Evaluación de Proveedores o la Gestión y Control de Indicadores.

Otra de las herramientas de SGC es ISOLUCIÓN v.3 Sistema Integral para Planeación, Implantación, Administración y Mantenimiento del Sistema de Gestión de Calidad, está diseñado sobre tecnología Web y su acceso es vía Internet o a través de su red interna (Intranet), permitiendo hacer una eficiente distribución de la información, los recursos y las actividades del sistema de gestión de calidad. Apoyando así su operación y funcionamiento, reduciendo costos y aumentando el impacto de productividad sobre la organización. Está conformado por 15 módulos los cuales conforman la organización del sistema entre

ellos está el Sistema de Gestión Documental, Manual de Calidad, Listados Maestros, Auditorías, Talento Humano, entre otros.

Estas herramientas a pesar de que presenta una estructura correcta en su organización no satisface, es decir, no cumple con las características de lo que realmente se pretende con la aplicación que se quiere analizar y diseñar, para luego implementar, ya que se enfocará en los problemas fundamentales que afectan al grupo de calidad para poder obtener una organización óptima.

En la UCI antes de empezar cada proyecto productivo en la realización de sus productos se realiza una planificación para poder estimar que tiempo durará la fabricación del software. Además se tiene en cuenta los riesgos que se pueden presentar en la elaboración del mismo. Los proyectos para su organización se basan en el uso de metodologías donde independientemente la que utilice determinado proyecto, es necesario realizar planificaciones para alcanzar lo deseado del producto para de esta forma ir encontrando la manera de evaluar la calidad del mismo.

Entre las herramientas más utilizadas en la universidad para la planificación están el Microsoft Excel, Microsoft Proyect, etc. En la facultad 8 en el grupo de calidad a raíz de tener como principales herramientas las antes mencionadas y hacerse tedioso a la hora de elaborar la planificación en la cual no existe un planificador para su elaboración y los profesores tienen que emplear tiempo en la elaboración de un plan; se ha decidido realizar un software que permita una serie de facilidades en todas las áreas de trabajo, con el objetivo de obtener una mayor organización en la panificación de la confección de un producto software. Además la planificación contribuye a actividades ordenadas y con un propósito. Todos los esfuerzos están apuntados hacia los resultados deseados.

1.3 Calidad de software

El término calidad ha ido evolucionando de forma continua a lo largo de su desarrollo. Muchos autores han dado su definición, la cual de forma general plantea que: "Es la totalidad de aspectos y características de un producto o servicio que se refieren a su capacidad para satisfacer necesidades dadas en la adecuación de sus objetivos'".[1]

De forma general Pressman define la calidad de software como:

Calidad de software: Concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo explícitamente documentados, y con las características implícitas que se espera de todo software desarrollado profesionalmente.[2]

Reuniendo de forma conjunta las características expuestas anteriormente se puede decir, que la calidad de software no es más que, todos los requisitos que necesariamente deben estar presente en un software de forma documentada, donde a través de su seguimiento y mediante revisiones este cuente con lo requerido logrando de este modo proporcionar la satisfacción del cliente.

1.4 Planificación

En todo el transcurso de la fabricación de cualquier producto software es de suma importancia realizar una adecuada planificación, permitiendo, una organización favorable para permitir el cumplimiento de entrega del producto y contribuyendo a la calidad del mismo. Para ello se puede ver las siguientes definiciones:

“Es el proceso consciente de selección y desarrollo del mejor curso de acción para lograr el objetivo.” [3]

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

“Es el proceso de seleccionar información y hacer suposiciones respecto al futuro para formular las actividades necesarias para realizar los objetivos organizacionales”. [3]
“Consiste en decidir con anticipación lo que hay que hacer, quién tiene que hacerlo, y cómo deberá hacerse”. [3]

Según las definiciones anteriores se llega a la conclusión que ***planificación*** es la manera de anticipar de forma organizada qué se va hacer, quién tiene que hacerlo y cómo deberá hacerse, teniendo en cuenta los recursos con que se disponen y los factores que puedan influir en el alcance de sus objetivos.

1.5 Gestión de la calidad

La gestión de la calidad es un conjunto de actividades donde se determina la calidad, los objetivos y las responsabilidades, las cuales le brindan el funcionamiento como un todo a un sistema de calidad en una empresa donde se implanta por medios tales como la planificación, el control, el aseguramiento (garantía) y la mejora, en el marco del sistema de calidad.[4]

Es la función directiva que determina y aplica la política de la calidad expresada formalmente por la Dirección. Comprende el conjunto de actividades mediante las cuales se alcanza la aptitud al uso, independientemente de donde se realicen.[5]

Gestionar es coordinar todos los recursos disponibles para conseguir unos objetivos. El fin de la calidad es satisfacer al cliente y al proveedor con el mínimo coste combinado.

La gestión de la calidad comprende tres procesos, planificación, organización y control.

En conjunto los procesos de la calidad incluyen todas las maneras de organización en la cual estará estructurada la gestión de la calidad, la cual va a estar destinada principalmente a los clientes. Además esta gestión estará dirigida por un líder que será el encargado de mantener la organización en su equipo de desarrollo; viéndose así, la participación de este unido al personal de trabajo, donde se tomarán decisiones para integrar toda la información que se pueda recopilar de forma organizada y contribuyendo a la organización de los procesos de la gestión de la calidad.

De manera más completa la ***gestión de la calidad*** no puede estar ajena a ninguna institución que se dedique a la producción de productos informáticos. Además contribuye a una buena organización y orientación del trabajo en la producción de los productos de software.

1.5.1 Aseguramiento de la calidad

El aseguramiento de calidad del software es el conjunto de actividades planificadas y sistemáticas necesarias para aportar la confianza en que el producto (software) satisfará los requisitos dados de calidad.

Antes de comenzar el desarrollo de una aplicación se realiza un plan de aseguramiento de calidad del software, el cual es diseñado antes de comenzar su desarrollo. El mismo está presente en:

- ✓ Métodos y herramientas de análisis, diseño, programación y prueba.
- ✓ Inspecciones técnicas formales en todos los pasos del proceso de desarrollo del software.
- ✓ Estrategias de prueba multiescala.
- ✓ Control de la documentación del software y de los cambios realizados.
- ✓ Procedimientos para ajustarse a los estándares (y dejar claro cuando se está fuera de ellos).
- ✓ Mecanismos de medida (métricas).
- ✓ Registro de auditorías y realización de informes.

En las actividades que se pueden desarrollar para el aseguramiento de la calidad del software se tiene:

- ✓ Métricas de software para el control del proyecto
- ✓ Verificación y validación del software a lo largo del ciclo de vida [4]

Pressman plantea una serie de actividades que se deben realizar para el mejor funcionamiento antes del comienzo del proyecto en cuanto el Aseguramiento de la calidad las cuales son:

El establecimiento del plan de calidad del proyecto: Se realiza en las primeras etapas del proyecto y es un documento que planifica y rige todas las actividades de aseguramiento de la calidad así como la forma de aplicación en el proyecto.

La participación en el desarrollo de la descripción del proceso de software del proyecto: Es tarea del equipo de aseguramiento de la calidad la revisión del proceso que se lleva a cabo en el proyecto vigilando que se ajuste a las políticas y los estándares internos del software.

La revisión de las actividades de ingeniería del software para verificar su ajuste al proceso de software definido: El grupo de aseguramiento de la calidad identifica, documenta y sigue la pista de las desviaciones desde el proceso y verifica que se han hecho las correcciones.

Auditoría de los productos de software designados para verificar el ajuste con los definidos como parte del proceso del software: El grupo de aseguramiento de la calidad revisa los productos

seleccionados; identifica, documenta y sigue la pista de las desviaciones; verifica que se han hecho las correcciones, e informa periódicamente de los resultados de su trabajo al gestor del proyecto.

Asegurar la documentación de los productos de software: Documentar debidamente toda actividad que se realice en el proyecto es una práctica importante que se debe seguir para obtener un producto de calidad. El equipo de aseguramiento de la calidad está encargado de desarrollar una estrategia para la revisión de la documentación que se genera.

Registrar los desajustes en concordancia con los requisitos: Consiste en darle seguimiento a estos errores hasta que sean resueltos.[6]

De forma más concisa se puede observar que ***Aseguramiento de la calidad*** no es más que la forma en la que se planifica de forma sistémica un proyecto asegurando que cuente con los requisitos establecidos por el cliente y este obtenga un producto final de acuerdo a lo requerido por el mismo. Esto se puede obtener mediante auditorías periódicas permitiendo de esta manera el conocimiento de en qué estado se encuentra el producto hasta un momento determinado, y entre más rápido se detecten los problemas se evitará que se propaguen los errores en el transcurso del proceso de desarrollo del software.

1.6 Sistemas de Gestión de la calidad (SGC)

Un **sistema de gestión de la calidad** es la forma en la que una empresa o institución dirige y controla todas las actividades que están asociadas a la calidad.

Las **partes** que componen el sistema de gestión son:

1. **Estructura organizativa:** departamento de calidad o responsable de la dirección de la empresa.
2. Cómo se **planifica** la calidad.
3. Los **procesos** de la organización.
4. **Recursos** que la organización aplica a la calidad.
5. **Documentación** que se utiliza.[7]

Que una empresa tenga implantado un sistema de gestión de la calidad, sólo quiere decir que esa empresa gestiona la calidad de sus productos y servicios de una forma ordenada, planificada y controlada.

El enfoque a través de un sistema de gestión de la calidad anima a las organizaciones a analizar los requisitos del usuario, definir los procesos que contribuyen al logro de productos aceptables para el cliente y a mantener estos procesos bajo control. Un sistema de gestión de la calidad puede proporcionar el marco de referencia para la mejora continua con objeto de incrementar la probabilidad de aumentar la satisfacción del cliente y de otras partes interesadas. Proporciona confianza tanto a la organización como a sus clientes, de su capacidad para proporcionar productos que satisfagan los requisitos de forma coherente.

Un enfoque para desarrollar e implementar un sistema de gestión de la calidad comprende diferentes etapas tales como:

a) Determinar las necesidades y expectativas de los clientes y de otras partes interesadas.
b) Establecer la política y objetivos de la calidad de la organización.
c) Determinar los procesos y las responsabilidades necesarias para el logro de los objetivos de la calidad.
d) Determinar y proporcionar los recursos necesarios para el logro de los objetivos de la calidad.
e) Establecer los métodos para medir la eficacia y eficiencia de cada proceso.
f) Aplicar estas medidas para determinar la eficacia y eficiencia de cada proceso.
g) Determinar los medios para prevenir no conformidades y eliminar sus causas.
h) Establecer y aplicar un proceso para la mejora continua del sistema de gestión de la calidad.[7]

Un enfoque similar es también aplicable para mantener y mejorar un sistema de gestión de la calidad ya existente. Una organización que adopte el enfoque anterior genera confianza en la capacidad de sus procesos y en la calidad de sus productos, y proporciona una base para la mejora continua. Esto puede conducir a un aumento de la satisfacción de los clientes y de otras partes interesadas y al éxito de la organización.

Para ello se tiene en cuenta los siguientes tipos de documentos los cuales son utilizados en los sistemas de gestión de la calidad:

a) Documentos que proporcionan información coherente, interna y externamente, acerca del sistema de gestión de la calidad de la organización; tales documentos se denominan manuales de la calidad.
b) Documentos que describen cómo se aplica el sistema de gestión de la calidad a un producto, proyecto o contrato específico; tales documentos se denominan planes de la calidad.
c) Documentos que establecen requisitos; tales documentos se denominan especificaciones.
d) Documentos que establecen recomendaciones o sugerencias; tales documentos se denominan guías.
e) Documentos que proporcionan información sobre cómo efectuar las actividades y los procesos de manera coherente; tales documentos pueden incluir procedimientos documentados, instrucciones de trabajo y planos.
f) Documentos que proporcionan evidencia objetiva de las actividades realizadas o resultados obtenidos; tales documentos son conocidos como registros.[7]

Cada organización determina la extensión de la documentación requerida y los medios a utilizar. Esto depende de factores tales como el tipo y el tamaño de la organización, la complejidad e interacción de los procesos, la complejidad de los productos, los requisitos de los clientes, los requisitos reglamentarios que sean aplicables, la competencia demostrada del personal y el grado en que sea necesario demostrar el cumplimiento de los requisitos del sistema de gestión de la calidad.

1.7 Análisis y Gestión de Riesgos

Según Pressman, riesgo de software ha traído consigo debates sobre su definición adecuada, hay acuerdo común en que el riesgo siempre implica dos características:

- ✓ ***Incertidumbre****: el* acontecimiento que caracteriza al riesgo puede o no puede ocurrir; por ejemplo, no hay riesgos de un 100 por 100 de probabilidad.
- ✓ ***Pérdida:*** si el riesgo se convierte en una realidad, ocurrirán consecuencias no deseadas o pérdidas.

Cuando se analizan los riesgos es importante cuantificar el nivel de incertidumbre y el grado de pérdidas asociado con cada riesgo. Para hacerlo, se consideran diferentes categorías de riesgos los cuales son:

- ✓ Los riesgos del proyecto: amenazan al plan del proyecto.
- ✓ Los riesgos técnicos: amenazan la calidad y la planificación temporal del software que hay que producir.
- ✓ Los riesgos del negocio: amenazan la viabilidad del software a construir.

Las Fuerzas Aéreas de Estados Unidos [AFC88] han redactado un documento que contiene excelentes directrices para identificar riesgos del software y evitarlos. El enfoque de las Fuerzas Aéreas requiere que el gestor del proyecto identifique los controladores del riesgo que afectan a los componentes de riesgo del software: rendimiento, coste, soporte y planificación temporal.

En el contexto de este estudio, los componentes de riesgo se definen de la siguiente manera:

- ✓ Riesgo de rendimiento: el grado de incertidumbre con el que el producto encontrará sus requisitos y se adecue para su empleo pretendido.
- ✓ Riesgo de coste: el grado de incertidumbre que mantendrá el presupuesto del proyecto.
- ✓ Riesgo de soporte: el grado de incertidumbre de la facilidad del software para corregirse, adaptarse y ser mejorado.
- ✓ Riesgo de la planificación temporal: el grado de incertidumbre con que se podrá mantener la planificación temporal y de que el producto se entregue a tiempo.[2]

Integrando las características de una manera más entendible los riesgos no son más que problemas que pueden ocurrir en todo el transcurso del proceso de desarrollo de un producto. Además estos pueden ser medibles dependiendo del software a realizar a través de métricas. Los riesgos deben tenerse en cuenta desde la fase de inicio del proceso de desarrollo de un software para evitar consecuencias que afecten a la calidad del producto, principalmente a la hora de realizar la planificación en la cual uno de los riesgos que la afecta es la organización de los recursos humanos y tecnológicos. De esta forma si se tiene controlado los distintos riesgos, el producto saldrá con la calidad requerida por el cliente.

1.8 Estimación

Una de las actividades cruciales del proceso de gestión del proyecto es la planificación. A la hora de realizar una planificación se obtienen estimaciones del esfuerzo humano requerido, de la duración

cronológica, del proyecto y del costo. Se han desarrollado varias técnicas de estimación para el desarrollo de software, aunque cada una tiene sus puntos fuertes y débiles, todas tienen en común los siguientes atributos:

1. Se han de establecer de antemano el ámbito del proyecto.
2. Como bases para la realización de estimaciones se usan métricas del software de proyectos pasados.
3. El proyecto se desglosa en partes más pequeñas que se estiman individualmente.[8]

En general estimar no es más que dividir el tiempo para la realización de un sistema a través de la planeación, es decir, ver en qué tiempo se podría realizar una tarea o actividad para obtener una eficiencia en la entrega y desarrollo del producto.

1.9 Tendencias de las técnicas de captura de requisitos

Según IEEE (The Institute of Electrical and Electronics Engineers) los requisitos son la condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo.

Los requisitos deben cumplir con diferentes características las cuales son:

- ✓ Especificado por escrito.
- ✓ Posible de probar o verificar.
- ✓ Conciso.
- ✓ Completo.
- ✓ Consistente.
- ✓ No ambiguo.

En la ingeniería de requerimientos existen actividades a realizar las cuales se encuentran divididas en 4 partes: identificación, análisis, especificación y validación, y serán explicadas a continuación cada una de ellas.

Identificación: Es el nombre comúnmente dado a las actividades involucradas en el descubrimiento de los requerimientos del sistema. Los analistas de requerimientos deben trabajar junto al cliente para

descubrir el problema que el sistema debe resolver, los diferentes servicios que el sistema debe prestar, las restricciones que se pueden presentar, etc.

Análisis: En esta etapa se leen los requerimientos, se conceptúan, se investigan, se intercambian ideas con el resto del equipo, se resaltan los problemas, se buscan alternativas y soluciones, y luego se van fijando reuniones con el cliente para discutir los requerimientos.

Especificación: En esta fase se documentan los requerimientos acordados con el cliente, en un nivel apropiado de detalle. En la práctica, esta etapa se va realizando conjuntamente con el análisis, se puede decir que la especificación es el "pasar en limpio" el análisis realizado previamente aplicando técnicas y/o estándares de documentación, como la notación UML (Lenguaje de Modelado Unificado), que es un estándar para el modelado orientado a objetos, por lo que los casos de uso y la obtención de requerimientos basada en casos de uso se utiliza cada vez más para la obtención de requerimientos.

Validación: La validación es la etapa final de la Ingeniería de Requerimientos. Su objetivo es, ratificar los requerimientos, es decir, verificar todos los requerimientos que aparecen en el documento especificado para asegurarse que representan una descripción, por lo menos, aceptable del sistema que se debe implementar. Esto implica verificar que los requerimientos sean consistentes y que estén completos.[9]
Dentro de los requerimientos existen técnicas que son factibles a la hora de la obtención del mismo, se pueden citar algunas de ellas:

Entrevistas y Cuestionarios: Se emplean para reunir información proveniente de personas o de grupos. Durante la entrevista, el analista conversa con el encuestado; el cuestionario consiste en una serie de preguntas relacionadas con varios aspectos de un sistema.

Sistemas existentes: Esta técnica consiste en analizar distintos sistemas ya desarrollados que estén relacionados con el sistema a ser construido. Por un lado, podemos analizar las interfaces de usuario, observando el tipo de información que se maneja y cómo es manejada, por otro lado también es útil analizar las distintas salidas que los sistemas producen (listados, consultas, etc.), porque siempre pueden surgir nuevas ideas sobre la base de estas.

Lluvia de ideas (Brainstorm): Este es un modelo que se usa para generar ideas. La intención en su aplicación es la de generar la mayor cantidad posible de requerimientos para el sistema.

Las reglas básicas a seguir son:

- ✓ Los participantes deben pertenecer a distintas disciplinas y, preferentemente, deben tener mucha experiencia. Esto trae aparejado la obtención de una cantidad mayor de ideas creativas.
- ✓ Conviene suspender el juicio crítico y se debe permitir la evolución de cada una de las ideas, porque si no se crea un ambiente hostil que no alienta la generación de ideas.
- ✓ Por más locas o salvajes que parezcan algunas ideas, no se las debe descartar, porque luego de maduradas probablemente se tornen en un requerimiento sumamente útil.
- ✓ A veces ocurre que una idea resulta en otra idea, y otras veces se pueden relacionar varias ideas para generar una nueva.
- ✓ Escribir las ideas sin censura.

Prototipos: Los prototipos son simulaciones del posible producto, que luego son utilizados por el usuario final, permitiéndonos conseguir una importante retroalimentación en cuanto a si el sistema diseñado con base a los requerimientos recolectados le permite al usuario realizar su trabajo de manera eficiente y efectiva.

Casos de Uso: Los casos de uso son una técnica para especificar el comportamiento de un sistema. Estos permiten describir la posible secuencia de interacciones entre el sistema y uno o más actores, en respuesta a un estímulo inicial proveniente de un actor, es una descripción de un conjunto de escenarios, cada uno de ellos comenzado con un evento inicial desde un actor hacia el sistema. La mayoría de los requerimientos funcionales, sino todos, se pueden expresar con casos de uso.[9]

Luego de realizar una investigación donde se trata de las diferentes técnicas en la ingeniería de requerimientos se pretenderá para la elaboración de este trabajo emplear técnicas como entrevistas y tormenta de ideas para una mejor obtención de los requisitos especificándose en los capítulos posteriores lo tratado en cada de unas de estas técnicas; incluyendo además más adelante la técnica de casos de usos para especificar la interacción entre actores y el sistema.

1.10 Metodologías de desarrollo de software

En un proyecto a la hora de realizar el desarrollo de software se definen metodologías para precisar Quién debe hacer Qué, Cuándo y Cómo debe hacerlo, de ahí la importancia de la mejor selección de la metodología a utilizar. En la actualidad no existe una metodología que sea universal. Las particularidades de cada proyecto (equipo de desarrollo, recursos, etc.) exigen que el proceso sea configurable. Existen hoy en día diferentes metodologías orientadas a objetos las cuales son muy utilizadas en proyectos productivos, entre ellas tenemos: Rational Unified Process (RUP), Extreme Programming (XP) y Microsoft Solution Framework (MSF) entre otras.

1.10.1 Rational Unified Process

El proceso unificado de desarrollo de software fue publicado en el año 1998 como resultado de varios años de experiencia, éste es un proceso de desarrollo de software que unido al Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada en la actualidad para el análisis, implementación y documentación de sistemas orientados a objetos.

La metodología RUP se divide en 4 fases fundamentales en su desarrollo:

- ✓ **Inicio:** El objetivo en esta etapa es determinar la visión del proyecto.
- ✓ **Elaboración:** En esta etapa el objetivo es determinar la arquitectura óptima.
- ✓ **Construcción:** El objetivo es llegar a obtener la capacidad operacional inicial.
- ✓ **Transición:** El objetivo es llegar a obtener el release (versión) del proyecto.

En RUP las actividades son agrupadas en grupos lógicos definiéndose 9 flujos de trabajo principales. Los 6 primeros son conocidos como flujos de ingeniería y los tres últimos como de apoyo. Los Flujos de trabajo son:

Modelamiento del negocio: Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.

Requerimientos: Define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.

Análisis y diseño: Describe cómo el sistema será realizado a partir de la funcionalidad prevista y las restricciones impuestas (requerimientos), por lo que indica con precisión lo que se debe programar.

Implementación: Define cómo se organizan las clases y objetos en componentes, cuáles nodos se utilizarán y la ubicación en ellos de los componentes y la estructura de capas de la aplicación.

Prueba (Testeo): Busca los defectos a lo largo del ciclo de vida.

Instalación: Produce release del producto y realiza actividades (empaque, instalación, asistencia a usuarios, etc.) para entregar el software a los usuarios finales.

Administración del proyecto: Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.

Administración de configuración y cambios: Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización/actualización concurrente de elementos, control de versiones, etc.

Ambiente: Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización.

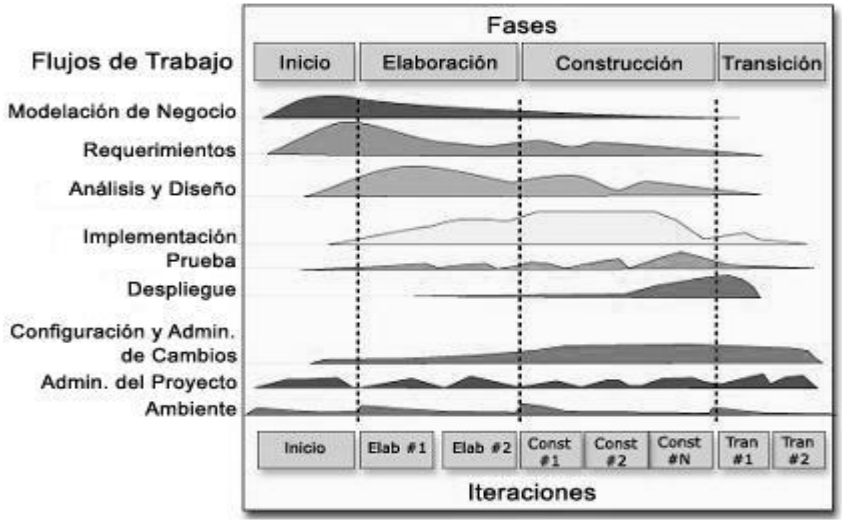


Figura 1: Rational Unified Process (RUP)

Los aspectos definitorios del Proceso Unificado de Rational pueden resumirse en tres características claves:

1. Dirigido por Casos de Uso.
2. Centrado en la arquitectura.
3. Iterativo e incremental.[2]

Esto es lo que hace único al proceso unificado. Para hacer que estas ideas funcionen, se necesita un proceso polifacético, que tenga en cuenta ciclos, fases, flujos de trabajo, gestión de riesgos, control de la calidad, gestión de proyectos y gestión de configuración.

El proceso unificado de desarrollo entre otras facilidades proporciona una visión disciplinada de una guía para ordenar las actividades y responsabilidades dentro de un equipo de desarrollo de software. Otras características que brinda RUP es el dirigir las tareas del desarrollador por separado, y del equipo como un todo. Además la realización de mejores prácticas de software a través de plantillas y herramientas que lo guían en todas las actividades del desarrollo del software. Es un producto que unifica las disciplinas en lo que a desarrollo de software se refiere, incluyendo modelado de negocio, manejo de requerimientos, componentes de desarrollo, ingeniería de datos, manejo y configuración de cambios, y pruebas. Con RUP aseguramos que todos los miembros de nuestro equipo compartan un lenguaje común, proceso y visión de cómo desarrollar software. RUP, también es un proceso configurable. Partiendo de que no todos los proyectos de software utilizan el mismo proceso, se puede decir que el proceso unificado se adecua a todas las organizaciones grandes o pequeñas.

1.10.2 Programación Extrema

A través de los años las metodologías empleadas por los proyectos evolucionan de manera muy rápida, por lo que la metodología de desarrollo de software de cualquier proyecto necesita ser adaptada a sus necesidades y circunstancias específicas y que ninguna metodología se remite solo a reglas a realizar mecánicamente.

Extreme Programming (XP) o programación extrema es una de las metodologías ágiles creada por Kent Beck y la cual está centrada en potenciar las relaciones interpersonales como clave para el éxito en

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

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

El juego de la planificación. Hay una comunicación frecuente el cliente y los programadores. El equipo técnico realiza una estimación del esfuerzo requerido para la implementación de las historias de usuario y los clientes deciden sobre el ámbito y tiempo de las entregas y de cada iteración.

Entregas pequeñas. Producir rápidamente versiones del sistema que sean operativas, aunque no cuenten con toda la funcionalidad del sistema. Esta versión ya constituye un resultado de valor para el negocio. Una entrega no debería tardar más de 3 meses.

Metáfora. El sistema es definido mediante una metáfora o un conjunto de metáforas compartidas por el cliente y el equipo de desarrollo. Una metáfora es una historia compartida que describe cómo debería funcionar el sistema (conjunto de nombres que actúen como vocabulario para hablar sobre el dominio del problema, ayudando a la nomenclatura de clases y métodos del sistema).

Diseño simple. Se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto.

Pruebas. La producción de código está dirigida por las pruebas unitarias. Éstas son establecidas por el cliente antes de escribirse el código y son ejecutadas constantemente ante cada modificación del sistema.

Refactorización (Refactoring). Es una actividad constante de reestructuración del código con el objetivo de remover duplicación de código, mejorar su legibilidad, simplificarlo y hacerlo más flexible para facilitar los posteriores cambios. Se mejora la estructura interna del código sin alterar su comportamiento externo.

Programación en parejas. Toda la producción de código debe realizarse con trabajo en parejas de programadores. Esto conlleva ventajas implícitas (menor tasa de errores, mejor diseño, mayor satisfacción de los programadores.).

Propiedad colectiva del código. Cualquier programador puede cambiar cualquier parte del código en cualquier momento.

Integración continúa. Cada pieza de código es integrada en el sistema una vez que esté lista. Así, el sistema puede llegar a ser integrado y construido varias veces en un mismo día.

40 horas por semana. Se debe trabajar un máximo de 40 horas por semana. No se trabajan horas extras en dos semanas seguidas. Si esto ocurre, probablemente está ocurriendo un problema que debe corregirse. El trabajo extra desmotiva al equipo.

Cliente in-situ. El cliente tiene que estar presente y disponible todo el tiempo para el equipo. Este es uno de los principales factores de éxito del proyecto XP. El cliente conduce constantemente el trabajo hacia lo que aportará mayor valor de negocio y los programadores pueden resolver de manera inmediata cualquier duda asociada. La comunicación oral es más efectiva que la escrita.

Estándares de programación. XP enfatiza que la comunicación de los programadores es a través del código, con lo cual es indispensable que se sigan ciertos estándares de programación para mantener el código legible.

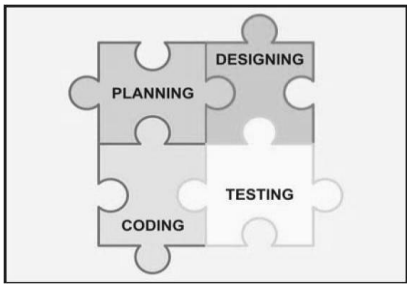


Figura 2 Metodología Programación Extrema

Otras características de esta metodología es que no hay un interés sincero por todas las partes de que los proyectos tengan éxitos, además cuando el equipo de trabajo es muy grande no se recomienda utilizar esta metodología.[10]

1.10.3 Microsoft Solution Framework (MSF)

El Microsoft Solutions Framework es una metodología manejada por escenarios que incorpora las prácticas probadas desarrolladas en Microsoft, además proporciona un sistema de modelos, disciplinas, conceptos, principios, y pautas para dar soluciones a empresas que diseñan y desarrollan de una manera que se asegure de que todos los elementos de un proyecto, tales como personas, procesos, y herramientas, puedan ser manejados con éxito. No es un modelo jerárquico, y es donde los equipos organizados bajo este MSF son pequeños y multidisciplinarios, en los cuales los miembros comparten responsabilidades y balancean las destrezas del equipo para mantenerse enfocados en el proyecto que están desarrollando. MSF no fuerza al desarrollador a utilizar una metodología específica (Cascada, ágil), pero les permite decidir qué método utilizar.[11]

1.10.4 Fundamentación de la selección de la metodología de desarrollo de software a utilizar

Para todo proyecto es fundamental la utilización de una metodología para asegurar la calidad y la organización tanto de su equipo de desarrollo como del producto final. Estas no deben ser pasos a seguir a la hora de emplearlas sino utilizarlas según las características y necesidades que persigue el proyecto de acuerdo a las exigencias del mismo. Se puede ver que no es conveniente la utilización de la metodología XP ya que no permite que la realización de las versiones del producto no excedan a los 3 meses, por lo que no se ajusta a las características del proyecto; donde a raíz de lo anteriormente, es necesario para realizar el sistema de gestión para la calidad en la facultad 8 utilizar una metodología que favorezca a la organización, y en la cual los estudiantes tengan un mayor dominio. En este caso se decidió que la metodología a utilizar sea RUP. La cual constituye una de las más abarcadoras para el análisis, implementación, documentación de sistemas orientados a objetos, entre otras. Además brinda facilidades en cuanto la organización, documentación, y una de sus características es que sirve de guía para ordenar las actividades y responsabilidades dentro del equipo de desarrollo del software y por último

las personas que se encuentran involucradas en la realización del software presentan un alto dominio de la metodología.

1.11 Herramientas CASE

1.11.1 Rational Rose

Rational Rose es una herramienta software para el Modelado Visual mediante UML de sistemas de software. UML permite trabajar en diseños de base de datos, con capacidad de representar la integración de los datos y los requerimientos de aplicación a través de diseños lógicos y físicos. Permite especificar, analizar, diseñar el sistema antes de codificarlo. Además tiene capacidad para la integración con otras herramientas de desarrollo del Rational (Eclipse, Java, .NET, Linux). Rational Rose presenta capacidad de análisis de calidad del código y características de control por separado de componentes que permite una administración más granular y el uso de modelos. Dos características populares de Rational Rose son su capacidad de proporcionar el desarrollo iterativo e ingeniería ida-vuelta. Otra de sus particularidades es, la generación de la documentación automáticamente. También presenta en su estructura carpetas en las cuales te indican que tienes que poner dentro de cada una de ellas. Soporta el sistema operativo Windows 2000, Windows NT, Windows XP.[8]

1.11.2 Visual Paradigm

Visual Paradigm es una herramienta que se utiliza para el modelado UML la cual soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientado a objetos, construcción, pruebas y despliegue. Esta herramienta CASE (Computer Aided Software Engineering o Ingeniería de Software Asistida por Computadora), ofrece un entorno de creación de diagramas para UML; el modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, para la mejora y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML; además su diseño está centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad; uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación; capacidades de ingeniería directa (versión profesional) e inversa; modelo y código que permanece sincronizado en todo el ciclo de desarrollo; disponibilidad de múltiples versiones, para cada

necesidad; disponibilidad de integrarse en los principales IDEs (Integrated Development Environment); disponibilidad en múltiples plataformas.[8]

1.11.3 Selección de la herramienta CASE para el modelado del sistema

A pesar de que Rational Rose tiene características similares al Visual Paradigm se escogió Visual Paradigm para la realización de este sistema, por la flexibilidad que presenta, permitiendo organización al desarrollador. Además la herramienta Visual Paradigm presenta soporte tanto en Windows como en Linux y por las condiciones en que se encuentra nuestro país donde se está emigrando para Software libre esta es una gran ventaja. De esta forma el proyecto de calidad se encuentra en un proceso de cambio de sistema operativo así como de las principales herramientas necesarias para el trabajo del mismo en el cual será conveniente la utilización de la herramienta de modelado Visual Paradigm ya que presenta soporte sobre Linux, y es una característica que lo diferencia del Rational Rose.

1.12 Desarrollo de Aplicaciones Web

1.12.1 Lenguajes del lado del Cliente

HTML

HTML (HyperText Markup Language) es un lenguaje muy sencillo que permite describir hipertexto, es decir, texto presentado de forma estructurada y agradable, con enlaces (hyperlinks) que conducen a otros documentos o fuentes de información relacionadas, y con inserciones multimedia (gráficos, sonido...). Una de las características esenciales de este lenguaje es la universalidad, y significa que prácticamente cualquier ordenador, independientemente del sistema operativo, puede leer o interpretar una página web. Sin embargo, el aspecto de dichas páginas depende del tipo de ordenador, del monitor, la velocidad de la conexión de Internet y, por último del navegador. Aunque no todos los navegadores muestran una misma página web de la misma forma. El lenguaje llamado HTML indica al navegador donde colocar cada texto, cada imagen o cada video y la forma que tendrán estos al ser colocados en la página.[12]

JavaScript

JavaScript es un lenguaje interpretado, es decir, no requiere compilación. El navegador del usuario se encarga de interpretar las sentencias JavaScript contenidas en una página HTML y ejecutarlas adecuadamente. Es soportado por la mayoría de los navegadores como Internet Explorer, Netscape, Opera, Mozilla Firefox, entre otros. Este lenguaje es orientado a eventos ya que cuando un usuario da clic sobre un enlace o mueve el puntero sobre una imagen se produce un evento. Mediante el uso de este lenguaje se pueden desarrollar scripts que ejecuten acciones en respuesta a estos eventos. La característica principal de Javascript, de hecho, es la de ser un lenguaje de scripting, pero, sobre todo, la de ser el lenguaje de scripting por excelencia y, sin lugar a dudas, el más usado.

CSS

Las Hojas de Estilo en Cascada (CSS) es una tecnología que permite controlar la presentación de los documentos en la Web. Usando CSS se pueden especificar muchos atributos de los elementos que conforman una página web, como por ejemplo el color del texto, márgenes, el tipo de letras, tamaño y color, además permite posicionar elementos de forma arbitraria dentro del documento.

El modo de funcionamiento de las CSS consiste en definir, mediante una sintaxis especial, la forma de presentación que le aplicaremos a:

- ✓ Un web entera, de modo que se puede definir la forma de toda la web de una sola vez.
- ✓ Un documento HTML o página, se puede definir la forma, en un pequeño trozo de código en la cabecera, a toda la página.
- ✓ Una porción del documento, aplicando estilos visibles en un fragmento de la página.
- ✓ Una etiqueta en concreto, llegando incluso a poder definir varios estilos diferentes para una sola etiqueta. Esto es muy importante ya que ofrece potencia en nuestra programación.

CSS permite aplicar al documento formato de un modo mucho más exacto como por ejemplo:

- ✓ Definir la distancia entre líneas del documento.
- ✓ Colocar elementos en la página con mayor precisión, y sin lugar a errores.
- ✓ Definir la visibilidad de los elementos, márgenes, subrayados, tachados.

Esta tecnología es bastante nueva, por lo que no todos los navegadores la soportan. En concreto, sólo los navegadores de Netscape versiones de la 4 en adelante y de Microsoft a partir de la versión 3 son capaces de comprender los estilos en sintaxis CSS.[3]

Java

Permite realizar aplicaciones tanto para ejecutar independientemente en un ordenador, exactamente como una aplicación corriente, como para incrustarlo en una página web. En este caso a estos programas se les llama applets. Java es un lenguaje multiplataforma, orientado a objetos y seguro. Eso quiere decir que si hacemos un programa en Java podrá funcionar en cualquier ordenador del mercado. Es una ventaja significativa para los desarrolladores de software, pues antes tenían que hacer un programa para cada sistema operativo, por ejemplo Windows, Linux, Apple, etc. Esto lo consigue porque se ha creado una Máquina de Java para cada sistema que hace de puente entre el sistema operativo y el programa de Java y posibilita que este último se entienda perfectamente. A través de Java no se pueden crear programas que incluyan algún tipo de código maligno. Con Java se puede programar páginas web dinámicas, con accesos a bases de datos, utilizando XML, con cualquier tipo de conexión de red entre cualquier sistema. En general, cualquier aplicación que deseemos hacer con acceso web se puede hacer utilizando Java. Dentro de sus principales ventajas esta simple, seguro, portable, rápido.

1.12.2 Lenguajes del lado del Servidor

PHP

El lenguaje PHP (acrónimo de Hypertext Pre Processor) es creado para desarrollar páginas web dinámicas, puede ser incrustado dentro de código HTML. Entre sus características se puede encontrar:

Gratuito. Al tratarse de software libre, puede descargarse y utilizarse en cualquier aplicación, personal o profesional, de manera completamente libre.

Gran popularidad. A raíz de que existen miles de desarrolladores y programadores que continuamente implementan mejoras en su código, se ha profundizado el amplio conocimiento de este lenguaje siendo así una gran ayuda para todos los que requieran la utilización del PHP, mediante la distribución de la documentación del mismo y debates en foros entre los que utilizan esta herramienta.

Enorme eficiencia. Con escaso mantenimiento puede soportar sin problema millones de visitas diarias.

Sencilla integración con múltiples bases de datos. Esencial para una aplicación Web verdaderamente dinámica, es una correcta integración con base de datos. Aunque MySQL es la base de datos que mejor trabaja con PHP, puede conectarse también a PostgreSQL, Oracle, dbm, filePro, interbasem o cualquier otra base de datos compatible con ODBC (Open Database Connectivity).

Versatilidad. PHP puede usarse con la mayoría de sistemas operativos, ya sea basados en UNIX (Linux, Solares, FreeBSD...), como con Windows, el sistema operativo de Microsoft.

Gran número de funciones predefinidas. A diferencia de otros lenguajes de programación, PHP fue diseñado especialmente para el desarrollo de páginas Web dinámicas. Por ello, está dotado de un gran número de funciones que simplifican enormemente tareas habituales como descargar documentos, enviar correos, trabajar con cookies y sesiones, etc.

A pesar de todas estas características también presenta diferentes ventajas: es libre, abierto; código fuente disponible; diseñado para la web multiplataforma, soporte nativo para prácticamente cualquier Base de Datos; perfecta integración del Apache-PHP-MySQL; bastante sencillo de aprender y utilizar; sintaxis clara y bien definida.

ASP

Las aplicaciones ASP (Active Server Pages) se definen como aquellas que utiliza una empresa para proporcionar servicios de software a otras de forma centralizada y a través de una red.
Presenta muchas ventajas:

- ✓ **Licencias**: el cliente no paga licencias. Simplemente se suscribe a un servicio por el que paga mensualmente.
- ✓ **Piratería**: es mucho más difícil de piratear que un programa clásico. Se puede decir que el 100% de los usuarios del servicio pagarán por él. Esto es una gran ventaja para los desarrolladores frente a las aplicaciones tradicionales.

Además presenta mínimo tiempo de desarrollo y puesta en marcha. (72 horas). Acceso Web a todas las aplicaciones desde cualquier lugar. Actualización permanente de las aplicaciones. Proyectos personalizados de bajo coste. Integración con las aplicaciones del cliente.[13]

JSP

La tecnología Java Server Pages (JSP) provee una manera fácil para crear páginas Web dinámicas y simplificar la tarea de construir aplicaciones Web que trabajen con una amplia variedad de servidores Web, servidores de aplicaciones, navegadores y herramientas de desarrollo.

Las características ofrecidas por JSP como alternativa a la generación de contenido dinámico para la Web se pueden resumir en:

Mejoras en el rendimiento:

- ✓ Utilización de procesos ligeros (hilos Java) para el manejo de las peticiones.
- ✓ Manejo de múltiples peticiones sobre una página.jsp en un instante dado.
- ✓ El contenedor servlet puede ser ejecutado como parte del servidor Web.
- ✓ Facilidad para compartir recursos entre peticiones (hilos con el mismo padre: servlet container).

Soporte de componentes reutilizables:

- ✓ Creación, utilización y modificaciones de JavaBeans del servidor.
- ✓ Los JavaBeans utilizados en páginas.jsp pueden ser utilizados en servlets, applets o aplicaciones Java.

Separación entre código de presentación y código de implementación:

- ✓ Cambios realizados en el código HTML relativos a cómo son mostrados los datos, no interfieren en la lógica de programación y viceversa.

División del trabajo:

- ✓ Los diseñadores de páginas pueden centrarse en el código HTML y los programadores en la lógica del programa.

- ✓ Los desarrollos pueden hacerse independientemente.
- ✓ Las frecuentes modificaciones de una página se realizan más eficientemente.[14]

1.12.3 Gestores de Bases de Datos

MySQL

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

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

Su principal objetivo de diseño fue la velocidad. Se sacrificaron algunas características esenciales en sistemas más "serios" con este fin.

Otra característica importante es que consume muy pocos recursos, tanto de CPU como de memoria.

Ventajas:

- ✓ Mayor rendimiento. Mayor velocidad tanto al conectar con el servidor como al servir selects y demás.
- ✓ Mejores utilidades de administración (backup, recuperación de errores, etc).
- ✓ Aunque se cuelgue, no suele perder información ni corromper los datos.
- ✓ Mejor integración con PHP.
- ✓ No hay límites en el tamaño de los registros.
- ✓ Mejor control de acceso, en el sentido de qué usuarios tienen acceso a qué tablas y con qué permisos.
- ✓ MySQL se comporta mejor que Postgres a la hora de modificar o añadir campos a una tabla "en caliente".

Inconvenientes:

- ✓ No soporta transacciones, "roll-backs" ni subselects.
- ✓ No considera las claves ajenas. Ignora la integridad referencial, dejándola en manos del programador de la aplicación.

PostgreSQL

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

1. Implementación del estándar SQL92/SQL99.
2. Soporta distintos tipos de datos, además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP...), cadenas de bits, etc. También permite la creación de tipos propios.
3. Incorpora una estructura de datos de arreglos.
4. Incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes, etc.
5. Permite la declaración de funciones propias, así como la definición de disparadores.
6. Soporta el uso de índices, reglas y vistas.
7. Incluye herencia entre tablas (aunque no entre objetos, ya que no existen), por lo que a este gestor de bases de datos se le incluye entre los gestores objeto-relacionales.
8. Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.

PostgreSQL intenta ser un sistema de bases de datos de mayor nivel que MySQL, a la altura de Oracle, Sybase o Interbase.

Ventajas:

- ✓ Por su arquitectura de diseño, escala muy bien al aumentar el número de CPUs y la cantidad de RAM.
- ✓ Soporta transacciones y desde la versión 7.0, claves ajenas (con comprobaciones de integridad referencial).
- ✓ Tiene mejor soporte para triggers y procedimientos en el servidor.

- ✓ Soporta un subconjunto de SQL92 mayor que el que soporta MySQL. Además, tiene ciertas características orientadas a objetos.

Inconvenientes:

- ✓ Consume más recursos y carga más el sistema.
- ✓ Límite del tamaño de cada fila de las tablas a 8k (se puede ampliar a 32k recompilando, pero con un coste añadido en el rendimiento).
- ✓ Es de 2 a 3 veces más lenta que MySQL.
- ✓ Menos funciones en PHP.

MySQL vs PostgreSQL ¿cuándo emplear cada una de ellas?

- ✓ MySQL es más rápida que PostgreSQL a la hora de resolver consultas.
- ✓ MySQL tiene mejor documentación y se ha orientado más a facilitarle la vida al desarrollador proporcionando mejores herramientas de administración.
- ✓ PostgreSQL ofrece una garantía de integridad en los datos mucho más fuerte que MySQL. Aunque sea más lenta respondiendo a una única consulta, PostgreSQL presenta una mejor escalabilidad y rendimiento bajo grandes cargas de trabajo.

1.12.4 Fundamentación de los lenguajes y gestor de base de datos a utilizar

Como lenguaje del lado del cliente se ha decidido utilizar HTML por ser un lenguaje sencillo y también CSS para darle la interfaz amigable de la página de acuerdo a lo que requiera su diseño, además javascript para el uso de validaciones. Como lenguaje del lado del servidor PHP versión 5.0, por ser multiplataforma ya que presenta soporte sobre los sistemas operativos de Linux, Windows, entre otras; además los estudiantes que desarrollarán el sistema presentan dominio en el lenguaje. Se escogió como gestor de base de datos PostgreSQL a pesar que presenta diferentes desventajas pero es un gestor que permite su integración con PHP y es conveniente su utilización para el desarrollo del sistema.

1.13 Conclusiones

Luego de haber realizado un estudio de las herramientas, metodologías, lenguajes tanto del lado del cliente como del lado del servidor, gestores de base de datos; además de estudiar las tendencias actuales relacionadas con lo referente a los sistemas de gestión para la calidad se puede elaborar a modo de

resumen todo lo propuesto a utilizar en este capítulo. De esta forma se puede observar que se han retomado los principales conceptos sirviendo de ayuda para la realización de este trabajo. Se llegó a concluir de manera general que la metodología a utilizar será el Proceso Unificado de Desarrollo (RUP). Como herramienta CASE Visual Paradigm, lenguaje de lado de cliente tendremos en conjunto HTML, CSS, JavaScript. Del lenguaje por parte del servidor PHP versión 5.0 por las posibilidades multiplataforma que este brinda, además por su posibilidad de uso gratuito y como gestor de base de datos PostgreSQL.

Capítulo II. Características del Sistema.

2.1 Introducción

En este capítulo serán tratados los principales procesos a automatizar así como el objeto de estudio del negocio de manera general. Se definirán las principales reglas del negocio que se deberán cumplir a la hora de la realización del sistema, además de describir los actores y trabajadores del negocio; y se abordarán sobre las entidades las cuales recogerán información conformándose de esta manera el modelo de objetos del mismo. También se especificarán los requisitos funcionales y no funcionales del sistema los cuales servirán de gran ayuda a los desarrolladores en la confección del mismo ya que a raíz de los requisitos saldrán las principales funcionalidades del sistema así como los actores que intervendrán en el mismo.

2.2 Aplicación de las principales actividades de la Ingeniería de Requisitos

Para facilitar la comprensión del negocio se realizó un estudio de las actividades fundamentales en la ingeniería de requerimientos donde se aplicaron y se obtuvieron resultados satisfactorios, dentro de estas actividades se encuentran:

- ✓ Identificación.
- ✓ Análisis.
- ✓ Especificación.
- ✓ Validación.

A continuación se explica de una manera detallada de cómo se desarrollaron estas actividades ligadas a las técnicas de recopilación de información utilizadas. Primeramente se desarrolló la primera actividad, cuyo objetivo era trabajar con el cliente para poder resolver la problemática, es decir, entender el negocio actual. Algunas de las técnicas empleadas para darle solución a esta actividad fueron las entrevistas a los clientes, y la tormenta de ideas para entender qué era lo que se pretendía en el negocio, y de esta forma ir recogiendo ideas para luego poder realizar un intercambio de estos pensamientos e ir proponiendo

nuevas, luego de conocer los principales procesos de negocio. Se trato de detallar al máximo cada aspecto del negocio para que no existieran interpretaciones erróneas.

Luego de identificado los factores de una primera etapa la cual permitió un gran avance para comprender el negocio se realizó un análisis donde después de haber compartido ideas se plasmaron los requerimientos para que de esta forma el sistema cumpla con sus funciones más importantes, y donde se pudieron realizar encuentros más formales con el cliente definiendo de esta manera los requerimientos funcionales y no funcionales que debiera cumplir el sistema.

Después de esta actividad se comienza la especificación de los requisitos propuestos para el sistema documentándolos en un nivel apropiado de detalle donde se pueden ir especificando los casos de usos más fundamentales a utilizar para la realización del software según lo requerido por el cliente.

Seguidamente, se llevó a cabo la validación de estos requerimientos donde ya estaba bien claro que realmente era lo que necesitaba el cliente, comprobándose de que no se había omitido ninguna funcionalidad requerida por este. De esta forma hay que tener en cuenta siempre lo que realmente se quiere a la hora de realizar alguna entrevista, ya que puede ser que no se especifique bien lo que se desea y entonces acarree muchos problemas un mal entendimiento entre el cliente y los desarrolladores o una mala comprensión del negocio.

2.3 Objeto de Estudio del Negocio

2.3.1 Objeto de automatización

La automatización del tiempo de los recursos se realizará mediante la gestión de la planificación lo cual se puede observar en la planificación de la pruebas donde a la hora de llegada una solicitud se podrá proceder verificando el tipo de prueba, la cantidad de estudiantes que se necesitan y las máquinas requeridas, de esta forma se realiza la gestión de la planificación donde se puedan buscar estudiantes disponibles. Verificando los diferentes riesgos que puedan presentar (Horario, Agenda Personal, etc.) y se envía un documento con la planificación de todos los recursos humanos y tecnológicos disponibles para esta actividad además de los responsables. Luego se envían notificaciones a todos los participantes.

Para la realización de las auditorías se tendrá que insertar los cronogramas de los diferentes proyectos de la facultad para que el planificador, de acuerdo lo que decida el grupo de calidad, informar al proyecto en qué momento se someterá a una auditoría, se consultará la disponibilidad de recursos humanos y tecnológicos verificando los diferentes riesgos que puedan presentar (Horario, Agenda Personal, etc.) y se envía un documento con la planificación de todos los recursos disponibles para esta actividad, responsables, el día de inicio y el de fin. Luego se envían notificaciones a todos los participantes.

Las revisiones se realizarán mediante la inserción de los cronogramas de los diferentes proyectos de la facultad para que el planificador pueda ver qué día será el adecuado para realizar una revisión, este después de verificado pasaría a realizar su propio cronograma, consultando cuáles riesgos se pueden presentar (Horario, Agenda Personal, etc.) y se envía un documento con la planificación de todos los recursos disponibles para esta actividad, así como los responsables, el día de inicio y el de fin. Luego se envían notificaciones a todos los participantes.

Asimismo se realizarán planificaciones del tiempo de máquina en la cual se consultan los diferentes riesgos que puedan ocurrir como: (Horario, Agenda Personal), tratando de que todos los integrantes del proyecto puedan realizar sus labores productivas de acuerdo a la sesión en que se encuentren disponibles.

2.3.2 Información que se maneja

Notificación de Auditoría_ Revisiones.

- ✓ Tipo de Notificación (auditoría o revisiones).
- ✓ Personas a la que se les va a informar sobre auditoría o revisiones.
- ✓ Tiempo de duración aproximado por horas o revisiones.
- ✓ Fecha de inicio de la auditoría o revisiones.
- ✓ Fecha de culminación de la auditoría o revisiones.
- ✓ Nombre del proyecto a auditar o revisar.
- ✓ Polo a auditar o revisar.

Agenda personal.

- ✓ Nombre y Apellidos.
- ✓ Descripción de la afectación.
- ✓ Día de la afectación.
- ✓ Turno en que se va a producir la afectación (Mañana, Tarde y Noche).
- ✓ Número de la máquina.

Distribución del Tiempo _ Máquina.

- ✓ Número de la Máquina.
- ✓ Nombre de Estudiante.
- ✓ Turno (Mañana, Tarde y Noche).
- ✓ Días de la Semana (lunes, martes, miércoles, jueves, viernes, sábado).

Cronogramas.

- ✓ Nombre del proyecto.
- ✓ Inicio de Prueba.
- ✓ Fin de Prueba.
- ✓ Número de Organización.

Solicitud de petición de Recursos.

- ✓ Cantidad Recursos humanos.
- ✓ Cantidad Recursos tecnológicos.
- ✓ Nombre del proyecto.
- ✓ Dirección de la aplicación.

Respuesta de petición de recursos.

- ✓ Cantidad Recursos humanos.
- ✓ Cantidad Recursos tecnológicos.
- ✓ Nombre del proyecto.
- ✓ Recursos Humanos Disponibles (RHD).

- ✓ Nombre y Apellidos de RHD.
- ✓ Recursos Tecnológicos Disponibles (RTD).
- ✓ Cantidad de máquinas de RTD.
- ✓ Ubicación de los RTD.

Información de Proyectos.

- ✓ Nombre y Apellidos del asesor de calidad
- ✓ Usuario.
- ✓ Teléfono.
- ✓ Número de Organización.
- ✓ Nombre del proyecto.
- ✓ Polo en que se encuentra el proyecto.

Plantilla del Proyecto.

- ✓ Nombre y Apellidos.
- ✓ Grupo.
- ✓ Apartamento.
- ✓ Teléfono.
- ✓ Usuario.
- ✓ Año.

Cronogramas de Proyectos.

- ✓ Nombre del proyecto
- ✓ Número de organización.

2.3.3 Propuesta de sistema

La herramienta que se propone desarrollar es un sistema para la gestión de la calidad de la facultad 8 la cual automatizará los procesos que se desarrollan actualmente en el grupo de calidad, este sistema se encuentra distribuido por cuatro módulos fundamentales:

Módulo de Planificación: Se procederá a la automatización de los recursos para la gestión de la planificación de las actividades del proyecto como son las pruebas, auditorías, revisiones y capacitación; además de la planificación del proyecto de manera general.

Módulo de Pruebas: Se realizará para permitir que los procesos de prueba que se desarrollan en el grupo de calidad presenten una organización en la realización de las pruebas de documentación y aplicación, dando posibilidad de la generación de reportes e informes una vez realizadas estas pruebas, tanto en la documentación como las pruebas exploratorias a las aplicaciones.

Módulo de Auditorías y Revisiones: Se requiere automatizar una serie de actividades que son necesaria en el proceso de auditoría o revisiones en donde se podrá guardar información obtenida de los proyectos así como las técnicas de recopilación de información.

Módulo de Capacitación: Pretende generar una serie de reportes que hasta el momento se realizan de manera manual, trayendo consigo gasto de tiempo, empleo de mano de obra innecesariamente, etc. Permitirá de manera óptima el desarrollo de los cursos del perfil de calidad.

2.4 Modelo del Negocio

2.4.1 Descripción del Negocio

El modelo de negocio que a continuación se describe va a responder a un conjunto de actividades que se realizan en los procesos de gestión de la calidad. De manera inicial el asesor de calidad y el jefe de prueba reciben una solicitud de prueba de los especialistas de calidad realizando la planificación del tiempo de las actividades de las pruebas y de los recursos necesarios para las mismas. En el proceso de auditorías y revisiones, se realizan internamente a los proyectos de la facultad partiendo de los cronogramas generales que posee cada proyecto permitiendo observar en qué momento será más recomendado realizar una auditoría o una revisión, para ir verificando que estos vayan en el camino correcto en la confección del producto software. También se realizan para mantener una organización dentro del grupo de calidad una planificación exhaustiva del tiempo de máquina, donde los integrantes del proyecto se guiaran para efectuar su trabajo según las sesiones que le corresponda. Esta planificación se efectuará teniendo en cuenta los posibles inconvenientes que puedan ocurrir, informándolos al que realiza

la planificación, donde actualmente se procede a la hora de que existe alguna dificultad al correo electrónico, permitiéndole a los integrantes exponer las posibles dificultades que surjan y en la cual se vea afectada el tiempo de máquina.

2.4.2 Reglas del Negocio

- ✓ No realizar la planificación en la sesión en la cual ese día se encuentre afectado por educación física (EF).
- ✓ Planificar de manera alterna la distribución del tiempo de máquina.
- ✓ Los estudiantes de 3er año deben tener 3 sesiones de trabajo es decir 12 horas/semana.
- ✓ Los estudiantes de 4to año deben tener 4 sesiones de trabajo es decir 16 horas/semana.
- ✓ Los estudiantes de 5to año deben tener 5 sesiones de trabajo es decir 20 horas/semana.
- ✓ Todos los que presenten alguna afectación tienen que registrarlo en su agenda personal en una fecha límite la cual será 2 días con antelación a la afectación.
- ✓ Tienen que estar ubicados en el sistema los cronogramas de los proyectos para realizar la planificación de las actividades de auditorías, revisiones y pruebas a los mismos.
- ✓ Tener actualizado el horario docente.
- ✓ No involucrar en ninguna otra actividad a los integrantes que estén realizando auditorías o revisiones.

2.4.3 Actores y trabajadores del Negocio

Tabla 1: Actores del Negocio

Actor del Negocio	Justificación
Planificador de Actividades	Persona que intervienen el proceso del negocio siendo este o el asesor de calidad o los jefes de equipos de trabajo donde a la hora de recibir una solicitud se encargan de que se realice la planificación de acuerdo con lo solicitado para la organización de su equipo de trabajo.

Tabla 2 Trabajadores del Negocio

Trabajadores del Negocio	Justificación
Planificador	Persona que interviene en el negocio a la hora de generar los procesos correspondientes en cada una de las actividades que se desarrollan en este.

2.5 Casos de Usos del Negocio

2.5.1 Casos de usos

Listado de casos de uso del negocio a tratar:

1. Planificar Tiempo _ Máquina.
2. Planificar Capacitación.
3. Planificar actividades del grupo de calidad.

2.5.2 Entidades del Negocio

Solicitud de petición de Recursos, Solicitud Respuesta de Recursos, Cronogramas, Horario, Notificación de Auditoría_ Revisiones, Distribución del Tiempo _ Máquina, Agenda Personal, Información de Proyectos, Cronogramas de Proyectos.

2.5.3 Diagrama de casos de usos del Negocio

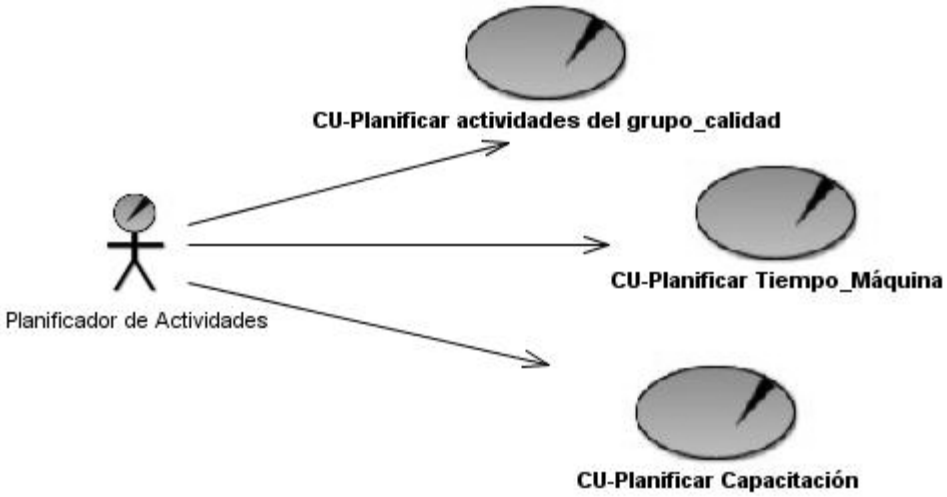


Figura 3 Diagrama de Casos de Uso del Negocio

El diagrama del modelo de objetos es la relación del conjunto básico de las entidades del negocio con los trabajadores del negocio.

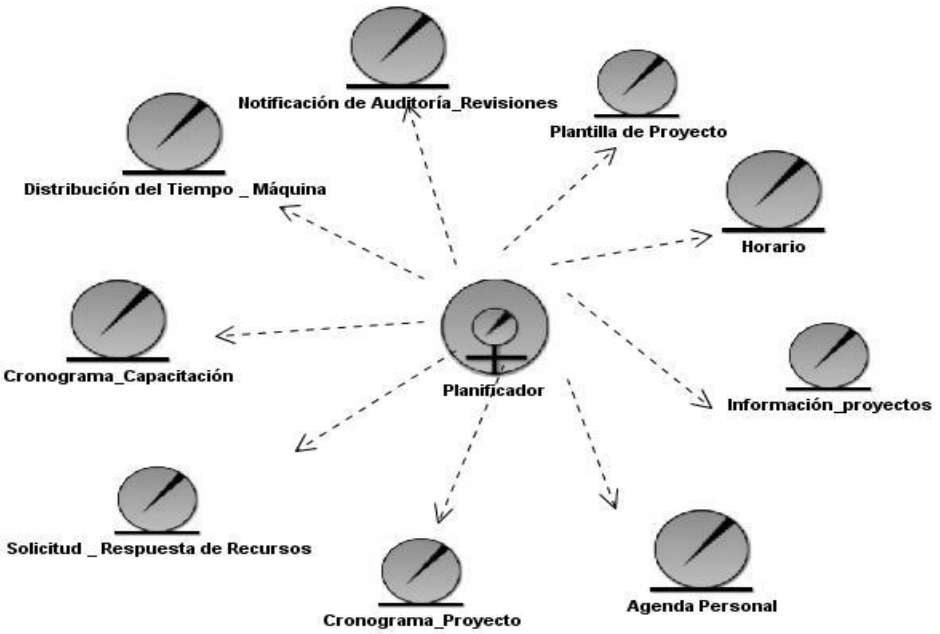


Figura 4 Diagrama de clases del Modelo de Objetos

2.6 Descripciones de casos de usos del Negocio

2.6.1 Caso de Uso Planificar Distribución _Tiempo_ Máquina.

Tabla 3 Descripción textual Caso de Uso Planificar Distribución _Tiempo_ Máquina

Caso de Uso del Negocio	
Actores	Planificador de Actividades.
Resumen	El caso de uso inicia cuando el asesor de calidad decide realizar la planificación de tiempo de máquina para la organización de los recursos humanos y tecnológicos.
Casos de Uso asociados	-

Acción del actor	**Respuesta del proceso de negocio**
1. El asesor de calidad decide que hay que realizar la planificación de tiempo de máquina.	
	2. El planificador reúne todos los riesgos que existan así como el horario, correos de estudiantes que se encuentren afectados etc. 3. El planificador realiza la planificación y la envía al asesor.
4. El asesor de calidad revisa la planificación para saber si está correcta. 5. En caso contrario lo vuelve a enviar con las indicaciones previas.	
	6. El planificador realiza los cambios correspondientes. 7. El planificador envía la planificación a los integrantes del proyecto.

Otras secciones	-

Mejoras propuestas	Se automatizará el proceso de planificación del tiempo de máquina donde se logrará una planificación más rápida y efectiva teniendo en cuanta todos los riesgos que puedan ocurrir en la aplicación sin necesidad de buscarlos en otro lugar.

2.6.2 Caso de Uso Planificar Capacitación.

Tabla 4 Descripción textual Planificar Capacitación

Caso de Uso del Negocio		
Actores	Planificador de Actividades.	
Resumen	El caso de uso inicia cuando el asesor de calidad o los jefes de equipo deciden que es necesario realizar una capacitación a los integrantes de proyecto y para ello se realiza una planificación en coordinación de en qué momento será el indicado para proceder a la misma.	
Casos de Uso asociados	-	
Acción del actor		**Respuesta del proceso de negocio**
1. El asesor de calidad o los jefes de equipos deciden sobre qué tema realizarán la capacitación.		
		2. El planificador informa en qué momento se puede realizar la planificación de la capacitación teniendo en cuenta los posibles riesgos como el horario etc.
3. El asesor de calidad y los jefes de equipos buscan a la persona encargada de realizar la capacitación que se encuentre disponible en ese horario indicado por la planificadora.		

	4. El planificador se encarga de verificar quiénes no han pasado la capacitación. 5. El planificador procede a realizar la planificación y enviarlo a los integrantes y al profesor encargado.
Otras secciones	-
Mejoras propuestas	Se automatizará el proceso de planificación de la capacitación donde se logrará una planificación más rápida y efectiva.

2.6.3 Caso de Uso Planificar actividades del grupo de calidad.

Tabla 5 Descripción textual Caso de Uso Planificar actividades del grupo de calidad

Caso de Uso del Negocio	
Actores	Planificador de Actividades.
Resumen	El caso de uso inicia cuando el Planificador de actividades recibe una solicitud de prueba, auditoría o revisiones de los especialistas de calidad o por decisión del grupo de realizar algunas de estas actividades a los diferentes proyectos de la facultad donde estos proceden a realizar la planificación de los recursos necesarios para las mismas.
Casos de Uso asociados	-
Acción del actor	**Respuesta del proceso de negocio**
1. El Planificador de actividades recibe solicitudes de prueba, auditoría o revisiones donde especifican todos los datos asociados con lo que se requiere en alguna de estas actividades etc.	
	2. El Planificador revisa de qué tipo de actividad

	es la solicitud. 3. Revisa las posibles afectaciones que puedan ocurrir. 4. Verifica el horario para ver quiénes son los afectados. 5. Realiza el cronograma de las actividades. 6. Envía el Cronograma de pruebas al Planificador de actividades.
7. El Planificador de actividades verifica que se encuentre correcto el cronograma según el horario. 8. Envía la planificación que se realizó al especialista de calidad para que exista información con lo que realmente se cuenta.	
	9. Los especialistas de calidad reciben dicha planificación.
Otras secciones	-
Mejoras propuestas	Se automatizará el proceso de planificación de las actividades de pruebas, auditorías o revisiones donde se logrará una planificación más rápida y efectiva teniendo en cuenta los riesgos que puedan ocurrir.

Para ver los diagramas de actividades referente a los procesos de negocio consultar Anexo 1.

2.7 Especificación de Requisitos

2.7.1 Requisitos Funcionales

RF1 Gestionar datos Integrantes del Proyecto calidad.

1.1 Insertar datos Integrantes.

1.2 Eliminar datos Integrantes.

1.3 Modificar datos Integrantes.

1.4 Mostrar datos integrantes.

1.4.1 Exportar a pdf o Excel la plantilla del proyecto.

RF2 Consultar Integrantes del Proyecto calidad.

2.1 Exportar a pdf o Excel la plantilla del proyecto.

RF3 Adicionar Cronogramas Proyectos.

RF4 Consultar Cronogramas Proyectos.

RF5 Gestionar Distribución Tiempo Máquina.

5.1 Asignar datos Tiempo Máquina.

5.2 Modificar datos Tiempo Máquina.

5.3 Mostrar datos Tiempo Máquina.

RF6 Consultar Distribución Tiempo Máquina.

6.1 Buscar datos Tiempo Máquina.

6.2 Mostrar datos Tiempo Máquina.

6.2.1 Exportar a pdf datos Tiempo Máquina.

RF7 Adicionar datos Agenda Personal.

RF8 Consultar datos Agenda Personal.

8.1 Mostrar datos Agenda Personal.

8.2 Eliminar Agenda Personal.

8.3 Actualizar datos Agenda personal.

RF9 Consultar Cronogramas (Auditorías, Revisiones, Pruebas).

9.1 Buscar cronogramas.

9.2 Mostrar cronogramas.

9.2.1 Exportar a pdf cronograma.

RF10 Gestionar Petición Recursos.

10.1 Insertar Recursos.

10.2 Modificar Recursos.

10.3 Eliminar Recursos.

10.4 Enviar Recursos.

RF11 Consultar respuesta de Petición de recursos.

11.1 Insertar respuesta.

11.2 Mostrar datos de la respuesta de petición.

11.2.1 Exportar a pdf respuestas de Petición Recursos.

RF12 Gestionar Información Proyectos.

12.1 Insertar Información Proyecto.

12.2 Modificar Información Proyecto.

12.3 Eliminar Información Proyecto.

12.4 Mostrar Información Proyecto.

RF13 Consultar Información Proyectos.

13.1 Mostrar datos Proyectos.

13.1.1 Exportar a pdf Información de Proyectos.

RF14 Gestionar Cronogramas (Auditorías, Revisiones, Pruebas).

14.1 Insertar datos al cronograma.

14.2 Modificar datos al cronograma.

14.3 Mostrar datos al cronograma.

2.7.2 Requisitos no funcionales

Apariencia o interfaz externa.

El sistema debe poseer una interfaz web amigable y sencilla, fácil para la interacción del usuario. Con colores en su interfaz que distingan a la facultad, sin saturación de colores ni imágenes.

Usabilidad.

El sistema podrá ser usado por todas aquellas personas de los proyectos de la facultad, integrantes del grupo de calidad y calidad UCI ya que estos tienen que interactuar con él.

Rendimiento.

Operaciones normales en dos hilos.
Respuestas rápidas en segundos al usuario.
Debe estar disponible las 24 horas del día.

Soporte.

La aplicación tendrá un período de tiempo en prueba para ver su funcionamiento y que cumpla con lo requerido dándole mantenimiento para luego brindar servicio de instalación. Además de la documentación necesaria para el uso o desarrollo de las funcionalidades de la herramienta.

Portabilidad.

El sistema debe ser Multiplataforma. Haciendo énfasis en Linux.

Seguridad.

Se permitirá la autenticación la cual será una contraseña de acceso para los usuarios autorizados. Además existirán autorizaciones según las funciones de trabajo.

Políticos-culturales.

La herramienta sólo podrá ser utilizada dentro de la UCI.

Las funcionalidades del sistema no deben contener palabras en otros idiomas.
Se deben poner las palabras lo más acorde a su entendimiento.

Legales.

Es un deber proteger la información por parte de las personas que tienen derecho de administración u otros que pongan en peligro la integridad y seguridad del sistema.

Confiabilidad.

Está prohibida la diseminación pública de la información por cualquier usuario a cualquier nivel.

Interfaz.

Las interfaces deben ser lo más entendible posible.
Todas las interfaces tienen que tener una forma de cerrarse o retornar al paso anterior.
Ninguna interfaz puede ir cargada de muchos colores evitando la desconcentración del usuario.

Software.

Tener como sistema Operativo Windows 98 o superior, Linux o Unix, en sus versiones de S.O servidores para el servidor del Web Services. Se necesita tener instalado el servidor de aplicaciones Web Apache en el servidor del Web Services. Se necesita tener instalado PostgreSQL en el servidor de Base de Datos.

Requerimientos de Hardware.

Se requiere de un servidor de 512 MB de RAM como espacio mínimo y 50 MB de espacio libre en disco duro.
Las computadoras para el proceso de pruebas deben estar conectadas a una red y tener al menos 128 MB de RAM.

Ayuda y documentación en línea.

La aplicación tiene que llevar un manual de ayuda.

2.8 Definición de los Casos de Usos del sistema

2.8.1 Definición de los Actores

Tabla 6 Actores del sistema

Actores del sistema.	**Justificación**.
Planificador	Encargado de interactuar con el sistema para poder realizar la gestión de los recursos para la planificación de las actividades de auditorías, revisiones, pruebas, tiempo de máquina y capacitación que se realizan en el grupo de calidad.
Usuarios internos	Son los integrantes del proyecto que necesitan interactuar con el sistema para obtener y dar información necesaria para el planificador.
Usuarios externos	Son los encargados de interactuar con el sistema para poder facilitar el trabajo al planificador, es decir, son los representantes de los proyectos que necesitan interactuar con el sistema.
Interesados	Son todos aquellos que interactúan con el sistema para que se puedan realizar determinadas acciones que son importantes para la planificación de las actividades, como el asesor de calidad, jefes de equipos, planificador, usuarios externos y usuarios internos.

2.8.2 Listado de casos de usos.

Tabla 7 Listado de Casos de usos del sistema

Casos de Usos de Sistema	Identificador	Arquitectónicamente Significativos.
Gestionar Integrantes del Proyecto.	CU-1	Significativo
Consultar Integrantes del proyecto.	CU-2	-

Adicionar Cronogramas Proyectos.	CU-3	Significativo
Consultar Cronogramas Proyectos.	CU-4	Significativo
Gestionar Distribución Tiempo Máquina.	CU-5	Significativo
Consultar Distribución Tiempo Máquina.	CU-6	-
Adicionar datos Agenda Personal.	CU-7	-
Consultar Agenda Personal.	CU-8	Significativo
Consultar Cronogramas (Auditorías, Revisiones, Pruebas).	CU-9	-
Gestionar Petición Recursos.	CU-10	Significativo
Consultar respuesta de Petición de Recursos.	CU-11	-
Gestionar Información Proyectos.	CU-12	Significativo
Consultar Información Proyectos.	CU-13	-
Gestionar Cronogramas (Auditorías, Revisiones, Pruebas).	CU-14	Significativo

2.8.3 Diagrama de Paquetes

Para la modelación de los casos de uso del sistema se decidió conformar cuatro paquetes por el criterio de empaquetamiento: Los Casos de Usos requeridos para dar soporte a un determinado actor del sistema.

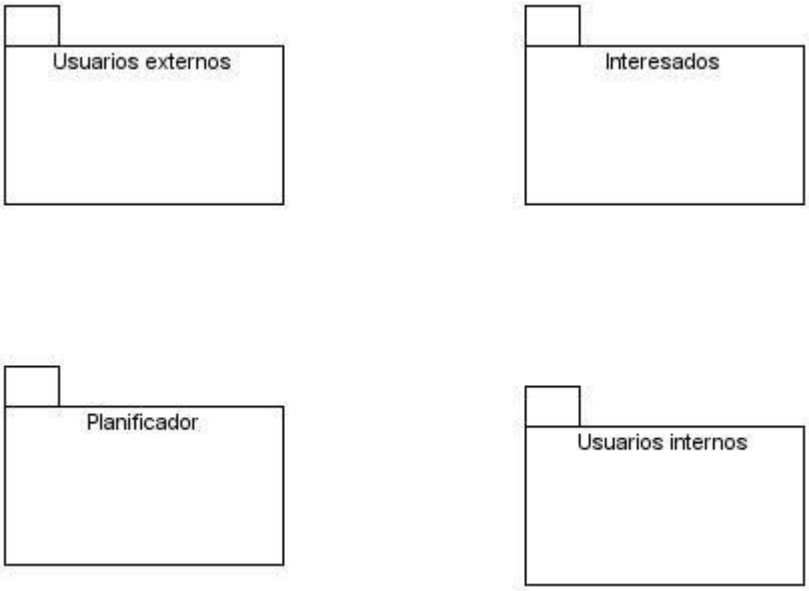


Figura 5 Diagrama de Casos de Usos por paquetes de actores

2.8.3.1 Paquete Planificador.

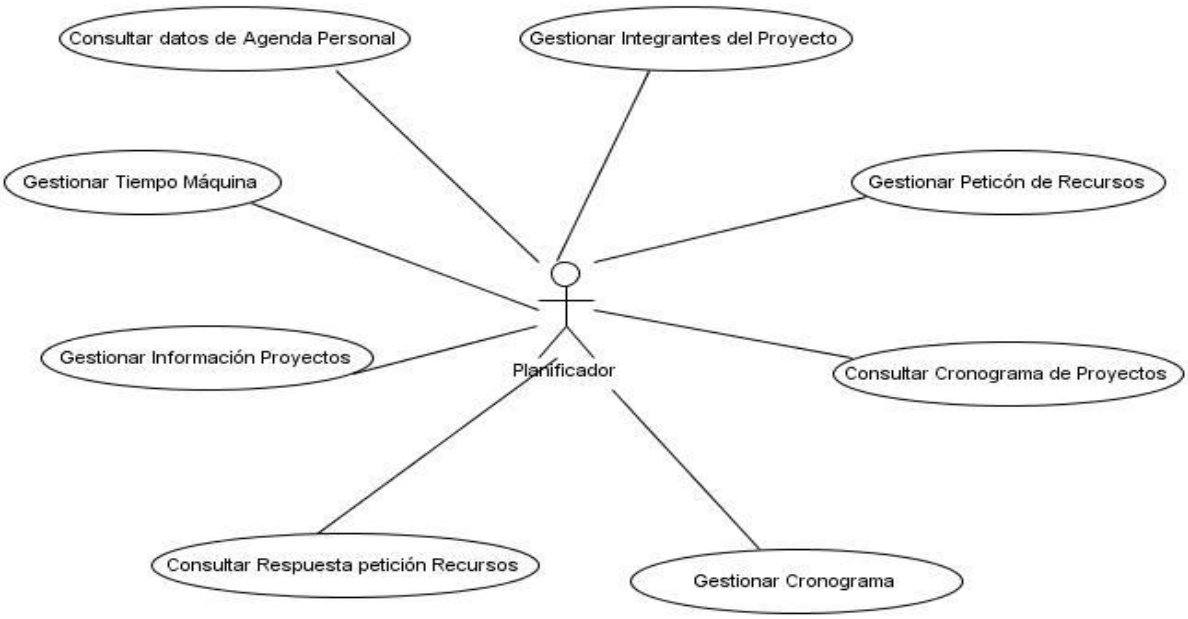


Figura 6 Diagrama de caso de uso del sistema paquete Planificador

2.8.3.2 Paquete Interesados.

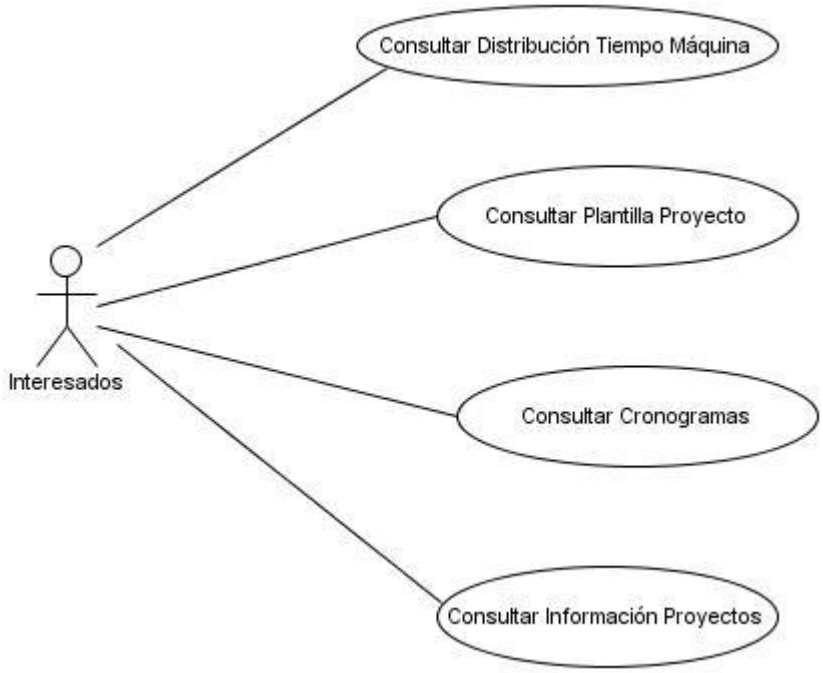


Figura 7 Diagrama de caso de uso del sistema paquete Interesados

2.8.3.3 Paquete Usuarios externos.

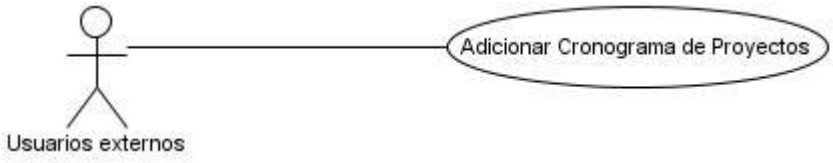


Figura 8 Diagrama de caso de uso del sistema Usuarios externos

2.8.3.3 Paquete Usuarios internos.

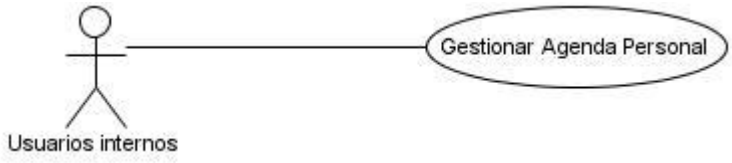


Figura 9 Diagrama de caso de uso del sistema Usuarios Internos

2.9 Casos de uso expandidos

2.9.1 Caso de Uso Gestionar Distribución Tiempo Máquina.

Tabla 8 Descripción textual Caso de Uso Gestionar Distribución Tiempo Máquina

Caso de Uso	Gestionar Distribución Tiempo Máquina
Actores:	Planificador
Resumen:	El caso de uso se inicia cuando el Planificador decide elaborar la distribución del tiempo de máquina del proyecto donde puede realizar diferentes acciones como asignar, mostrar, modificar, buscar. En caso de que seleccione la opción de asignar, el sistema da la posibilidad de realizar una asignación. Si el actor elige la opción de mostrar el sistema muestra el tiempo de máquina. Si el actor elige la opción de modificar el sistema permite la modificación y una vez realizados los cambios, guarda las modificaciones. El sistema permite ver una vista previa, con posibilidad de exportar a pdf para luego poder enviar el documento, terminando así el caso de uso.
Precondiciones:	Debe haberse generado el escritorio de trabajo del usuario autenticado. Según la acción de insertar, mostrar, modificar o buscar a realizar debe estar

<table>
<tr><td></td><td colspan="2">seleccionada la acción que se desee realizar.</td></tr>
<tr><td>Referencias</td><td colspan="2">RF 5</td></tr>
<tr><td>Prioridad</td><td colspan="2">Alta</td></tr>
<tr><td colspan="3">Flujo Normal de Eventos</td></tr>
<tr><td colspan="2">Acción del actor</td><td>Respuesta del Sistema</td></tr>
<tr><td colspan="2">1. El caso de uso se inicia cuando el actor selecciona la opción de realizar una acción sobre la distribución del tiempo de máquina.</td><td></td></tr>
<tr><td colspan="2"></td><td>2. Este brinda la posibilidad de realizar las acciones:
• Asignar tiempo de máquina.
• Modificar datos. Ver sección 1 “Modificar datos de la distribución del tiempo de máquina”.
• Mostrar datos. Ver sección 2 “Mostrar datos de la distribución del tiempo de máquina”.</td></tr>
<tr><td colspan="2">3. Selecciona la opción de asignar tiempo de máquina.</td><td></td></tr>
<tr><td colspan="2"></td><td>4. Permite realizar la asignación a través de los datos siguientes:
• Número de la Máquina.
• Nombre de Estudiante.
• Turno.
• Días de la Semana.
• Semanas.
Y permite:
• Guardar datos.
• Cancelar datos.</td></tr>
</table>

5. Selecciona la opción de guardar datos.	
	6. Muestra un mensaje de información "Se ha realizado la asignación correctamente". 7. El caso de uso termina.
Prototipo de Interfaz	
Flujo Alterno 5.a "El actor decide cancelar la operación"	
Acción del Actor	**Respuesta del Sistema**
	5.a.1 Elimina los datos creados. 5.a.2 Muestra un mensaje de información "Se ha cancelado la operación". 5.a.3 Regresa a la vista anterior. 5.a.4 El caso de uso termina.
Sección 1 "Modificar datos de la distribución del tiempo de máquina".	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de modificar los datos de la distribución del tiempo de máquina.	
	2. Muestra los datos y le permite realizar modificaciones a los siguientes datos: • Turno. • Días de la semana. • Semana. Y permite: • Guardar los cambios realizados • Cancelar la operación.
3. Modifica los datos que considere y selecciona opción de guardar.	

	4. Muestra un mensaje de información “Se han modificado los datos correctamente.” 5. Muestra los datos modificados. 6. El caso de uso termina.
Prototipo de Interfaz	
Flujo Alterno 3.a “El actor decide cancelar la operación”	
Acción del Actor	**Respuesta del Sistema**
	3.a.1 Elimina los datos creados. 3.a.2 Muestra un mensaje de información “Se ha cancelado la operación”. 3.a.3 Regresa a la vista anterior. 3.a.4 El caso de uso termina.
Sección 2 “Mostrar datos de la distribución del tiempo de máquina”.	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de mostrar todos los datos de la distribución del tiempo de máquina.	
	2. Muestra los datos. Y permite: • Exportar a PDF. • Salir de la vista actual.
3. Selecciona la opción de salir de la vista actual.	
	4. Muestra la vista anterior. 5. El caso de uso termina.
Prototipo de Interfaz	
Flujo Alterno 3.a “El actor decide Imprimir o exportar a pdf”	
Acción del Actor	**Respuesta del Sistema**
	3.a.1 Realiza la operación de exportar a pdf.

	3.a.2 Accede a al opción de imprimir. 3.a.3 Muestra un mensaje “Se encuentra realizando la operación ” 3.a.4 El caso de uso termina.
Poscondiciones	-

2.9.2 Caso de Uso Gestionar Información Proyectos.

Tabla 9 Descripción textual Caso de Uso Gestionar Información Proyectos

Caso de Uso	Gestionar Información Proyectos.
Actores:	Planificador
Resumen:	El caso de uso se inicia cuando el Planificador decide elaborar una plantilla donde esté la información de los proyectos que se necesite, en ella se podrá realizar diferentes acciones como insertar, mostrar, modificar, eliminar. En caso de que seleccione la opción de insertar, el sistema brinda la posibilidad de incluir los datos que se necesiten. Si el actor elige la opción de mostrar el sistema muestra los datos. Si el actor elige la opción de modificar el sistema va a permitir realizar la modificación y una vez realizados los cambios, guardará las modificaciones. El sistema permite ver una vista previa, con posibilidad de exportar a pdf para luego poder enviar el documento, terminando así el caso de uso.
Precondiciones:	Debe haberse generado el escritorio de trabajo del usuario autenticado. Según la acción de insertar, ver, modificar, eliminar o buscar a realizar debe estar seleccionada la acción a realizar.
Referencias	RF 14
Prioridad	Alta
Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. El caso de uso se inicia cuando el actor selecciona la opción de realizar una	

acción sobre la Información de los proyectos.	
	2. Este brinda la posibilidad de realizar las acciones: • Insertar información de los proyectos. • Modificar datos. Ver sección 1 “Modificar datos de la información de los proyectos”. • Mostrar datos. Ver sección 2 “Mostrar datos de la información de los proyectos”. • Eliminar datos. Ver sección 3 “Eliminar datos de la información de los proyectos”.
3. Selecciona la opción de Insertar datos de la información de los proyectos.	
	4. Permite introducir los siguientes datos: • Nombre y Apellidos del líder de calidad. • Usuario. • Teléfono. • Nombre del proyecto. • Polo. Y permite: • Guardar los datos. • Cancelar la operación.
5. Selecciona la opción de guardar datos.	
	6. Muestra un mensaje de información “Se han guardado los datos correctamente”. 7. El caso de uso termina.
Prototipo de Interfaz	
Flujo Alterno 5.a “El actor decide cancelar la operación”	

Acción del Actor	**Respuesta del Sistema**
	5.a.1 Elimina los datos creados. 5.a.2 Muestra un mensaje de información "Se ha cancelado la operación". 5.a.3 Regresa a la vista anterior. 5.a.4 El caso de uso termina.
Sección 1 "Modificar datos de la distribución del tiempo de máquina".	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de modificar los datos de la información de los proyectos.	
	2. Muestra los datos y le permite realizar modificaciones. Y permite: • Guardar los cambios realizados. • Cancelar la operación.
3. Modifica los datos que considere y selecciona opción de guardar.	
	4. Muestra un mensaje de información "Se han modificado los datos correctamente." 5. Muestra los datos modificados. 6. El caso de uso termina.
Prototipo de Interfaz	
Flujo Alterno 3.a "El actor decide cancelar la operación"	
Acción del Actor	**Respuesta del Sistema**
	3.a.1 Elimina los datos creados. 3.a.2 Muestra un mensaje de información "Se ha cancelado la operación". 3.a.3 Regresa a la vista anterior. 3.a.4 El caso de uso termina.

Sección 2 "Mostrar datos de la información de los proyectos".	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de mostrar todos los datos de la información de los proyectos.	
	2. Muestra los datos. Y permite: • Exportarlos a PDF. • Salir de la vista actual.
3. Selecciona la opción de salir de la vista actual.	
	4. Muestra la vista anterior. 5. El caso de uso termina.
Prototipo de Interfaz	
Flujo Alterno 3.a "El actor decide Imprimir o exportar a pdf"	
Acción del Actor	**Respuesta del Sistema**
	3.a.1 Realiza la operación de exportar a pdf. 3.a.2 Accede a al opción de imprimir. 3.a.3 Muestra un mensaje "Se encuentra realizando la operación " 3.a.4 El caso de uso termina.
Sección 3 "Eliminar datos de la información de los proyectos".	
Acción del Actor	**Respuesta del Sistema**
1. El actor decide eliminar algún dato de la información de proyectos y selecciona la opción.	
	2. Elimina los datos seleccionados. 3. Muestra un mensaje "Se han eliminado los datos

	correctamente". 4. Regresa a la vista anterior. 5. Termina el caso de uso.
Poscondiciones	-

Para ver las descripciones textuales de los casos de usos arquitectónicamente significativos consultar Anexo 2.

2.10 Conclusiones

En este capítulo se observaron las principales características del negocio así como del sistema, donde se desarrolló un estudio de los principales procesos del negocio permitiendo seleccionar los actores y trabajadores, además de ver las entidades que interactuaban en el mismo para una mejor comprensión. Se identificaron los requisitos funcionales y no funcionales dando paso a un entendimiento eficiente para el desarrollo del sistema. A través de los requisitos se realizó la selección de los casos de usos que darían paso a las funcionalidades al sistema con la descripción textual de cada uno de ellos el cual permitirá la comprensión y la solución más exacta a la hora de desarrollar dicho sistema.

Capítulo III. Análisis y Diseño.

3.1 Introducción

A raíz del capítulo anterior según la descripción y definición de las diferentes funcionalidades que deberá seguir el desarrollador a la hora de la implementación del sistema, se definirá de manera más formal el desarrollo del sistema a implementar. Este capítulo tiene como objetivo primordial realizar la concepción general del análisis y diseño del sistema propuesto, donde se presentan los diagramas de clases del análisis, diagramas de clases de estereotipos Web así como sus descripciones textuales, además se mostrará el diagrama de modelo de datos y el diagrama de despliegue para una mejor comprensión del sistema.

3.2 Análisis

En el análisis se refinan los requisitos que se describieron en la captura de los mismos refinándolos y estructurándolos, además proporciona una estructura centrada en el mantenimiento en aspectos tales como la flexibilidad ante los cambios y la reutilización; por lo que esto no se ve solamente para la parte en la ingeniería de requerimientos sino también en el diseño y la implementación.

Propósito del análisis:

El propósito del análisis no es más que refinar los requisitos seleccionados en la captura de los mismos pero en este flujo de trabajo tratarlos con mayor profundidad utilizando el lenguaje del desarrollador para describir los resultados. Para de esta forma razonar más sobre los aspectos internos del sistema.

3.2.1 Modelo de clases de análisis

Presenta una jerarquía de paquetes proporcionando la vista interna al sistema el cual se encuentra estructurado por clases. Ofrece una especificación de los requisitos más precisa, y se involucra utilizando el lenguaje de los desarrolladores. Define realizaciones de casos de usos donde cada una de ellas representa el análisis de un caso de uso del modelo de casos de usos. Además permite facilitar la comprensión, preparación, su modificación y en general su mantenimiento; además se puede considerar como la primera aproximación al modelo de diseño.

3.2.2 Diagrama de clases del análisis

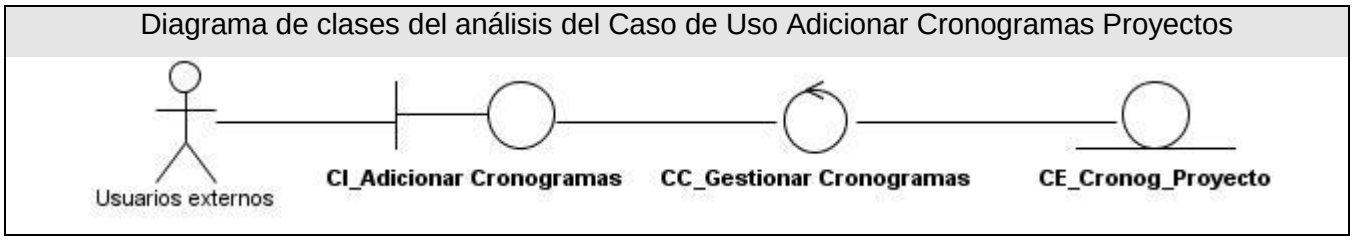


Figura 10 Diagrama de clases del análisis del Caso de Uso Adicionar Cronogramas Proyectos

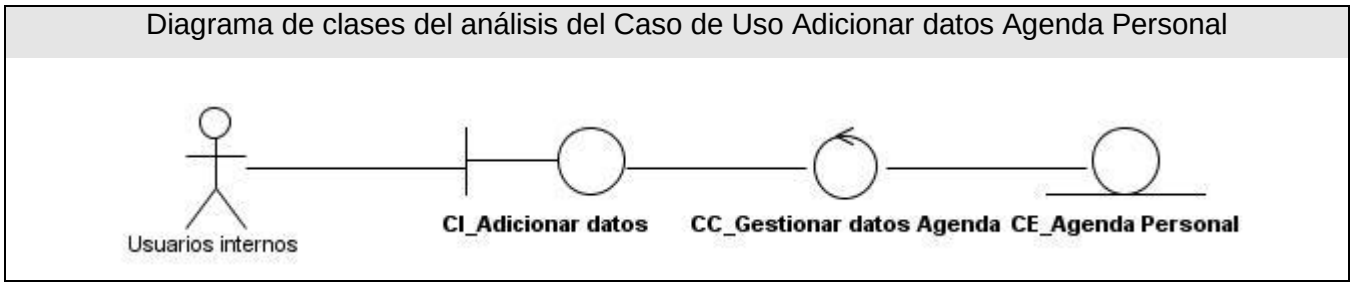


Figura 11 Diagrama de clases del análisis del Caso de Uso Adicionar datos Agenda Personal

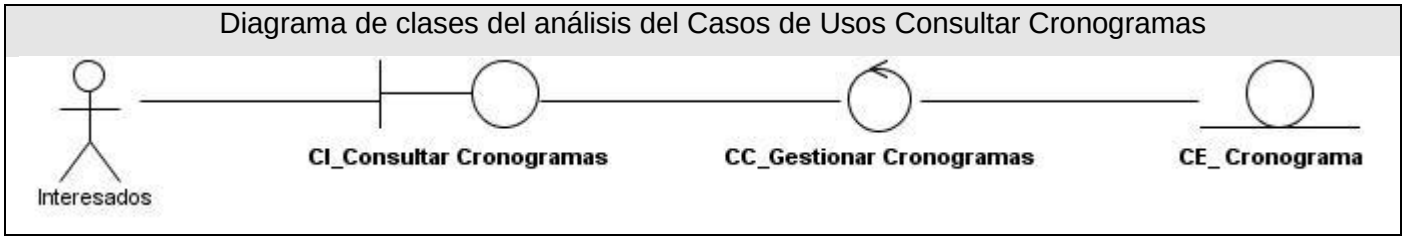


Figura 12 Diagrama de clases del análisis del Casos de Usos Consultar Cronogramas

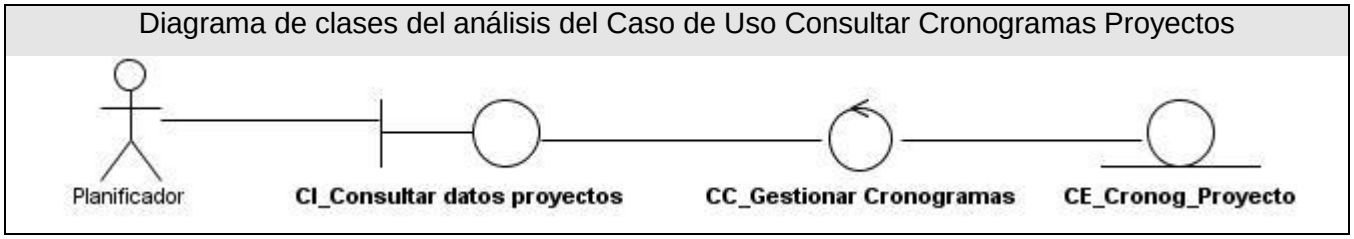


Figura 13 Diagrama de clases del análisis del Caso de Uso Consultar Cronogramas Proyectos

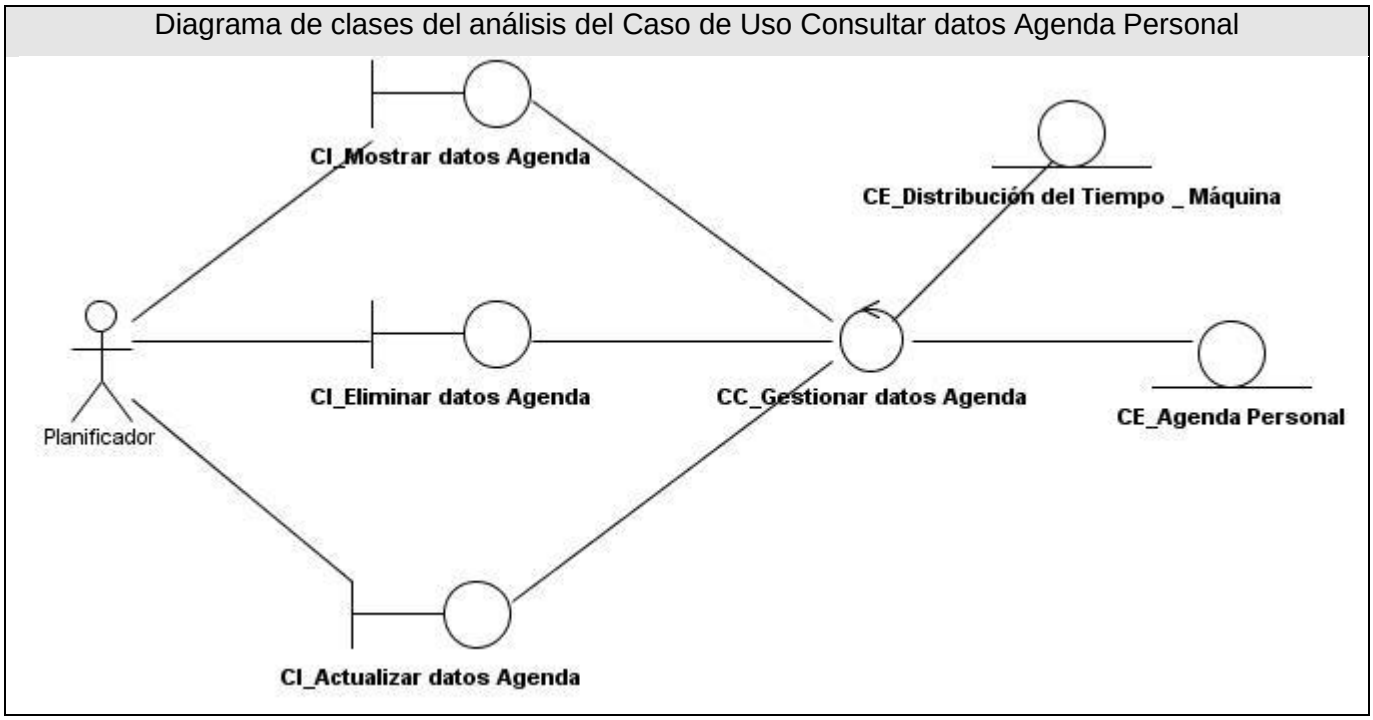


Figura 14 Diagrama de clases del análisis del Caso de Uso Consultar datos Agenda Personal

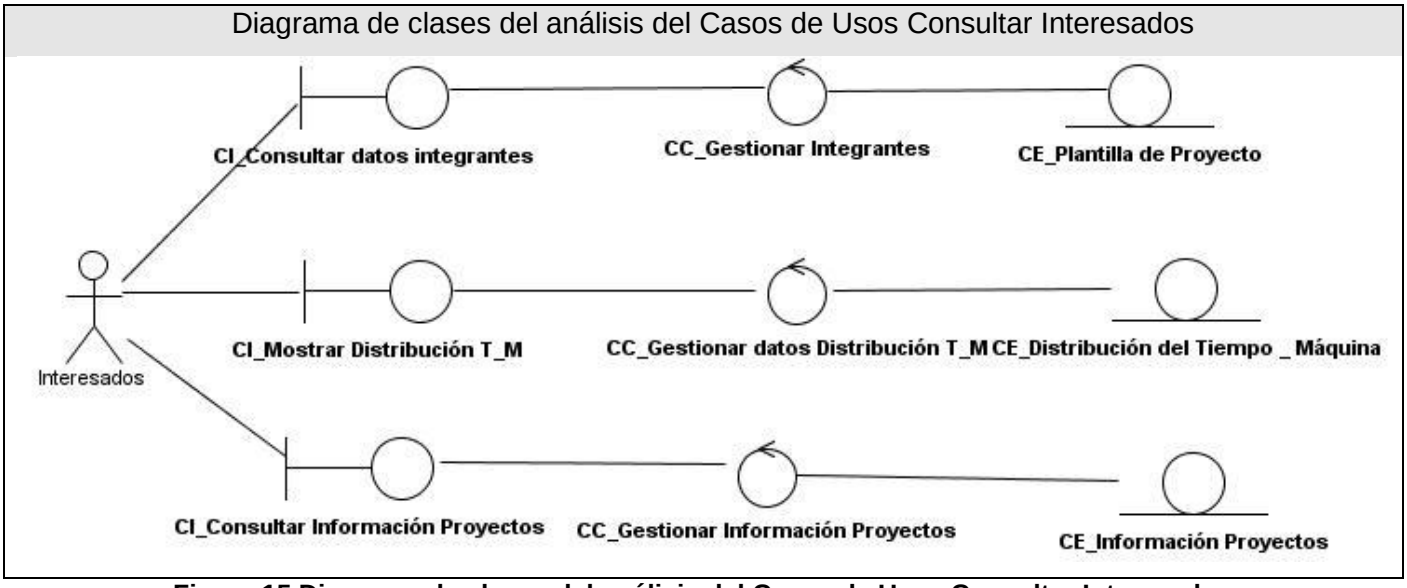


Figura 15 Diagrama de clases del análisis del Casos de Usos Consultar Interesados

Diagrama de clases del análisis del Caso de Uso Consultar Respuesta Petición Recursos

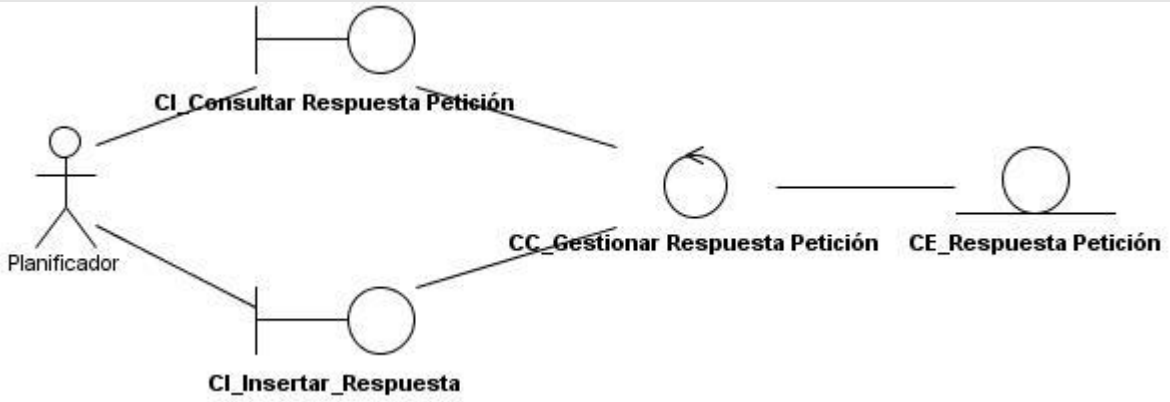


Figura 16 Diagrama de clases del análisis del Caso de Uso Consultar Respuesta Petición Recursos

Diagrama de clases del análisis del Casos de Usos Gestionar Cronogramas Planificador

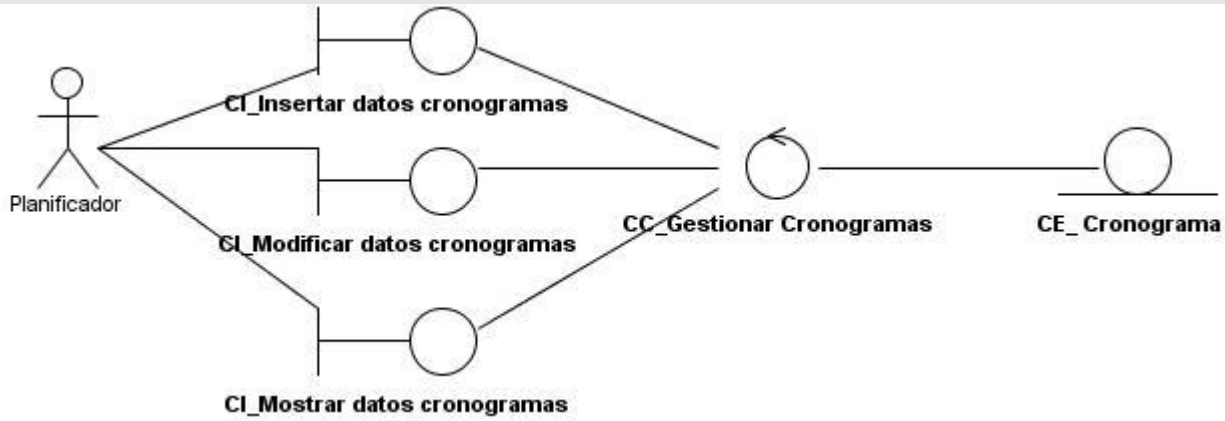


Figura 17 Diagrama de clases del análisis del Casos de Usos Gestionar Cronogramas Planificador

Diagrama de clases del análisis del Caso de Uso Gestionar Distribución del Tiempo Máquina

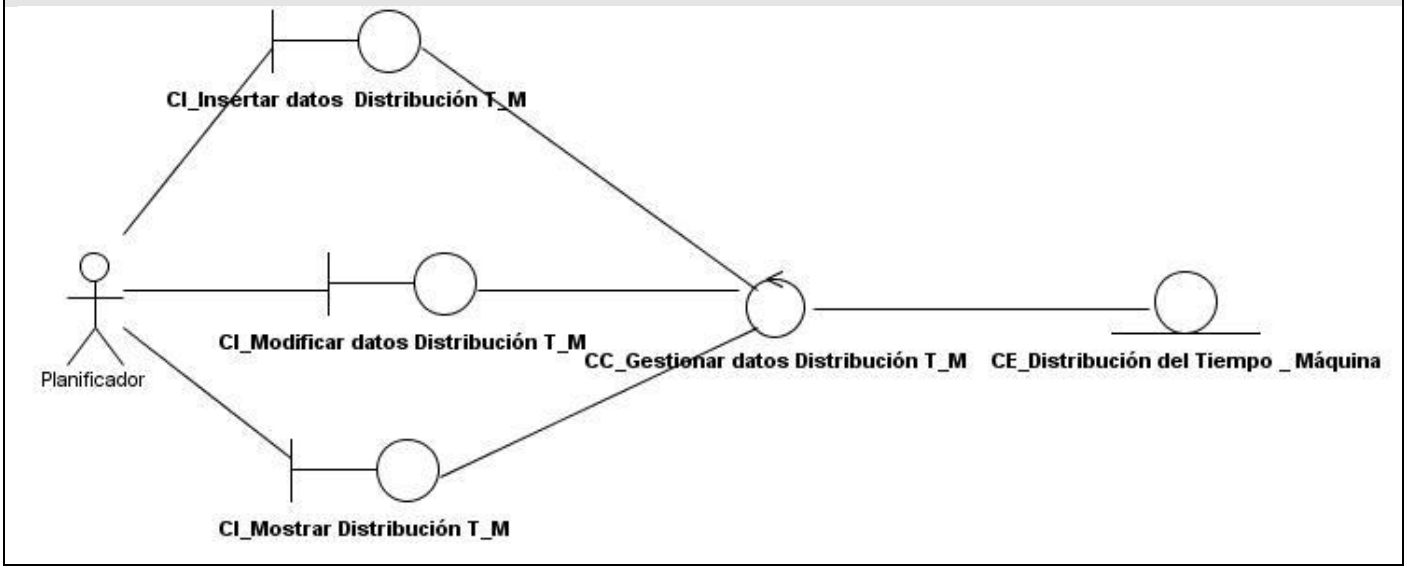


Figura 18 Diagrama de clases del análisis del Caso de Uso Gestionar Distribución del Tiempo Máquina

Diagrama de clases del análisis del Caso de Uso Gestionar Información Proyectos

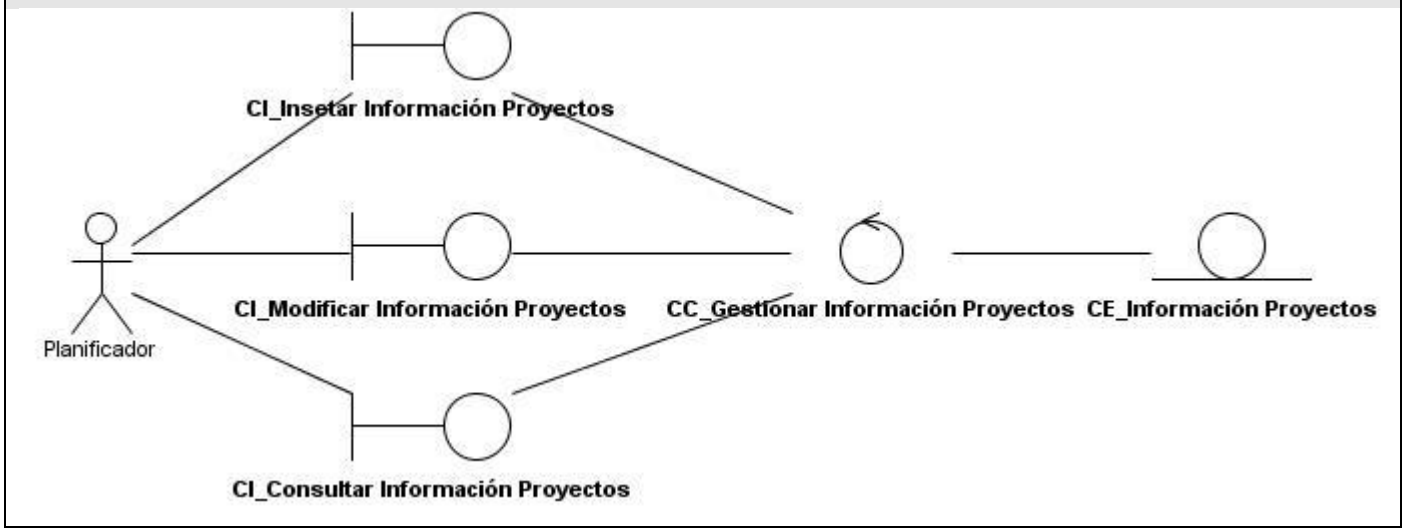


Figura 19 Diagrama de clases del análisis del Caso de Uso Gestionar Información Proyectos

Diagrama de clases del análisis del Caso de Uso Gestionar Integrantes Proyecto

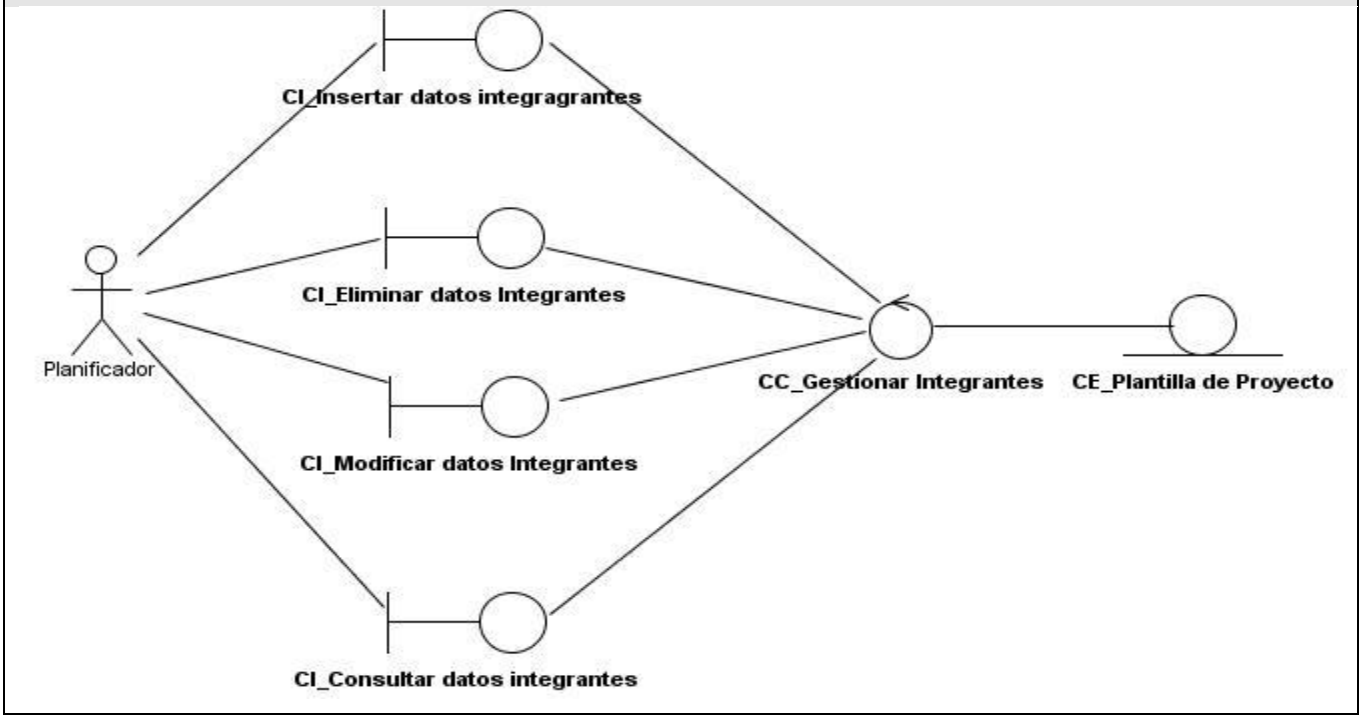


Figura 20 Diagrama de clases del análisis del Caso de Uso Gestionar Integrantes Proyecto

Diagrama de clases del análisis del Caso de Uso Gestionar Petición Recursos

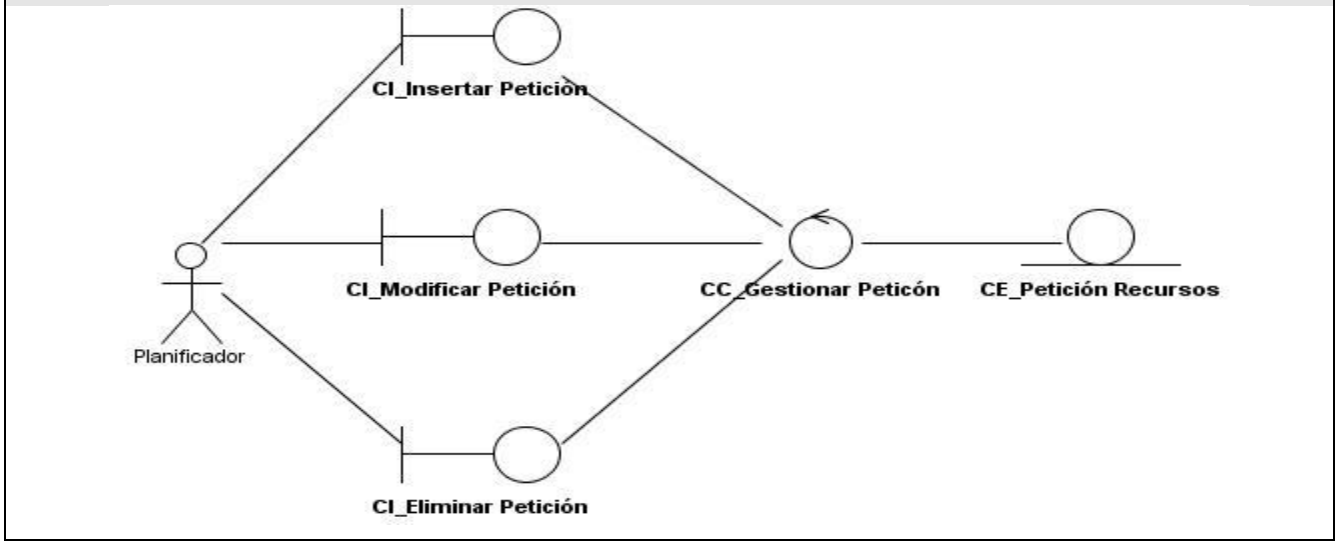


Figura 21 Diagrama de clases del análisis del Caso de Uso Gestionar Petición Recursos

3.3 Diseño

En el diseño modelamos el sistema y encontramos su forma para que soporte los requisitos funcionales incluyendo en este los requisitos no funcionales o especializados. La entrada principal al diseño es el resultado del análisis la cual es la encargada de realizar una comprensión detallada de los requisitos dándole paso al diseño para su estructura detallada del sistema presentando la base para la implementación en la fase de construcción.

Propósito del diseño:

El propósito del diseño según Jacobson, Booch, Rumbaugh es:

- ✓ Adquirir una comprensión en profundidad de los aspectos relacionados con los requisitos y restricciones relacionadas con los lenguajes de programación, componentes reutilizables, sistemas operativos, tecnologías de distribución y concurrencia, tecnologías de interfaz de usuario, tecnologías de gestión de transacciones, etc.
- ✓ Además crear una entrada apropiada y un punto de partida para las actividades de implementación subsiguientes capturando los requisitos o subsistemas individuales, interfaces y clases.
- ✓ Ser capaces de descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo, teniendo en cuenta la posible concurrencia.
- ✓ Capturar la interfaces entre los subsistemas antes del ciclo de vida del software.
- ✓ Ser capaces de visualizar y reflexionar sobre el diseño utilizando una notación común.

Crear una abstracción sin costuras de la implementación del sistema, en el sentido de que la implementación es un refinamiento directo del diseño que rellena lo existente sin cambiar la estructura.

3.3.1 Patrones de diseño

Un patrón no es más que soluciones dadas para resolver problemas en determinados contextos, entendiéndose por contextos las situaciones en las cuales es aplicable un determinado patrón. Además es

una pareja de problema-solución que presenta un nombre para facilitar la comunicación y que es ajustable a diferentes argumentos.

Un patrón de diseño describe clases y objetos permitiendo mediante la comunicación entre ellos la manera de resolver problemas de diseño en un contexto común.

3.3.2 ¿Por qué usar patrones?

Las principales características por la que se requiere usar patrones son:

- ✓ Reutilización de código.
- ✓ Flexibilidad en los cambios del software.
- ✓ Permiten diseñar un software eficiente orientado a objetos.

3.3.3 Patrones utilizados

Patrones GRASP

Los patrones GRASP (General Responsibility Asignment Software Patterns) describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones.

A continuación se presentan los principales patrones de asignación de responsabilidades, con el objetivo a cumplir en cada uno de ellos:

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

Creador: El objeto B tiene la responsabilidad de crear una instancia A si:

- ✓ B agrega los objetos A.
- ✓ B contiene los objetos A.
- ✓ B registra las instancias de los objetos A.
- ✓ B utiliza específicamente los objetos A.

- ✓ B tiene los datos de inicialización que serán transmitidos a A cuando este objeto sea creado (así que B es un Experto respecto a la creación de A).
- ✓ B es un creador de los objetos A.

Si existe más de una opción, prefiera la clase B que agregue o contenga la clase A.

Bajo Acoplamiento: El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases. Soporta el diseño de clases más independientes, que reducen el impacto de los cambios, y también más reutilizables por lo que es un patrón evaluativo.

Alta Cohesión: Es una medida de cuan relacionadas y enfocadas están las responsabilidades de una clase. Una gran funcionalidad soporta una mayor capacidad de reutilización y es un patrón evaluativo.

Controlador: Presenta la responsabilidad del manejo de mensaje de los eventos de un sistema a una clase para centralizar sus tareas. [17]

Patrón Fachada

El patrón Facade o Fachada es de la familia de los patrones GOF (Gound of four) y representa un patrón estructural este proporciona una interfaz unificada de alto nivel para un subsistema, que oculta las interfaces de bajo nivel de las clases que lo implementan. Con esto se consiguen dos objetivos fundamentales: hacer el subsistema más fácil de usar y desacoplar a los clientes de las clases del subsistema.

Su objetivo es aislar a los clientes de las interfaces de bajo nivel del subsistema colocando entre ambos una clase denominada genéricamente “fachada” del subsistema, cuya interfaz pública recoja precisamente la semántica de los servicios ofrecidos por el subsistema que interesan a los clientes más habituales.

El valor que añade esta clase es el ofrecer a los clientes una forma única y simplificada de acceder a los servicios más generales del subsistema. Para ello, los clientes enviarán mensajes solo a la fachada, y esta se encargará de poner en funcionamiento la maquinaria del subsistema para conseguir el objetivo pretendido y devolver al cliente los resultados.

Aplicación:

- ✓ Cuando se desee dotar de una interfaz sencilla y usable a un subsistema complejo. Una fachada proporciona una vista por defecto de la funcionalidad del subsistema suficiente para la mayoría de los programadores.
- ✓ Cuando se detecten demasiadas dependencias entre las clases clientes de una abstracción y las clases que implementan esta abstracción en un subsistema. En este caso debe introducirse una fachada que permita diseñar a los clientes y otros subsistemas para que dependan de una interfaz y no de una implementación.
- ✓ Cuando se quiera estructurar un sistema en subsistemas siguiendo un *patrón de capas*.
- ✓ Será de gran ayuda dotar de una fachada a cada nivel de subsistemas y utilizarla como punto de acceso al mismo. De este modo se simplificará al máximo el mantenimiento de las dependencias entre niveles.
- ✓ Cuando se tenga un subsistema que ofrece una funcionalidad muy rica y compleja y un conjunto significativo de clientes que solo necesitan usar una parte reducida de la misma. [Fernández, 2005]

Patrón de acceso a datos (DAO)

El patrón DAO es una solución al problema del diferencial de impedancia entre un programa de aplicación orientado a objetos y una base de datos relacional, empleando únicamente la interfaz de programación (API) nativa del manejador de base de datos, o algún otro sustituto como el ODBC, DBI, etc.

Las clases DAO acceden a la fuente de datos y la encapsula para los objetos clientes. Entendiendo que oculta tanto la fuente como el modo de acceder a ella, logrando así desacoplar la lógica de negocios de la lógica de acceso a datos. Esto permite que la fuente de datos pueda cambiar y no es necesario cambiar la lógica del negocio, solo las API que utiliza la clase DAO para acceder a la fuente. Con esto se obtiene varias ventajas:

1) Se tiene un paliativo al problema del diferencial de impedancia (transparencia).
2) Se baja en nivel de acoplamiento entre clases, reduciendo la complejidad de realizar cambios.
3) Se aísla las conexiones a la fuente de datos en una capa fácilmente identificable y mantenerle.[15]

Patrón Modelo-Vista-Controlador

Modelo Vista Controlador (MVC) es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón MVC se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página, el controlador es el Sistema de Gestión de Base de Datos y el modelo es el modelo de datos.

- ✓ **Modelo**: Esta es la representación específica de la información con la cual el sistema opera. La lógica de datos asegura la integridad de estos y permite derivar nuevos datos.
- ✓ **Vista**: Este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.
- ✓ **Controlador**: Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

Ventajas

- ✓ Múltiples vistas del mismo modelo.
- ✓ Vistas sincronizadas.
- ✓ Flexibilidad para cambiar las vistas y los controladores.
- ✓ La aplicación puede soportar distintos tipos de interfaz de usuario.
- ✓ Aumenta en gran medida el nivel de reusabilidad de código. Facilita una evolución continuada de los sistemas, sin puntos de ruptura, ya que un cambio en un sistema afectará a uno o más componentes pero nunca afectará significativamente al "core" de la aplicación.
- ✓ Facilidad de desarrollo y acortamiento del “Tiempo de Comercialización” gracias a la paralelización de tareas.

Inconvenientes

- ✓ Complejidad creciente.
- ✓ Cambios innecesarios. Puede ser que no todas las vistas estén interesadas en los cambios.
- ✓ Conexión entre la vista y el controlador. Hay que usar los dos a la vez.

- ✓ Si cambia el interfaz del modelo, hay que cambiar todas las vistas y todos los controladores.
- ✓ Acceso ineficiente a los datos en la vista. Puede necesitar varias llamadas al modelo para actualizar todos sus datos.

Tanto la vista como el controlador son específicos de una plataforma. Algunas herramientas de diseño de interfaces de usuario incorporan parte del procesamiento de eventos entrada. El controlador deja de ser necesario.

3.3.4 Diagramas de secuencia por realización de casos de usos.

Los diagramas de secuencia muestran la interacción entre los objetos mediante transferencias de mensajes entre objetos o subsistemas.

3.3.4.1 Diagramas de secuencia "Gestionar Distribución del Tiempo Máquina"

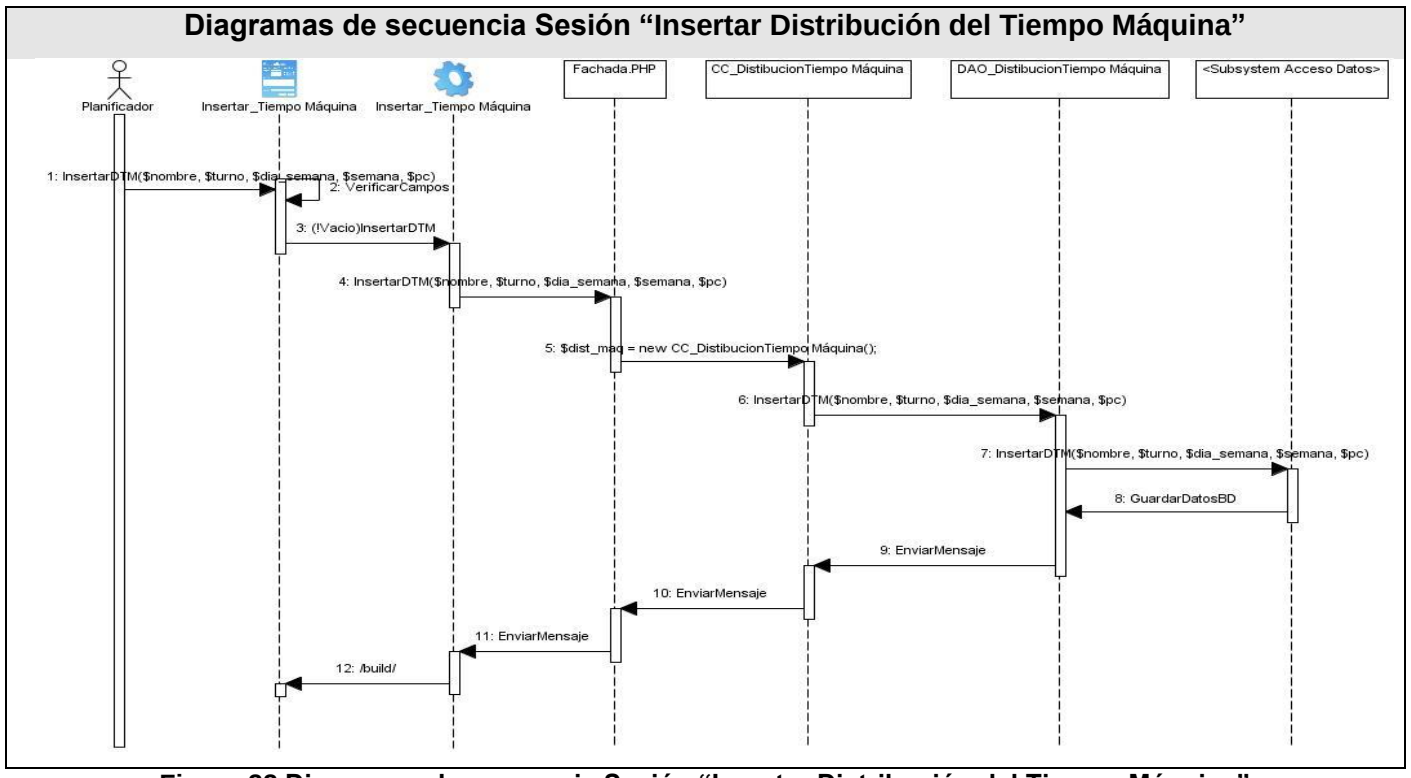


Figura 22 Diagramas de secuencia Sesión "Insertar Distribución del Tiempo Máquina"

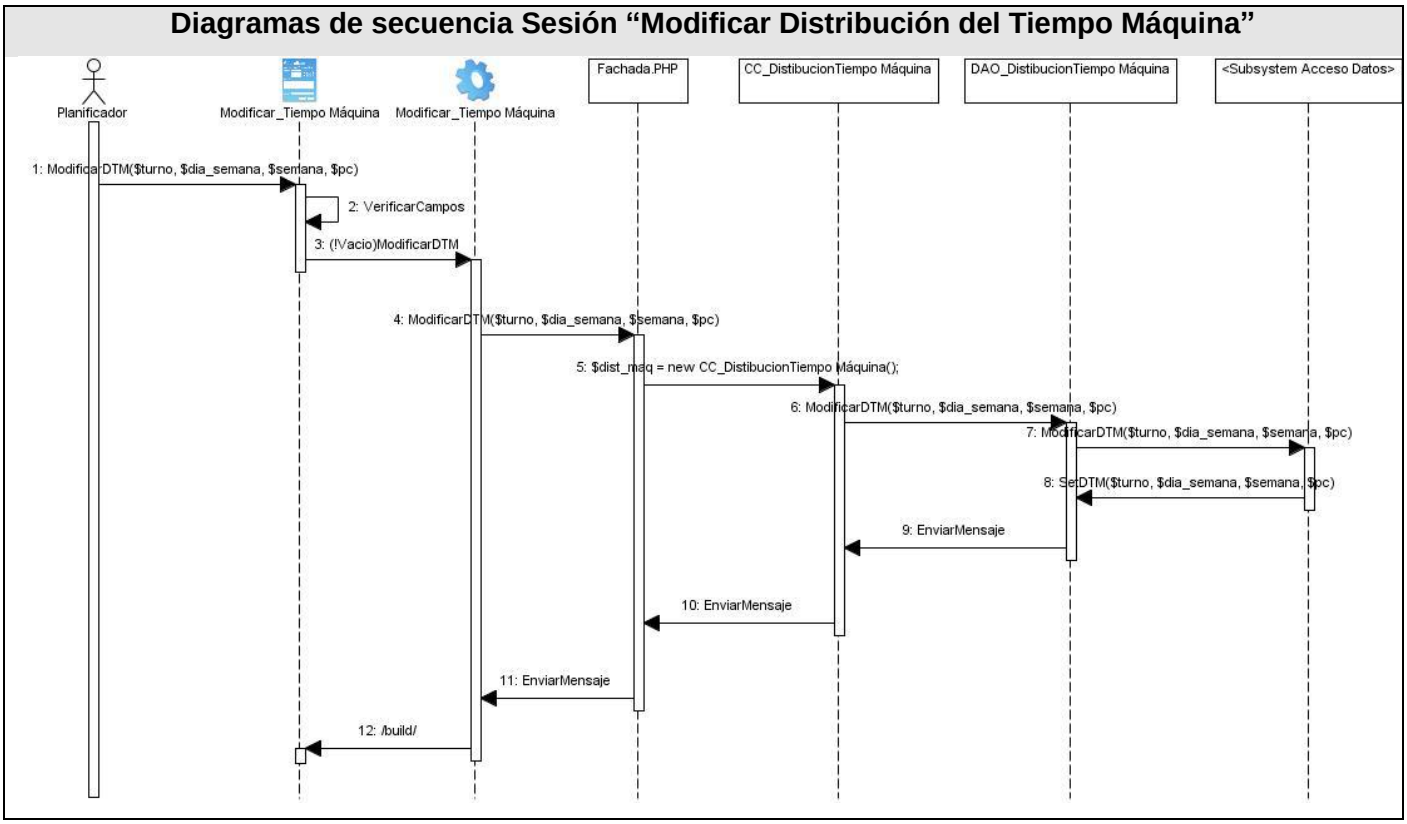


Figura 23 Diagramas de secuencia Sesión “Modificar Distribución del Tiempo Máquina”

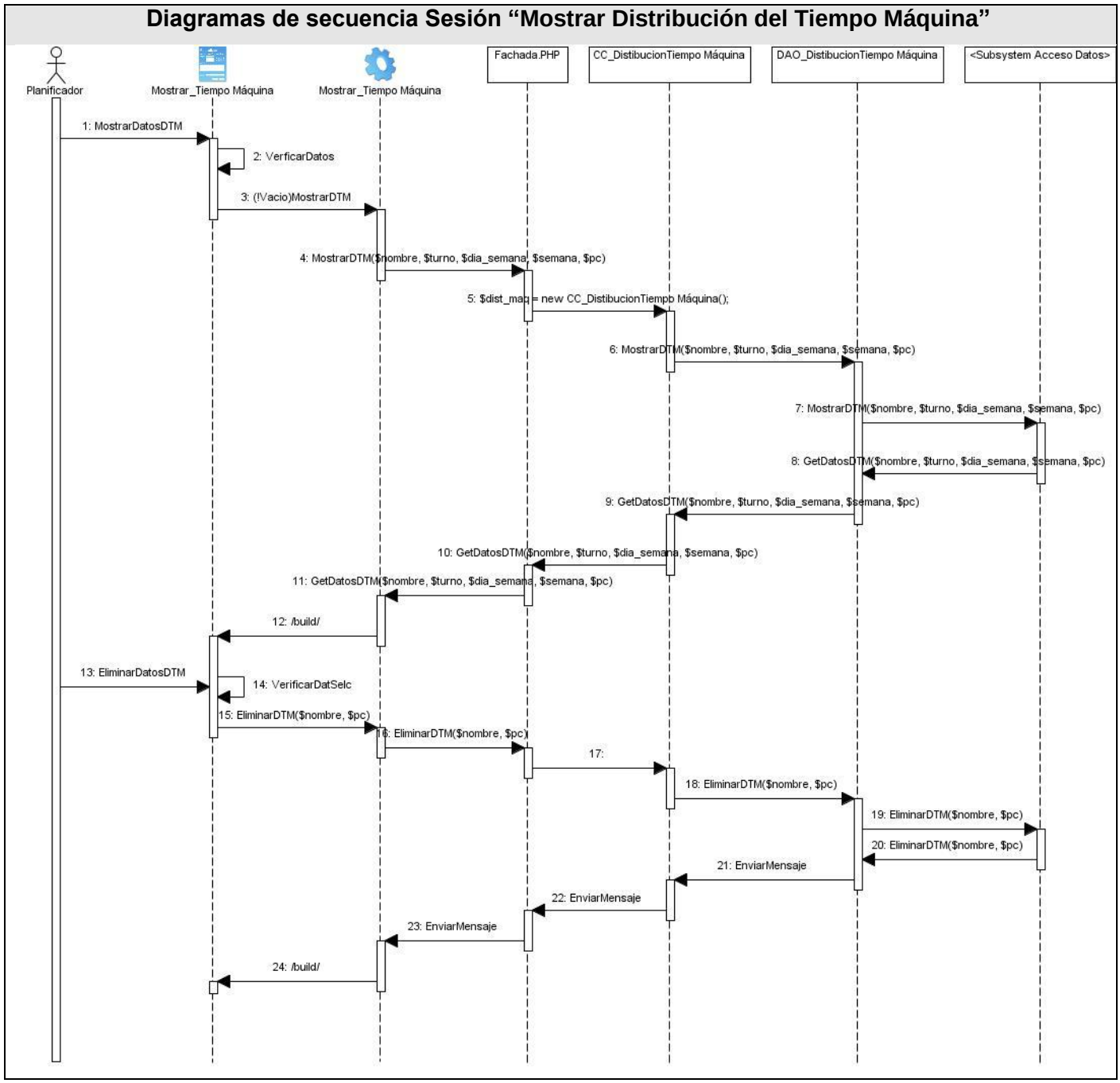


Figura 24 Diagramas de secuencia Sesión “Mostrar Distribución del Tiempo Máquina”

3.3.4.2 Diagramas de secuencia “Gestionar datos Integrantes del Proyecto”

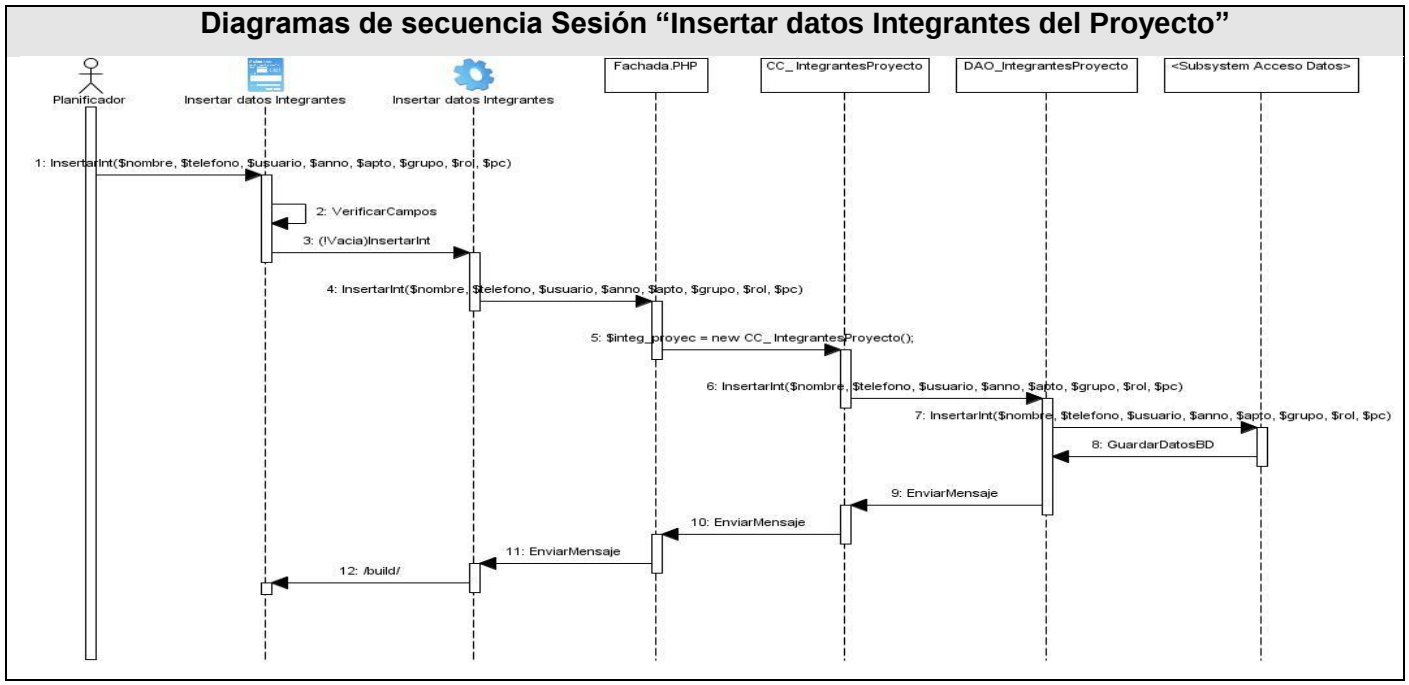


Figura 25 Diagramas de secuencia Sesión "Insertar datos Integrantes del Proyecto"

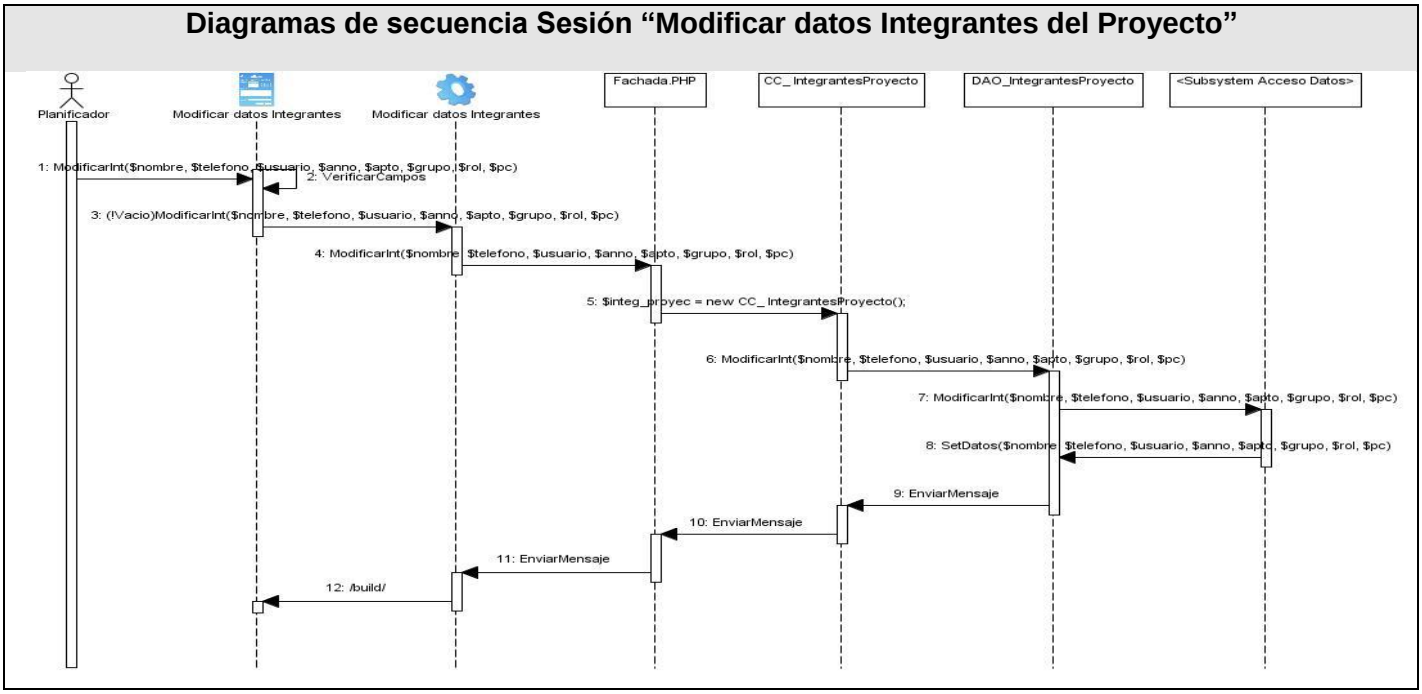


Figura 26 Diagramas de secuencia Sesión "Modificar datos Integrantes del Proyecto"

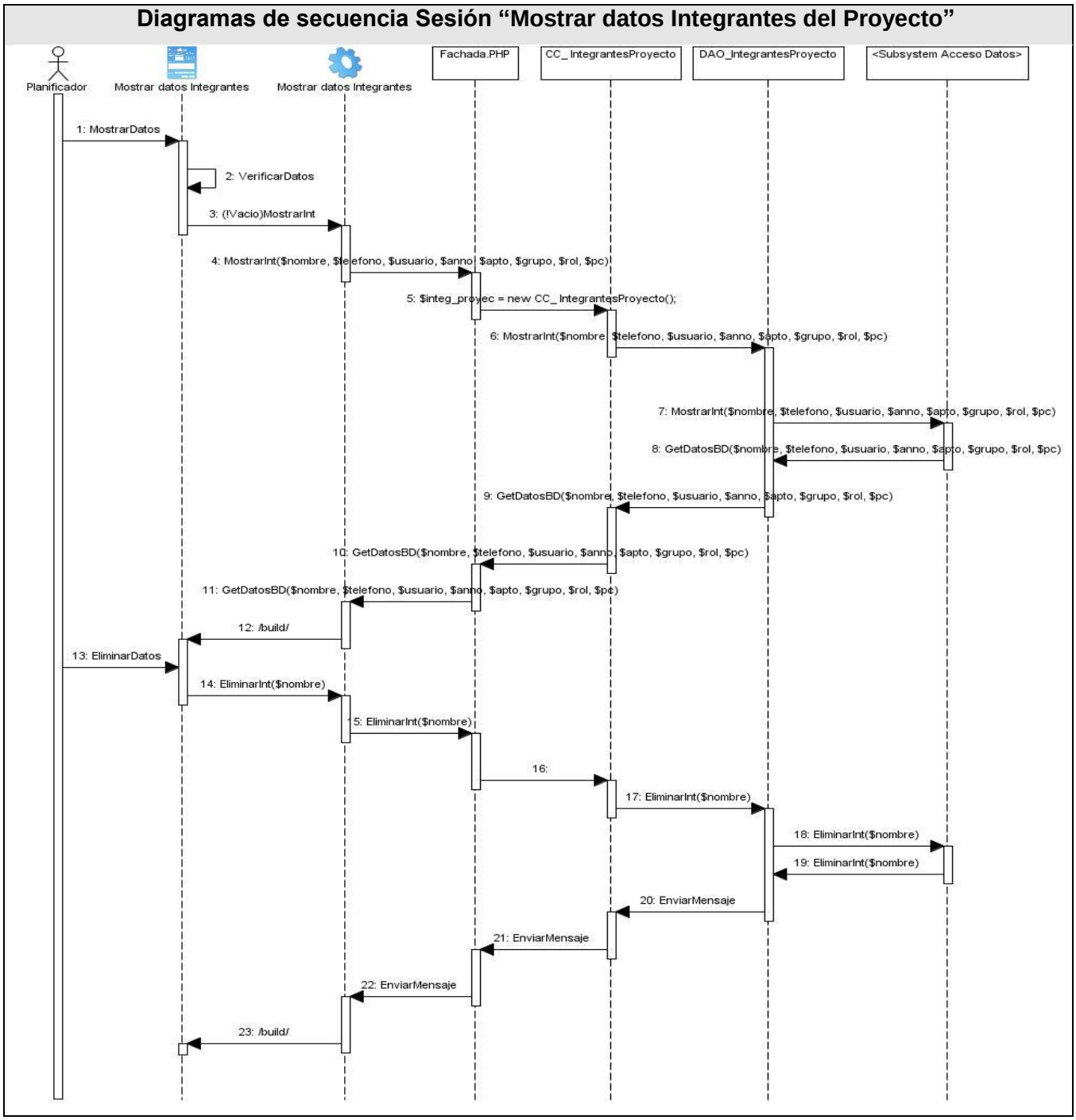


Figura 27 Diagramas de secuencia Sesión "Mostrar datos Integrantes del Proyecto"

Para ver los diagramas de secuencias restantes más significativos consultar Anexo 3.

3.3.5 Diagrama de clases

Un diagrama de clases muestra de manera detallada la especificación de las clases de una aplicación.

Diagrama de clases Web para el caso de uso "Gestionar Distribución del Tiempo Máquina"

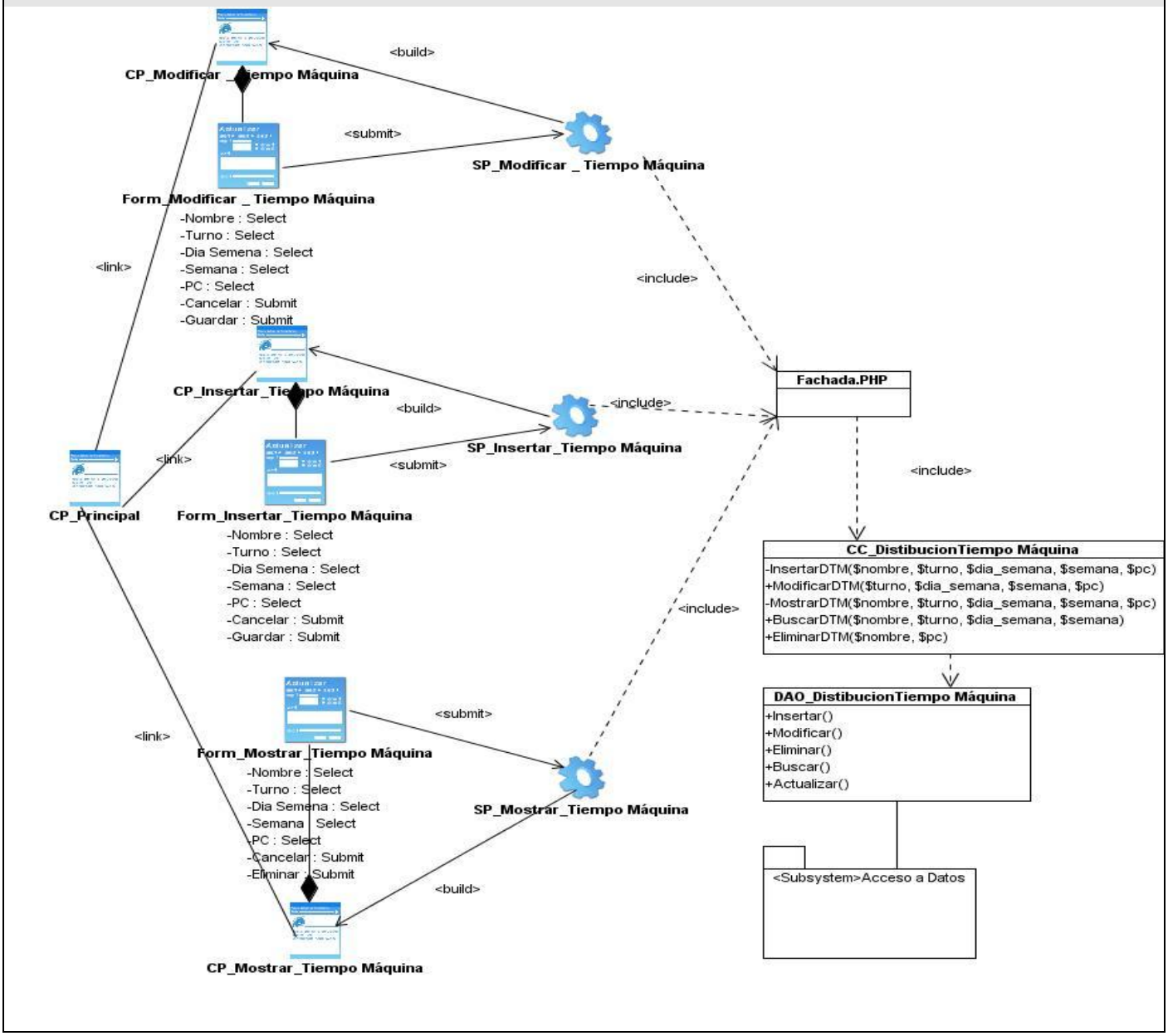


Figura 28 Diagrama de clases Web para el caso de uso "Gestionar Distribución del Tiempo Máquina"

Diagrama de clases Web para el caso de uso "Gestionar datos Integrantes del Proyecto"

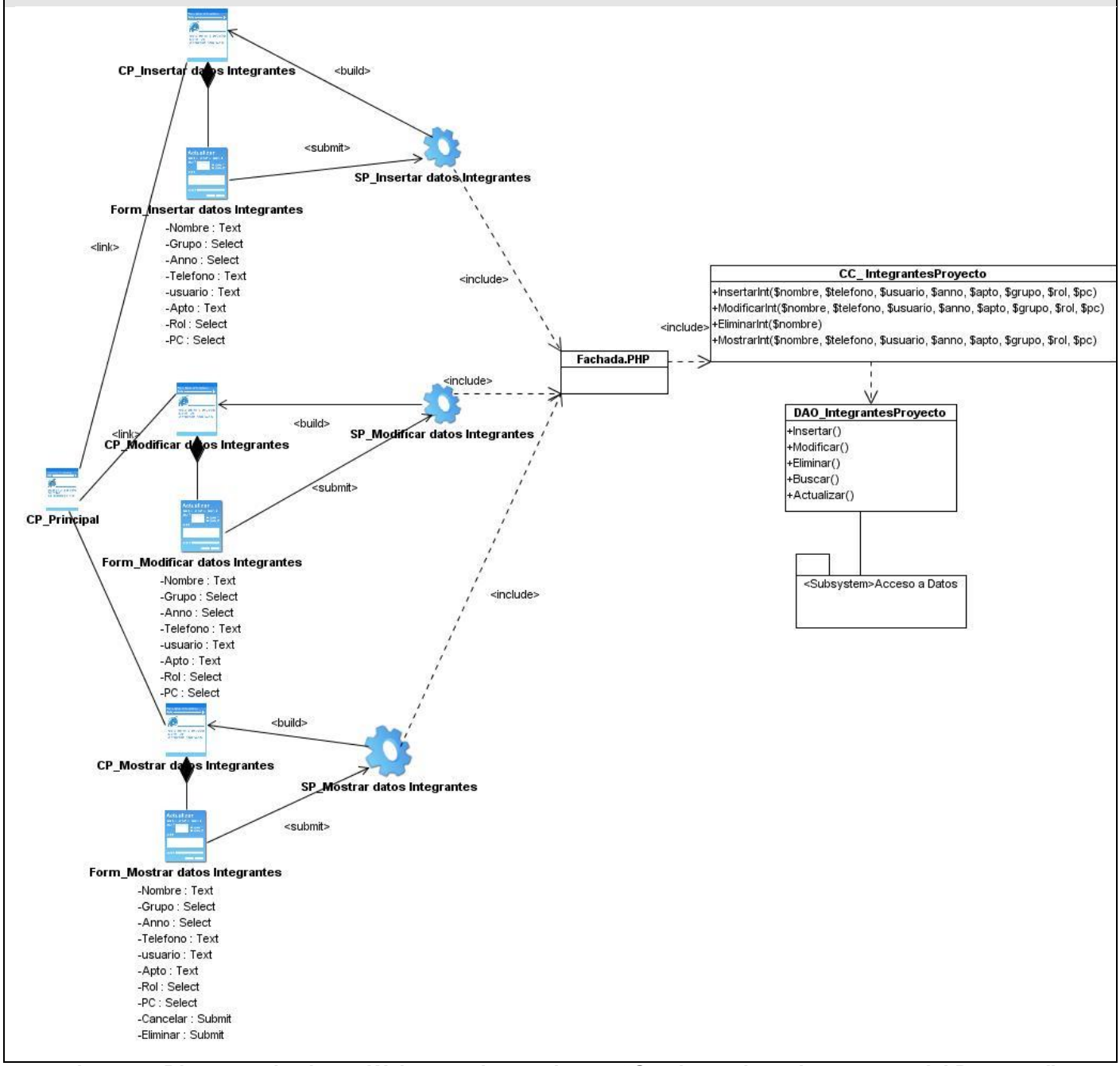


Figura 29 Diagrama de clases Web para el caso de uso "Gestionar datos Integrantes del Proyecto"

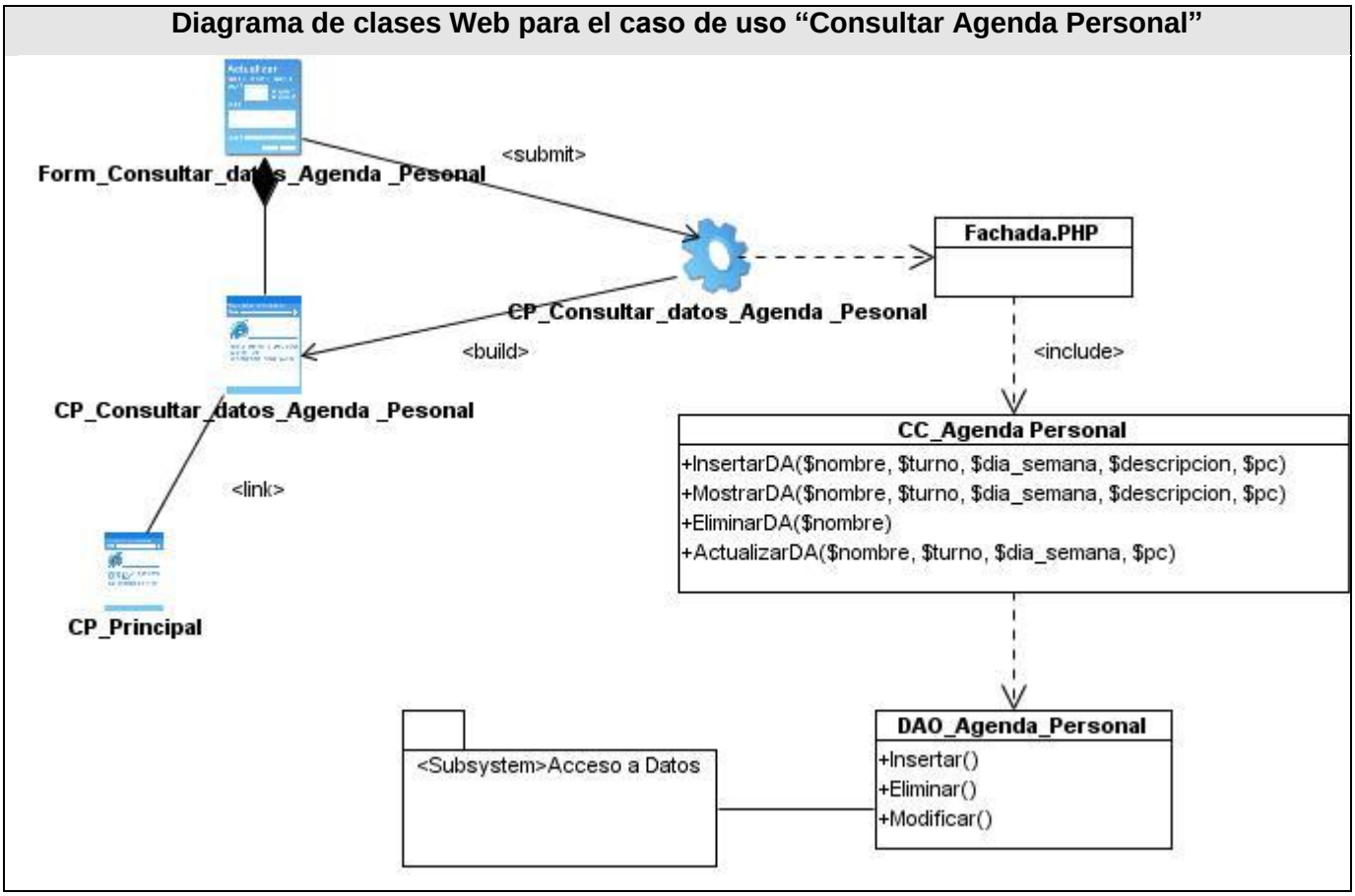


Figura 30 Diagrama de clases Web para el caso de uso "Consultar Agenda Personal"

Para ver los diagramas de clases web restantes más significativos consultar Anexo 4.

3.3.6 Descripción textual de las clases Web

La descripción textual de las clases del diseño ayuda a la comprensión del objetivo y a la responsabilidad que ocupa cada una dentro del desarrollo del sistema, es decir, con la descripción se puede especificar que es lo que realiza cada funcionalidad además de la información que se maneja en cada una de ellas.

3.3.6.1 Clase Controladora CC_Agenda Personal

Tabla 10 Clase Controladora CC_Agenda Personal

Nombre	CC_Agenda Personal
Tipo de clase	Controladora

Atributo	**Tipo**
Para cada responsabilidad	
Nombre	InsertarDA($nombre, $turno, $dia_semana, $descripción, $pc) MostrarDA($nombre, $turno, $dia_semana, $descripción, $pc) EliminarDA($nombre) ActualizarDA($nombre, $turno, $dia_semana, $pc) Listado_Afectaciones()
Descripción	Controla todo lo relacionado con la adición de las afectaciones que puedan surgir así como para poder realizar la distribución del tiempo de máquina.

3.3.6.2 Clase Controladora CC_DistibuciónTiempo Máquina

Tabla 11 Controladora CC_DistibuciónTiempo Máquina

Nombre	**CC_DistibuciónTiempo Máquina**
Tipo de clase	Controladora
Atributo	**Tipo**
Para cada responsabilidad	
Nombre	InsertarDTM($nombre, $turno, $dia_semana, $semana, $pc) ModificarDTM($turno, $dia_semana, $semana, $pc) MostrarDTM($nombre, $turno, $dia_semana, $semana, $pc) BuscarDTM($nombre, $turno, $dia_semana, $semana) EliminarDTM($nombre, $pc)
Descripción	Controla todo lo relacionado con la adición y manejo de las principales funcionalidades de la distribución de tiempo de máquina.

3.3.6.3 Clase Controladora CC_ Información Proyectos

Tabla 12 Clase Controladora CC_ Información Proyectos

Nombre	**CC_ Información Proyectos**	
Tipo de clase	Controladora	
Atributo		**Tipo**
Para cada responsabilidad		
Nombre	InsertarInf($nombre, $apellidos, $usuario, $teléfono, $non_proyect, $polo) ModificarInf($nombre, $apellidos, $usuario, $teléfono, $non_proyect, $polo) MostrarInf($nombre, $apellidos, $usuario, $teléfono, $non_proyect, $polo) EliminarInf($non_proyect) Listar_Inf()	
Descripción	Controla todo lo relacionado con la adición y manejo de las principales funcionalidades de la información de los proyectos de la facultad.	

3.3.6.4 Clase Controladora CC_ IntegrantesProyecto

Tabla 13 Clase Controladora CC_ IntegrantesProyecto

Nombre	**CC_ IntegrantesProyecto**	
Tipo de clase	Controladora	
Atributo		**Tipo**
Para cada responsabilidad		
Nombre	InsertarInt($nombre, $teléfono, $usuario, $año, $apto, $grupo, $rol, $pc) ModificarInt($nombre, $teléfono, $usuario, $año, $apto, $grupo, $rol, $pc) EliminarInt($nombre) MostrarInt($nombre, $teléfono, $usuario, $año, $apto, $grupo, $rol, $pc) Listado_Integrantes()	
Descripción	Controla todo lo relacionado con la adición y manejo de las principales funcionalidades de los integrantes del proyecto de calidad.	

3.3.6.5 Clase Controladora CC_Cronogramas

Tabla 14 Clase Controladora CC_Cronogramas

Nombre	**CC_Cronogramas**	
Tipo de clase	Controladora	
Atributo		**Tipo**
Para cada responsabilidad		
Nombre	InsetarCR($nomb_proy, $f_inic, $f_fin) ModificarCR($nomb_proy, $f_inic, $f_fin) MostrarCR($nomb_proy, $f_inic, $f_fin) EliminarCR($nomb_proy) List_Cronog()	
Descripción	Controla todo lo relacionado con la adición y manejo de las principales funcionalidades a la hora de la realización de los cronogramas de auditorias, revisiones y pruebas.	

3.3.6.6 Clase Controladora CC_ Petición Recursos

Tabla 15 Clase Controladora CC_ Petición Recursos

Nombre	**CC_ Petición Recursos**	
Tipo de clase	Controladora	
Atributo		**Tipo**
Para cada responsabilidad		
Nombre	InsertarRec($cantRH, $cantRT, $diraplic, $nombproy, $correo) ModificarRec($cantRH, $cantRT, $diraplic, $nombproy, $correo) EliminarRec($nomproy) MostrarRec($cantRH, $cantRT, $diraplic, $nombproy, $correo)	
Descripción	Controla todo lo relacionado con la adición y manejo de las principales funcionalidades a la hora de realizar la petición de los recursos necesarios para	

	determinado objetivo.

3.3.6.6 Clase Controladora Fachada

Tabla 16 Clase Controladora Fachada

Nombre	**Fachada**	
Tipo de clase	Interfaz(PHP, servidor)	
Atributo		**Tipo**
agen_pers		CC_Agenda Personal
dist_maq		CC_DistibuciónTiempo Máquina
integ_proyec		CC_ IntegrantesProyecto
cronog		CC_Cronogramas
inf_proyec		CC_ Información Proyectos
pet_rec		CC_ Petición Recursos
Para cada responsabilidad		
Nombre	InsertarDA(nombre, turno, dia_semana, descripción, pc) MostrarDA($nombre, $turno, $dia_semana, $descripción, $pc) EliminarDA($nombre) ActualizarDA($nombre, $turno, $dia_semana, $pc) Listado_Afectaciones() InsertarDTM($nombre, $turno, $dia_semana, $semana, $pc) ModificarDTM($turno, $dia_semana, $semana, $pc) MostrarDTM($nombre, $turno, $dia_semana, $semana, $pc) BuscarDTM($nombre, $turno, $dia_semana, $semana) EliminarDTM($nombre, $pc) InsertarInf($nombre, $apellidos, $usuario, $teléfono, $non_proyect, $polo) ModificarInf($nombre, $apellidos, $usuario, $teléfono, $non_proyect, $polo) MostrarInf($nombre, $apellidos, $usuario, $teléfono, $non_proyect, $polo) EliminarInf($non_proyect) Listar_Inf()	

	InsertarInt($nombre, $teléfono, $usuario, $año, $apto, $grupo, $rol, $pc) ModificarInt($nombre, $teléfono, $usuario, $año, $apto, $grupo, $rol, $pc) EliminarInt($nombre) MostrarInt($nombre, $teléfono, $usuario, $año, $apto, $grupo, $rol, $pc) Listado_Integrantes() InsetarCR($nomb_proy, $f_inic, $f_fin) ModificarCR($nomb_proy, $f_inic, $f_fin) MostrarCR($nomb_proy, $f_inic, $f_fin) EliminarCR($nomb_proy) List_Cronog() InsertarRec($cantRH, $cantRT, $diraplic, $nombproy, $correo) ModificarRec($cantRH, $cantRT, $diraplic, $nombproy, $correo) EliminarRec($nomproy) MostrarRec($cantRH, $cantRT, $diraplic, $nombproy, $correo) Listar_Petición()
Descripción	Estas funciones son las encargadas de recibir los datos y distribuirlos a las clases controladoras correspondientes de acuerdo a la acción solicitada desde las páginas servidoras superiores.

3.3.6.7 Clase Entidad CE_ Petición Recursos

Tabla 17 Clase Entidad CE_ Petición Recursos

Nombre	CE_ Petición Recursos
Tipo de clase	Entidad
Atributo	**Tipo**
cantRH	Int
cantRT	Int
diraplic	String
nombproy	String
correo	String

Para cada responsabilidad	
Nombre	Petición Recursos (pcantRH, pcantRT, pdiraplic, pnombproy, pcorreo) Get_ cantRH () Get_ cantRT() Get_ Diraplic() Get_ Nombproy() Get_ correo() Set_ cantRH () Set_ cantRT() Set_ Diraplic() Set_ Nombproy() Set_ correo()
Descripción	Clase entidad que representa la tabla petición_rec en la base de datos.

3.3.6.8 Clase Entidad CE_ Integrantes Proyecto

Tabla 18 Clase Entidad CE_ Integrantes Proyecto

Nombre	CE_ Integrantes Proyecto
Tipo de clase	Entidad
Atributo	**Tipo**
año	Int
apto	Int
nombre	String
teléfono	String
usuario	String
grupo	Int
rol	String
pc	Int
Para cada responsabilidad	
Nombre	Integrantes Proyecto(pnombre, ptelefono, pusuario, paño, papto, pgrupo, prol,

	ppc) Get_ nombre () Get_ telefono() Get_ usuario() Get_ año() Get_ apto() Get_ grupo () Get_ rol() Get_ pc() Set _ nombre () Set _ telefono() Set _ usuario() Set _ año() Set _ apto() Set _ grupo () Set_ rol() Set_ pc()
Descripción	Clase entidad que representa la tabla integrantes_proy en la base de datos.

3.3.6.9 Clase Entidad CE_ Distribución_T_M

Tabla 19 Clase Entidad CE_ Distribución_T_M

Nombre	CE_ Distribución_T_M
Tipo de clase	Entidad
Atributo	**Tipo**
nombre	String
turno	String
semana	Int
dia_semana	String
pc	Int

Para cada responsabilidad	
Nombre	Distribución_T_M (pnombre, pturno, pdia_semana, psemana,ppc) Get_ nombre () Get_ turno() Get_ dia_semana() Get_ semana() Get_ pc() Set _ nombre () Set _ turno () Set _ dia_semana () Set _ semana () Set_ pc()
Descripción	Clase entidad que representa la tabla distrib_tiempo_máquina en la base de datos.

3.3.6.10 Clase Entidad CE_ Agenda Personal

Tabla 20 Clase Entidad CE_ Agenda Personal

Nombre	CE_ Agenda Personal
Tipo de clase	Entidad
Atributo	**Tipo**
nombre	String
turno	String
semana	Int
dia_semana	String
pc	Int
descripción	String
Para cada responsabilidad	
Nombre	Agenda Personal(pnombre, pturno, pdia_semana, pdescripción, ppc) Get_ nombre ()

	Get_ turno() Get_ dia_semana() Get_ descripción () Get_ pc() Set _ nombre () Set _ turno () Set _ dia_semana () Set _ descripción () Set_ pc()
Descripción	Clase entidad que representa la tabla agenda_personal en la base de datos.

3.3.6.11 Clase Entidad CE_ Información_Proyectos

Tabla 21 Clase Entidad CE_ Información_Proyectos

Nombre	CE_ Información_Proyectos	
Tipo de clase	Entidad	
Atributo		**Tipo**
nombre		String
teléfono		String
usuario		String
apellidos		String
polo		String
non_proyect		String
Para cada responsabilidad		
Nombre	Información_Proyectos(pnombre, papellidos, pusuario, ptelefono, pnon_proyect, ppolo) Get_ nombre () Get_ telefono() Get_ usuario() Get_ apellidos()	

	Get_ non_pryect() Get_ polo () Set _ nombre () Set _ telefono() Set _ usuario() Set _ apellidos () Set_ non_proyect() Set_ polo()
Descripción	Clase entidad que representa la tabla informac_proyectos en la base de datos.

3.3.6.12 Clase Entidad CE_ Cronogramas

Tabla 22 Clase Entidad CE_ Cronogramas

Nombre	CE_ Cronogramas	
Tipo de clase	Entidad	
Atributo		**Tipo**
nomb_proy f_inic f_fin		String Int Int
Para cada responsabilidad		
Nombre	Cronograma(pnomb_proy, pf_inic, pf_fin) Get_ nom_proy () Get_ f_inic() Get_ f_fin() Set _ nom_proy () Set _ f_inic() Set _ f_fin()	
Descripción	Clase entidad que representa la tabla cronograma en la base de datos.	

3.3.6.13 Clase de acceso a datos Base_Datos

Nombre	Base_Datos	
Tipo de clase	DAO	
Atributo		**Tipo**
Servidor		String
Usuario		String
Password		String
BaseDatos		String
Conexión		String
Error		String
Ado_DB		ADONewConexión
Para cada responsabilidad		
Nombre	Base_Datos(s=SERVIDOR, u=USUARIO, p=PASSWORD, bd=BASEDATOS, mt_MOTOR_BD) Abrir_Conexión () Cerrar_Conexión() Insertar(sql) Modificar (sql) Eliminar(sql) Ejecutar_consulta(sql) Ejecutar_inserción(sql) Empezar_Transacción() Terminar_Transacción() Ocurrió_Error_Transacción() Forzar_Transacción()	
Descripción	Clase encargada de realizar la conexión a la base datos y ejecutar las consultas utilizando el paquete ADO.	

3.3.6.14 Clase de acceso DAO_ Agenda Personal

Tabla 23 Clase de acceso DAO_ Agenda Personal

Nombre	DAO_Agenda Personal	
Tipo de clase	DAO	
Atributo		**Tipo**
Para cada responsabilidad		
Nombre	InsertarDA(nombre, turno, dia_semana, descripción, pc) MostrarDA($nombre, $turno, $dia_semana, $descripción, $pc) EliminarDA($nombre) ActualizarDA($nombre, $turno, $dia_semana, $pc) Listado_Afectaciones()	
Descripción	Clase encargada de realizar todas las consultas a la base datos referentes al manejo de la agenda personal.	

3.3.6.15 Clase de acceso DAO_ DistibuciónTiempo Máquina.

Tabla 24 Clase de acceso DAO_ DistibuciónTiempo Máqu

Nombre	**DAO_DistibuciónTiempo Máquina**	
Tipo de clase	DAO	
Atributo		**Tipo**
Para cada responsabilidad		
Nombre	InsertarDTM($nombre, $turno, $dia_semana, $semana, $pc) ModificarDTM($turno, $dia_semana, $semana, $pc) MostrarDTM($nombre, $turno, $dia_semana, $semana, $pc) BuscarDTM($nombre, $turno, $dia_semana, $semana) EliminarDTM($nombre, $pc)	
Descripción	Clase encargada de realizar todas las consultas a la base datos referentes al manejo de la distribución de tiempo de máquina.	

3.3.6.16 Clase de acceso DAO_ Información Proyectos

Tabla 25 Clase de acceso DAO_ Información Proyectos

Nombre	**DAO_ Información Proyectos**
Tipo de clase	DAO
Atributo	**Tipo**
Para cada responsabilidad	
Nombre	InsertarInf($nombre, $apellidos, $usuario, $teléfono, $non_proyect, $polo) ModificarInf($nombre, $apellidos, $usuario, $teléfono, $non_proyect, $polo) MostrarInf($nombre, $apellidos, $usuario, $teléfono, $non_proyect, $polo) EliminarInf($non_proyect) Listar_Inf()
Descripción	Clase encargada de realizar todas las consultas a la base datos referentes al manejo de la información de los proyectos de la facultad.

3.3.6.17 Clase de acceso DAO_ IntegrantesProyecto

Tabla 26 Clase de acceso DAO_ IntegrantesProyecto

Nombre	**DAO_ IntegrantesProyecto**
Tipo de clase	DAO
Atributo	**Tipo**
Para cada responsabilidad	
Nombre	InsertarInt($nombre, $teléfono, $usuario, $año, $apto, $grupo, $rol, $pc) ModificarInt($nombre, $teléfono, $usuario, $año, $apto, $grupo, $rol, $pc) EliminarInt($nombre) MostrarInt($nombre, $teléfono, $usuario, $año, $apto, $grupo, $rol, $pc) Listado_Integrantes()

Descripción	Clase encargada de realizar todas las consultas a la base datos referentes al manejo de los integrantes del proyecto de calidad.

3.3.6.18 Clase de acceso DAO_ Cronogramas

Tabla 27 Clase de acceso DAO_ Cronogramas

Nombre	**DAO_Cronogramas**	
Tipo de clase	DAO	
Atributo		**Tipo**
Para cada responsabilidad		
Nombre	InsetarCR($nomb_proy, $f_inic, $f_fin) ModificarCR($nomb_proy, $f_inic, $f_fin) MostrarCR($nomb_proy, $f_inic, $f_fin) EliminarCR($nomb_proy) List_Cronog()	
Descripción	Clase encargada de realizar todas las consultas a la base datos referentes al manejo de los cronogramas de auditorías, revisiones y pruebas.	

3.3.6.19 Clase de acceso DAO_ Petición Recursos

Tabla 28 Clase de acceso DAO_ Petición Recursos

Nombre	**DAO_ Petición Recursos**	
Tipo de clase	DAO	
Atributo		**Tipo**
Para cada responsabilidad		
Nombre	InsertarRec($cantRH, $cantRT, $diraplic, $nombproy, $correo) ModificarRec($cantRH, $cantRT, $diraplic, $nombproy, $correo) EliminarRec($nomproy)	

	MostrarRec($cantRH, $cantRT, $diraplic, $nombproy, $correo)
Descripción	Clase encargada de realizar todas las consultas a la base datos referentes al manejo de la petición de los recursos.

3.4 Subsistema de Acceso a datos

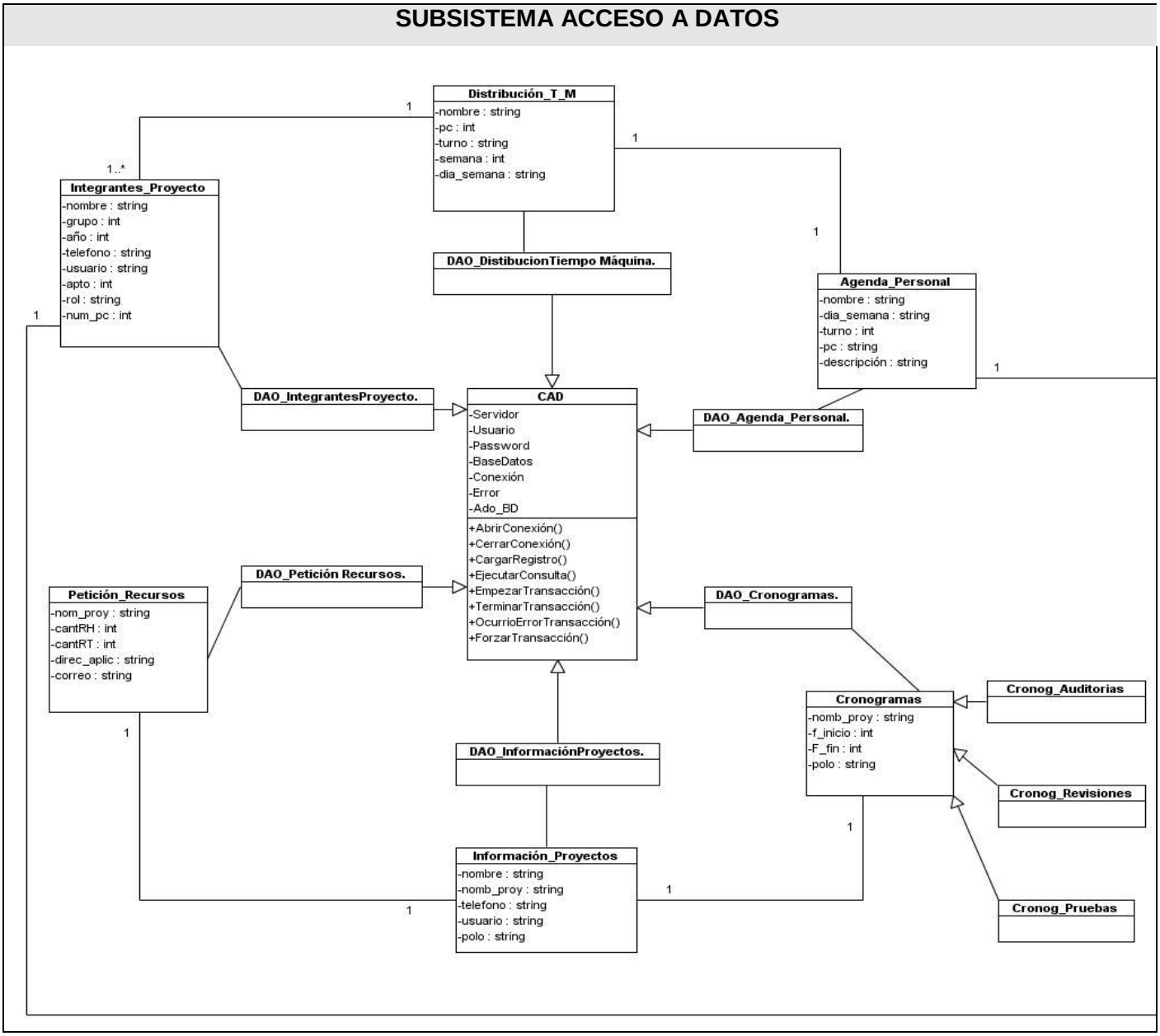


Figura 31 Subsistema de Acceso a datos

3.5 Diseño de la Base de Datos

3.5.1 Diagrama de clases persistentes.

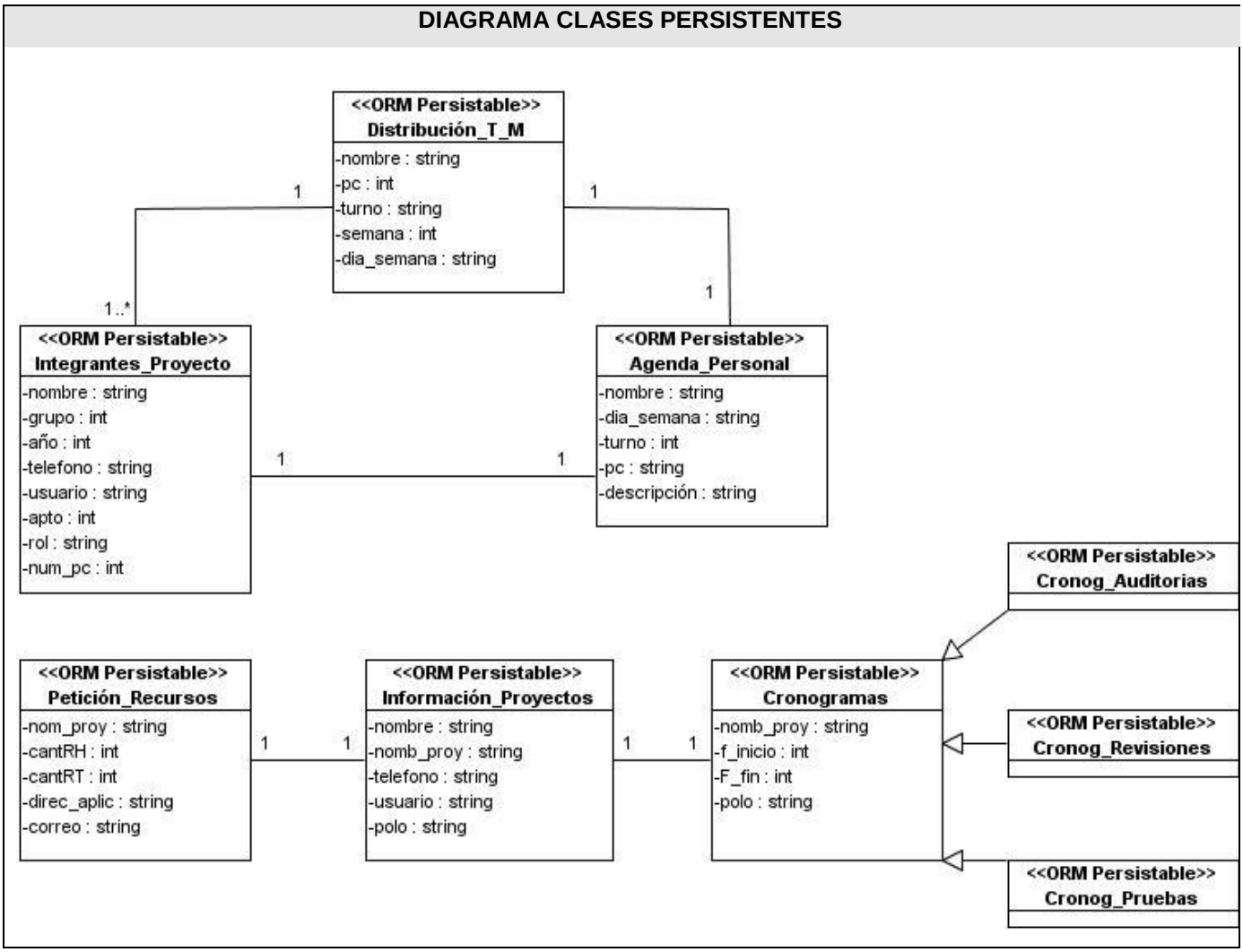


Figura 32 Diagrama de clases persistentes

3.5.2 Diagrama Entidad Relación

DIAGRAMA ENTIDAD RELACION

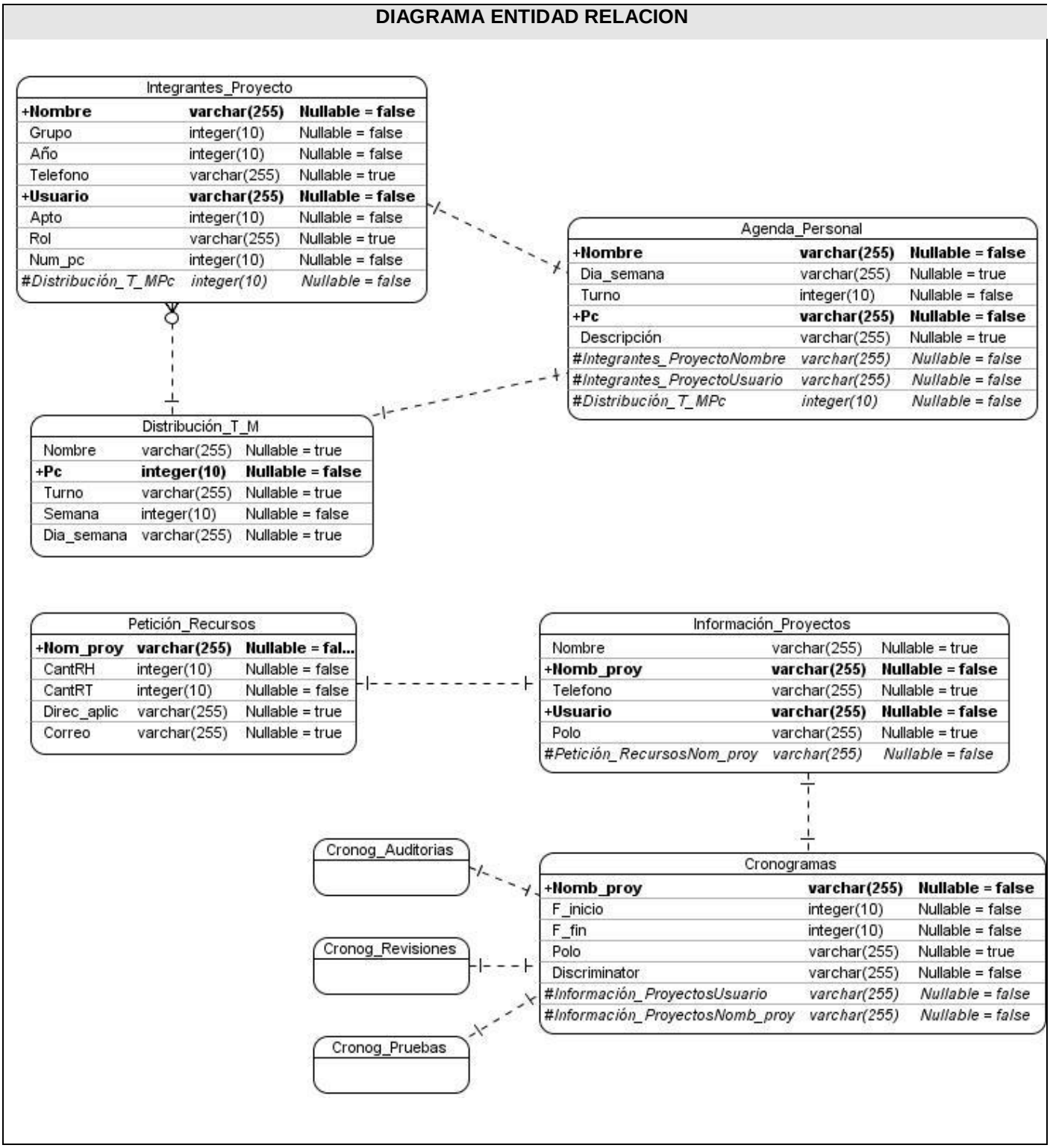


Figura 33 3.5.2 Diagrama Entidad Relación

3.6 Diagrama de Despliegue

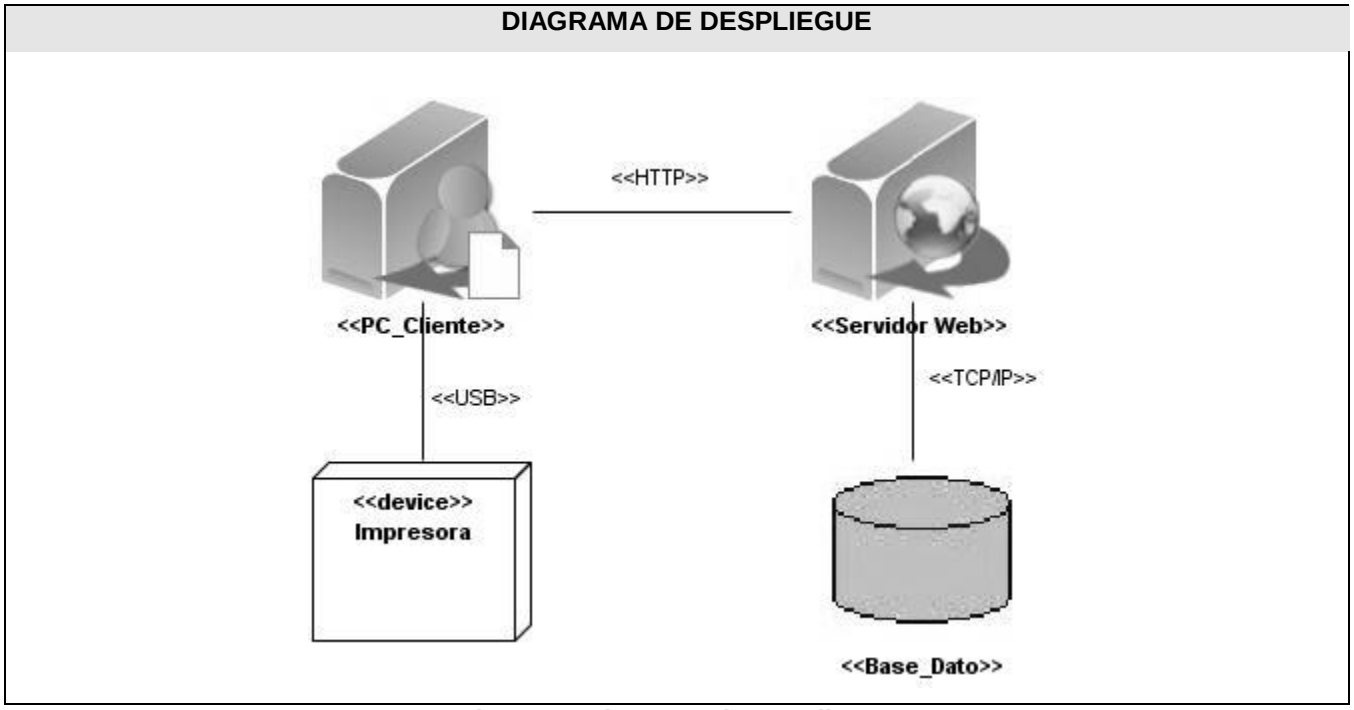


Figura 34 Diagrama de Despliegue

3.7 Definiciones de diseño que se apliquen

3.7.1 Interfaz

El diseño de la interfaz no es más que la fachada de un producto a la hora de interactuar con los usuarios donde estos a raíz de la búsqueda de algún aspecto que requieran el sistema le enviaran la respuesta correcta, donde este mantendrá colores claros y las letras de un mismo tamaño permitiendo la uniformidad del mismo.

Para el diseño de la interfaz se tuvieron en cuenta algunos aspectos fundamentales los cuales garantizan la usabilidad en los diseños para aplicaciones Web:

1. Búsqueda estable.

2. Proporcionar atajos a usuarios expertos.
3. Ofrecer información de retroalimentación.
4. Diseñar diálogos que conduzcan a una conclusión.
5. Permitir deshacer acciones fácilmente.

Por otra parte se mantendrá uniformidad presentando el menú en la parte izquierda donde aparecerán los diferentes módulos y los formularios se encontrarán centrados en las páginas permitiendo una organización en sus datos.

3.7.2 Tratamiento de errores

La eficacia del manejo de errores en cualquier sistema garantizará el buen funcionamiento de este y a su vez la información que se maneja. Para el manejo de errores se utilizará Javascript el cual es un lenguaje que es utilizado principalmente para la validación de los datos pretendiendo con este el menor número posible de errores. Además estos errores son especificados durante la interacción del usuario a través de una interfaz mostrando cual fue el error existente. También se muestran mensaje de confirmación de errores para el caso de que el usuario no requiere terminar determinada acción así como mensajes para indicar si la acción fue desarrollada con éxito o no.

3.7.3 Seguridad

La seguridad del sistema se basará en la autenticación por parte de los usuarios con determinados permisos donde todos los usuarios accederán a través de su usuario y contraseña.

3.7.4 Concepción de la ayuda

La ayuda se encontrará en todo momento en la aplicación ya que estará disponible en todas las páginas donde el usuario se encuentre. Para ello en cada página según donde se encuentre se mostrará la ayuda referente a lo que el usuario necesite como la explicación de los mensajes de error que surjan en un determinado momento donde el usuario se encuentre en interacción con la aplicación.

3.8 Conclusiones

En este capítulo se logró realizar un estudio profundo del análisis y diseño donde se describieron y se mostraron las clases del diseño de estereotipos web el cual permitirá a los desarrolladores una mejor comprensión del sistema. Además se realizaron los diagramas de secuencias por realizaciones de casos de usos destacando los más significativos, es decir las funcionalidades más importantes. Se realizó el diseño de la base de datos con sus clases persistentes y sus relaciones, así como el modelo entidad relación y el diagrama de despliegue que se pretende utilizar.

Conclusiones

El grupo de calidad de la facultad 8 necesita incorporar para su estructura y organización un sistema que contribuya al logro de una mejor organización tanto del personal como de las actividades que realiza para el desarrollo de sus objetivos. Por lo cual, con la confección de este trabajo de diploma se han obtenido una serie de artefactos que harán posible la implementación de lo que realmente se desea, es decir, un sistema para la gestión de la Calidad en la facultad 8 que facilitará:

- ✓ Una eficiente organización mediante la planificación correcta de los integrantes del proyecto.
- ✓ Realización de consultas para obtener información de los estudiantes que se encuentran trabajando en un momento determinado.
- ✓ Información acerca de los estudiantes que se hallan afectados por problemas personales o en otras actividades.
- ✓ Aumentará la interacción con los proyectos de la facultad mediante la obtención de los cronogramas de ejecución de cada uno de ellos, para verificar el cumplimiento de los hitos establecidos por los mismos velando de esta forma por la mejora de la calidad del producto.
- ✓ Información oportuna a los usuarios de la ocurrencia de determinado evento.

De esta manera se obtendrá una adecuada estructura del proyecto y mayor eficiencia en el manejo de sus datos.

Recomendaciones

A medida del avance de este trabajo se fueron cumpliendo los objetivos trazados dándole solución a los problemas existentes pero es necesario realizar algunas recomendaciones para las personas que le darán continuidad a dicho trabajo dentro de las cuales se tiene:

- ✓ Realizar un estudio y análisis mas profundo de los procesos restantes que sean necesarios en el módulo de planificación con el objetivo de ir integrando aún más las funcionalidades del sistema.
- ✓ Añadir más datos a los reportes que se generan para una mayor utilidad y eficiencia.
- ✓ Profundizar en el estudio de las tecnologías actuales para asegurar bien la utilización de las mismas una vez que se comience el desarrollo de la aplicación.
- ✓ Realizar el flujo de trabajo de implementación y prueba para validar el sistema.
- ✓ A la hora del desarrollo de la aplicación implementar la seguridad del mismo según el módulo y el rol del usuario, además a través de la autenticación permitir mostrar en el menú solamente lo necesario para el usuario autenticado.

Bibliografía

[1]Departamento de Control de Calidad y Auditoría Informática. Control de Calidad en los Sistemas, Junio 2000.

[2]Alejandra Cechich, Richard Moore. Un Modelo Formal de Patrones Orientados a Objetos.

[3]Departamento de técnicas de programación (UCI). Conferencia #1 Principios de Programación Web, 2008.

[4]Departamento de técnicas de programación (UCI). Conferencia #2 Programacion del cliente, 2008.

[5]ISO 9001. Sistemas de gestión de la calidad.

[6]José H. Canós, Patricio Letelier y Mª Carmen Penadés. Métodologías Ágiles en el Desarrollo de Software, DSIC -Universidad Politécnica de Valencia.

[7]Lovelle, J. M. C. Calidad del Software, 21 de Octubre de 1999.

[8]Moral, A. C. SERVLETS Y JSP. Universidad de Salamanca, Departamento de Informática y Automática, mayo, 2003.

[9]S.Pressman., R. Quinta edición Ingeniería de Software Un enfoque Práctico.

[10]Daniel Pecos. PostGreSql vs MySQL. [Disponible en: http://www.netpecos.org/docs/mysql_postgres/index.html]

[11]JavaHispano. 2008 [Disponible en: http://www.javahispano.org/contenidos/es/mysql_vs_postgresql_cuando_emplear_cada_una_de_ellas_11/?menuId=MINDS&onlypath=true]

[12]Desarrolloweb.com. [Disponible en: http://www.desarrolloweb.com/articulos/497.php]

[13]Fernández, J. N. D. P. R. (2005). El Patrón Fachada. Departamento de Informática y Automática. España, Universidad de Salamanca. 2007.

[14]Artola, L. Metodologías Agiles, 2007. [Disponible en: http://www.programania.net/diseno-de-software/ventajas-de-asp-application-server-provider/]

[15]Jesús López. ApuntesGestion.com, 2008. [Disponible en: http://www.apuntesgestion.com/2008/08/13/definicion-planificacion]

[16]J.M.Juran, Jesús Nicolau Medina, Mercedes Gozalbes Ballester. Juran y la planificación para la calidad, Ediciones Díaz de Santos, 1990, 299 páginas.

[17]ISO 9000. Sistemas de gestión de la calidad Conceptos y vocabulario.

[18]Graig Larman. UML y Patrones. Introducción al análisis y diseño orientado a objetos.

[19]Ivar Jacobson, Grady Booch, James Rumbaugh. El proceso Unificado de Desarrollo de Software.

[20]Rational Rose: Procedimientos básicos para desarrollar un proyecto con UML. 2006, [Disponible en: http://www.vico.org/TallerRationalRose.pdf.]

[21]Instituto Nacional de Estadística Informática. Herramientas CASE.

Referencias Bibliográficas

1.Departamento de Control de Calidad y Auditoría Informática. Control de Calidad en los Sistemas, Junio 2000.

2.S.Pressman., R., Quinta edición Ingeniería de Software Un enfoque Práctico.

3.Jesús López. ApuntesGestion.com, 2008. [Disponible en: http://www.apuntesgestion.com/2008/08/13/definicion-planificacion].

4.Lovelle, J.M.C., Calidad del Software. 21 de Octubre de 1999.

5.Manual de Gestión para jefes de servicios clínicos. .

6.Pressman, R.S., Un Enfoque Práctico. , ed. M.G. Hill. 2002.: Ingeniería de Software.

7.ISO 9000. Sistemas de gestión de la calidad Conceptos y vocabulario.

8.Métricas, Estimación y Planificación en Proyectos de Software. [Disponible en: http://www.willydev.net/InsiteCreation/v1.0/descargas/willydev_planeasoftware.pdf]

9.Chaves, M.A., La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. Julio de 2006.

10.José H. Canós, Patricio Letelier y Mª Carmen Penadés, Métodologías Ágiles en el Desarrollo de Software: DSIC -Universidad Politécnica de Valencia.

11.Sergio José Villaneda Ávila. Introducción a Microsoft Solutions Framework. 2005.

12.Departamento de técnicas de programación (UCI). Conferencia #1 Principios de Programación Web, 2008.

13.Artola, L. Metodologías Agiles, 2007. [Disponible en: http://www.programania.net/diseno-de-software/ventajas-de-asp-application-server-provider/]

14.Moral, A.C., SERVLETS Y JSP. Universidad de Salamanca: Departamento de Informática y Automática, mayo, 2003.

15.Ceyusa. Software Libre para Guanajuato. 2006.

16.Ricardo J. Vargas Del Valle, J.P.M.G., Programación por Capas.

17.Graig Larman. UML y Patrones. Introducción al análisis y diseño orientado a objetos.

Anexos

Anexo 1 Diagrama de actividades caso de uso Planificar Distribución _ Tiempo _ Máquina.

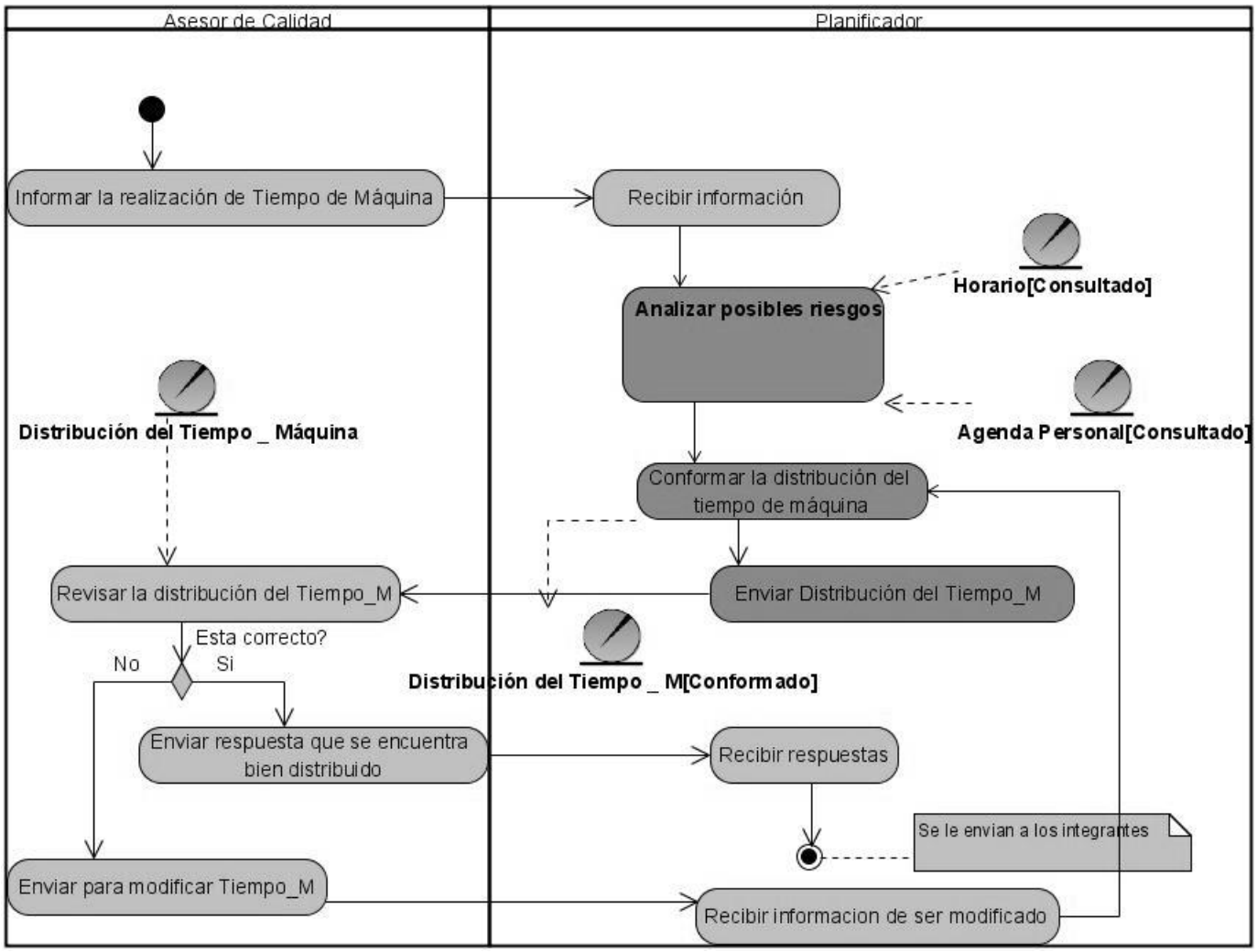


Diagrama de actividades caso de uso Planificar Capacitación.

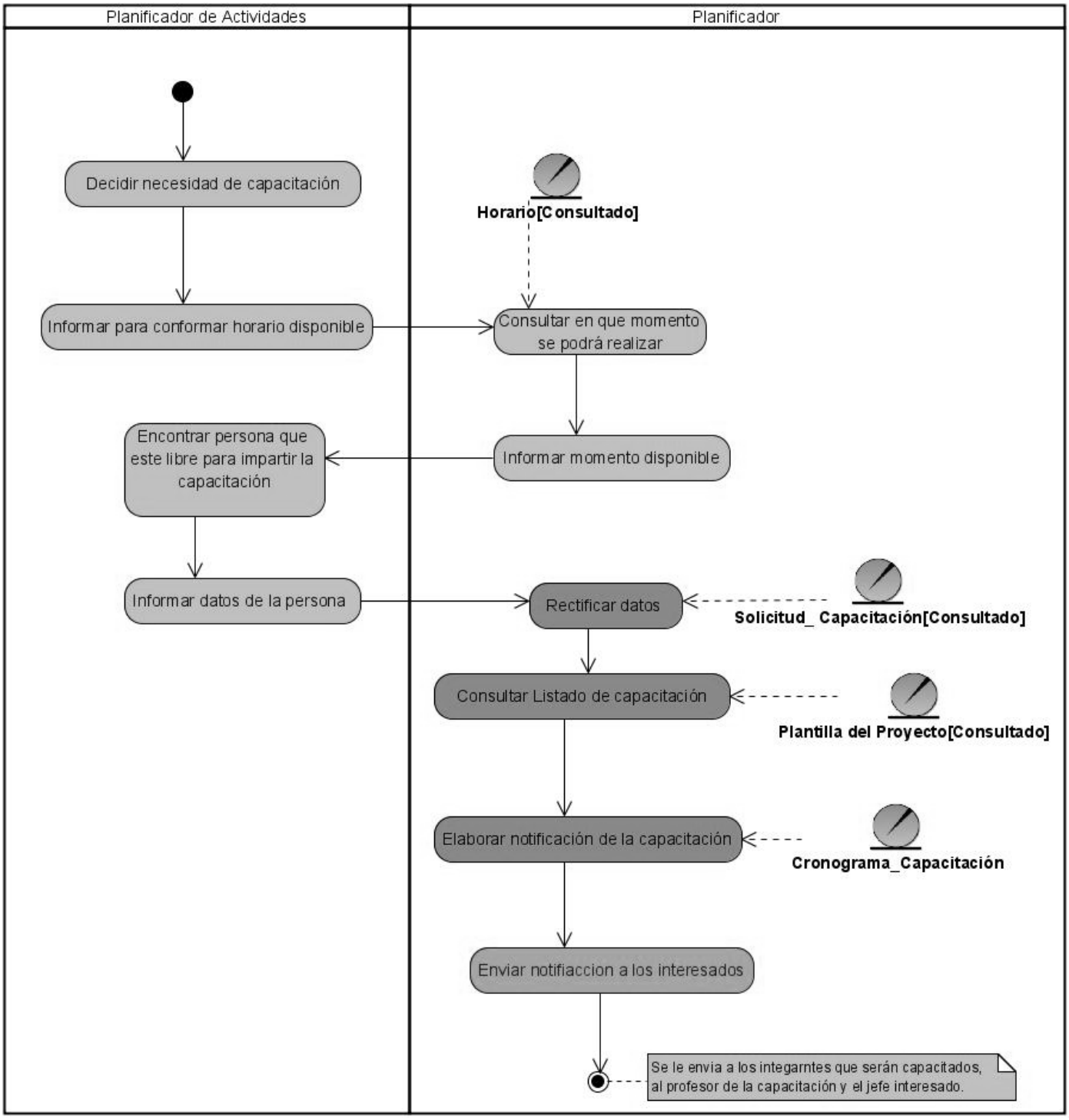
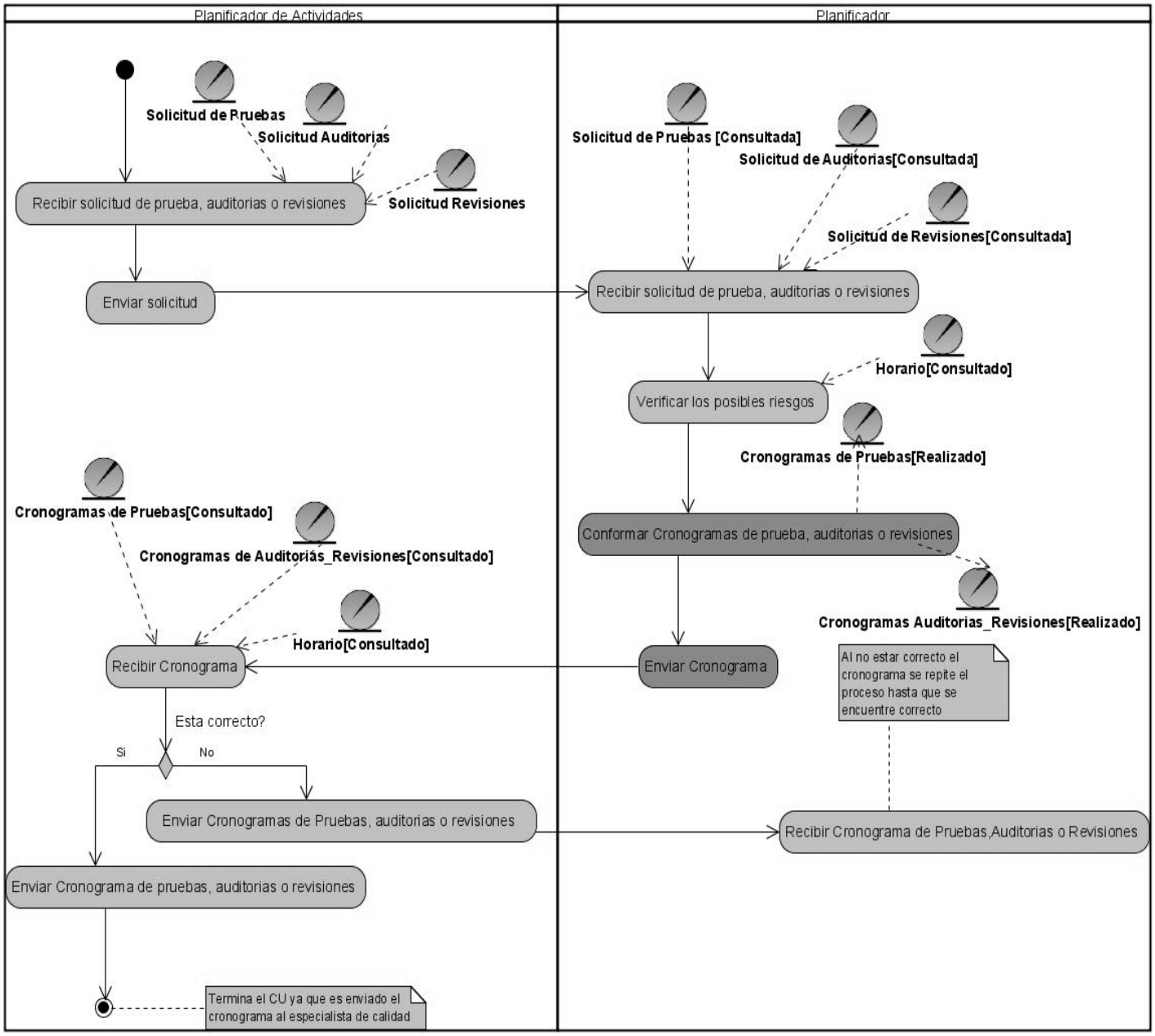

Planificador de Actividades
Planificador
Decidir necesidad de capacitación
Horario[Consultado]
Informar para conformar horario disponible
Consultar en que momento se podrá realizar
Encontrar persona que este libre para impartir la capacitación
Informar momento disponible
Informar datos de la persona
Rectificar datos
Solicitud_ Capacitación[Consultado]
Consultar Listado de capacitación
Plantilla del Proyecto[Consultado]
Elaborar notificación de la capacitación
Cronograma_Capacitación
Enviar notifiaccion a los interesados
Se le envia a los integarntes que serán capacitados, al profesor de la capacitación y el jefe interesado.

Diagrama de actividades caso de uso Planificar actividades del grupo de calidad.



Anexo 2 Descripciones textuales de casos de usos arquitectónicamente significativos.

Caso de Uso	Gestionar Integrantes del Proyecto calidad.
Actores:	Planificador
Resumen:	El caso de uso se inicia cuando el Planificador decide verificar la plantilla del proyecto donde puede realizar diferentes acciones como insertar, modificar, mostrar, eliminar. En caso de que seleccione la opción de insertar, el sistema dará la posibilidad de incluir los datos que se necesiten para llenar o actualizar en esta plantilla. Si el actor elige la opción de modificar le permitirá realizar la modificación y una vez realizados los cambios, guardará las modificaciones. El sistema permite ver una vista previa, con posibilidad de exportar a pdf para poder imprimir y donde luego podrá enviar el documento, terminando así el caso de uso.
Precondiciones:	Debe haberse generado el escritorio de trabajo del usuario autenticado. Según la acción de insertar, mostrar, modificar o eliminar a realizar debe estar seleccionada la acción previamente que se quiere.
Referencias	RF 1
Prioridad	Alta

Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. El caso de uso se inicia cuando el actor selecciona la opción de realizar una acción sobre la plantilla del proyecto.	
	2. Este brinda la posibilidad de realizar las acciones: • Insertar nuevo dato de Integrantes. • Modificar datos de Integrantes. Ver sección 1 “Modificar datos de Integrantes”. • Eliminar datos de Integrantes. Ver sección 2

	"Eliminar datos de Integrantes". • Mostrar datos de Integrantes. Ver sección 3 "Mostrar datos de Integrantes".
3. Selecciona la opción de insertar un nuevo dato en la plantilla.	
	4. Permite incluir los datos siguientes: • Nombre y Apellidos • Teléfono. • Apto. • Usuario. • Año. • Rol. • Grupo. Y permite: • Guardar datos de integrantes. • Cancelar datos de integrantes.
5. Selecciona la opción de guardar datos.	
	6. Muestra un mensaje de información "Se ha insertado un nuevo elemento." 7. El caso de uso termina.
Prototipo de Interfaz	
Flujo Alterno 5.a "El actor decide cancelar la operación"	
Acción del Actor	**Respuesta del Sistema**
	5.a.1 Elimina los datos creados. 5.a.2 Muestra un mensaje de información "Se ha cancelado la operación". 5.a.3 Regresa a la vista anterior. 5.a.4 El caso de uso termina.
Sección 1 "**Modificar datos de la plantilla".**	

Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de modificar los datos de integrantes.	
	2. Muestra los datos de la plantilla y brinda la posibilidad de cambiar sus valores ya sea introduciendo nuevos o seleccionando nuevos datos. Permite además: • Guardar datos • Cancelar la operación de la plantilla.
3. Modifica los datos que considere y selecciona opción de guardar.	
	4. Muestra un mensaje de información “Se han modificado los datos correctamente.” 5. Muestra los datos modificados. 6. El caso de uso termina.
Prototipo de Interfaz.	
Flujo Alterno 3.a “El actor decide cancelar la operación”	
Acción del Actor	**Respuesta del Sistema**
	3.a.1 Elimina los datos creados. 3.a.2 Muestra un mensaje de información “Se ha cancelado la operación”. 3.a.3 Regresa a la vista anterior. 3.a.4 El caso de uso termina.
Sección 2* “Eliminar datos de la plantilla”*.	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de eliminar algún integrante o dato.	

	2. Muestra el o los elementos a eliminar. 3. Muestra un mensaje de información "Se eliminarán los elementos seleccionados. ¿Desea continuar?". Y permite: • Eliminar elementos seleccionados. • Cancelar operación.
4. Selecciona la opción eliminar elementos seleccionados.	
	5. Muestra un mensaje de información "Se han eliminado los elementos seleccionados" 6. Regresa a la vista anterior. 7. El caso de uso termina.
Prototipo de Interfaz.	
Flujo Alterno 4.a "El actor decide cancelar la operación"	
Acción del Actor	**Respuesta del Sistema**
	4.a.1 Elimina los datos creados. 4.a.2 Muestra un mensaje de información "Se ha cancelado la operación". 4.a.3 Regresa a la vista anterior. 4.a.4 El caso de uso termina.
***Sección 3* " Mostrar datos de integrantes"**	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de mostrar datos.	
	2. Muestra los datos. Y permite: • Exportarlos a PDF. • Salir de la vista actual.

3. Selecciona la opción de salir de la vista actual.	
	4. Muestra la vista anterior. 5. El caso de uso termina.
Prototipo de Interfaz.	
Flujo Alterno 3.a "El actor decide Imprimir o exportar a pdf"	
Acción del Actor	**Respuesta del Sistema**
	3.a.1 Realiza la operación de exportar a pdf. 3.a.2 Accede a al opción de imprimir. 3.a.3 Muestra un mensaje "Se encuentra realizando la operación " 3.a.4 El caso de uso termina.
Poscondiciones	-

Caso de Uso	Adicionar Cronogramas de Proyectos
Actores:	Usuarios externos
Resumen:	El caso de uso se inicia cuando los usuarios externos necesitan insertar el cronograma del proyecto para que se puedan realizar otras operaciones.
Precondiciones:	Debe haberse generado el escritorio de trabajo del usuario autenticado. Según la acción debe estar seleccionado previamente lo que se quiere.
Referencias	RF 3
Prioridad	Alta
Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. El caso de uso se inicia cuando el actor selecciona la opción de realizar una acción sobre el cronograma de proyectos.	

	2. Este brinda la posibilidad de subir un archivo de tipo excel/project. Y permite: • Guardar los datos. • Cancelar la operación.
3. Selecciona la opción de guardar datos.	
	4. Muestra un mensaje de información “Se han guardado los datos correctamente”. 5. El caso de uso termina.
Prototipo de Interfaz	
Flujo Alterno 3.a “El actor decide cancelar la operación”	
Acción del Actor	**Respuesta del Sistema**
	3.a.1 Elimina los datos creados. 3.a.2 Muestra un mensaje de información “Se ha cancelado la operación”. 3.a.3 Regresa a la vista anterior. 3.a.4 El caso de uso termina.
Poscondiciones	-

Caso de Uso	Consultar Cronogramas Proyectos
Actores:	Planificador
Resumen:	El caso de uso inicia cuando el Planificador decide consultar el listado de cronogramas de los proyectos, permitiendo buscar un determinado cronograma de proyecto y mostrar el que se desee para realizar otras operaciones.
Precondiciones:	Debe haberse generado el escritorio de trabajo del usuario autenticado.
Referencias	RF3, RF 4
Prioridad	Alta

Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. El caso de uso inicia cuando accede a la opción consultar cronogramas de proyectos.	
	2. Brinda la posibilidad de seleccionar los datos de la búsqueda. Y permite: • Realizar una búsqueda a partir del dato seleccionado. • Mostrar datos. Ver sección 1 “Mostrar datos” • Cancelar la operación.
3. Realizar la búsqueda según el dato seleccionando.	
	4. Realiza la búqueda según el dato: • Proyecto. Y permite: • Mostrar los datos. Ver sección 1 “Mostrar datos” 5. El caso de uso termina.
Prototipo de Interfaz	
Flujo Alterno 3.a “El actor decide cancelar la operación”	
Acción del Actor	**Respuesta del Sistema**
	3.a.1 Elimina los datos creados. 3.a.2 Muestra un mensaje de información “Se ha cancelado la operación”. 3.a.3 Regresa a la vista anterior. 3.a.4 El caso de uso termina.
***Sección 1* “ Mostrar datos ”**	
Acción del Actor	**Respuesta del Sistema**

1. Selecciona la opción de mostrar los datos.	
	2. Muestra los datos. Y permite: • Exportar a pdf. • Salir de la vista actual.
3. Selecciona la opción de salir de la vista actual.	
	4. Muestra la vista anterior. 5. El caso de uso termina.
Prototipo de Interfaz	
Flujo Alterno 3.a "El actor decide Imprimir o exportar a pdf"	
Actor del Sistema	***Respuesta del Sistema***
	3.a.1 Realiza la operación de exportar a pdf. 3.a.2 Accede a al opción de imprimir. 3.a.3 Muestra un mensaje "Se encuentra realizando la operación " 3.a.4 El caso de uso termina.
Poscondiciones	-

Caso de Uso	Consultar datos Agenda Personal
Actores:	Planificador
Resumen:	El caso de uso se inicia cuando el Planificador luego de elaborar la distribución del tiempo de máquina del proyecto surgen afectaciones por parte de los involucrados en dicha distribución donde se le brinda la posibilidad de modificar su tiempo de máquina siempre que sea posible y además registrar sus afectaciones.
Precondiciones:	Debe haberse generado el escritorio de trabajo del usuario autenticado. Según la acción de mostrar, eliminar o actualizar a realizar debe estar

	seleccionado previamente lo que se quiere.
Referencias	RF7, RF 8
Prioridad	Alta

Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. El caso de uso se inicia cuando el actor selecciona la opción de ver las afectaciones de la agenda personal.	
	2. Este brinda la posibilidad de realizar las acciones: • Actualizar datos de la agenda. • Mostrar datos de la agenda. Ver sección 1 “Mostrar datos de la agenda”. • Eliminar datos de la agenda. Ver sección 2 “Eliminar datos de la agenda”.
3. Selecciona la opción de actualizar la agenda.	
	4. Brinda la posibilidad de realizar modificaciones en los siguientes datos: • Dia de la semana. • Turno. • Nombre y Apellidos. • No PC. • Para ello Ver CU-Gestionar Distribución Tiempo de máquina. 5. Termina el caso de uso.
6. Selecciona la opción de actualizar la agenda.	
	7. Brinda la posibilidad de realizar

	modificaciones en los siguientes datos: • Día de la semana. • Turno. • Nombre y Apellidos. • No PC. • Para ello Ver CU-Gestionar Distribución Tiempo de máquina. 8. Termina el caso de uso.
Prototipo de Interfaz	
Sección 1 "Mostrar datos de la agenda".	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de mostrar datos de la agenda.	
	2. Muestra los datos de la agenda: • Nombre y Apellidos. • Día de la afectación. • Turno. • Descripción de la afectación. Y permite: • Regresar a la vista anterior.
3. Selecciona ir a la vista anterior.	
	4. Muestra la vista anterior. 5. Termina el caso de uso.
Sección 2 "Eliminar datos de la agenda".	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de eliminar datos de la agenda.	
	2. Muestra el o los elementos a eliminar.

	3. Muestra un mensaje de información “Se eliminarán los elementos seleccionados. ¿Desea continuar?”. Y permite: Eliminar elementos seleccionados. 4. Termina el caso de uso.
Prototipo de Interfaz.	
Poscondiciones	-

Caso de Uso	Gestionar Petición Recursos
Actores:	Planificador
Resumen:	El caso de uso se inicia cuando el Planificador decide verificar la petición de los recursos que necesita donde puede realizar diferentes acciones como insertar, modificar, eliminar, enviar. En caso de que seleccione la opción de insertar, el sistema dará la posibilidad de incluir los datos que se necesiten. Si el actor elige la opción de modificar le permitirá realizar la modificación y una vez realizados los cambios, guardará las modificaciones, terminando así el caso de uso.
Precondiciones:	Debe haberse generado el escritorio de trabajo del usuario autenticado. Según la acción de insertar, modificar a realizar debe estar seleccionado previamente lo que se quiere.
Referencias	RF 13
Prioridad	Media
Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**
1. El caso de uso se inicia cuando el actor selecciona la opción de realizar una acción en petición de	

recursos.	
	2. Este brinda la posibilidad de realizar las acciones: • Insertar nuevo dato. • Modificar datos. Ver sección 1 “Modificar datos”. • Eliminar datos de petición. • Enviar datos. Ver sección 2”Enviar datos”.
3. Selecciona la opción de insertar un nuevo dato.	
	4. Permite insertar los datos siguientes: • Cantidad Recursos humanos. • Cantidad Recursos tecnológicos. • Dirección de la aplicación. • Nombre del proyecto. • Correo del líder de calidad. Y permite: • Guardar datos. • Cancelar datos.
5. Selecciona la opción de guardar datos.	
	6. Muestra un mensaje de información “Se han insertado nuevos elementos.” 7. El caso de uso termina.
Prototipo de Interfaz	
Flujo Alterno 5.a “El actor decide cancelar la operación”	
Acción del Actor	**Respuesta del Sistema**
	5.a.1 Elimina los datos creados. 5.a.2 Muestra un mensaje de información “Se ha

	cancelado la operación". 5.a.3 Regresa a la vista anterior. 5.a.4 El caso de uso termina.
Sección 1 "**Modificar datos".**	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción de modificar los datos de la petición de recursos.	
	2. Muestra los datos de la petición y brinda la posibilidad de cambiar sus datos: Permite además: • Guardar datos. • Cancelar la operación.
Prototipo de Interfaz.	
Flujo Alterno 2.a "El actor decide cancelar la operación"	
Acción del Actor	**Respuesta del Sistema**
	2.a.1 Elimina los datos creados. 2.a.2 Muestra un mensaje de información "Se ha cancelado la operación". 2.a.3 Regresa a la vista anterior. 2.a.4 El caso de uso termina.
Sección 2 "Enviar datos".	
Acción del Actor	**Respuesta del Sistema**
	1. Permite enviar la solicitud de petición según el correo del líder. 2. Regresa a la vista anterior. 3. Termina el caso de uso.
Poscondiciones	-

Anexo 3 Diagramas de secuencias por realización de casos de usos.

Diagrama de secuencia Consultar Agenda Personal.

DIAGRAMA DE SECUENCIA CONSULTAR AGENDA PERSONAL

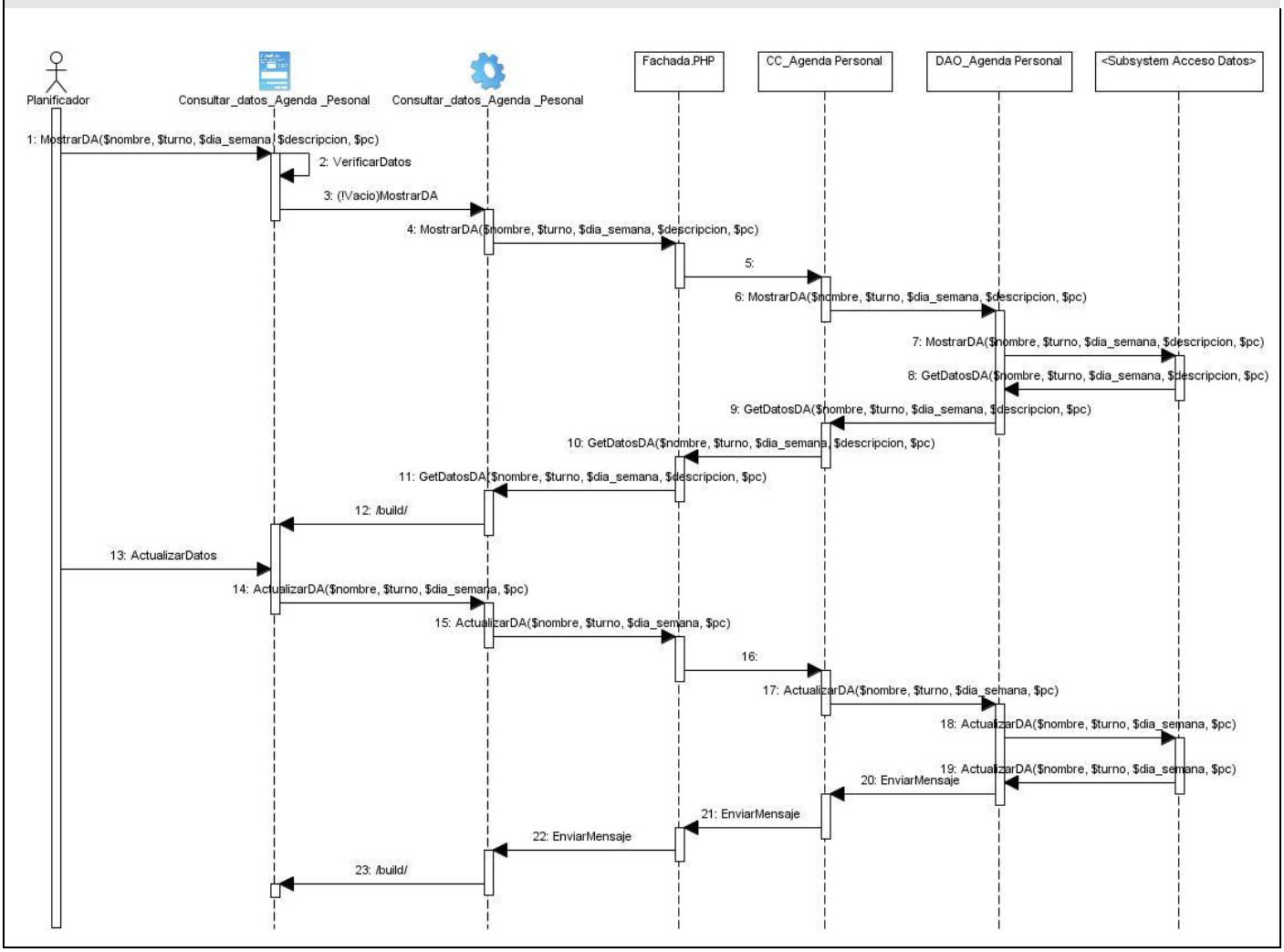


Diagrama de secuencia Gestionar Petición Recursos.

DIAGRAMA DE SECUENCIA INSERTAR PETICION DE RECURSOS

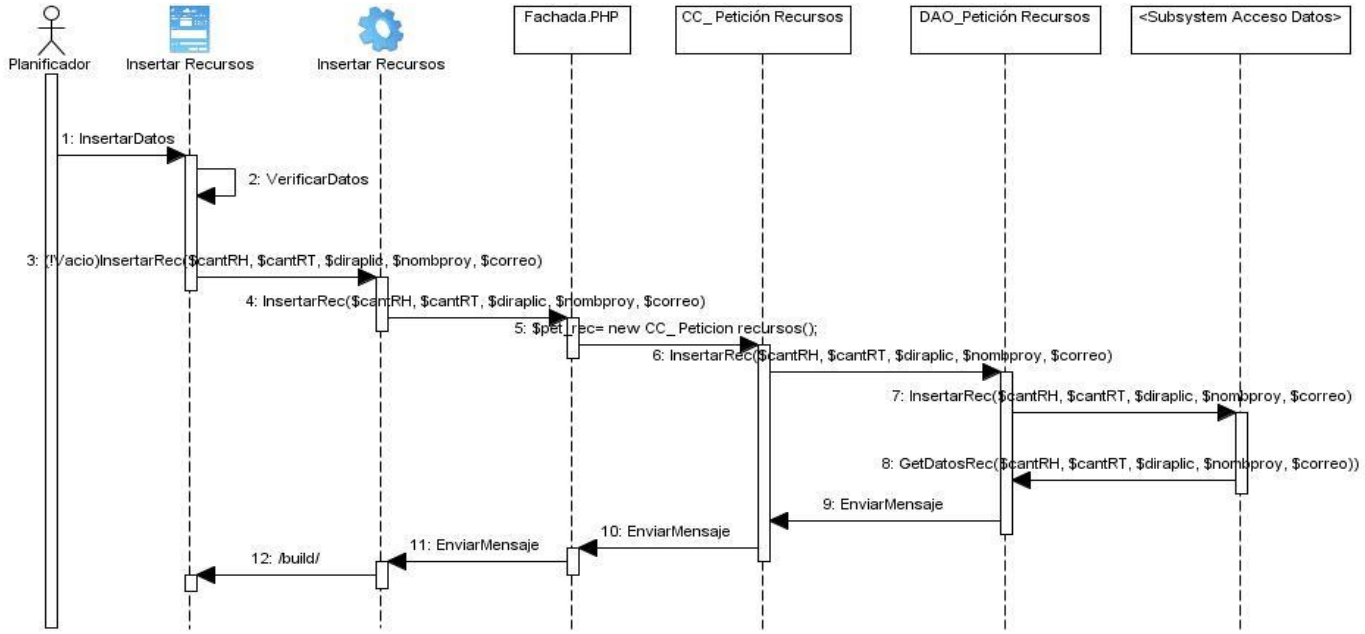


DIAGRAMA DE SECUENCIA MODIFICAR PETICION DE RECURSOS

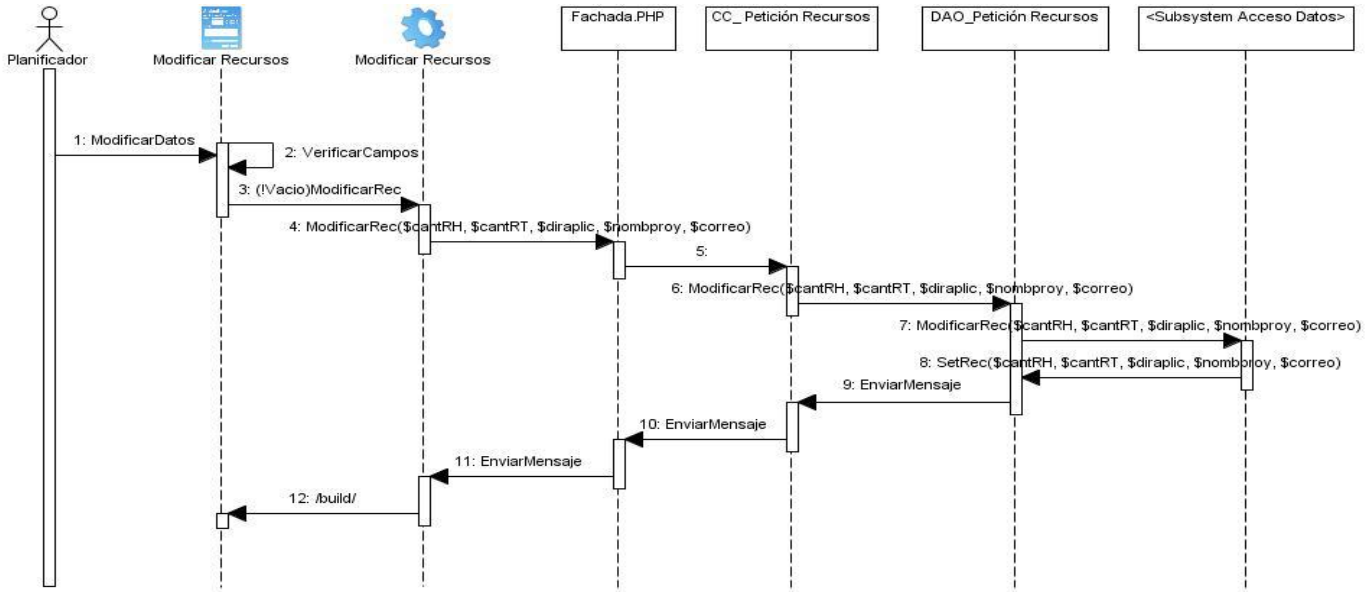


DIAGRAMA DE SECUENCIA MOSTRAR PETICION DE RECURSOS

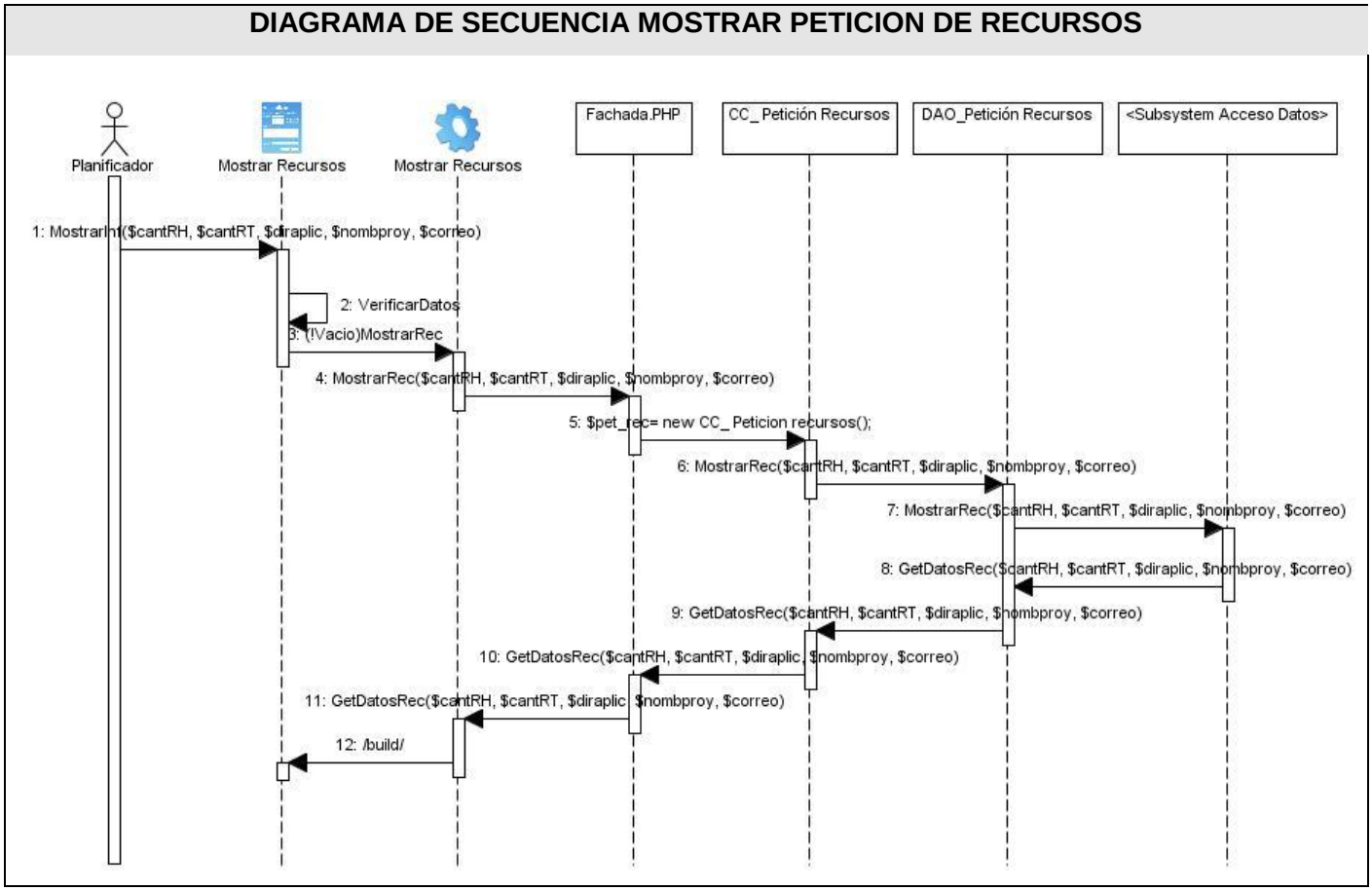


Diagrama de secuencia Gestionar Información Proyectos.

DIAGRAMA DE SECUENCIA INSERTAR INFORMACION PROYECTOS

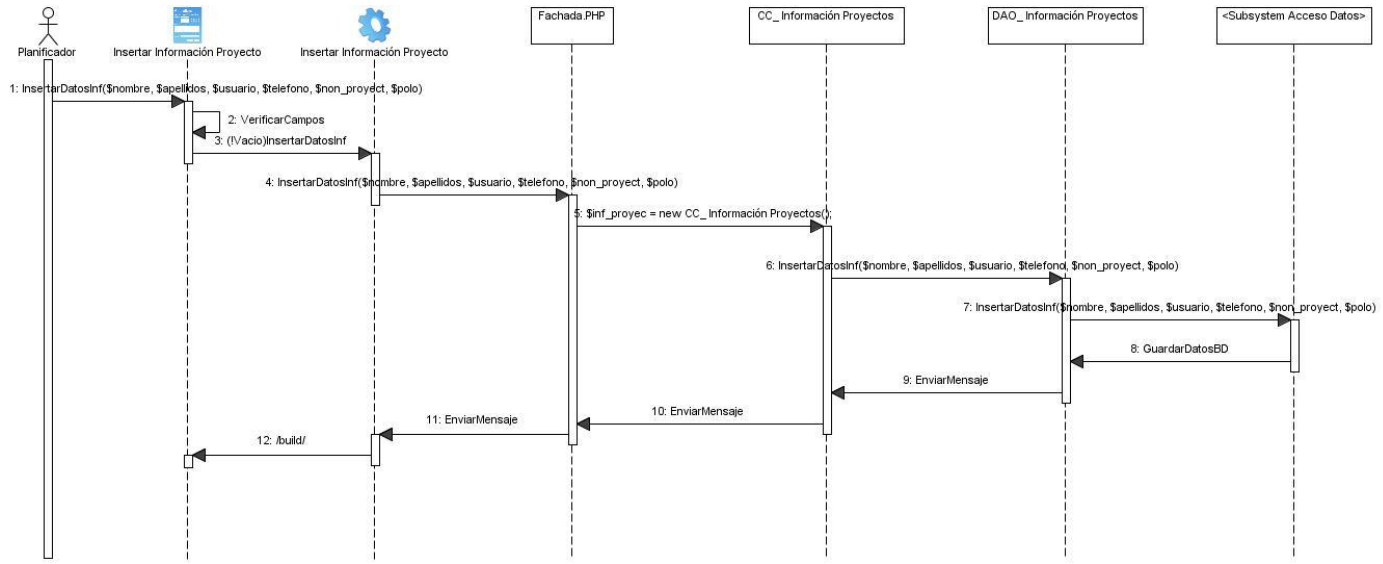


DIAGRAMA DE SECUENCIA MODIFICAR INFORMACION PROYECTOS

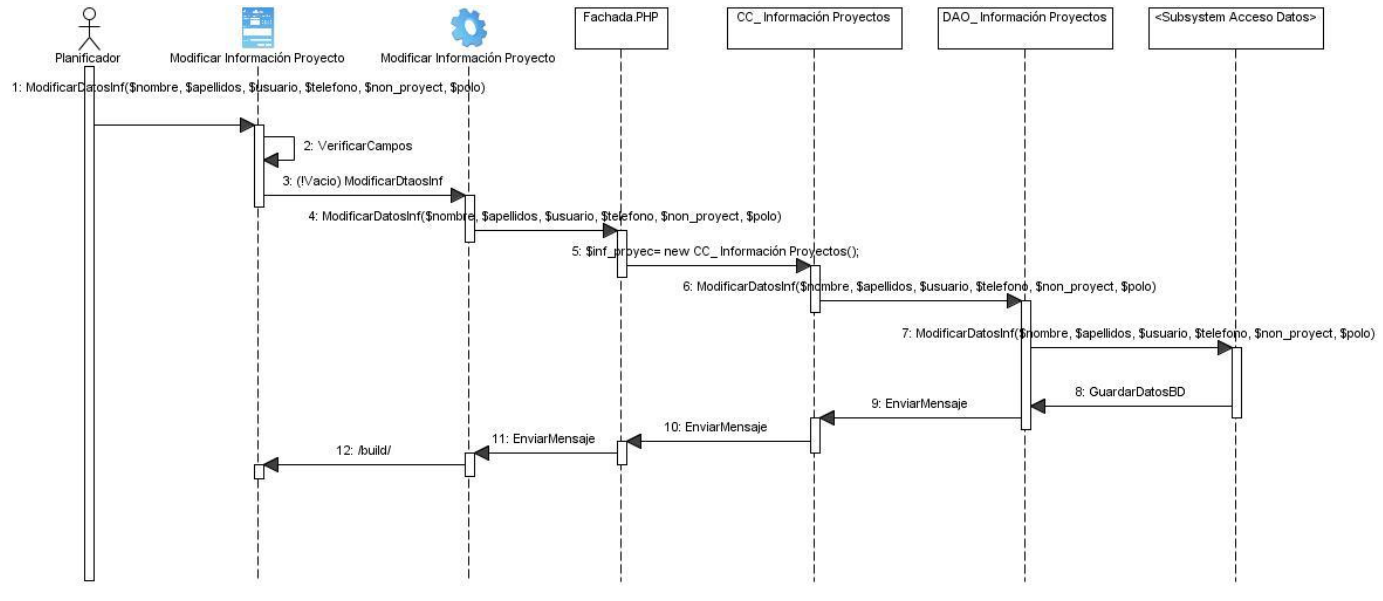


DIAGRAMA DE SECUENCIA MOSTRAR INFORMACION PROYECTOS

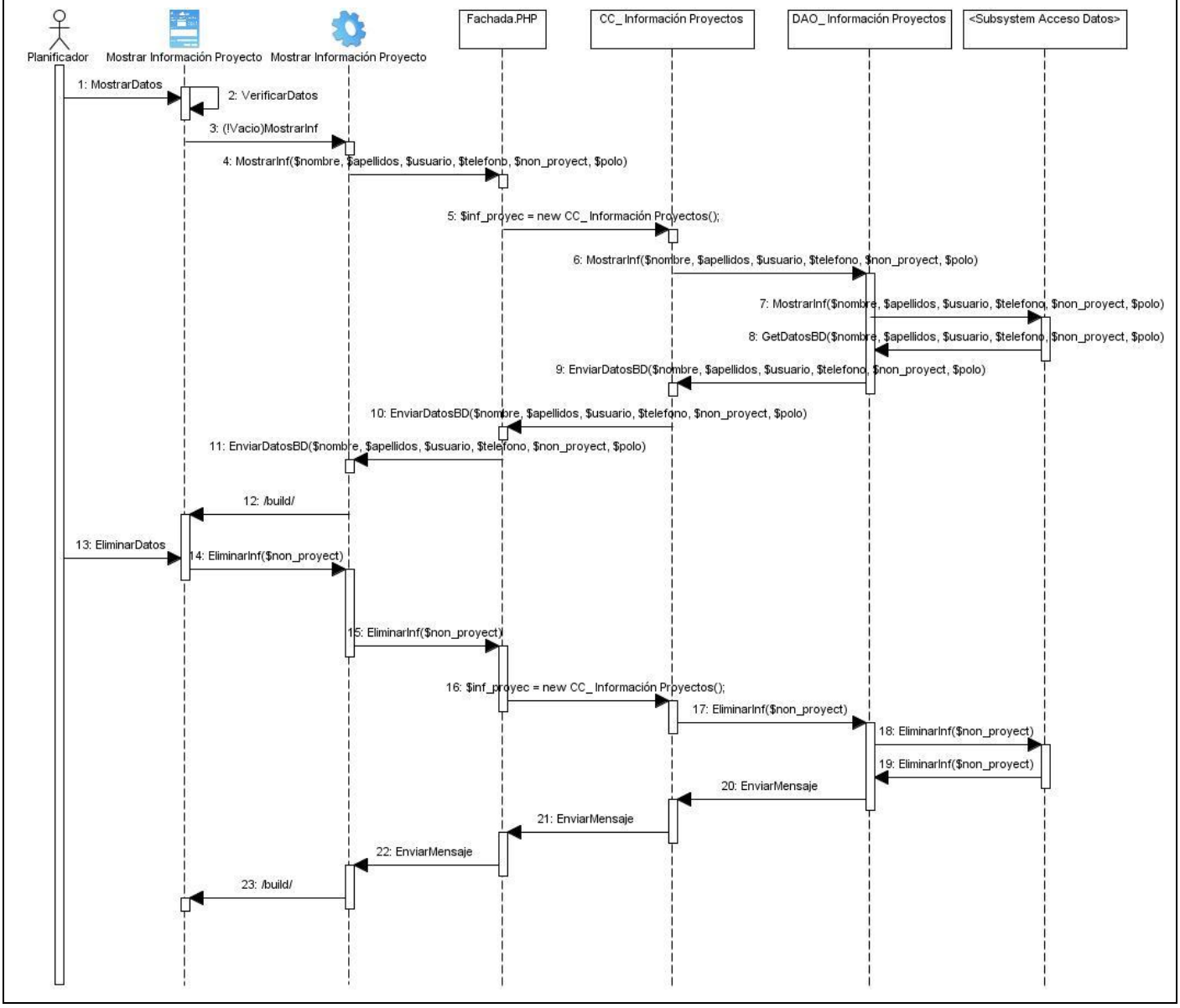


Diagrama de secuencia Gestionar Cronogramas de Pruebas.

DIAGRAMA DE SECUENCIA INSERTAR CRONOGRAMA PRUEBAS

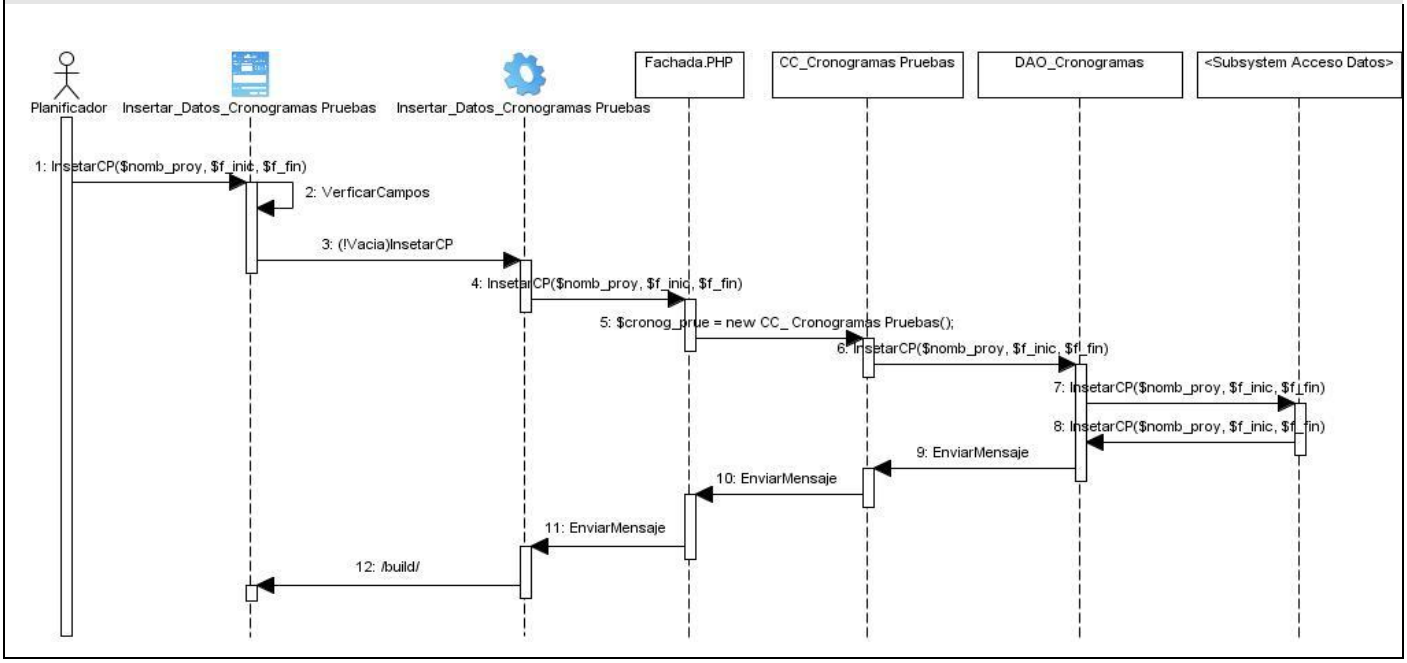


DIAGRAMA DE SECUENCIA MODIFICAR CRONOGRAMA PRUEBAS

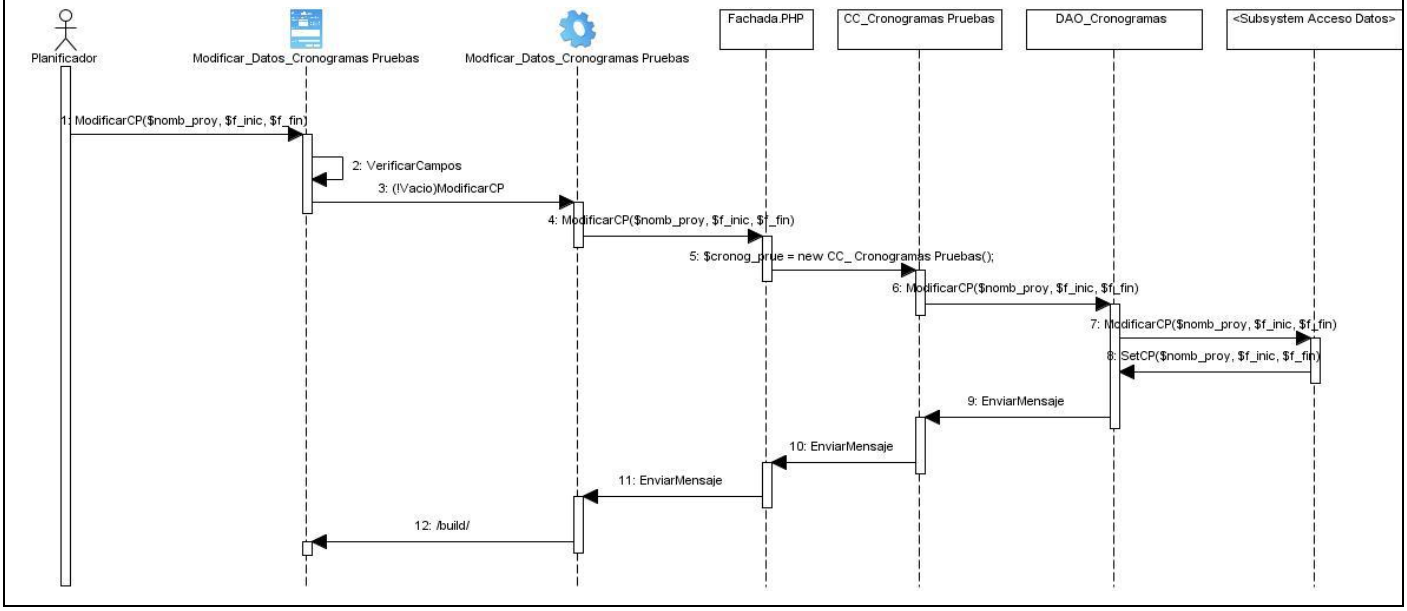
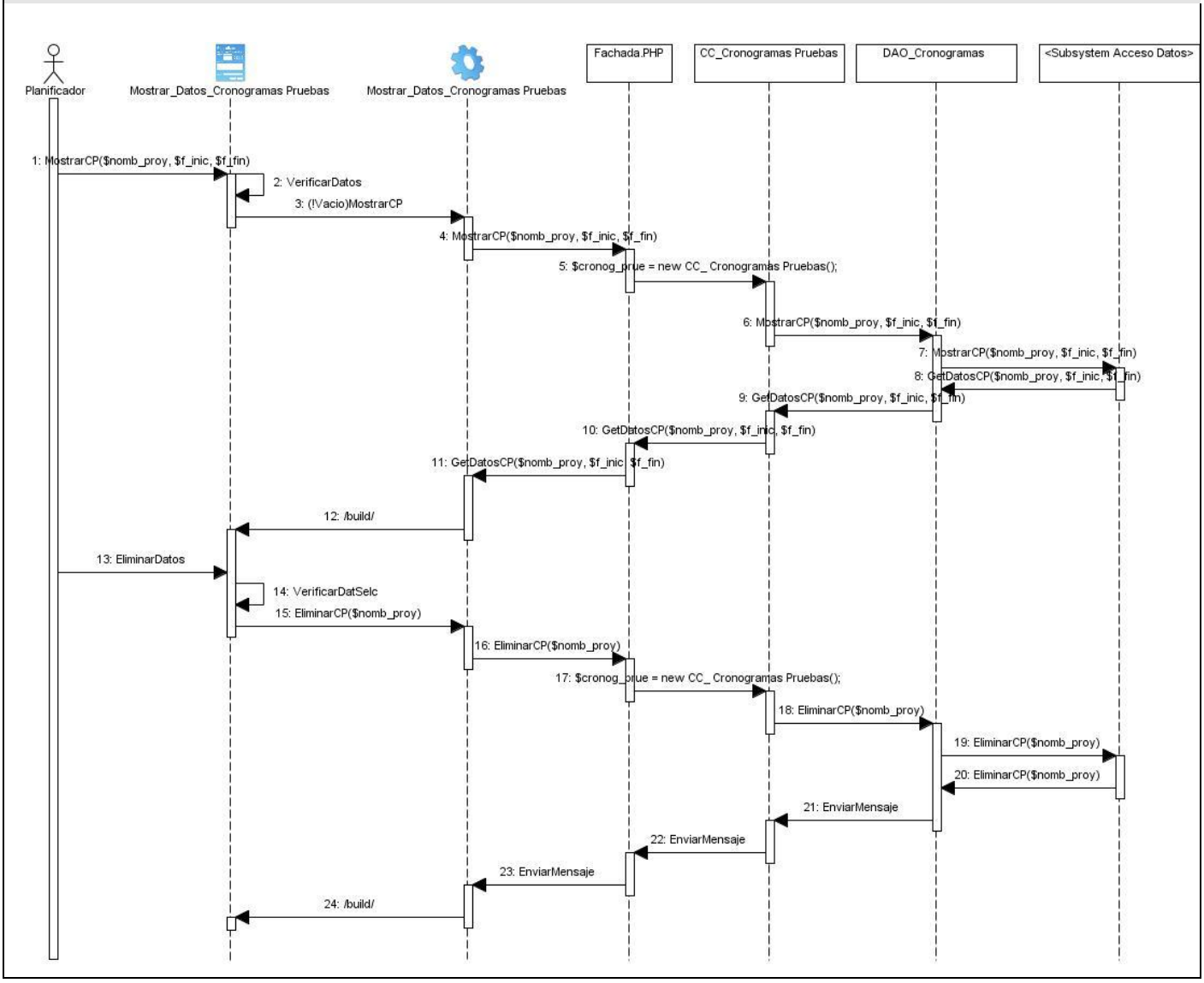


DIAGRAMA DE SECUENCIA MOSTRAR CRONOGRAMA PRUEBAS



Anexo 4 Diagrama de clases de estereotipos web significativos.

Diagrama de clases web Gestionar Petición Recursos.

DIAGRAMA DE CLASES WEB GESTIONAR PETICION DE RECURSOS

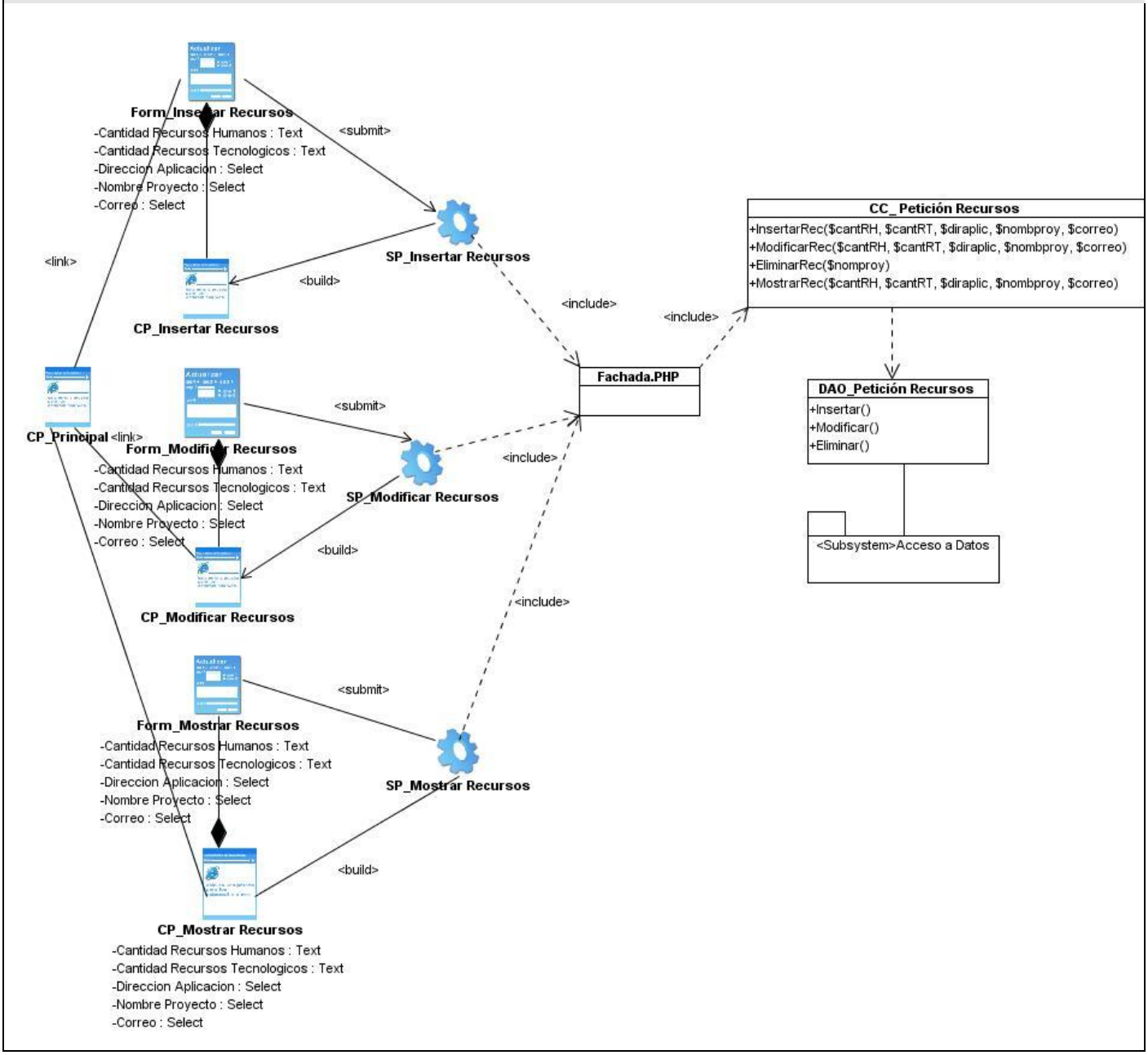


Diagrama de clases web Gestionar Información Proyectos.

DIAGRAMA DE CLASES WEB GESTIONAR INFORMACION PROYECTOS

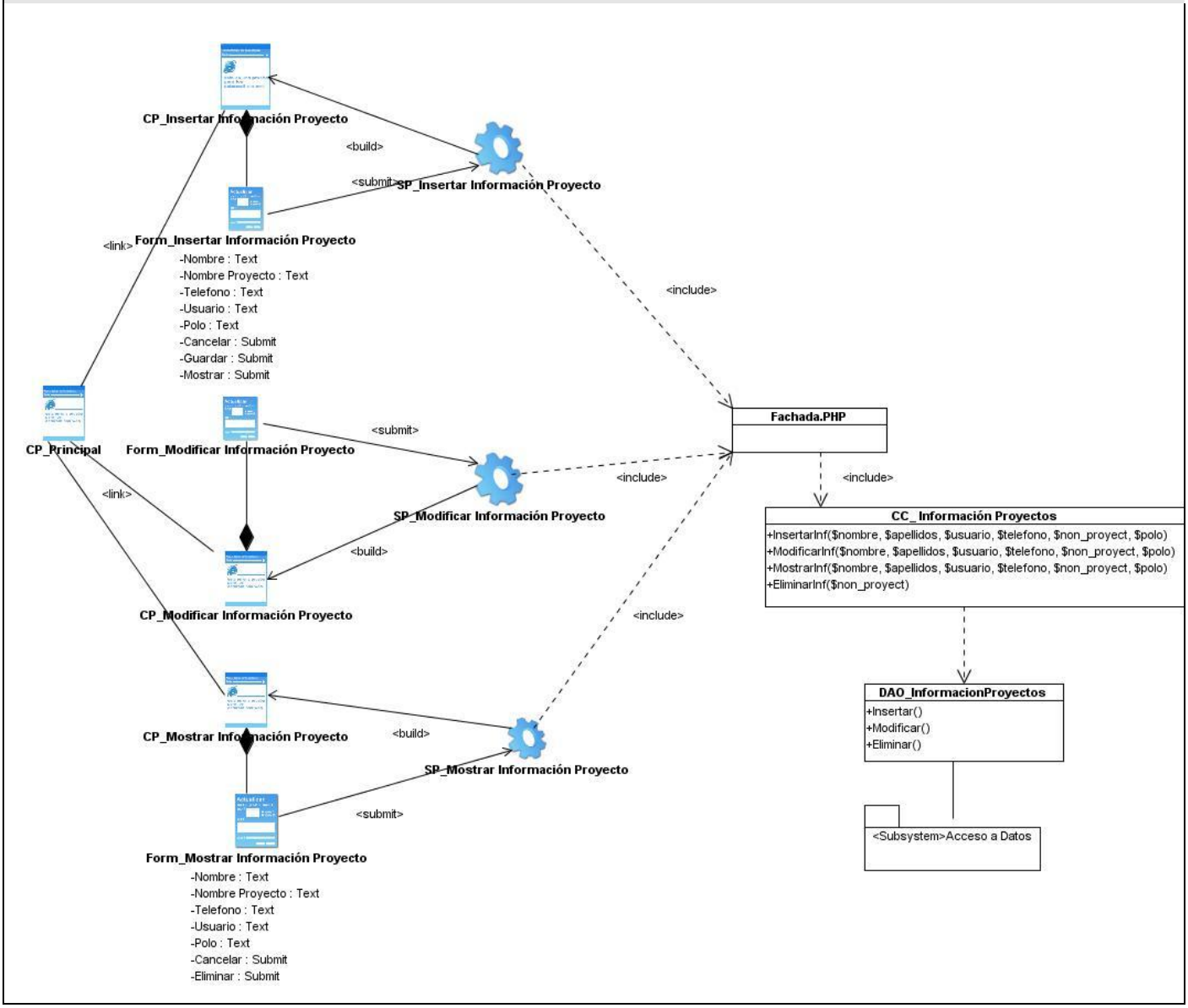
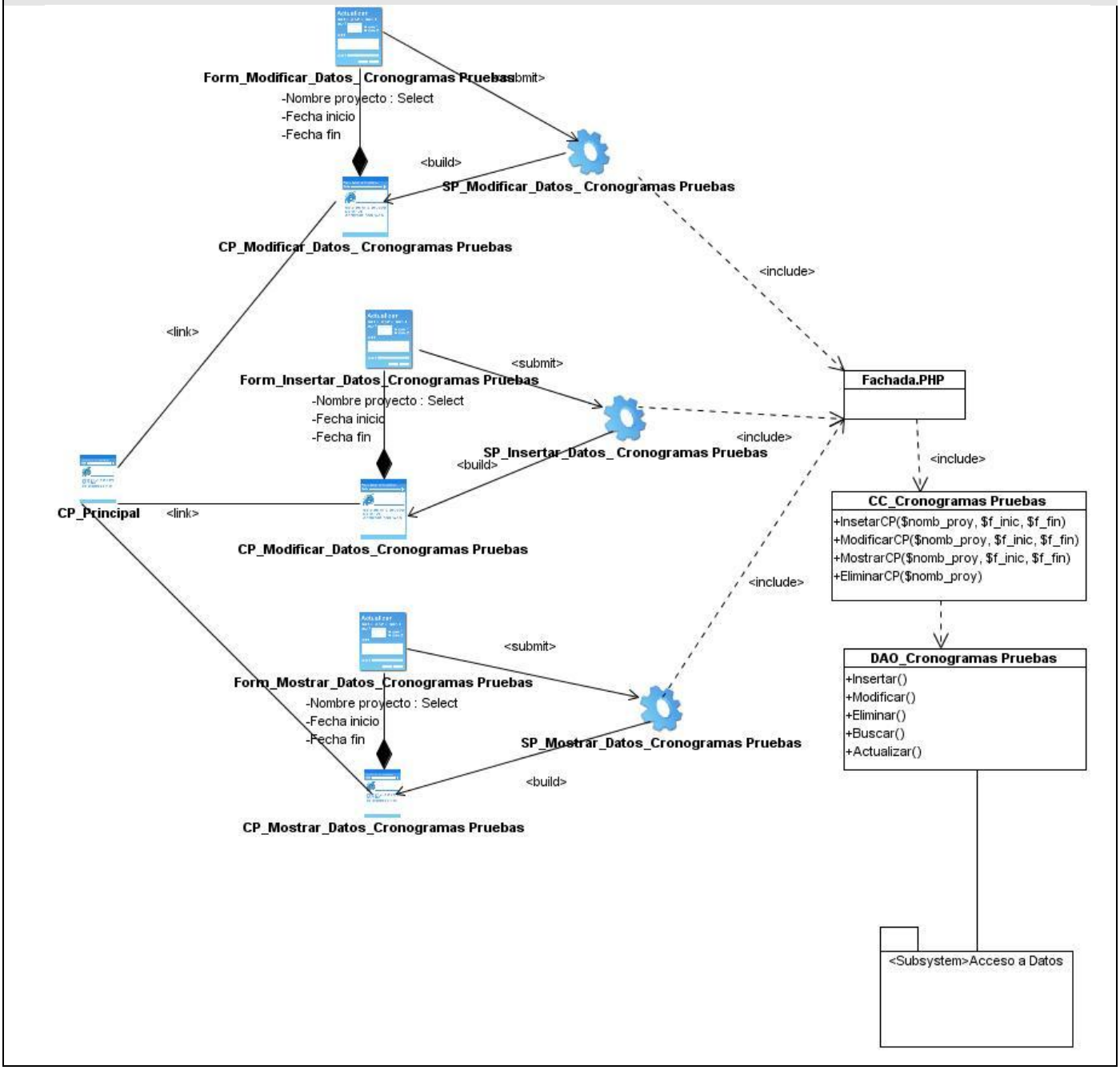


Diagrama de clases web Gestionar cronograma de pruebas.

DIAGRAMA DE CLASES WEB GESTIONAR CRONOGRAMA DE PRUEBAS



Anexo 5 Otros patrones estudiados.

Programación por capas.

Existen innumerables prácticas de programación, dependiendo del tipo de software a desarrollar; una de las más utilizadas es la programación por capas la cual divide el código fuente según su funcionalidad.

Esta se encuentra dividida en n-capas pero generalmente se conoce como tres capas: la capa de presentación o frontera, la capa de lógica de negocio o control, y la capa de datos.

De esta manera se asegura de que exista una programación organizada en su proyecto beneficiándose mediante esta en la reducción de costos por tiempos de manera que se podrá avanzar de forma más segura en su desarrollo. Además presenta facilidad a la hora de realizar actualizaciones donde se podrán realizar diferentes tareas en partes específicas para cada capa del proyecto al igual que pueden ampliarse según las necesidades que surjan.

Las capas se encuentran estructuradas de la siguiente manera:

Capa de Presentación o Frontera: Las interfaces deben ser consistentes con la información que se requiere, no se deben utilizar más campos de los necesarios, así como la información requerida tiene que ser especificada de manera clara y concisa, no debe haber más que lo necesario en cada formulario y por último, las interfaces deben satisfacer los requerimientos del usuario, por lo cual no se debe excluir información solicitada por el usuario final y no se debe incluir información no solicitada por el mismo. Esta capa se comunica únicamente con la capa de Reglas de Negocio o Control.

Lógica de Negocio o Control: Es llamada capa de reglas de negocio porque en esta se definen todas las reglas que se deben cumplir para una correcta ejecución del programa. Recibe los datos que ingresó el usuario del sistema mediante la capa de presentación, luego los procesa y crea objetos según lo que se necesite hacer con estos datos.

Capa de Datos: Es en esta capa donde se definen las consultas a realizar en la base de datos, tanto las consultas simples como las consultas complejas para la generación de reportes más específicos. Esta

capa envía la información directamente a la capa de lógica de negocio para que sea procesada e ingresada en objetos según se necesite.[16]

Se pueden ver las ventajas que se obtienen de esta arquitectura donde según Pressman escribe que al implementar este modelo de programación, se asegura un trabajo de forma ordenada y separada, debido a que sigue el principio de “divide y vencerás”.

Al igual se pueden observar desventajas que surgen sino se tienen un balance entre el número de capas y subcapas que componen el programa. Trayendo este pérdida de eficiencia, realización de trabajo innecesario o redundante entre capas, gasto de espacio de la aplicación debido a la expansión de las capas, o bien una alta dependencia entre los objetos y capas que contradice el objetivo principal del modelo.

Glosario de Términos

Auditorías: Visión formal y sistemática para determinar hasta qué punto una organización está cumpliendo sus objetivos establecidos por la gerencia, así como para identificar los que requieran mejorarse.

ASP: Páginas activas del servidor.

Applets: Son programas escritos en Java que sirven para dar vida a las páginas web.

CASE: Significado de sus siglas en inglés Ingeniería de Software Asistida por Computadora.

CSS: Significado de sus siglas en inglés Hojas de Estilos en Cascada.

Eclipse: Plataforma de desarrollo open source basada en Java.

HTML: Significado de sus siglas en inglés Lenguaje De Etiquetación De Hipertexto.

Incertidumbre: Expresión del grado de desconocimiento de una condición futura.

Java: El Entorno de Programación Java es un proyecto de la compañía Sun Microsystems que presenta una posible aproximación a lo que serán los lenguajes de programación. Sus objetivos se resumen en dotar de herramientas de programación adecuadas para desarrollar aplicaciones que se ejecuten en entornos heterogéneos, conectados a través de redes de comunicaciones. Estas aplicaciones deben ser eficientes, podrán crecer dinámicamente, en función de las necesidades, y serán operativas en una gran variedad de ordenadores.

JSP: Significado de las siglas en inglés Paginas de servidor java.

Metodología: Es estudiar los métodos para luego determinar cuál es la más adecuada a aplicar.

Métodos: Palabra que proviene del término griego methodos (camino o vía) y que se refiere al medio utilizado para llegar a un fin.

Métricas de software: Cualquier medida relacionada con un sistema, proceso o documentación de software.

MSF: Significado de sus siglas en inglés Armadura de Solución de Microsoft.

PHP: Significado de sus siglas en inglés Procesador de hipertexto.

Prototipos: Primer ejemplar que se fabrica de una figura, un invento u otra cosa, y que sirve de modelo para fabricar otros iguales, o molde original con el que se fabrica.

Requerimientos: Descripción de una condición o capacidad que debe cumplir un sistema, ya sea derivada de una necesidad de usuario identificada, o bien, estipulada en un contrato, estándar, especificación u otro documento formalmente impuesto al inicio del proceso.

Riesgos: Constituye una probabilidad de daño o condición de vulnerabilidad, siempre dinámica e individual. No implica necesariamente daño, no otorga certezas, ni relaciones directas y deterministas, sino una probabilidad de ocurrencia de éste.

RUP: Significado de sus siglas en inglés Proceso Unificado de software.

SGC: Sistemas de gestión de la calidad.

UML: Lenguaje Unificado de Modelado.