UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 7

Trabajo de Diploma para Optar por el Título de

Ingeniero en Ciencias Informáticas

Sistema para la Gestión Administrativa de la Facultad 7. Módulo para la Gestión de Reuniones.

Autores:
Alfredo Betancourt Escalona
Miguel Alejandro Martínez Rodríguez

Tutor:
Ing. Alejandro Martínez Brito
Ing. Yanoksy Durañona Yero

Ciudad de la Habana, julio de 2010
"Año 52 de la Revolución"

RESUMEN

El presente trabajo tiene como objetivo fundamental viabilizar el proceso de gestión de reuniones en la Facultad 7 de la Universidad de las Ciencias Informáticas. Surgió por la necesidad que presentó la facultad de mantener un control de las reuniones y cumplimiento de los acuerdos tomados en ellas.

La dirección del Sistema Integral de Gestión de Administrativa de la Facultad definió las tecnologías y herramientas a utilizar en la implementación del sistema. Como Metodología de desarrollo se define Rational Unified Proccess (RUP). Como lenguaje de Modelado Lenguaje de Modelado Unificado (UML 2.0). Como herramienta de modelado Enterprise Architect (EA). Como gestor de bases de datos se tiene PostgreSQL. Como lenguaje de programación se utiliza Python, Django como marco de desarrollo web y WingIDE como entorno de desarrollo. Además se implementa el patrón de arquitectura Model View Template.

Con la puesta en práctica del sistema se espera facilitar el proceso de realización de reuniones de la facultad, así como el almacenamiento para posteriores consultas de los acuerdos, opiniones, temas tratados y el control de la asistencia a las mismas.

Tabla de Contenidos

Tabla de Contenidos

INTRODUCCIÓN

En el mundo, el desarrollo de la informática, las comunicaciones y de las tecnologías en general, se ha expandido a todos los sectores de la sociedad. Lo que facilita a la mayoría de las organizaciones de tipo empresariales e instituciones optar por la informatización de gran parte de sus procesos de trabajo, con el fin de que estos se realicen de forma más rápida y eficiente.

Cuba tampoco es ajena a este aspecto tan necesario para el control y mejoramiento de diversos sectores de la sociedad. Por eso es que el gobierno se ha propuesto entre sus principales tareas la informatización en diversos sectores tanto económicos como sociales. La Universidad de las Ciencias Informáticas(UCI) se ha trazado estrategias para informatizar la gestión de la información administrativa. A pesar de esto se conoce que se mantienen un alto número de procesos dentro de la misma que aún trabajan con la información de forma manual, haciendo más lento el proceso.

En específico la Facultad 7 forma parte en la informatización que se viene llevando a cabo en la Universidad, entre las actividades que se realizan en esta resultando provechoso su mejoramiento se encuentra la Gestión de Reuniones. La misma parte de la persona o colectivo interesado en realizar una reunión y esta procede a gestionar el local, fecha en la que se efectuará, personal designado a participar y elaboración de un orden del día a debatir. Finalizada la reunión existen acuerdos tomados en ella que tienen un seguimiento y control.

Actualmente en la Facultad 7, se presentan dificultades a la hora de llevar a cabo un seguimiento y control de las actividades que se programan en reuniones convocadas, ya sean docentes, productivas o extracurriculares en cada una de las estructuras existentes en la misma. Esto trae como consecuencia que en la mayoría de los casos se reciban

indicaciones de diferentes instancias de la Institución ya sean de otros departamentos, de la producción e incluso de la propia universidad, lo que conlleva a un retraso y descoordinación en el uso del sistema de mensajería como resultado de la invocación de dichas reuniones. La falta de control de gran parte de las actividades asignadas por los directivos, provoca en la mayoría de los casos, que no se lleven a cabo la revisión de los acuerdos y actividades pendientes con la mayor rapidez y eficacia posible debido a que:

1- Existen ineficiencias a la hora de divulgar las informaciones necesarias tales como: plan de trabajo, avisos, convocatorias, actas de información, planificación de actividades, y chequeo del cumplimiento de las mismas.

2- Actualmente la planificación de los eventos de información y reuniones de directivos en general se efectúan de forma manual (Clásica), haciendo uso de medios que no son los suficientemente eficaces para garantizar que las actividades con estas características se desarrollen correctamente en tiempo y espacio.

3- Para el correcto orden y funcionamientos de las instituciones este tipo de actividad debe ser llevado a cabo por la forma tradicional en que se desarrollan dichos eventos. Estos generan demoras en la asistencia y el control del cumplimiento de estas actividades, se evidencian retrasos en la divulgación y desacuerdos en distintos puntos de los que tratan los sectores dependiendo del tema que traten cada uno de ellos.

Dichas situaciones hace que se vea afectada la toma de decisiones por no contar con toda la planificación necesaria y debidamente organizada para ello. A raíz de la **Situación Problémica** antes expuesta se formula el siguiente **Problema Científico**:

¿Cómo viabilizar el proceso de la gestión de reuniones y acuerdo en la facultad #7 de la Universidad de las Ciencias Informáticas?

El problema se enmarca en el **objeto de estudio** el proceso de gestión de la información administrativa en la Universidad de las Ciencias Informáticas.
El **campo de acción** el proceso de gestión de la información relacionada con la gestión de reuniones en la facultad 7 de la Universidad de las Ciencias Informáticas.

Para la solución del problema se plantea como **objetivo general**: Desarrollar un sistema informático que permita viabilizar la gestión de la información que se genera en los procesos relacionados con la gestión de reuniones en la Facultad 7 de la Universidad de las Ciencias Informáticas.

Para dar cumplimiento al objetivo planteado se proponen las siguientes **tareas de la investigación:**

- Analizar los procesos de negocio asociados a la gestión de la información relacionada con la gestión de reuniones en la facultad 7 de la Universidad de las Ciencias Informáticas.
- Realizar un análisis acerca de los sistemas informáticos existentes a nivel nacional e internacional referentes a la gestión de reuniones.
- Asimilar la arquitectura definida para el desarrollo del Sistema Integral de Gestión Administrativa.
- Modelar los procesos actuales de la gestión de la información relacionada con la gestión de reuniones en la Facultad 7 de la Universidad de las Ciencias Informáticas.
- Especificar los Requisitos de Software.
- Realizar el modelado de casos de uso del sistema.
- Diseñar el sistema informático utilizando la arquitectura definida para el desarrollo del Sistema Integral de Gestión Administrativa.

- Realizar el diseño de la base de datos.
- Implementar el sistema informático.

La actualidad y necesidad del trabajo hacen necesaria la elaboración de un software que garantice:

- Una solución de software integrada al Sistema para la Gestión Administrativa de la facultad que posibilite la gestión de reuniones y seguimiento de acuerdos de las mismas en la facultad.
- El control de asistencia en las reuniones.
- La confección del orden del día.
- La creación de la lista de acuerdos.
- Emitir reportes de asistencia y del cumplimiento de los acuerdos tomados.
- La implantación y validación de la solución esperada.

El presente trabajo, está estructurado en cuatro capítulos, distribuidos de la siguiente forma:

Capítulo 1. Fundamentación Teórica: problemas que motivan la investigación, sistemas existentes vinculados al campo de acción. Tendencias y Tecnologías Actuales a Considerar: descripción de tendencias y tecnologías seleccionadas, para el desarrollo de la propuesta de solución.

Capítulo 2.Características del Sistema: definición de los procesos, actores, trabajadores, casos de uso del negocio, diagramas de clases del modelo de objetos del negocio; requisitos funcionales y no funcionales; actores y casos de uso del sistema.

Capítulo 3.Análisis y Diseño del Sistema: descripción del diseño a través de diagramas de clases análisis y diseño de la aplicación; en la realización de los diagramas de diseño se tienen en cuenta estereotipos web, que describen la relación entre las páginas. Se

obtiene el diagrama de clases persistentes para generar a partir del mismo la base de datos y se definen, además, los principios de diseño.

Capítulo 4. Implementación: descripción de los modelos de implementación, los diagramas de despliegue y de componentes. Se describe como se implementan en términos de componentes y su organización y dependencia entre nodos físicos, en los que funcionará la aplicación.

Capítulo 1. Fundamentación Teórica

El presente capítulo tiene como objetivo fundamental abordar distintos temas que sirven de soporte teórico al sistema a implementar, así como el estado del arte del sistema y se hace alusión a los antecedentes de los sistemas que han tenido relación con la gestión de reuniones en Cuba.

Además se realiza un análisis de la metodología a utilizar, el servidor de bases de datos, el servidor Web, y los lenguajes utilizados en la aplicación. Finalmente, se hace referencia a las herramientas que son necesarias en la ingeniería y desarrollo de la aplicación.

1.1. Sistemas para la Gestión de Reuniones a nivel Internacional

Se han identificado sistemas que incluyen módulos gestores de reuniones a nivel internacional, herramientas potentes que son capaces de gestionar desde planes de trabajo hasta eventos familiares.

Efficient Calendar

El software Efficient Calendar es un calendarizador, planeador y programa de recordatorios, que es elegante además muy fácil de usar. Múltiples vistas del calendario, tales como las vistas de Día, Semana, Mes y Año además ofrece una lista disponible para que pueda organizar y realizar un mejor seguimiento de sus eventos. Al establecer una sub-tarea en cualquier nivel, el usuario puede organizar de la mejor manera y manejar o administrar la lista de \"cosas por hacer\". Todas las citas, reuniones y tareas pueden ser recordadas en el momento configurado.

Efficient Reminder

Es una aplicación profesional, elegante y práctica para la administración de citas, reuniones, feriados y eventos. Con la ayuda del programa Efficient Reminder será capaz de ponerse al corriente con las reuniones y coordinar las mismas.

Labor Plan

Entre sus muchas opciones se encuentran múltiples configuraciones de cuadrantes, turnos, permisos, vacaciones, horarios por turnos y fechas, personal mínimo de cada equipo de trabajo en cada turno, días fijos para librar, turnos válidos para cada empleado, equipos de trabajo, movimientos de personal, etc. Además también ofrece cuatro tipos de listados relacionados con las fichas de personal, los movimientos, los turnos y las estadísticas laborales.

Dejando de lado las características generales, Labor Plan también dispone de algunas herramientas adicionales como copias de seguridad, control de acceso por usuario y contraseña, y una completa ayuda online.

Principales características de los sistemas internacionales

Ofrecen múltiples vistas del calendario, como la vista diaria y la vista mensual.

Todas las citas registradas, reuniones, eventos y tareas podrán ser recordadas de la forma en que fueron configuradas en forma de mensajes o avisos (sonoros o visuales).

El producto le proporciona características como una búsqueda rápida, funciones de copiar y pegar eventos y también importaciones masivas.

Proporcionan soporte para otras características diferentes, como por ejemplo la importación las de días festivos predefinidos, etiquetas para eventos específicos de diferentes colores, manejo de eventos mediante las prioridades de los mismos y la papelera de reciclaje.

Principales deficiencias de los sistemas internacionales

Son programas enfocados principalmente en la planificación de eventos familiares o de tipo personal, son privadas en su mayoría y no se centran en la organización de una reunión sino en el proceso de notificación al usuario.

1.2. Sistemas para la Gestión de Reuniones en Cuba

VIR 120

Implementado para mejorar el Plan de trabajo de la universidad de la habana, permite el control de actas, plan de trabajo, plan de reuniones, reserva de locales, chequeo de actividades, chequeo de medidas y sus seguimientos.

Debido a las características que presenta la Universidad de las Ciencias Informáticas propias y únicas en la enseñanza superior la implantación del VIR 120 no es factible por lo que solo se utilizará como ejemplo para la implementación de algunas funcionalidades del sistema.

1.3. Tecnologías

Teniendo en cuenta las necesidades vistas y las características del entorno donde se aplicará la solución propuesta, se realizó un estudio de las tecnologías actuales posibles a emplear, descritas a continuación.

1.3.1 Aplicaciones Web

Una Aplicación Web es una especialización de las aplicaciones Cliente/Servidor, están generalmente estructuradas como una aplicación en tres capas, son conocidas como aplicaciones servidor. En su forma más general, el navegador web es la primera capa, la segunda capa realiza la función de intermediaria entre la primera y tercera capa y se caracteriza por el uso de tecnología web dinámica ejemplos: CGI (Common Gateway Interface o Interfaz de Entrada Común), PHP (Hypertext Preprocessor o Procesador de Hipertexto), ASP (Active Server Pages o Servidor de Páginas Activas), Java y la tercera capa se conoce como capa de base de datos. [2]

Ventajas:

Permite a los clientes o usuarios migrar de sistema operativo o cambiar de hardware libremente sin afectar el funcionamiento de las aplicaciones servidor.

No se requieren complicadas combinaciones Hardware/Software para utilizar estas aplicaciones, los requerimientos mínimos consisten en un cómputo con un buen navegador web.

Facilita el trabajo a larga distancia, permite trabajar desde cualquier PC con conexión a Internet.

Desventajas:

Necesidad de conexión permanente y rápida a Internet hacen que el acceso a estas aplicaciones no esté al alcance de todos.

Elementos de interacción muy limitados.

1.3.2 Lenguaje Unificado de Modelado (UML)

UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software. UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. Debido a que el UML es un lenguaje, cuenta con reglas para combinar tales elementos. En lugar de indicarle cuáles son los elementos y las reglas, véase directamente los diagramas ya que se utiliza para hacer el análisis del sistema. UML es un lenguaje que permite la modelación de sistemas con tecnología orientada a objetos. [3]

UML 2.0 es la mayor revisión que se le ha hecho a UML desde la versión 1.0. El modelo conceptual ha sido reestructurado completamente y nuevos diagramas han sido incorporados. Los diagramas tradicionales también han sido mejorados. La nueva versión permitirá a los fabricantes de herramientas CASE, proporcionar a los analistas, arquitectos y desarrolladores, herramientas cada vez más potentes, que les permitan aprovechar mejor los modelos y como consecuencia generar una mayor cantidad de código, reduciendo significativamente el ciclo de desarrollo de sus aplicaciones.

1.3.2 Lenguajes de Programación

Al mundo encontrarse inmerso en un desarrollo continuo, son muchos los lenguajes para programar Aplicaciones Web que han ido surgiendo, en la actualidad estos lenguajes se dividen en:

Lenguajes del lado del Servidor: son los lenguajes que son reconocidos, ejecutados e interpretados por el propio servidor y son enviados al cliente en un lenguaje comprensible.

Lenguajes del lado del Cliente: independiente del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio.

Entre los diferentes lenguajes de programación se encuentra **Python,** el cual se caracteriza por ser:

Programación orientada a objetos.

Programación estructurada.

Programación funcional.

Multiplataforma.

Lenguaje de alta calidad.

Aplicaciones y librerías independientes.

Framework

En el desarrollo de software, un framework es una estructura conceptual y tecnológica de soporte definida, normalmente con artefactos o módulos de software concretos, con base en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Los objetivos principales que persigue un Framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones. Un Framework Web, podemos definirlo como un conjunto de componentes por ejemplo clases en java, descriptores y archivos de configuración en XML que componen un diseño reutilizable que facilita y agiliza el desarrollo de Aplicaciones Web. [4]

Django

Como Framework utilizamos el Django y sus características son:

Internacionalización de lenguaje (Interfaz).

Sistema de BD robusto (API).

Sistema extensible de plantillas.

Sistema incorporado de "vistas genéricas“.

Posee gran comunidad de desarrollo.

Sistema de redirección de URLs.

Documentación incorporada.

1.3.3 Sistemas Gestores de Base de Datos (SGBD)

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

Definir una base de datos: especificar tipos, estructuras y restricciones de datos.

Construir la base de datos: guardar los datos en algún medio controlado por el mismo SGBD.

Algunas de las características deseables en un Sistema Gestor de Base de Datos son:

Control de la redundancia.

Restricción de los accesos no autorizados: cada usuario debe tener permisos de acceso y autorización.

Cumplimiento de las restricciones de integridad: el SGBD debe ofrecer recursos para definir y garantizar el cumplimiento de las restricciones de integridad.

PostgreSQL.

PostgreSQL es un sistema de gestión de base de datos relacional orientada a objetos y libre, publicado bajo la licencia BSD.

Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una sola empresa sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo. Dicha comunidad es denominada el PGDG (PostgreSQL Global Development Group).

PostgreSQL está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo. Posee muchas características que tradicionalmente solo se podían ver en productos comerciales de alto calibre.

Las principales mejoras en PostgreSQL incluyen: [5]

- ✓ Los bloqueos de tabla han sido sustituidos por el control de concurrencia multiversión, el cual permite a los accesos de solo lectura continuar leyendo datos consistentes durante la actualización de registros, y permite copias de seguridad en caliente desde pg_dump mientras la base de datos permanece disponible para consultas.
- ✓ Se han implementado importantes características del motor de datos, incluyendo subconsultas, valores por defecto, restricciones a valores en los campos (constraints) y disparadores (triggers).
- ✓ Se han añadido características adicionales que cumplen el estándar SQL92, incluyendo claves primarias, identificadores entrecomillados, forzado de tipos cadenas literales, conversión de tipos y entrada de enteros binarios y hexadecimales.
- ✓ Los tipos internos han sido mejorados, incluyendo nuevos tipos de fecha/hora de rango amplio y soporte para tipos geométricos adicionales.

- ✓ La velocidad del código del motor de datos ha sido incrementada aproximadamente en un 20-40%, y su tiempo de arranque ha bajado el 80% desde que la versión 6.0 fue lanzada.

1.4. Arquitectura

Modelo-Vista-Controlador (MVC)

Es una de las guías más usadas para el diseño de arquitecturas de aplicaciones que ofrezcan una fuerte interactividad con usuarios. Este patrón organiza la aplicación en tres modelos separados:

Modelo: representa los datos de la aplicación y las reglas de negocio.

Vistas: representa a los formularios de entrada y salida de la información.

Controlador: representa al conjunto de controladores o páginas servidoras que procesan las peticiones de los usuarios y controla el flujo de ejecución del sistema.

Model- Template-View (MTV)

En el patrón de diseño MTV:

- M significa Model (Modelo), la capa de acceso a la base de datos. Esta capa contiene toda la información sobre los datos: cómo acceder a estos, cómo validarlos, cuál es el comportamiento que tiene, y las relaciones entre los datos.
- T significa Template (Plantilla), la capa de presentación. Esta capa contiene las decisiones relacionadas a la presentación: como algunas cosas son mostradas sobre una página web u otro tipo de documento.
- V significa View (Vista), la capa de la lógica de negocios. Esta capa contiene la lógica que accede al modelo y la delega a la plantilla apropiada: puedes pensar en esto como un puente entre el modelo y las plantillas.

Si se está familiarizado con otros frameworks de desarrollo web MVC, como Ruby on Rails, quizás considere que las vistas de Django pueden ser el controlador y las plantillas

de Django pueden ser la vista. Esto es una confusión desafortunada a raíz de las diferentes interpretaciones de MVC. En la interpretación de Django de MVC, la vista describe los datos que son presentados al usuario; no necesariamente el cómo se mostrarán, pero si cuáles datos son presentados. En contraste, Ruby on Rails y frameworks similares sugieren que el trabajo del controlador incluya la decisión de cuales datos son presentados al usuario, mientras que la vista sea estrictamente el cómo serán presentados y no cuáles.

Ninguna de las interpretaciones es más ¨correcta¨ que otras. Lo importante es entender los conceptos subyacentes.

1.5. Metodología

Proceso Unificado de Desarrollo (RUP)

El Proceso Unificado de Desarrollo del Software más conocido por sus siglas RUP fue publicado en 1988 como resultado de varios años de experiencia. Es un proceso para el desarrollo de un software que define claramente *quién, cómo, cuándo y qué debe hacerse* en el proyecto. RUP es dirigido por casos de uso, centrado en la arquitectura e iterativo e incremental.

RUP cubre el ciclo de vida de desarrollo de un proyecto y toma en cuenta las mejores prácticas a utilizar en el modelo de desarrollo de software. Es un proceso basado en componentes, que utiliza el Lenguaje Unificado de Modelado (UML) para preparar todos los esquemas de un sistema software.

El presente capítulo ha posibilitado conocer y adentrarse en el mundo actual de la gestión de reuniones. Se identificaron un grupo de resultados de otros sistemas que persiguen el mismo objetivo de informatización, cada uno de ellos con características comunes pero con marcadas diferencias con la UCI debido al complejo nivel organizativo que presenta cada facultad y con funcionalidades que no entran dentro del alcance del presente trabajo; por lo que solo se utilizarán estos sistemas como ejemplo o guía para el desarrollo de la solución propuesta.

Además se asimilaron las principales tendencias, tecnologías y herramientas definidas por la Facultad que se utilizarán tanto en el modelado y la implementación del software, como para la generación de documentos.

Capítulo 2. Características del Sistema

La calidad de un producto de software, está estrechamente relacionada con la calidad que se inicie su proceso de desarrollo. Se basa en comprender la estructura de la organización en la cual se va a implantar el producto, comprender los problemas actuales y asegurar que los consumidores, usuarios finales y desarrolladores tengan un entendimiento común de la organización.

En el presente capítulo se abordan características del sistema como: definición de los procesos, actores, trabajadores, casos de uso del negocio, diagramas de clases del modelo de objetos del negocio; requisitos funcionales y no funcionales; actores y casos de uso del sistema.

2.1 Descripción del flujo de trabajo en la Gestión de Reuniones.

El análisis del flujo de trabajo en la gestión de reuniones permite conocer su funcionamiento para producir uno o varios resultados. Dicho análisis revela problemas potenciales tales como: la circulación de información doble, pérdida y acumulación de documentos, entre otros.

En la gestión de reuniones, el flujo comienza cuando el interesado en realizar una reunión busca el horario óptimo con la menor cantidad de afectaciones del personal implicado. Luego se procede a reservar un local disponible para esta hora, acto seguido se envía la notificación por correo electrónico al personal que debe estar presente en la misma. Se precisa la fecha, hora y lugar, dicha notificación debe de enviarse con 72 horas de antelación. Al realizarse la reunión se elabora el acta, con diferentes puntos a los que debe de dársele continuidad. Además se recoge todo lo analizado así como la asistencia y el por qué de las ausencias que también deben de contabilizarse. Además del chequeo de acuerdos es posterior y cuenta con la fecha de cumplimiento el cual acepta el personal implicado y debe de verificarse.

2.2 Análisis crítico de ejecución de los procesos actuales

El proceso de gestión de reunión que se viene llevando se ve afectado por un grupo de aspectos que entorpecen la obtención de buenos resultados. Existen ineficiencias a la hora de divulgar las informaciones necesarias tales como: plan de trabajo, avisos, convocatorias, actas de información, planificación de actividades, y chequeo del cumplimiento de las mismas. Actualmente la planificación de los eventos de información y reuniones de directivos en general se efectúan de forma manual (Clásica), haciendo uso de medios que no son los suficientemente eficaces para garantizar que las actividades con estas características se desarrollen correctamente en tiempo y espacio.

2.3 Objeto de Automatización

Para minimizar los problemas existentes en la gestión de reuniones se propone la creación de una aplicación web ¨Módulo para la Gestión de Reuniones¨ la cual como su nombre lo indica es un módulo del ¨ Sistema para la Gestión Administrativa de la Facultad 7¨. Los procesos objetos de automatización del sitio son la confección del orden del día, la gestión de reuniones, el control de asistencia a las mismas así como la emisión de los reportes y la confección de la lista de Acuerdos.

2.4 Actores del Negocio

Un actor del negocio representa un individuo, grupo, entidad, organización, máquina o sistema de información externos con los que interactúa el negocio [6]. Lo que se modela como actor es el rol que se juega cuando se interactúa con el negocio para beneficiarse de sus resultados.

2.5 Trabajadores del Negocio

Un trabajador (rol), define un comportamiento o responsabilidades de un individuo o grupo de individuos trabajando en equipo, en el contexto de una organización de ingeniería de software. [7]

Trabajadores del Negocio	Descripción
Administrador	Es el encargado de revisar y chequear los eventos que en este se programen, de revisar el funcionamiento del sistema, posibles errores y encargado de darle soporte a la aplicación e informar a los usuarios registrados en la aplicación sobre algún problema que tenga la misma.
Usuario	El usuario una vez autenticado, tiene la posibilidad de utilizar la mayoría de las herramientas que brinda el sistema y de hecho todas las que implica el módulo, que son gestionar, crear, eliminar, modificar y verificar a conveniencia los datos que brinde dicho módulo.

En el caso a tratar en la aplicación fue más fácil determinar Procesos del negocio antes de someter un diagrama de caso de uso del negocio.

Los mismos fueron descritos en el capítulo anterior y se hará referencia a ellos nuevamente.

- ❖ Planificar reunión.
- ❖ Control de asistencia.
- ❖ Confección del orden del día.
- ❖ Confección de la lista de acuerdos.
- ❖ Emisión de reportes.

Partiendo de los ya mencionados procesos del negocio es posible modelar uno o varios diagramas para dichos procesos pero para ellos se deben identificar también los Casos de uso del sistema como tal.

2.6 Casos de uso del sistema

Descripción general:

La identificación de los casos de uso del sistema se basa en los procesos del negocio que se lograron identificar, ya que el sistema como tal es una composición de módulos en el cual el "Módulo Gestor de Reuniones" está incluido. En los casos de uso que se deben informatizar dada la importancia y complejidad de los mismos en su versión manual obligan al equipo de desarrollo a organizar y describir el proceso de caso de uso del Sistema junto a sus respectivos diagramas de clases del diseño con respecto al estereotipo WEB.

❖ **CU: Gestionar reunión.**	
Objetivo	Gestionar una o varias reuniones
Actores	Usuario
Resumen	Inicia cuando la persona que esta autenticada en el sistema pueda optar por planificar una reunión, esto teniendo en cuenta que son directivos de la facultad, especificando la hora, fecha, lugar o local, entidad, grupo o departamento, el carácter y objetivo de la reunión. Cambios incluso hasta de las fechas planificadas, pero solo deberá ser aquel que planifique la reunión o con total acceso a los datos. RF1
Complejidad	Alta
Prioridad	Crítico
Precondiciones	La facilidad que brinda al usuario han sido registradas
Postcondiciones	Se crea la reunión.

❖ CU: Gestionar del orden del día.	
Objetivo	Gestionar el orden del día de una reunión
Actores	Usuario
Resumen	Inicia cuando la persona que esta a cargo de la reunión ya tenga de ante mano planificado los puntos que se medirán en la reunión. El sistema debe ser capaz de brindar al usuario la confección del orden del día ya sea mediante una plantilla, modelo, acta , no necesariamente en la extensión .doc, quizás para mejor organización sea necesario definir el acta en el mismo asistente como modelo a llevar cada vez que se vaya a gestionar los puntos del orden del día.
Complejidad	Media
Prioridad	Estándar
Precondiciones	Que ya se haya gestionado la reunión.
Postcondiciones	Se genera el orden del día.

❖ CU: Emisión de reportes.	
Objetivo	Emitir reportes.
Actores	Sistema
Resumen	La emisión de reportes estaría orientada a los chequeos de acuerdos en reuniones que fueron efectuadas, ya que estos deben incluir en la confección de la lista de acuerdos “Una fecha tope para el cumplimiento del mismo” al cual de no darse respuesta debería emitir un mensaje de retraso o incumplimiento. RF 8

Complejidad	Alta
Prioridad	Crítica
Precondiciones	La fecha de cumplimiento del acuerdo esté obsoleta.
Postcondiciones	Se genera un reporte

❖ **CU: Gestionar Actas.**	
Objetivo	Gestionar el acta de una reunión
Actores	Usuario
Resumen	El usuario tendrá la opción de gestionar las actas mediante un modelo ya definido en la aplicación para este trabajo. El usuario podrá añadir una nueva acta, modificar una ya existente o eliminar la que considere innecesaria por motivos convenientes, un acta de manera directa está relacionada con una reunión y de paso esta posee un orden del día, y al confeccionar un acta la asistencia de todo el personal que asistió debe estar reflejada en ella, así como los acuerdos que en esta se tomen. RF6
Complejidad	Media
Prioridad	Estándar
Precondiciones	Que ya se haya gestionado la reunión.
Postcondiciones	Se crea el acta de una reunión.

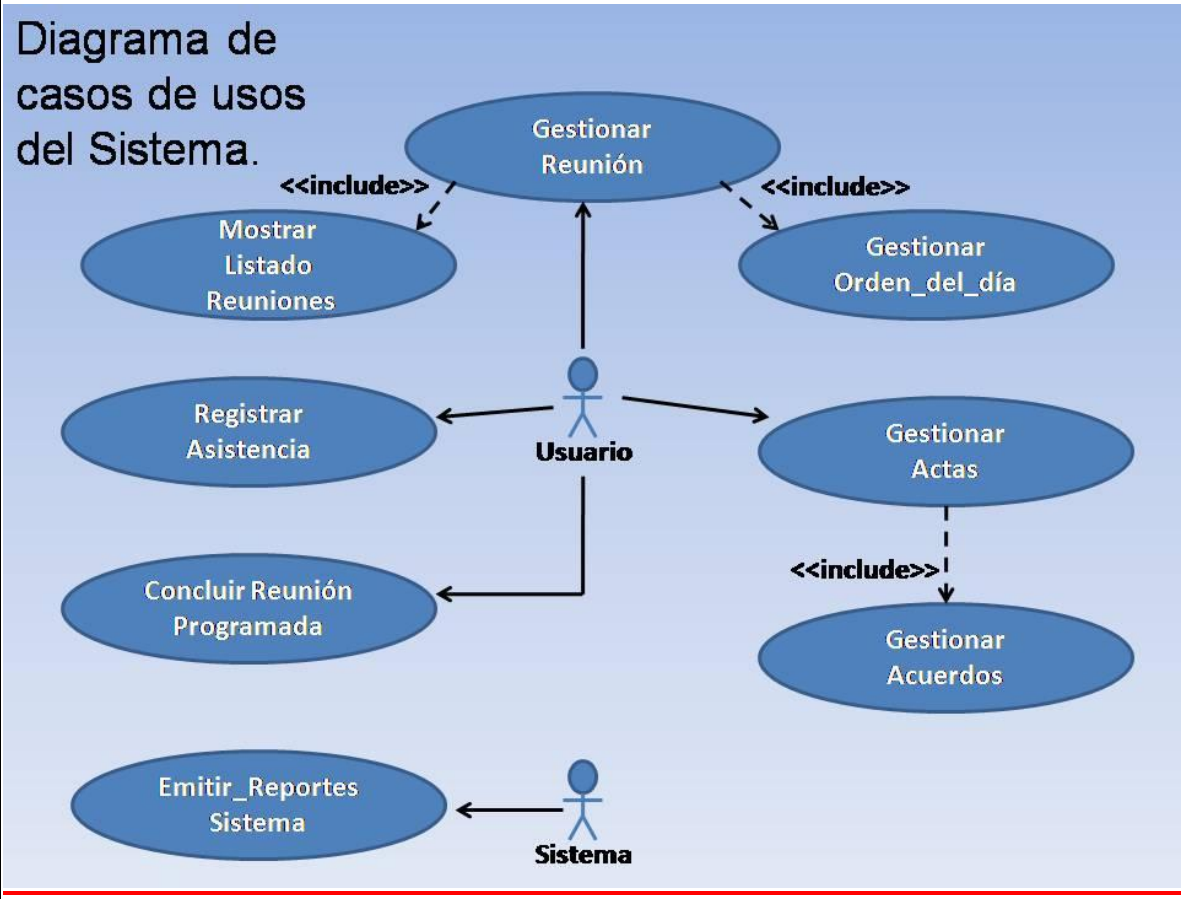
❖ **CU: Control de asistencia.**

Objetivo	Poseer el control de asistencia.
Actores	Usuario
Resumen	Dada una lista conformada en sus inicios de forma manual, el sistema debe permitir crear una o varias listas donde se almacenen las personas que accedieron a dicha reunión, y de no haber asistido debe notificarse las causas. La opción asistencia debe permanecer visible desde el listado de las reuniones ”Mostrar Reuniones”, donde el usuario pueda ver directamente las personas que asistieron, a dicho encuentro. RF9
Complejidad	Media
Prioridad	Estándar
Precondiciones	Que ya se haya gestionado la reunión.
Postcondiciones	Se crea la lista de participantes.

❖ CU: Gestionar acuerdos.	
Objetivo	Crear los acuerdos que se tomen en cada evento.
Actores	Usuario
Resumen	Al igual que el orden del día, cada punto discutido debe ser reflejado, por tanto la confección de la lista de acuerdo debe llevar de entrada la confección del orden del día, con las opciones de notificar en cada una de ellas los acuerdos que fueron tomados. La lista de acuerdos debe poseer de hecho la garantía de poder ser marcada como cumplidos o sencillamente permanecer como pendientes dentro del marco de una fecha que se designe que el mismo deba cumplirse. De no ser así, estos son considerados retrasados. RF: 7
Complejidad	Media
Prioridad	Estándar
Precondiciones	Que ya se haya gestionado la reunión.
Postcondiciones	Se crea la lista de acuerdos.

Trabajadores del Sistema	Descripción
Administrador	Es el encargado de revisar y chequear los eventos que en este se programen, de revisar el funcionamiento del sistema, posibles errores y encargado de darle soporte a la aplicación e informar a los usuarios registrados en la aplicación sobre algún problema que tenga la misma.
Usuario	Todos los usuarios tienen la posibilidad de ver las reuniones creadas y las fechas a efectuarse, pero los mismos solo podrán eliminar y modificar las que hayan sido creadas por él. Las demás opciones del módulo podrán ser efectuadas en correspondencia a lo anterior, o sea que podrán agregar el acta a las reuniones creadas por ellos y podrán ver todas las actas así como los listados de asistencia.
Sistema	El propio sistema como tal es el encargado de notificar a los administradores cuando un acuerdo no ha sido cumplido en el tiempo establecido o cuando una reunión ya esta pasada de fecha para su futura finalización.

Diagrama de Casos de Uso del Sistema



2.7 Especificación de los requisitos de software.

Los requerimientos de software son condiciones o capacidades que debe tener el sistema para satisfacer las necesidades de un cliente, estos deben ser especificados por escrito.

2.7.1 Requerimientos Funcionales.

Los requerimientos funcionales son aquellos requisitos que, desde el punto de vista de las necesidades del usuario, debe cumplir el sistema y que están fuertemente ligados a las opciones del programa.

RF1 -- Registrar Reunión.

RF2 -- Mostrar Listado de Reuniones.

RF3 -- Modificar Reunión.

RF4 – Eliminar Reunión.

RF5 – Gestionar Listado del Orden del día.

RF6 – Registrar Actas.

RF7 – Registrar Acuerdos

RF8 – Emitir reportes.

RF9 – Registrar asistencia.

Rf-10 – Concluir Una Reunión Programada.

2.7.2 Requerimientos No Funcionales.

Los requerimientos no funcionales son características que describen alguna forma o restricción para la realización de algún requerimiento (funcionalidad) o conjunto de ellas e inclusive todos los requerimientos. Se consideran los atributos del sistema, propiedades que debe tener el producto. A continuación se muestran los requerimientos no funcionales:

Seguridad

La herramienta protege la información que se maneja otorgando los permisos necesarios.

Integridad

El sistema se encargará de controlar los diferentes niveles de acceso en el sitio, se identificará al usuario antes de que pueda realizar cualquier acción sobre el sistema. Se realizarán validaciones de la información tanto en el cliente como en el servidor, no obstante los usuarios acceden de manera rápida y operativa al sistema sin que los requerimientos de seguridad se conviertan en un retraso para ellos.

Soporte

Se requiere que el producto reciba mantenimiento ante cualquier fallo que ocurra y que sea de fácil instalación.

Diseño

Aplicación web se desarrollará con tecnología para creación de páginas web dinámicas Python y con gestor de base de datos PostgreSQL.

Interfaz

Interfaces de usuario

La interfaz no contendrá muchas imágenes para no demorar las respuestas al usuario. El diseño de la interfaz es sencillo y claro de usar con reconocimiento visual a través de elementos visibles que identifiquen cada una de sus acciones. Es formal, serio y con una navegación sugerente, todo esto teniendo en cuenta el fin con el que se desarrolla la aplicación.

Requerimientos de Hardware

Se necesitarán como requerimientos mínimos una PC con procesador Pentium II o superior, una memoria RAM de 128 MB o superior, disco duro de 20 GB o más, admite cualquier velocidad de red, aunque se recomienda un ancho de banda mayor de 54kb/s.

Portabilidad

Es compatible con varios navegadores de internet como Mozilla Firefox, Google Chrome y Opera**.**

Restricciones

Se utilizará UML 2.0 para lograr una mejor documentación del sistema y como herramienta de apoyo Enterprise Architec, como lenguaje de programación Python y gestor de base de datos PostgreSQL.

En el presente capítulo se ha realizado un análisis de como ocurre el flujo de la información entre las diferentes esferas de la facultad tanto productivas como administrativas y los profesores. De la información obtenida, se definieron las necesidades de funcionamiento de la aplicación a implementar. A partir de los requerimientos funcionales y no funcionales, se definieron los Casos de Usos del Sistema, que son las funcionalidades que se van a implementar y se describió además, cada uno de los Casos de Usos del Sistema.

Capítulo 3. Diseño del Sistema

Cada paso durante el ciclo de desarrollo del proyecto cumple un importante papel para obtener el producto final, en este caso se desarrolla el Flujo de Trabajo de Análisis y Diseño. Se explicará la estructura del sistema, la arquitectura a utilizar, así como la representación gráfica de los diagramas de clases del análisis, de interacción y de clases del diseño con estereotipos web.

3.1 Arquitectura

Modelo-Vista-Controlador (MVC)

Fue diseñada para reducir el esfuerzo de programación necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos. Sus características principales son que el modelo, las vistas y los controladores se tratan como entidades separadas; esto hace que cualquier cambio producido en el modelo se refleje automáticamente en cada una de las vistas.

Este modelo de arquitectura presenta varias ventajas:

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

Hay un API (Interfaz de Programación de Aplicaciones) muy bien definido; cualquiera que use el API, podrá reemplazar el modelo, la vista o el controlador, sin aparente dificultad.

La conexión entre el modelo y sus vistas es dinámica; se produce en tiempo de ejecución, no en tiempo de compilación.

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

monolítica típica de muchos programas Java. Todos tienen un Marco (Frame) que contiene todos los elementos, un controlador de eventos, y una gran cantidad de cálculos y la presentación del resultado. Ante esta perspectiva, hacer un cambio aquí no es nada trivial.

Definición de las partes

El **Modelo** es el objeto que representa los datos del programa. Maneja los datos y controla todas sus transformaciones. El modelo no tiene conocimiento específico de los controladores o de las vistas, ni siquiera contiene referencias a ellos. Es el propio sistema el que tiene encomendada la responsabilidad de mantener enlaces entre el modelo y sus vistas, y notificar a las vistas cuando cambia el modelo.

La **Vista** es el objeto que maneja la presentación visual de los datos representados por el modelo. Genera una representación visual del modelo y muestra los datos al usuario. Interactúa con el modelo a través de una referencia al propio modelo.

El **Controlador** es el objeto que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el modelo. Cuando se realiza algún cambio, entra en acción, bien sea por cambios en la información del modelo o por alteraciones de la vista. Interactúa con el modelo a través de una referencia al propio modelo.

3.2 Modelo de Diseño

El modelo de diseño es un modelo de objetos que describe la realización de los casos de uso, y sirve como una abstracción del modelo de implementación y el código fuente. Es usado como una entrada inicial en las actividades de implementación y prueba.

Propósito: el modelo de diseño es una abstracción de la implementación del sistema. Es usado para concebir un documento del diseño del sistema de software. Es abarcador, compuesto por artefactos que engloban todas las clases del diseño, subsistemas, paquetes, colaboraciones, y las relaciones entre ellos. [8]

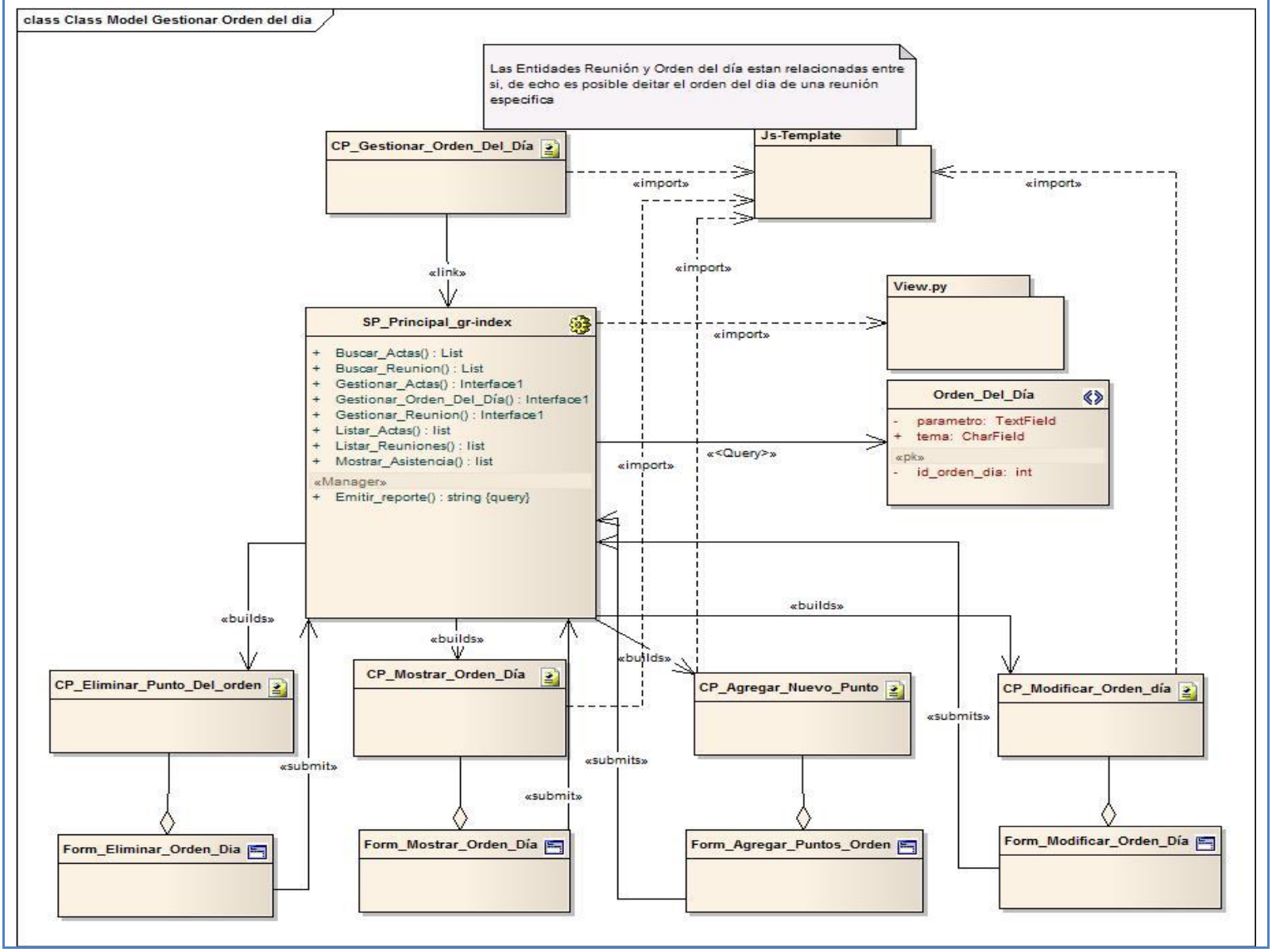
Descripción General de los diagramas de clase del diseño con estereotipo WEB Utilizando el MVT (Model View Template)

Las clases interfaces pueden importar o heredar de una clase Template Java Script, puesto se define una general. La página server page importa o hereda también de una clase general llamada VIEW.py y esta contiene en si las llamadas a métodos para los casos de uso a desarrollar. La clase View.py (la vista) contiene las funcionalidades completas a desarrollar (métodos) no así en la ingeniería común.

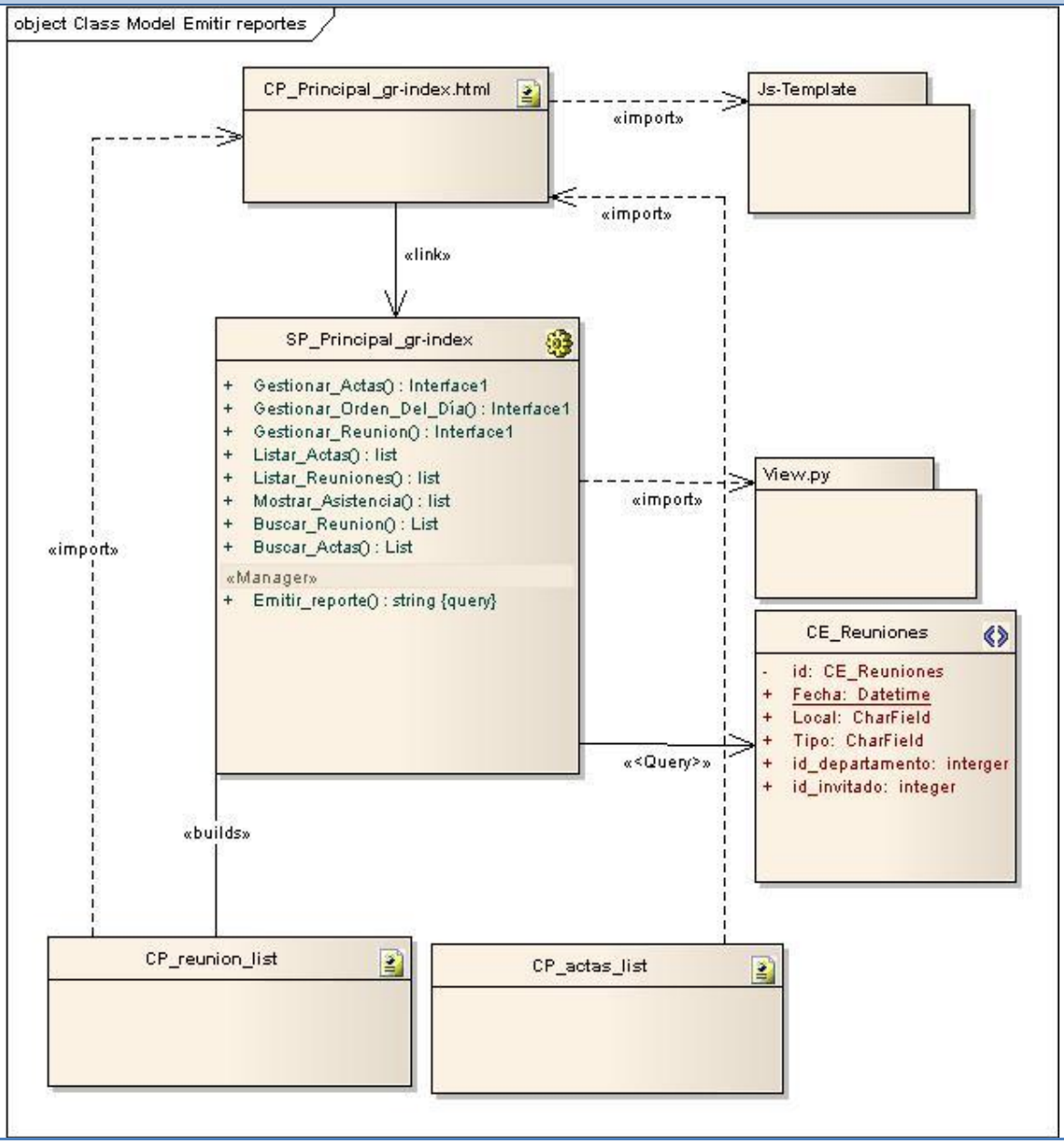
La clase de acceso a datos es innecesaria dado que las clases entidades CE, contiene un patrón llamado Active Record, que implica que las clases entidades porten funcionalidades generales como la salva, la inserción, la modificación de las celdas, campos, tipos de datos entre otros. Cada clase interfaz que se genere después de hacer un pedido GET o Set al sistema hereda de la JS Template original.

La Server page original construye dependiendo de la opción seleccionada por el usuario una clase en al cual esta contiene formularios, pero a diferencia de la ingeniería común esta, puede hacerse un submit a ella misma variando en la interfaz el Java Script solamente. Server page.py en este caso Gestor_de_reuniones.py

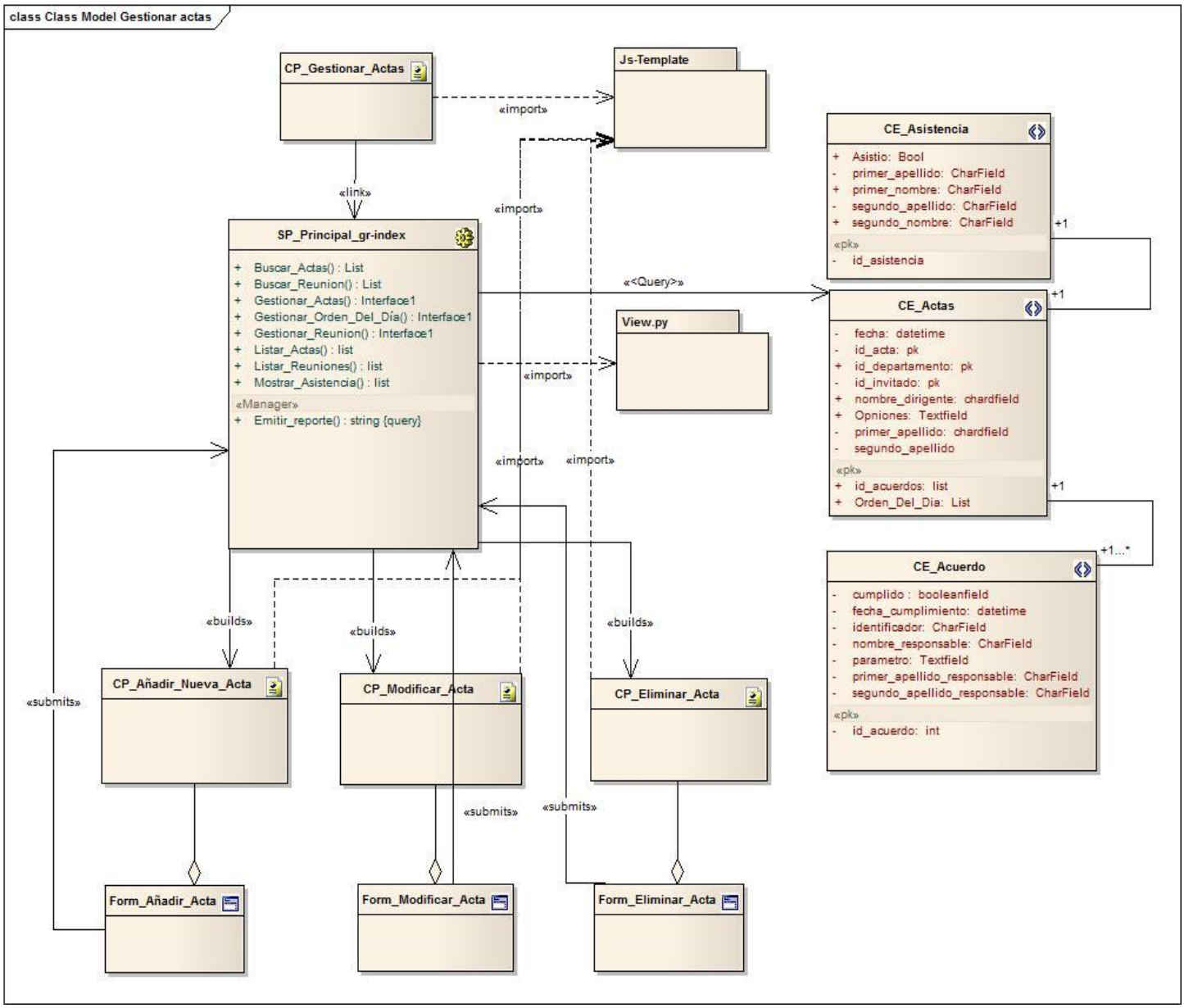
Diagrama de Clases del diseño para el CUS: Gestionar del orden del día.



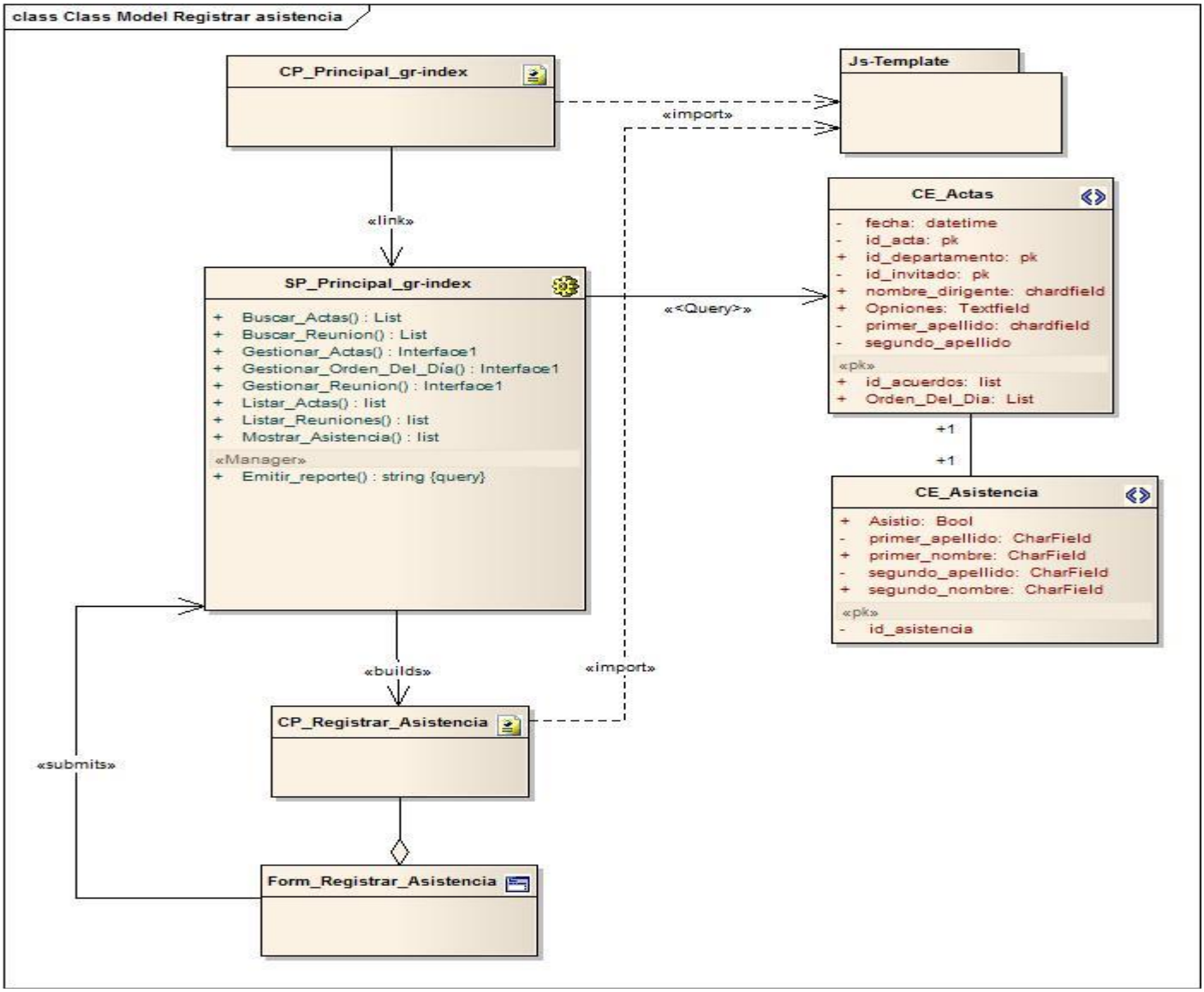
Modelo de Clases del diseño para el CUS: Emitir Reportes.



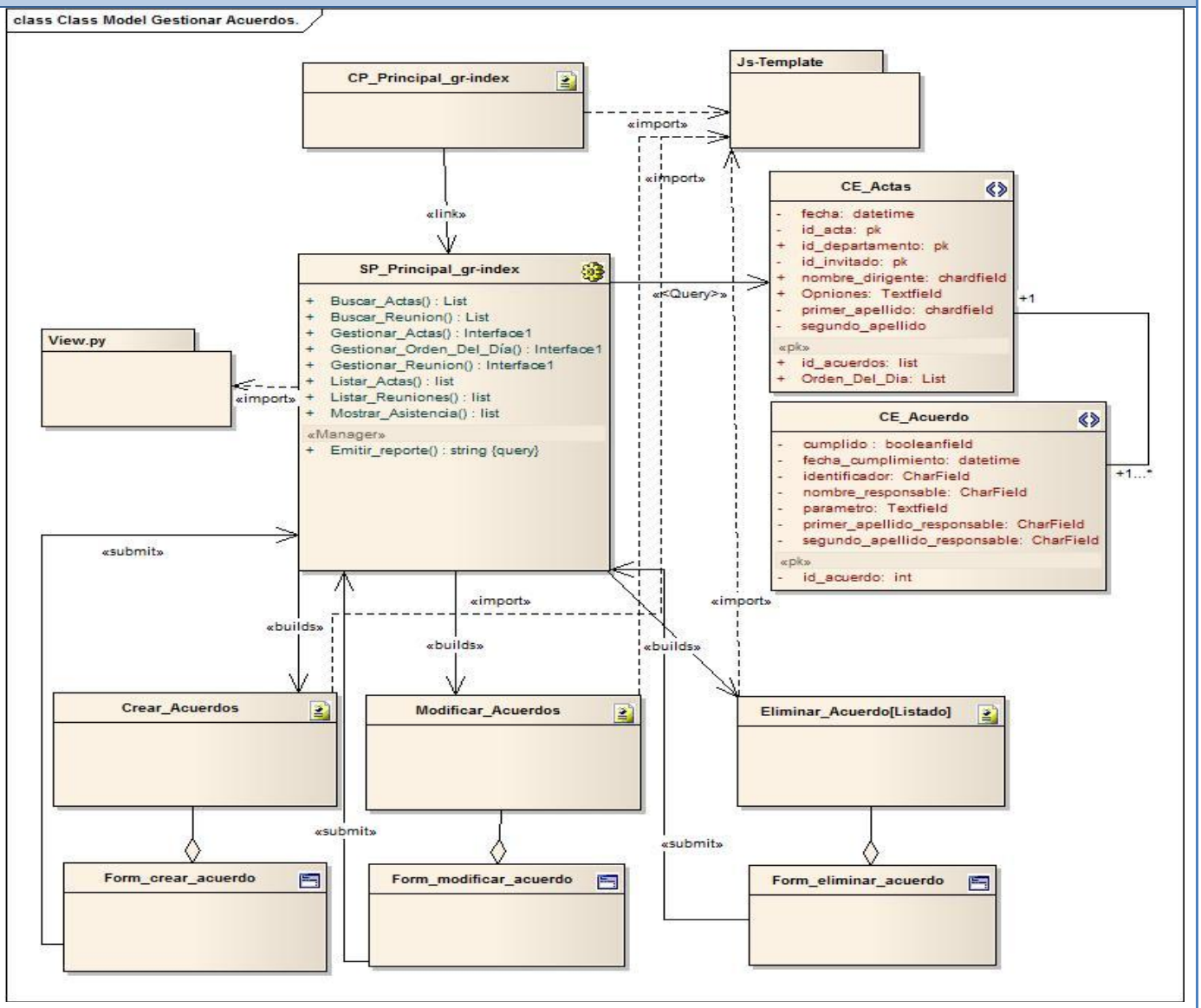
Modelo de Clases del diseño para el CUS: Gestionar Actas



Modelo de Clases del diseño para el CUS: Gestionar asistencia.



Modelo de Clases del diseño para el CUS: Gestionar acuerdos.



Descripción de las entidades.

Reunión		
Descripción	Contiene la información de las distintas reuniones que se han planificado en el la aplicación a partir de las distintas áreas existentes.	
Atributo	**Tipo**	**Descripción**
id _ reunión	integer	Identificador de la Reunión.
Departamento	string	Área docente-productiva que efectuará la reunión.
Dirigente	varchar	Nombre del planificador.
Fecha	string	Fecha en la que tendrá lugar le evento.
Local	varchart	Área o lugar donde se realizará la reunión.
Tema	string	Tema central a discutir en la reunión.
Actas		
Descripción	Contiene la información de las distintas reuniones que se han efectuado en las distintas áreas existentes.	
Atributo	**Tipo**	**Descripción**
id_Reunión	integer	Identificador de la Reunión a que pertenece el acta.
Orden _día	List	Lista de parámetros que se discutieron en la reunión.
Opiniones	List	Lista de argumentos dados por los participantes.
Fecha	string	Fecha en la que tendrá lugar le evento.
Local	varchart	Área o lugar donde se realizará la reunión.
Tema	string	Tema central a discutir en la reunión.
Asistencia		
Descripción	Contiene la asistencia de los participantes de una reunión especifica efectuada en la facultad.	

Atributo	Tipo	Descripción
Asistencia	bool	Medidor true o false para asistentes.
Nombre	varchart	Nombre del participante.
Apellidos	varchart	Apellido del participante.
Departamento	string	Fecha en la que tendrá lugar le evento.
Lista_asistencia	List	Listado de asistencia de la reunión.
Orden_Dia		
Descripción	Contiene los parámetros que se medirán en una reunión específica efectuada en la facultad.	
Atributo	**Tipo**	**Descripción**
Parámetro	integer	Numero del aspecto que se medirá.
Tema	varchart	Tema del punto a tratar.
Argumento	string	Argumento inicial para discutir el tema.
Listado_orden_día	List	Listado de parámetros que se medirán en la reunión.
Acuerdos		
Descripción	Contiene los acuerdos que se tomaron como resultado del análisis de los parámetros efectuados en la reunión.	
Atributo	**Tipo**	**Descripción**
Nombre	varchart	Nombre del participante que propone un acuerdo.
Apellidos	string	Apellidos del participante que propone un acuerdo.
Encargado	string	Persona encargada de cumplir el acuerdo tomado.
Fecha	string	Fecha límite en que se debe cumplir el acuerdo.
Listado_acuerdos	List	Listado de parámetros que se medirán en la reunión.

Capítulo 3.Diseño del Sistema

En este capítulo se ha completado el flujo de análisis y diseño con la realización de los diagramas de clases del análisis, los diagramas de clases de diseño y los diagramas de secuencia de los casos de uso del sistema, proporcionando una idea completa de lo que realmente es el software, materializando con precisión los requerimientos del cliente. Los diagramas y especificaciones de diseño que se proponen constituyen una guía que puede ser fácilmente leída y comprendida por los desarrolladores del sistema.

Capítulo 4. Implementación

Uno de los flujos de trabajo que define RUP como metodología de desarrollo, es el de implementación. En este flujo se materializa la creación del analista, se refina la arquitectura y se obtiene una versión del producto. Sus propósitos se basan generalmente en definir la organización del código, implementar las clases y objetos en forma de componentes, probar los componentes desarrollados e integrarlos en un sistema ejecutable.

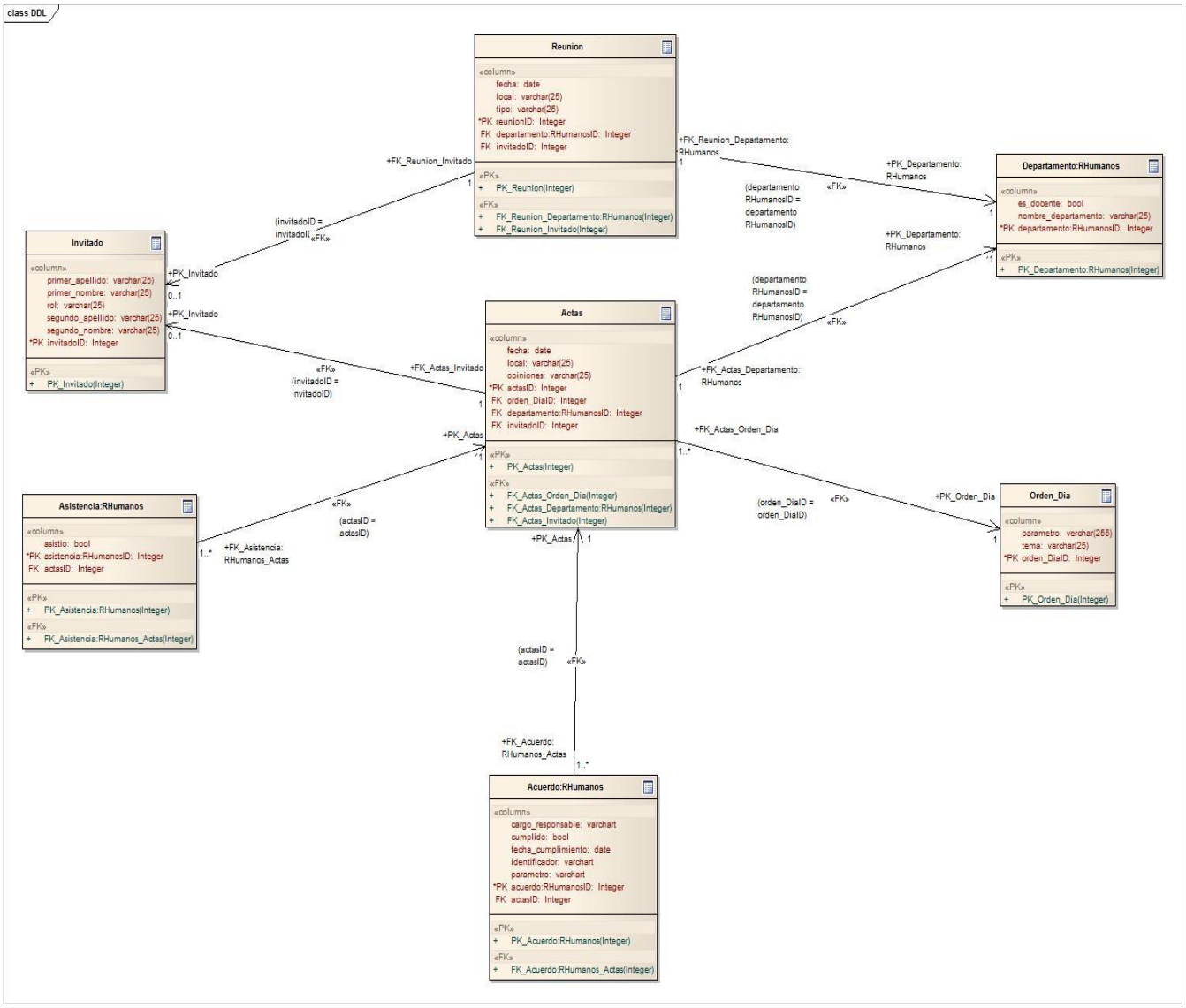
Este capítulo tiene por objetivo describir los diferentes elementos del modelo de datos así como la representación gráfica de los diagramas de componentes y el diagrama de despliegue.

4.1 Modelo de Datos.

Al diseñar una base de datos y gestionar su información, se plasma una parte del mundo real en las tablas, registros y campos ubicados en el ordenador; creando un modelo semejante a la realidad. El principal paso para crear las tablas de la base de datos es diseñar un modelo de datos.

Este modelo es un artefacto de gran importancia dentro del proceso que incluye generar la base de datos. Proporciona una representación visual y física de los datos persistentes del sistema, que finalmente formarán las tablas de la base de datos. A partir del diagrama de clases persistentes que se diseña, se obtiene el modelo de datos. Cada clase que se diseña en el diagrama de clases persistentes constituye una entidad del modelo de datos.

Modelo de Datos



Descripción de las Tablas.

NOMBRE: Reunión		
Descripción: Almacena los datos correspondientes a las reuniones		
Atributo	**Tipo**	**Descripción**
Id_reunión	integer	Llave primaria: representa el identificador de una reunión.
fecha	date	Contiene la fecha para la cual se programa la reunión.
local	varchar	Contiene el local donde tendrá lugar el evento o reunión.
tipo	varchar	Contiene el tipo de reunión que tendrá lugar. Ejemplo: (Balance, Formación, rendición de cuentas…)
departamentos	integer	Contiene los departamentos que gestiona el módulo de ”Recursos Humanos”.
invitado	integer	Contiene los invitados que son gestionados por el usuario.

NOMBRE: Orden_día		
Descripción: Almacena los datos del orden del día a analizar en las reuniones programadas.		
Atributo	**Tipo**	**Descripción**
id_orden_día	integer	Llave primaria: representa el identificador de una orden del día específico.
tema	varchar	Titulo o tema que se abordará en la reunión programada.
parámetro	varchar	Contiene el índice de los puntos a medir en la reunión.

NOMBRE: Invitado
Descripción: Posee los datos los invitados y en especial al invitado particular de cada

reunión.		
Atributo	**Tipo**	**Descripción**
id_invitado	integer	Llave primaria: representa el identificador de un invitado.
primer_nombre	varchar	Nombre del invitado.
segundo_nombre	varchar	2do nombre del invitado, este campo no es obligatorio, puesto que no todas las personas tienen un segundo nombre.
primer_apellido	varchar	Contiene el 1er apellido del invitado.
segundo_apellido	varchar	Contiene el segundo apellido del invitado.
rol	bool	Contiene el rol o cargo que posee dicho invitado.

NOMBRE: Departamento:RHumanos		
Descripción: Almacena los datos correspondientes a departamentos que tendrán una reunión programada.		
Atributo	**Tipo**	**Descripción**
id_reunión_departamento	integer	Llave primaria: representa el identificador entre una reunión y un departamento..
Nombre_departamento	varchar	Nombre del departamento perteneciente al módulo de “Recursos Humanos”, del cual se importan dichos departamentos.
es_docente	bool	Confirma si el departamento es un departamento docente o no.

NOMBRE: Actas		
Descripción: Almacena los datos correspondientes a las actas que son generadas como resultado de las reuniones programadas.		
Atributo	**Tipo**	**Descripción**
id_acta	integer	Llave primaria: representa el identificador de un acta.

fecha	date	Contiene la fecha en la que se confecciona dicha acta.
local	varchar	Posee el local donde se efectuó la reunión.
departamento	integer	Contiene el departamento que llevó a cabo la reunión.
invitado	integer	Contiene la persona que fue invitada especial a presencial dicha reunión.
listado_orden_dia	integer	Contiene el orden del día que se programó para la reunión, con sus respectivos puntos de análisis.
opiniones	varchar	Posee las opiniones de los participantes de la reunión respecto a los puntos de análisis del orden del día.
listado_asistencia	integer	Contiene el listado de personas que asistieron a la reunión.

NOMBRE: Acuerdo:RHumanos		
Descripción: Almacena los datos correspondientes a los acuerdos en las distintas reuniones.		
Atributo	**Tipo**	**Descripción**
id_acuerdo	integer	Llave primaria: representa el identificador de un premio.
identificador	varchar	Contiene más bien el tema al que hace referencia dicho acuerdo. Ejemplo (entrevista, análisis, chequeo).
fecha_cumplimiento	date	Posee la fecha de plazo para efectuar el cumplimiento del acuerdo propuesto.
cargo_responsable	varchar	Salva el cargo que tiene el responsable, ya sea docente o extra docente.

parámetro	varchar	Salva los datos propuestos o características del acuerdo, según propongan los participantes de la reunión.
cumplido	bool	Afirma si el acuerdo se declara como cumplido o no cumplido. Dependiendo de este estado se generan reportes de los mismos por el sistema.

NOMBRE: Actas_Listado_Acuerdos		
Descripción: Almacena los acuerdos pertenecientes a cada acta en particular.		
Atributo	**Tipo**	**Descripción**
id_ actas_listado_acuerdos	IntegerField	Llave primaria: representa el identificador de un acta y uno o muchos acuerdos.
id_acta	IntegerField	Llave foránea: contiene el id del acta
id_acuerdo	IntegerField	Llave foránea: contiene el id del acuerdo o los acuerdos perteneciente(s) a dicha acta.

NOMBRE: Asistencia:RHumanos		
Descripción: Almacena los datos de las una persona que asisten a las reuniones.		
Atributo	**Tipo**	**Descripción**
id_asistencia	integer	Llave primaria: representa el identificador de un listado de asistencia.
asistió	bool	Afirma si la persona asistió o no a la reunión, puesto que se debe registrar el total del personal.

NOMBRE: Actas_Listado_Asistencia
Descripción: Almacena los datos del listado de asistencia perteneciente a las reuniones correspondientes.

Atributo	Tipo	Descripción
id_actas_listado_asistencia	integer	Llave primaria: representa el identificador de un acta y su listado de asistencia.
id_acta	integer	Llave primaria: representa el identificador de un acta.
id_asistencia	integer	Llave primaria: representa el identificador de un listado de asistencia perteneciente a dicha acta.

4.2 Modelo Implementación.

El modelo de implementación está compuesto por una colección de componentes y subsistemas. Estos componentes incluyen ficheros ejecutables, de código fuente o cualquier otro tipo necesario para obtener la versión del sistema y llevarlo al despliegue. Describe como se organizan los componentes de acuerdo con los mecanismos de estructuración disponibles en el entorno de la implementación.

4.3 Diagrama de Despliegue

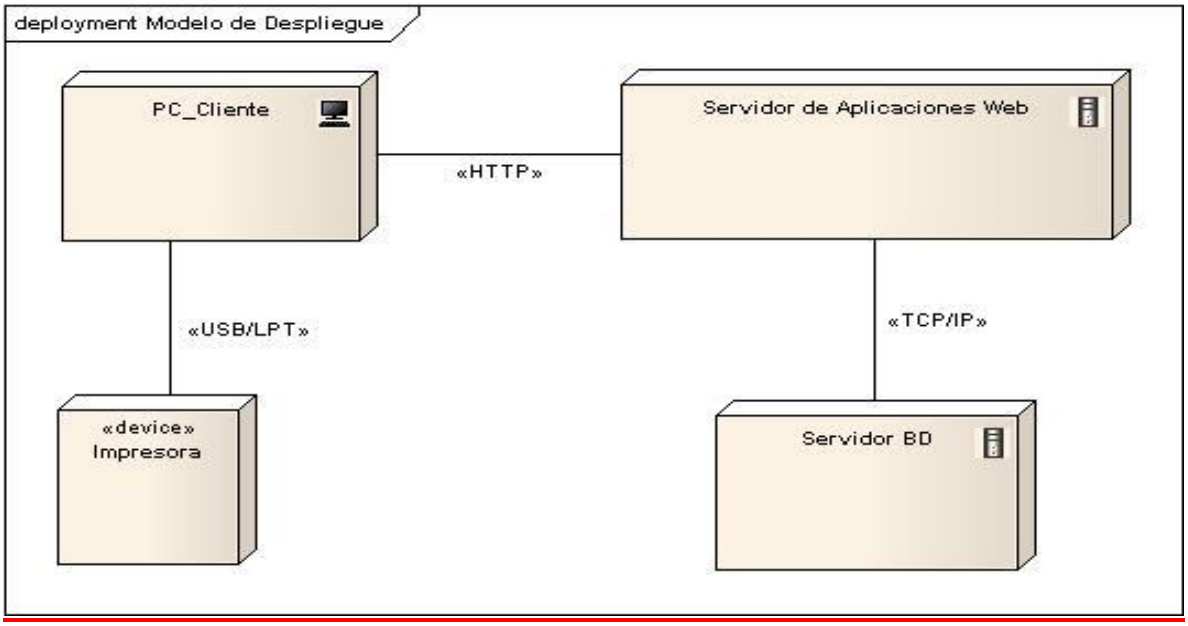
Los diagramas de despliegue muestran las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Un nodo es un recurso de ejecución tal como un computador, un dispositivo o memoria. Los estereotipos permiten precisar la naturaleza del equipo: [9]

- ✓ Dispositivos
- ✓ Procesadores

- ✓ Memoria

Los diagramas de despliegue muestran la configuración en funcionamiento del sistema, incluyendo su hardware y su software. Para cada componente de un diagrama de despliegue se deben documentar las características técnicas requeridas, el tráfico de red esperado, el tiempo de respuesta requerido. [10]

Diagrama de Despliegue



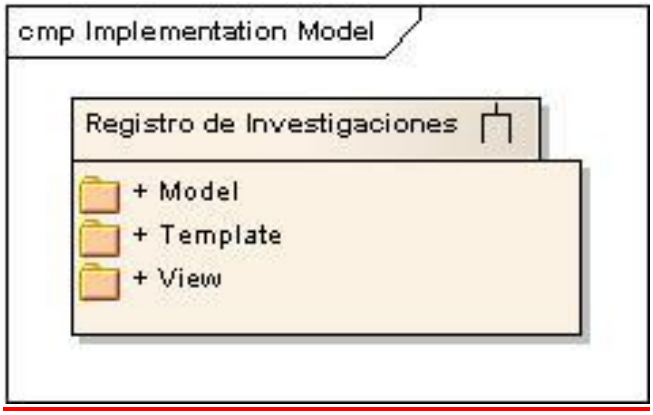
4.4 Diagrama de Componentes

Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes de software, sean éstos de código fuente, binarios o ejecutables. Desde el punto de vista del diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión del software, la reutilización y las

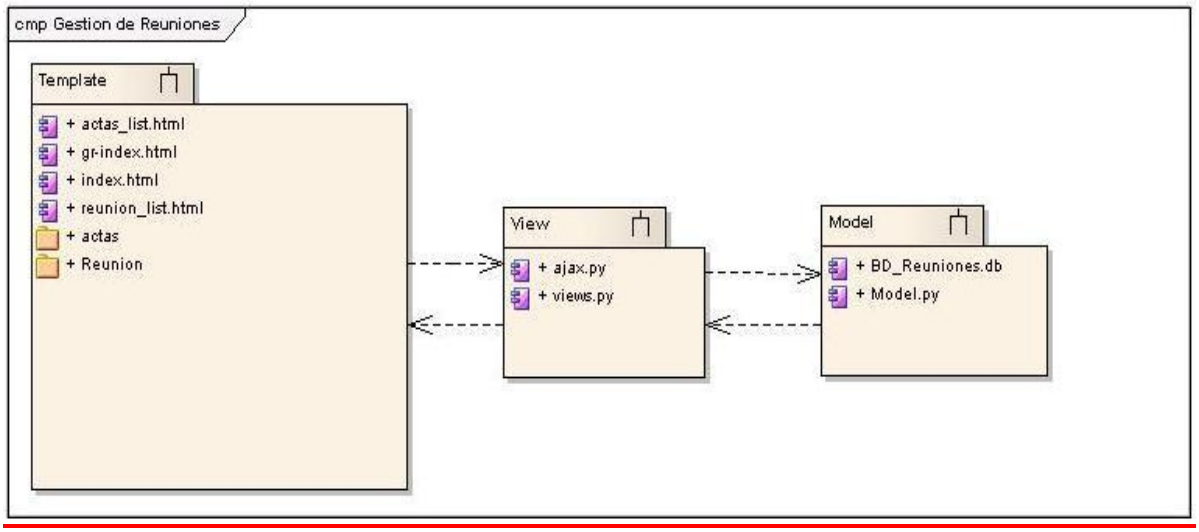
restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo.[11]

El sistema cuenta con tres subsistemas de implementación: Vistas, Controladoras y Modelo, estos están estructurados según el patrón arquitectónico MVC. En el subsistema Vistas se encuentran encapsulados los componentes que permiten la interacción directa con los usuarios del sistema. El subsistema Controladoras contiene todas las clases que manipulan los eventos del usuario y realizan peticiones al modelo para mostrarlas en las vistas. El subsistema Modelo, agrupa las clases que interactúan con la base de datos y velan por el cumplimento de las reglas del negocio.

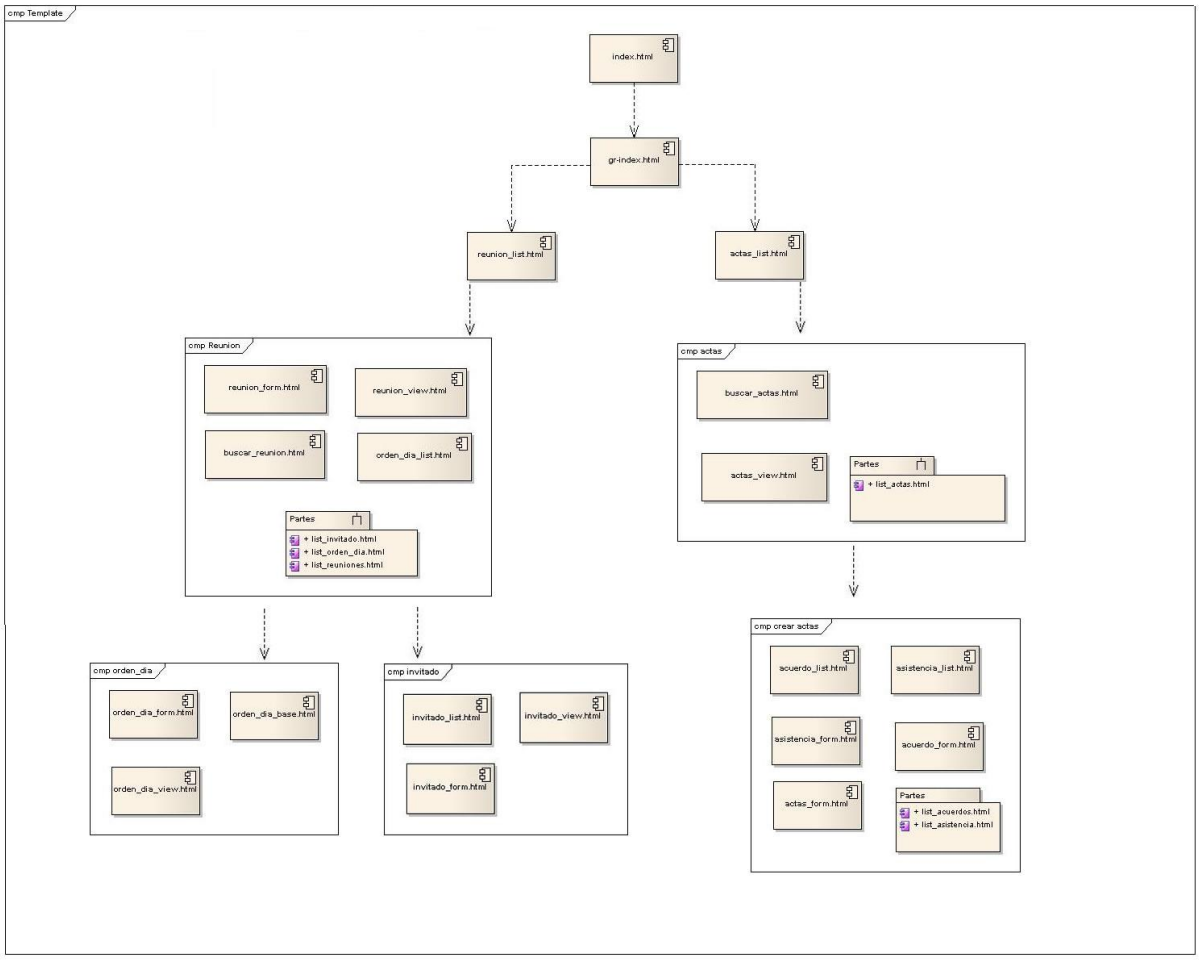
Modelo de Componentes



Modelo de Componentes <Ampliado>



Subsistema Templates <Ampliado>



4.5 Tratamiento de Errores

El sistema muestra mensajes de error, que son de fácil comprensión para el usuario. Se pueden observar cuando se insertan datos y no se entran los campos que son obligatorios, ocurriendo lo mismo cuando se inserta información errónea en los campos.

Capítulo 4. Implementación

En este capítulo se ha completado el flujo de implementación y despliegue con la realización del modelo de datos describiendo las principales tablas del mismo, los diagramas de componentes y el diagrama de despliegue. Además de proporcionar una idea completa de como se lleva a cabo la seguridad del sistema así como el tratamiento de errores y los estándares y estilos a utilizar como estrategias de codificación.

Conclusiones

Luego de un estudio de las diferentes aplicaciones que han sido elaboradas y diseñadas para gestionar reuniones, se demostró que las mismas no se ajustan a la Universidad.

Se identificaron los principales procesos que se llevan a cabo en la planificación, realización y seguimiento de una reunión**;** determinándose la creación del “Sistema para la Gestión Administrativa de la Facultad 7. Módulo para la Gestión de Reuniones”, asimilándose las herramientas a utilizar las cuales son: PostgreSQL de Gestor de Base de Datos, Python como Lenguaje de Programación, como Metodología de Desarrollo de Software el Proceso Unificado de Desarrollo más conocido por sus siglas RUP, UML 2.0 como Lenguaje de Modelado.

Se modelaron los flujos de trabajo propuestos por el Proceso Unificado de Desarrollo: Modelado del Negocio, Requerimientos, Diseño del Sistema e Implementación.

Se implementó el módulo “Sistema para la Gestión Administrativa de la Facultad 7. Módulo para la Gestión de Reuniones”.

RECOMENDACIONES

Teniendo como base los resultados de esta investigación y la experiencia adquirida durante el desarrollo de la misma, se proponen las siguientes recomendaciones:

- Brindar la posibilidad de gestionar reuniones también a los estudiantes para así tener un mayor control y eficacia en las reuniones de la facultad.
- Adicionar una funcionalidad que permita la reserva de locales la cual no solo se encargara de guardar el local donde será producida la reunión sino que también comprobará que el mismo se encuentre disponible para la hora prevista a desarrollarse la misma.
- Implantar el sistema en todas las facultades de la Universidad de las Ciencias Informáticas.

REFERENCIAS BIBLIOGRÁFICAS

[1] IPFisio. [En línea] [Citado el: 20 de Noviembre 2009.]

http://www.ipfisio.com/inicio

[2] Ídem a la referencia 1

[3] Ídem a la referencia 1

[4] Ídem a la referencia 1

[5] Ídem a la referencia 1

[6] **Ingeniería de Software 1**. Fase de Inicio. Modelo del Negocio. UCI. curso 2009_2010. Conferencia # 2.

[7]Proceso Unificado de Desarrollo de Software. [En línea] [Citado el : 5 de Diciembre del 2009]. http://www.chaco.gov.ar/UTN/disenodesistemas/apuntes/oo/ApunteRUP.pdf**.**

[8] **Ingeniería de Software 2**. "Continuación del FT Análisis y Diseño. Modelo de Diseño." UCI. curso 2009_20010. Conferencia # 1.

[9] Diagrama de despliegue [En línea] [Citado el : 10 de Marzo del 2010]

http://virtual.usalesiana.edu.bo/web/practica/archiv/despliegue.doc

[10] Diagrama de Componentes [En línea] [Citado el : 15 de Marzo del 2010]

http://tvdi.det.uvigo.es/~avilas/UML/node49.html

[11] **Fowler, Martín.** *UML Gota a Gota.* 1999.

BIBLIOGRAFÍA

Ingeniería de Software 1. Fase de Inicio. Modelo del Negocio. UCI. curso 2009_2010. Conferencia # 2.

Ingeniería de Software 2. "Continuación del FT Análisis y Diseño. Modelo de Diseño." UCI. curso 2009_2010. Conferencia # 1.

Portal Proceso Unificado de Desarrollo de Software. [En línea] [Citado el : 5 de Diciembre del 2008].

http://www.chaco.gov.ar/UTN/disenodesistemas/apuntes/oo/ApunteRUP.pdf**.**

Portal "HelMakers". [En línea] [Citado el: 22 de enero de 2010.] http://helpmaker.uptodown.com

Portal "pixelco.us blog". [En línea] [Citado el: 7 de febrero de 2010.]

http://pixelco.us/blog/codeigniter-framework-php

Portal Diagrama de despliegue [En línea] [Citado el : 10 de Marzo del

2010] http://virtual.usalesiana.edu.bo/web/practica/archiv/despliegue.doc

Portal Diagrama de Componentes [En línea] [Citado el : 15 de Marzo del 2010]

http://tvdi.det.uvigo.es/~avilas/UML/node49.html

Portal "PHP Nuke". [En línea] [Citado el: 17 de Marzo de 2010.] http://www.adelat.org/media/docum/nuke_publico/lenguajes_del_lado_servidor_o_cliente.html

Portal de Desarrollo Web. [En línea] [Citado el: 22 de Marzo de 2010.] http://www.desarrolloweb.com/articulos/25.php

GLOSARIO DE TÉRMINOS

Aplicación Web: especialización de las aplicaciones Cliente/Servidor, están comúnmente estructuradas como una aplicación en tres capas, son conocidas como aplicaciones servidor.

Arquitectura Cliente/Servidor: es un modelo para el desarrollo de sistemas de información, en el que las transacciones se dividen en elementos independientes que cooperan entre sí para intercambiar información, servicios o recursos.

Casos de Usos (CU): es una secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema.

DHTML: Dynamic HTML/HTML Dinámico.

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

HTML: HyperText Markup Language/Lenguaje Hipertexto de Marcado. Lenguaje usado para escribir documentos para servidores World Wide Web. Es una aplicación de la ISO Standard 8879:1986. Es un lenguaje de marcas. Los lenguajes de marcas no son equivalentes a los lenguajes de programación aunque se definan igualmente como "lenguajes". Son sistemas complejos de descripción de información, normalmente documentos, que se pueden controlar desde cualquier editor ASCII.

HTTP: HyperText Transfer Protocol/ Protocolo de Transferencia de Hipertextos. Modo de comunicación para solicitar páginas Web.

Hardware: Componentes electrónicos, tarjetas, periféricos y equipo que conforman un sistema de computación; se distinguen de los programas (software) porque son tangibles.

Glosario de Términos

Informática: disciplina que estudia el tratamiento automático de la información utilizando dispositivos electrónicos y sistemas computacionales.

Internet: Sistema de redes de computación ligadas entre sí, con alcance mundial, que facilita servicios de comunicación de datos como registro remoto, transferencia de archivos, correo electrónico y grupos de noticias. Internet es una forma de conectar las redes de computación existentes que amplía en gran medida el alcance de cada sistema participante.

Linux: Es el nombre de un núcleo, pero se suele denominar con este nombre a un sistema operativo de libre distribución software libre (y de código abierto), donde el código fuente está disponible públicamente y cualquier persona, con los conocimientos informáticos adecuados, puede libremente estudiarlo, usarlo, modificarlo y redistribuirlo.

Ofrece una completa solución abierta para las tecnologías Web y estándares de hoy, incluyendo la accesibilidad y servicios Web.

Microsoft: Compañía que manufactura los sistemas de operación DOS y Windows.

MySQL: Es un sistema de gestión de bases de datos relacional que cuentan con todas las características de un motor de BD comercial: transacciones atómicas, triggers, replicación, llaves foráneas entre otras. Su ingeniosa arquitectura lo hace extremadamente rápido y fácil de personalizar.

MVC: Modelo Vista Controlador.

Paquete: mecanismo de propósito general para organizar elementos en grupos.

PHP: Hypertext Preprocessor/Preprocesador de Hipertexto. Es un lenguaje script del lado del servidor que permite crear y ejecutar aplicaciones Web dinámicas e interactivas. Con PHP se pueden combinar páginas HTML y scripts. Con el objetivo de crear aplicaciones potentes.

PostgreSQL: es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (ORDBMS) libre.

Glosario de Términos

RUP: Rational Unified Process (Proceso Unificado de Desarrollo). Metodología para el desarrollo de Software.

Software: es el conjunto de los programas de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de computación.

Software Libre: es el software que una vez copiado puede ser modificado, cambiado, mejorado y redistribuido libremente.

Sitio Web: es un conjunto de páginas web, típicamente comunes a un dominio de Internet o subdominio en la World Wide Web en Internet.

SGBD: Sistema de Gestión de Bases de Datos. Es el software que permite la utilización y/o la actualización de los datos almacenados en una (o varias) base(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez.

SOAP: Simple Object Access Protocol /Protocolo Estándar de Acceso a Objetos

UML: Unified Modeling Language/Lenguaje Unificado de Modelado. Es una notación estándar para modelar objetos del mundo real como primer paso en el desarrollo de programas orientados a objetos. Es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema de software.

WEB (WWW): red de documentos HTML intercomunicados y distribuidos entre servidores del mundo entero.