Universidad de las Ciencias Informáticas

Sistema de Gestión de Informatización de la Facultad 8. Módulo para la Gestión de Planificación Docente.

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

Autores:

Yoandrys Verdecia Álvarez

Yoisbel Hurtado Portal

Tutor:

Ing.Sandra Gutiérrez Secades

Ciudad de La Habana, Cuba
Junio, 2007

DECLARACIÓN DE AUTORÍA

Declaramos que Yoandrys Verdecia Álvarez y Yoisbel Hurtado Portal somos los únicos autores de este trabajo y autorizamos 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 firmamos la presente a los ____ días del mes de junio del 2007.

_____	_____
Firma del Autor	Firma del Autor
Yoandrys Verdecia Álvarez	Yoisbel Hurtado Portal

_____

Firma del Tutor

Ing. Sandra Gutiérrez Secades

Pensamiento

"¿Por qué esta magnífica tecnología científica, que ahorra trabajo y nos hace la vida mas fácil, nos aporta tan poca felicidad? La repuesta es ésta, simplemente: porque aún no hemos aprendido a usarla con tino."

Albert Einstein.

Agradecimientos

Nuestro más sagrado agradecimiento:

A nuestra Revolución y comandante en jefe Fidel por avernos dado esta posibilidad, por sentirnos más comprometidos con la Revolución.

A nuestra tutora, nuestros más sagrados agradecimientos por su calidez, sugerencias y confianza

A Maria Elena por su tiempo dedicado, por su comprensión, dedicación y apoyo incondicional.

Especialmente a nuestros padres quienes siempre nos enseñaron lo correcto

A todos nuestro mayor reconocimiento y gratitud……

Dedicatoria

Yoisbel

Creo firmemente que la vida nos regala muchas oportunidades, deseos y sueños.
A mis queridos padres regalo mi esfuerzo, mis alegrías y penas, ellos que con sólo su presencia me llenan de vida, alegrías y ganas de luchar cada día más, por ellos, por su bienestar y por mí.
A mis fieles amigos, les agradezco de corazón por estos años de estudio que han estado a mi lado; por sus palabras de aliento y por su apoyo en cada momento.
A DIOS, quien marca mi rumbo y mi vida, quien me da fortaleza y amor cada día, la claridad para seguir mejorando en esta sociedad de exigencias y metas.
A todos gracias por escuchar mis pensamientos, por comprender mis sueños, por convivir con mis defectos y por quererme durante todos estos años.

Yoandrys
Especialmente a quienes me enseñaron a vivir, soñar y caminar por el camino correcto, mis amados padres, Maria Esperanza Álvarez y Félix Gonzáles
A DIOS por haberme dado la posibilidad y salud para llegar a mi propósito.
A toda mi familia por su inmenso amor, comprensión, apoyo y por creer en mí.
A mis amigos, aquí donde todos estamos separados de la familia es donde estuvieron conmigo apoyándome y dándome fuerzas para seguir adelante.Gracias por permitirme ser parte de ustedes.
A quien es parte de mi en todo momento, mi novia Aliannys, por su amor, apoyo y compañía en cada etapa del camino recorrido juntos y, también en aquellos momentos difíciles
A mis más allegados compañeros, Ismael, Leonardo, Yoisbel, Pacheco, Daniel, Onel, Eduardo, Leyvis, Jakelin en fin a todos gracias por su amistad, comprensión y compañía.
A mis abuelos, por estar orgullosos de mí.

Gracias a todos!!
Gracias por ayudarme a lograrlo…….

Resumen

La planificación docente es en la actualidad un proceso complejo y amplio, donde día a día se encuentran situaciones difíciles y de soluciones engorrosas. Debido a lo anteriormente expresado y teniendo en cuenta que la UCI[1] no se encuentra exenta de dicha situación, se concibe el siguiente trabajo con la finalidad de automatizar la gestión de la planificación docente para la facultad 8 de este centro de estudios universitarios. El presente se ha centrado en el análisis, diseño e implementación de un sistema para mejorar todo lo relacionado con la planificación docente de la facultad 8 de la UCI[1] en cuanto a eficiencia se refiere, así como buscar diferentes alternativas para minimizar el tiempo de búsquedas de información, acceso a ella en cualquier momento y desde cualquier lugar dentro del alcance del sistema. El presente trabajo brinda una propuesta de aplicación Web para automatizar los principales procesos asociados con la planificación docente de la facultad. Para llevar a cabo este proyecto se siguieron los pasos que propone el Proceso Unificado de Desarrollo de Software, además se utilizaron algunos lenguajes y tecnologías, entre ellos, del lado del servidor php y del lado del cliente javaScript, ajax y CSS**;** para el almacenamiento de datos se puso en marcha un servidor de Postgres conjuntamente con la arquitectura de tres capas para el uso de la modelación del tipo Modelo-Vista Controlador.

[1]Universidad de las Ciencias Informáticas

INDICE

INTRODUCCIÓN

La planificación implica un proceso consciente de estudio y selección del mejor curso de acción a seguir, frente a una variedad de alternativas posibles y factibles de acuerdo a los recursos disponibles; no es más que la concepción anticipada de una actividad de acuerdo a una evaluación racional entre fines y medios que debe afrontarse de manera adecuada para que al final de la misma se pueda hablar de éxito. Cada vez que se intente prever un comportamiento futuro y se tomen las medidas necesarias, se está planificando.

Todo curso académico en la Universidad de las Ciencias Informáticas (UCI), posee, entre sus particularidades, la del incremento notable de la matrícula, lo que hace cada vez más difícil el proceso de control y atención de los estudiantes, así como de los profesores que cada año se incorporan a su claustro.

En el caso específico de la Facultad 8 se palpan también estas situaciones, por lo que se plantea la necesidad de implementación de un Módulo para la Planificación Docente, que, unido a otros, conforman el Sistema de Gestión de Información de la Facultad, facilitando así el trabajo de los implicados en los asuntos de la misma y elevando la calidad de la información de los estudiantes y profesores, de manera que pueda accederse a ella con mayor rapidez y eficiencia. Este sistema sirve como base de análisis para otros similares en la universidad, y en el país en general, pues hasta el momento todo se procesa manualmente, lo que trae consigo muchas agravantes. Además de agilizar el proceso, podrán conocerse los distintos indicadores de los estudiantes, tanto en la beca como en la docencia.

Situación Problémica

En toda universidad y por qué no, también en otro tipo de enseñanza, es conocido que el problema de la Planificación Docente que se manipula es complejo, debido a la planificación de horarios, locales, pruebas, preparación metodológicas, entre otras actividades; y con todo ello una serie de detalles que hace que el trabajo de la secretaria o planificadora docente sea engorroso y muy lento.

La Facultad 8 de la Universidad de las Ciencias Informáticas (UCI) presenta dicho problema, debido a la manipulación manual y almacenamiento en papeles de toda esta actividad; además de utilizar software como Microsoft Excel, Microsoft Word, entre otros, para el almacenamiento de manera digital, pero que no son los más eficientes para estos fines, pues trae consigo las siguientes agravantes:

• El cúmulo de información es cada vez más grande y llegará el momento en que se necesite obtener de manera más rápida la información referente a planificación de horarios, pruebas, locales, entre otros.
• La posibilidad de que todos puedan acceder a la información desde cualquier sitio no existe, pues la misma carece de una publicación en la red.
• Existe la posibilidad del deterioro o pérdida de los documentos físicos donde se encuentra información valiosa, por lo que sería engorrosa replanificarlo y obtenerlo en el tiempo requerido.

Problema Científico

De la situación problémica anteriormente planteada se deriva el siguiente problema científico:
¿Cómo eliminar las dificultades de la planificación docente de la facultad mediante la creación de un software de Gestión?

Aportes Prácticos

Los aportes prácticos esperados son:
• Centralización de toda la información referente a los procesos que se desarrollan en la planificación docente.
• Rapidez en la comunicación y en las búsquedas de información por parte del personal disminuyendo su carga de trabajo, debido a la automatización de los procesos que se realizan.
• Optimizar el anterior sistema de planificación docente de la facultad.

Objeto de Estudio

Por lo expresado anteriormente el objeto de estudio es el Proceso de Gestión de Planificación Docente en la Universidad de las Ciencias Informáticas.

Campo de Acción

Derivándose de lo anteriormente expresado que el campo de acción es sin duda alguna:
Proceso de gestión de planificación docente en la facultad 8 de la Universidad de las Ciencias Informáticas.

Hipótesis

Si se desarrolla un sistema de gestión para la planificación docente en la facultad entonces permitirá que todo el trabajo de planificación docente sea más rápido, eficiente y de una manera más sencilla y factible.

Objetivo General

Realizar el análisis, diseño e implementación de un sistema automatizado que gestione el proceso de planificación docente en la Facultad 8 de la Universidad de las Ciencias Informáticas.

Objetivos Específicos

- ✓ Procesar toda la información vinculada con la gestión de la planificación docente.
- ✓ Realizar Análisis, Diseño, Implementación y Prueba del Sistema de Planificación Docente propuesto.
- ✓ Confeccionar la documentación que recopile la información del Sistema de Gestión de Planificación Docente (SGPD) y que sirva de apoyo para posteriores versiones.

Tareas

Dentro de las tareas a realizar se platean las siguientes:

1- Estudio sobre el funcionamiento de las actividades docentes.

2- Documentación de las herramientas adecuadas, de acuerdo a la necesidad de los clientes, para la futura elaboración del sistema.

3- Indagación del modo de trabajo y la necesidad del lugar donde implemente la solución que se proponga.

4-Identificación de las necesidades de los clientes.

5-. Implementación del sistema y pruebas al mismo.

El presente trabajo, está estructurado en 5 capítulos, a continuación se muestra una breve descripción de cada uno de ellos:

Capítulo 1- Fundamentación Teórica: Describe interesantes cuestiones a tener en cuenta tales como: sistemas existentes con alguna similitud con el que se desea realizar, cuestiones teóricas necesarias para la comprensión del trabajo y el conocimiento de las posibles herramientas a utilizar, describiéndolas y abordando las principales características que fundamenten el porqué se escogieron.

Capítulo 2- Modelo del Negocio: Muestra una rica y detallada descripción del negocio actual, descripción de los procesos, actores, trabajadores, casos de uso del mismo, relación entre actores y procesos que se relacionan, diagramas de clases del modelo de objetos del negocio.

Capítulo 3- Requisitos: Define con alto grado de detalle lo que el sistema debe hacer realmente y características que debe de cumplir a través de los requisitos funcionales y no funcionales que se muestran en el capítulo, además describe los actores y casos de uso del sistema.

Capítulo 4- Análisis y Diseño: Define 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, además describe el análisis y diseño del sistema a través de diagramas y modelos; por ejemplo, para el análisis, el cual no es obligatoriamente necesario para pasar al diseño, pero sí importante por si en algún momento se desea cambiar para otro lenguaje, se realizó el diagrama de análisis de cada uno de los procesos del sistema a automatizar, además se muestra la realización de todo el diseño de este sistema, así como diagramas de clases Web.

Capítulo 5- Implementación y Prueba: 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. Además se implementa el producto y se realizan pruebas para buscar los defectos a los largo del ciclo de vida.

CAPÍTULO 1
Fundamentación Teórica

1.1 INTRODUCCION

En el presente capítulo se realiza una descripción del objeto de estudio y se muestran los procesos que serán objeto de automatización. Es significativo el análisis de los Sistemas automatizados existentes que estén vinculados al campo de acción de la problemática en cuestión. Recoge además las tendencias y tecnologías actuales sobre las cuales se apoya la siguiente propuesta.

1.2 Objeto de estudio

La Universidad de la Ciencias Informáticas es un centro docente - productivo formado por diez facultades donde el objeto de estudio es la facultad 8.

1.2.1 Objetivos estratégicos de la organización

La Universidad de la Ciencias Informáticas es una institución universitaria para la formación integral y continua de profesionales comprometidos con la patria, los ideales y principios de la Revolución, portadores de elevados valores socio - humanistas que garanticen la continuidad del socialismo. Desarrolla investigaciones científicas de relevancia nacional e internacional en la rama de la informática, promoviendo con la utilización de las tecnologías apropiadas, el desarrollo sostenible de la sociedad en un ambiente participativo y de mutuo compromiso de trabajadores y estudiantes, con una destacada labor extensionista y de promoción en la sociedad cubana.

Es espacio de reflexión, creación de conocimientos científicos y tecnológicos y de formación de valores, para coadyuvar a la continuidad histórica de la Revolución Cubana y al enriquecimiento cultural de la sociedad, desempeñando un papel principal en el desarrollo, difusión y aplicación de las ciencias informáticas en la formación integral y continúa de los profesionales.

La ejecución de los procesos docente, investigativo y productivo permiten el cumplimiento de los objetivo y misión de la facultad, tales como la preparación de los estudiantes a nivel

universitario, la adquisición de conocimientos suficientes para enfrentarse a la investigación y a la producción de software.

1.2.2 Flujo actual de los procesos

El análisis del flujo de procesos permite reconocer como funciona realmente el negocio para producir uno o varios resultados. El resultado puede ser un producto, un servicio, una información o combinaciones de ellos. Analizar el flujo de los procesos permite revelar problemas potenciales tales como: los cuellos de botella, los pasos innecesarios, la circulación doble de la información, la duplicación del trabajo, solo por citar algunos.
La Facultad tiene un flujo que se conforma de los procesos que a continuación se mencionan:

- **Gestión administrativa**

Son aquellas que contienen una serie de funcionalidades que controla el personal administrativo de la facultad tales como:

- **Docente**

Planificador: Recibe el listado de los locales, p1, y de los profesores con sus respectivas afectaciones, posteriormente realiza la distribución de los mismos conforme a la planificación de los turnos de clases para cada grupo asignándole el local donde ha de recibir la actividad docente. Esta información la almacena en un documento digital (Excel) y lo envía a los profesores y a los estudiantes.
Si existen cambios, debido a algún incidente por enfermedad, cruces de horarios u otro acontecimiento extraordinario, los profesores, en previo acuerdo con los estudiantes, avisa al Planificador y se ponen de acuerdo eligiendo un día para dar la clase perdida y este último reajusta el horario y lo reenvía a los estudiantes. Los profesores también le informan los días que van a realizar la consulta para que entonces se designe el aula, por parte del Planificador, donde se va a efectuar la misma y el horario y esta información se le hará llegar oficialmente a los estudiantes. El Planificador también tiene conocimiento de información tal como los profesores de la facultad, departamento al que pertenece, asignatura que imparte, si es profesor guía o no, y de serlo, grupo al que atiende, entre otros detalles.

1.2.3 Análisis crítico de la ejecución de los procesos

A pesar de que el funcionamiento en la facultad de los procesos actuales no se interrumpe, no es tan eficiente como pudiera pues, por ejemplo, se pierde tiempo, al planificar en los documentos Excel con las informaciones que se necesitan para la estructura general del horario, así como al buscar las distintas informaciones de profesores como los p1 y afectaciones y al saber el horario de un grupo determinado. Esta subutilización de la tecnología que está al alcance del personal trae como consecuencia una demora innecesaria en el cumplimiento de las tareas de algunos trabajadores.

1.3 Procesos objeto de automatización

Los procesos que se desean automatizar son todos los que intervienen de una forma u otra en la facultad, que tienen que ver con la planificación docente.

1.4 Sistemas automatizados existentes vinculados al campo de acción

Muchas son las personas, instituciones y empresas que se interesan por la gestión de planificación y específicamente por la docente. Algunos de los productos y sistemas realizados que podemos encontrar que de una manera u otra le dan solución a problemas de esta índole son los mencionados a continuación:

Un reconocido sistema Web, Sistema Generador de Horarios (**SIGEH**), que brinda servicio en la universidad autónoma de baja California, para la Facultad de Ciencias facilitando la automatización de los horarios de diferentes carreras [0].

Dentro del amplio mundo de aplicaciones de escritorio, existe también diversidad de ejemplares por ejemplo:

- Generador de Horarios para Centros Docentes (**GENHOR**) este software se puede descargar, e instalar posteriormente brindando servicios que ahorran tiempo para la planificación de horarios escolares [1].
- Generador de Horarios para Centros de Enseñanza (**GHC**) es un software gratis que, facilita la organización, confección, edición y transferencia de horarios escolares semanales. Su principal característica es un potente motor que genera resultados óptimos. Puede utilizarse en Institutos de Educación Secundaria, colegios, centros de

Formación Profesional, Educación infantil y Primaria, Escuelas de Arte, Superiores, etc. Con la limitación de solo para 10 profesores [2].

- Generador de horarios (KRONOWIN-M-6) interesante software generador de horario y con algunas otras funcionalidades docentes de interés, creado por la empresa (ADOSSIS, S.A.) Sistemas Informáticos [3]
- Cronos, es un sistema generador de horarios de clase para las Unidades Educativas de Educación Básica y Diversificado [4].

1.5 Tendencias y tecnologías actuales

Teniendo en cuenta las necesidades evidenciadas en la situación problémica y las características que presenta el entorno en el que se pondrá en práctica la solución propuesta, se realizó un estudió de las tendencias y tecnologías actuales posibles a emplear, lo cual se describe a continuación.

1.5.1 Metodologías para el desarrollo del Software

- **RUP**

Cada día la producción de software busca adecuarse más a las necesidades del usuario. Para lograr la productividad del software se necesita un proceso que integre las múltiples facetas del desarrollo del mismo. Se hace necesario definir la metodología de ingeniería del software que guiará el proceso de automatización, se ha escogido el Proceso Unificado de Desarrollo de Software (RUP).

El Proceso Unificado de desarrollo del software, (Rational Unified Process, de ahí las siglas RUP), fue publicado en 1998 como resultado de varios años de experiencia (Rational Unified Process), 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. Sus principales características son**:**

- Forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo).
- Implementa las mejores prácticas en Ingeniería de Software, como son:

- Desarrollo iterativo
- Administración de requisitos
- Uso de arquitectura basada en componentes
- Control de cambios
- Modelado visual del software
- Verificación de la calidad del software

Además RUP es un producto de Rational (IBM), que se caracteriza por ser iterativo e incremental, centrado en la arquitectura y guiado por casos de uso. Incluyendo artefactos (que son los productos tangibles del proceso como por ejemplo, el modelo de casos de uso, el código fuente, entre otros) y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso).También divide el proceso de desarrollo en ciclos, teniendo un producto final al terminar cada ciclo, los cuales se dividen en fases que finalizan con un hito. Las fases fundamentales son:

- inicio: se hace un plan de fases, se identifican los principales casos de uso y se identifican los riesgos.
- elaboración: se hace un plan de proyecto, se completan los casos de uso y se eliminan los riesgos.
- construcción: se concentra en la elaboración de un producto totalmente operativo, eficiente y el manual de usuario.
- transición: se implementa el producto en el cliente y se entrena a los usuarios. Como consecuencia de esto suelen surgir nuevos requisitos a ser analizados [12].

- **Extreme Programing (XP)**

La programación extrema o Extreme Programming (XP) es un enfoque de la ingeniería de software formulado por Kent Beck. XP es la más destacada de los procesos ágiles de desarrollo de software. Es una de las metodologías de desarrollo de software más exitosas en la actualidad utilizada para proyectos de corto plazo La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. Las características fundamentales de esta metodología se basan en:

Desarrollo iterativo e incremental: Pequeñas mejoras, unas tras otras.

Pruebas Unitarias: Se basa en las pruebas realizadas a los principales procesos, de tal manera que adelantándonos en algo hacia el futuro, podamos hacer pruebas de las fallas que pudieran ocurrir. Es como si nos adelantáramos a obtener los posibles errores.

Refabricación: Se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.

Programación en pares: La cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento.

Frecuente interacción del equipo de programación con el cliente o usuario: Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.

Corrección de todos los errores antes de añadir nueva funcionalidad: Hacer entregas frecuentes.

Refactorización del código: Reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento.

Propiedad del código compartida: Este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto.

Simplicidad en el código: Es la mejor manera de que las cosas funcionen. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

La programación extrema propone empezar en pequeño y añadir funcionalidad con retroalimentación continua, manejar cambios que convierten en partes sustantivas el proceso, así como el costo del cambio no depende de la fase o etapa del proyecto, no se introducen funcionalidades antes si no son necesarias , los clientes y los usuarios se convierten en miembros del equipo. Además lo fundamental en este tipo de metodología es [13]:

- La comunicación, entre los usuarios y los desarrolladores.
- La simplicidad, al desarrollar y codificar los módulos del sistema.
- La retroalimentación, concreta y frecuente del equipo de desarrollo, el cliente y los usuarios finales.

1.5.2 Lenguajes de Modelados

El lenguaje de modelado de objetos es un conjunto estandarizado de símbolos y de modos de disponerlos para modelar (parte de) un diseño de software orientado a objetos que definen con gran calidad un mejor entendimiento del software por estas razones se escogió el Lenguaje unificado de Modelado (UML).

El Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; aún cuando todavía no es un estándar oficial, está apoyado en gran manera por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. UML es un "lenguaje" para especificar y no un método o un proceso, se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir. Es el lenguaje en el que está descrito el modelo. Se puede aplicar en una gran variedad de formas para soportar una metodología de desarrollo de software (tal como el Proceso Unificado de Rational) .Pero no especifica en sí mismo qué metodología o proceso usar. El UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas y proporciona un estándar que permite al analista de sistemas generar un anteproyecto de varias facetas que sean comprensibles para los clientes, desarrolladores y todos aquellos que estén involucrados en el proceso de desarrollo de forma general las principales características son:

• Lenguaje unificado para la modelación de sistemas.
• Tecnología orientada a objetos.
• El cliente participa en todas las etapas del proyecto.
• Corrección de errores viables en todas las etapas.
• Aplicable para tratar asuntos de escala inherentes a sistemas complejos de misión crítica, tiempo real y cliente/servidor [14]

1.5.3 Herramientas Case

Existen varias herramientas CASE (Computer-Aided Systems Engineering), que dan asistencia a analistas, ingenieros de software y desarrolladores durante el ciclo de vida de desarrollo de un software, pero es Rational Rose líder en el modelado del desarrollo de los proyectos y es esta precisamente la que se utiliza en la modelación de este proyecto. La herramienta fue desarrollada por los creadores de UML, utilizando la notación estándar en la arquitectura de software. Esta herramienta integra todos los elementos que propone La metodología RUP para cubrir el ciclo de vida de un proyecto.

El Rational Rose es la herramienta CASE desarrollada por los creadores de UML (Booch, Rumbaugh y Jacobson), que cubre todo el ciclo de vida de un proyecto: Concepción, formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases entregables.

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

Esta herramienta propone la utilización de cuatro tipos de modelos para realizar el diseño del sistema, utilizando una vista estática y otra dinámica de los modelos del sistema, uno lógico y otro físico. Permite crear y refinar estas vistas creando de esta forma un modelo completo que representa el dominio del problema y el sistema de software. Rational Rose utiliza un proceso de desarrollo iterativo controlado (controlled iterative process development), donde el desarrollo se lleva a cabo en una secuencia de iteraciones.

Cada iteración comienza con una primera aproximación del análisis, diseño e implementación para identificar los riesgos del diseño, los cuales se utilizan para conducir la iteración, primero se identifican los riesgos y después se prueba la aplicación para que estos se hagan mínimos.

Cuando la implementación pasa todas las pruebas que se determinan en el proceso, esta se revisa y se añaden los elementos modificados al modelo de análisis y diseño. Una vez que la actualización del modelo se ha modificado, se realiza la siguiente iteración. Rational Rose permite que haya varias personas trabajando a la vez en el proceso iterativo controlado, para ello posibilita que cada desarrollador opere en un espacio de trabajo privado que contiene el modelo completo y tenga un control exclusivo sobre la propagación de los cambios en ese espacio de trabajo. También es posible descomponer el modelo en unidades controladas e integrarlas con un sistema para realizar el control de proyectos que permite mantener la integridad de dichas unidades [15].

1.5.4 Lenguajes de Programación y Tecnologías

- **Lenguajes de Programación y Tecnologías del lado del Cliente**

El HTML Dinámico (DHTML) no es más que, en pocas palabras, una forma que tienen las páginas de aportar interactividad a las mismas. DHTML es una característica de Netscape Communicator 4.0, y Microsoft Explorer 4.0 y posteriores versiones de ambos navegadores, y está orientada al usuario. Es tarea del navegador mostrar y manipular las páginas Web.
El DHTML tiene la ventaja de que es una herramienta con la que se pueden crear efectos que requieren poco ancho de banda, a la hora de bajarlos de Internet y, son estos efectos los que aumentan la funcionalidad de la página. Se puede utilizar para crear animaciones, juegos, aplicaciones, para introducir nuevas formas de navegar a través de los sitios Web, y para crear un auténtico entramado de capas que con sólo el HTML sería imposible abordar. Aunque muchas de las características del DHTML se podrían duplicar con otras herramientas como Java o Flash, el DHTML ofrece la ventaja de que no requiere ningún tipo de plug-in para poder utilizarlo [7].

Javascript: Es un lenguaje de programación utilizado para crear pequeños algoritmos, funciones, etc., encargados de realizar acciones dentro del ámbito de una página Web.
Se trata de un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento. Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado.
Javascript es el siguiente paso, después del HTML, que puede dar un programador de la Web que decida mejorar sus páginas y la potencia de sus proyectos. Es un lenguaje de

programación bastante sencillo y pensado para hacer las cosas con rapidez, a veces con ligereza. Incluso las personas que no tengan una experiencia previa en la programación podrán aprender este lenguaje con facilidad y utilizarlo en toda su potencia con sólo un poco de práctica [7].

CSS: Es una tecnología que nos permite crear páginas Web de una manera más exacta. Gracias a las CSS somos mucho más dueños de los resultados finales de la página, pudiendo hacer muchas cosas que no se podía hacer utilizando solamente HTML, como incluir márgenes, tipos de letra, fondos, colores.
CSS son las siglas de Cascading Style Sheets, en español Hojas de estilo en Cascada. En este reportaje vamos a ver algunos de los efectos que se pueden crear con las CSS sin necesidad de conocer la tecnología entera [7].

AJAX: (Asynchronous Javascript and XML) traducido como indica el título, no es más que una forma de programar aplicaciones interactivas para Web. Esta evolución de DHTML se la ha denominado Web 2.0. Para ello utiliza XHTML y CSS para formatear la información, DOM (Document Object Model) para interactuar y visualizar dinámicamente la información, se apoya en XML, XSTL para manipular la información mostrada, el objeto XMLHttpRequest (no estándar) y Javascript para actualizar los datos sin necesidad de refrescar la página, y para manipular todas esas tecnologías. Ajax no es una tecnología en si misma, sino un término que se refiere al uso de un grupo de tecnologías [8]
Ventajas:

- Rapidez en las operaciones.
- Más cerca de crear realmente "Aplicaciones Web".
- Menos carga del servidor (menos transferencia).
- Menos ancho de banda (nos podemos ahorrar mucho dinero si tenemos muchas visitas) [8].

Desventajas:

- Demasiado código Ajax hace lento al navegador.
- No poder recomendar links específicos.
- Cambiando el estado de los links (GET)
- Falta de Integración con el Botón de Retroceder Página del Navegador.
- No dar señales inmediatas que los links se están cargando. [8].

Navegadores que no permiten AJAX:

- Opera 7 y anteriores
- Microsoft Internet Explorer para Windows versión 4.0 y anteriores
- Microsoft Internet Explorer para Macintosh, todas las versiones
- Dillo
- Navegadores basados en texto como Lynx y Links
- Navegadores para incapacitados visuales (braille) [8]

1.5.5 Lenguajes de Programación y tecnologías del lado del Servidor

Existe una multitud de lenguajes concebidos o no para Internet .Cada uno de ellos explota más a fondo ciertas características que lo hacen más o menos útiles para desarrollar distintas aplicaciones y además están en dependencia de lo que quiera hacer el cliente con el. En el dominio de la red, los lenguajes del lado del servidor más ampliamente utilizados para el desarrollo de páginas dinámicas son el ASP, PHP, PERL y JAVA.

PHP: (Hypertext Prepocessor), es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor.
Una de sus características más potentes es su soporte para gran cantidad de bases de datos. Entre las que se pueden mencionar InterBase, mSQL, MySQL, Oracle, Informix, PostgreSQL, entre otras. PHP también ofrece la integración con varias bibliotecas externas, que dan al desarrollador la posibilidad de realizar cualquier tarea, desde generar documentos en pdf (Portable Document Format) hasta analizar código XML (extensible Markup Language) y últimamente también para la creación de otro tipo de programas incluyendo aplicaciones con interfaz gráfica usando la librería GTK+.
Es software libre, lo que implica menos costos y servidores más baratos que otras alternativas. Es muy rápido y su integración con la base de datos MySQL y el servidor Apache, le permite constituirse como una de las alternativas más atractivas del mercado.

Es multiplataforma, funciona tanto para Unix (con Apache) como para Windows (con Microsoft Internet Información Server) de forma que el código que se haya creado para una de ellas no tiene por qué modificarse al pasar a la otra.

Su sintaxis está inspirada en C, ligeramente modificada para adaptarlo al entorno en el que trabaja, de modo que si se está familiarizado con esta sintaxis, le resultará muy fácil aprender PHP. Su librería estándar es realmente amplia, lo que permite reducir los llamados "costes ocultos", uno de los principales defectos de ASP (Active Server Pages).

PHP tiene una de las comunidades más grandes en Internet, con lo que no es complicado encontrar ayuda, documentación, artículos, noticias, y más recursos. Ofrece una solución simple y universal para las paginaciones dinámicas del Web de fácil programación. Su diseño elegante lo hace perceptiblemente más fácil de mantener y ponerse al día, a diferencia con el código de otros lenguajes.
Debido a su amplia distribución PHP esta perfectamente soportado por una gran comunidad de desarrolladores. Como producto de código abierto, PHP goza de la ayuda de un gran grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y se reparan rápidamente. El código se pone al día continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP.

Otros lenguajes como Perl (Practical Extraction and Report Language), ASP (Active Server Pages) y JSP (Java Server Pages) tienen características similares al PHP aunque poseen rasgos que los marcan y por ello los distingue, entre ellos podemos encontrar:

- ✓ Velocidad de ejecución: La velocidad es mayor en PHP, seguidos por PERL y JSP.
- ✓ Disponibilidad de recursos: Actualmente los más utilizados en la Internet son el PHP y el JSP, siendo más utilizado en la publicación de artículos y códigos de ejemplos. PHP tiene una de las comunidades más grandes en Internet, al igual que la de Java.
- ✓ Familiaridad con el lenguaje: En las universidades los lenguajes más utilizados por los programadores es el ASP y el PHP [7].

1.5.6 Sistemas Gestores de Base de Datos (SBGD)

El software de bases de datos ha experimentado un auge extraordinario a raíz de la progresiva informatización de casi la totalidad de las empresas de hoy día, rapidez, efectividad en los procesos y los grandes flujos de información son la necesidad más apremiante a la hora de optimizar servicios y productos.

Ante esta notable demanda de soluciones informáticas han surgido multitud de gestores de bases de datos, estos son programas que permiten manejar la información de modo sencillo y que prestan servicios para el desarrollo y el manejo de bases de datos [16].

Objetivos de un SBGD

En un ambiente multiusuario el SGBD ofrece un control centralizado de la información. Los objetivos que plantean estos sistemas están relacionados con la intención de evitar los problemas que existían en los sistemas de información orientados a los procesos. Los principales objetivos son:

- Evitar la redundancia de los datos, eliminando así la inconsistencia de los mismos.
- Mejorar los mecanismos de seguridad de los datos y la privacidad. Podemos distinguir cuatro tipos de contextos para usar mecanismos de seguridad: seguridad contra accesos indebidos a los datos, seguridad contra accesos no autorizados a la base de datos, seguridad contra destrucción causada por el entorno (fuego, inundación, robo, etc.), seguridad contra fallos del propio sistema (fallos del hardware, del software, etc.).
- Asegurar la independencia de los programas y los datos, es decir, la posibilidad de modificar la estructura de la base de datos (esquema) sin necesidad de modificar los programas de las aplicaciones que manejan esos datos.
- Mantener la integridad de los datos realizando las validaciones necesarias cuando se realicen modificaciones en la base de datos.
- Mejorar la eficacia de acceso a los datos, en especial en el caso de consultas imprevistas.

Funciones de los SGBD

Las principales funciones que debe realizar un SGBD son:

- La definición de los datos.
- La manipulación de los datos.
- Garantizar la seguridad e integridad de los datos.
- La gestión de las transacciones y el acceso concurrente.

Rendimiento

Para la medida del rendimiento existen monitores del SGBD que van encaminados a:
Averiguar los recursos de la máquina que se están consumiendo, proporcionan tiempos específicos del sistema gestor. Hay la posibilidad de realizar programas específicos que midan exactamente el conjunto de características deseado. Estos programas se basan, en la anotación de la hora del sistema antes y después de realizar las operaciones cuyo rendimiento se quiere medir [16].

Por todos los aspectos vistos anteriormente vimos que con respecto a nuestras necesidades se escogió como sistema de gestión de bases de datos a:

PostgreSQL: Sistema gestor que posee una serie de características positivas respecto a otros. Por ejemplo:
• Posee una gran escalabilidad. Es capaz de ajustarse al número de procesadores y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta (en algunos benchmarks se dice que ha llegado a soportar el triple de carga de lo que soporta MySQL).
• Implementa el uso de rollbacks, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz.
• Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos.
• Multiusuario, multiprogramado, con arquitectura cliente-servidor y control de privilegios de acceso.
Además, como dice un Ínternauta, “Yo creo que los logos que han elegido para cada uno son bastante ilustrativos. PostgreSQL es un elefante: muy robusto y que aguanta una tremenda carga. MySQL es el delfín: bonito, rápido, estilizado, pero con sus limitaciones.” [17]

Las principales mejoras en PostgreSQL incluyen:

• Los bloqueos de tabla han sido sustituidos por el control de concurrencia multi-versión, el cual permite a los accesos de sólo 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 [18].

Las características fundamentales de Postgres son:

- DBMS Objeto-Relacional
  PostgreSQL aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multi-versión, soporte multi-usuario, transactions, optimización de consultas, herencia, y arrays.
- Altamente_Extensible
  PostgreSQL soporta operadores, funcionales métodos de acceso y tipos de datos definidos por el usuario.
- Integridad Referencial
  PostgreSQL soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos.
- API Flexible
  La flexibilidad del API de PostgreSQL ha permitido a los vendedores proporcionar soporte al desarrollo fácilmente para el RDBMS PostgreSQL. Estas interfaces incluyen Object Pascal, Python, Perl, PHP, ODBC, Java/JDBC, Ruby, TCL, C/C++, y Pike.
- Lenguajes Procedurales
  PostgreSQL tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL. Otra ventaja de PostgreSQL es su habilidad para usar Perl, Python, o TCL como lenguaje procedural embebido.

- MVCC

  MVCC, o Control de Concurrencia Multi-Versión (Multi-Versión Concurrency Control), es la tecnología que PostgreSQL usa para evitar bloqueos innecesarios. Si alguna vez ha usado algún DBMS con capacidades SQL, tal como MySQL o Access, probablemente habrá notado que hay ocasiones en las una lectura tiene que esperar para acceder a información de la base de datos. La espera está provocada por usuarios que están escribiendo en la base de datos. Resumiendo, el lector está bloqueado por los escritores que están actualizando registros.

  Mediante el uso de MVCC, PostgreSQL evita este problema por completo. MVCC está considerado mejor que el bloqueo a nivel de fila porque un lector nunca es bloqueado por un escritor. En su lugar, PostgreSQL mantiene una ruta a todas las transacciones realizadas por los usuarios de la base de datos. PostgreSQL es capaz entonces de manejar los registros sin necesidad de que los usuarios tengan que esperar a que los registros estén disponibles.
- Cliente/Servidor

  PostgreSQL usa una arquitectura proceso-por-usuario cliente/servidor. Esta es similar al método del Apache 1.3.x para manejar procesos. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL.
- Write Ahead Logging (WAL)

  La característica de PostgreSQL conocida como Write Ahead Logging incrementa la dependencia de la base de datos al registro de cambios antes de que estos sean escritos en la base de datos. Esto garantiza que en el hipotético caso de que la base de datos se caiga, existirá un registro de las transacciones a partir del cual podremos restaurar la base de datos. Esto puede ser enormemente beneficioso en el caso de caída, ya que cualesquiera cambios que no fueron escritos en la base de datos pueden ser recuperados usando el dato que fue previamente registrado. Una vez el sistema ha quedado restaurado, un usuario puede continuar trabajando desde el punto en que lo dejó cuando cayó la base de datos.

MySQL: Es un sistema de gestión de bases de datos relacional, licenciado bajo la GPL de la GNU. Su diseño multihilo le permite soportar una gran carga de forma muy eficiente. MySQL

fue creada por la empresa sueca MySQL AB, que mantiene el copyright del código fuente del servidor SQL, así como también de la marca.

Aunque MySQL es software libre, MySQL AB distribuye una versión comercial de MySQL, que no se diferencia de la versión libre más que en el soporte técnico que se ofrece, y la posibilidad de integrar este gestor en un software propietario, ya que de no ser así, se vulneraría la licencia GPL.

Este gestor de bases de datos es, probablemente, el gestor más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración.

Sus principales Características son:

- Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multihilo.
- Gran portabilidad entre sistemas.
- Gestión de usuarios y contraseñas, manteniendo un buen nivel de seguridad en los datos.
- Soporta gran cantidad de tipos de datos para las columnas.
- Dispone de API's en gran cantidad de lenguajes (C, C++, Java, PHP, etc.).
- Soporta hasta 32 índices por tabla.
- Gestión de usuarios y passwords, manteniendo un muy buen nivel de seguridad en los datos.
- Es rapidísimo en búsqueda de BD.
- Multiusuario, multiprogramado, con arquitectura cliente-servidor y control de privilegios de acceso [19].

1.5.7 Servidores Web

Apache es un servidor Web desarrollado por el grupo Apache Según estudios realizados por diferentes empresas el servidor Web más utilizado en Internet es Apache. Los binarios y el código fuente de Apache se pueden usar y distribuir de forma libre y en las condiciones mencionadas en la licencia anterior [9].

Orígenes de Apache: Originalmente Apache eran una serie de parches para el servidor Web de la NCSA (National Center for Supercomputing Applications).
Estos parches fueron creados por webmasters para el servidor Web de la NCSA. Después de que se abandonara el desarrollo de dicho servidor varios de ellos se pusieron de acuerdo para coordinar sus esfuerzos y crear un único servidor Web [9].
Plataformas para las que está disponible Apache:
Apache está disponible para una gran multitud de plataformas [9]:

- FreeBSD, NetBSD, OpenBSD,...
- GNU/Linux
- Mac OS y Mac OS X Server
- Netware
- OpenStep/Match
- UNIX comerciales como AIX (R), Digital UNIX (R), HP-UX (R),IRIX (R), SCO (R), Solaris (R), SunOS (R), UnixWare (R)
- Windows (R)

Características de Apache [9]
Como ya hemos visto Apache funciona en casi todas las plataformas actuales. Debido a esto podemos escoger la plataforma que más se adapte a nuestras características, y también podemos cambiar de plataforma si en un momento determinado una plataforma nos ofrece más ventajas que la que estemos utilizando.
Algunas de las características más importantes que debemos de tener en cuenta son:
Permite la autenticación de usuarios en varias formas.
Permite el uso de bases de datos DBM Para la autenticación de usuarios. De esta forma se puede restringir el acceso a determinadas páginas de un sitio Web de una forma sencilla y de fácil mantenimiento.
Respuestas personalizadas ante errores del servidor.
Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor.
Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto.
Creación de contenidos dinámicos
Apache permite la creación de sitios Web dinámicos mediante:

- El USO de CGI's.

- El uso de Server Side Includes (SSI).
- El uso de lenguajes de Scripting como PHP, javascript, Python.
- El uso de Java y páginas jsp.
- Alta configurabilidad en la creación y gestión de logs
- Apache permite la creación de ficheros de log a medida del administrador.
- Apache utiliza el formato Common Log Format (CLF) para la generación de los logs de error. Este formato es usado por varios servidores Web y existen herramientas para el análisis de ficheros con este formato
- Gran escalabilidad

Se pueden extender las características de Apache hasta donde nuestra imaginación y conocimientos lleguen.
Apache soporta Dinamic Shared Object Gracias a ello se pueden construir módulos que le den nuevas funcionalidades que son cargadas en tiempos de ejecución.

1.6 Programación por capas

La programación por capas es un estilo de programación en la que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño, un ejemplo básico de esto es separar la capa de datos de la capa de presentación al usuario.

La ventaja principal de este estilo, es que el desarrollo se puede llevar a cabo en varios niveles y en caso de algún cambio sólo se ataca al nivel requerido sin tener que revisar entre código mezclado. Un buen ejemplo de este método de programación seria: Modelo de interconexión de sistemas Abiertos.

Además permite distribuir el trabajo de creación de una aplicación por niveles, de este modo, cada grupo de trabajo está totalmente abstraído del resto de niveles, simplemente es necesario conocer la API que existe entre niveles.

En el Diseño de sistemas informáticos actual se suele usar la arquitectura multinivel o Programación por capas. En dichas arquitecturas a cada nivel se le confía una misión simple, lo que permite el diseño de arquitecturas escalables (que pueden ampliarse con facilidad en caso de que las necesidades aumenten).

El diseño que más en boga actualmente es el diseño en tres niveles (o en tres capas).

Capas o niveles

- Capa de presentación: Es la que ve el usuario (hay quien la denomina "capa de usuario"), presenta el sistema al usuario, le comunica la información y captura la información del usuario dando un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). Esta capa se comunica únicamente con la capa de negocio.
- Capa de negocio: Es donde residen los programas ejecutables, recibiendo las peticiones del usuario y enviando las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) pues es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos para almacenar o recuperar datos de él. [20].
- Capa de datos: Es donde residen los datos. Está formada por uno o más gestor de bases de datos que realiza todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio. [20]

Todas estas capas pueden residir en un único ordenador (no sería lo normal), si bien lo más usual es que haya una multitud de ordenadores donde reside la capa de presentación (son los clientes de la arquitectura cliente/servidor). Las capas de negocio y de datos pueden residir en el mismo ordenador, y si el crecimiento de las necesidades lo aconseja se pueden separar en dos o mas ordenadores. Así, si el tamaño o complejidad de la base de datos aumenta, se puede separar en varios ordenadores los cuales recibirán las peticiones del ordenador en que resida la capa de negocio.

Si por el contrario fuese la complejidad en la capa de negocio lo que obligase a la separación, esta capa de negocio podría residir en uno o más ordenadores que realizarían solicitudes a una única base de datos. En sistemas muy complejos se llega a tener una serie de ordenadores sobre los cuales corre la capa de datos, y otra serie de ordenadores sobre los cuales corre la base de datos. [20]

En una arquitectura de tres niveles, los términos "capas" y "niveles" no significan lo mismo ni son similares.
El término "capa" hace referencia a la forma como una solución es segmentada desde el punto de vista lógico:

Presentación/ Lógica de Negocio/ Datos.
En cambio, el término "nivel", corresponde a la forma en que las capas lógicas se encuentran distribuidas de forma física. Por ejemplo:

Una solución de tres capas (presentación, lógica, datos) que residen en un solo ordenador (Presentación+lógica+datos). Se dice, que la arquitectura de la solución es de tres capas y un nivel.
Una solución de tres capas (presentación, lógica, datos) que residen en dos ordenadores (presentación+lógica, lógica+datos). Se dice que la arquitectura de la solución es de tres capas y dos niveles.

Una solución de tres capas (presentación, lógica, datos) que residen en tres ordenadores (presentación, lógica, datos). La arquitectura que la define es: solución de tres capas y tres niveles.

1.7 El patrón Modelo-Vista-Controlador (MVC)

El patrón Modelo-Vista-Controlador (MVC) descompone una aplicación interactiva en tres grandes bloques:
El modelo contiene los datos y la funcionalidad de la aplicación. Es independiente de la representación de los datos.
Los objetos de negocio son los objetos del Modelo que implementan la lógica de negocio. Sus funciones fundamentales son:

- Realizar la validación de los datos introducidos por el usuario, tanto sintáctica (numérico, fecha, etc.) como lógica (importe menor que saldo, etc.)
- Ejecutar la petición realizada por el usuario. Para ello podrá valerse de transacciones contra Sistemas Host, consultas a bases de datos, consultas a proveedores de contenidos, etc.

- Generar los objetos que utilizarán las Vistas para mostrar los resultados obtenidos.
- Garantizar la navegación y el flujo de pantallas correcto.

Las vistas muestran la información al usuario de una cierta forma. Existen todas las que se necesite definir, es la encargada de interaccionar con el usuario y se corresponde con lo que tradicionalmente se conoce como Interfaz de Usuario.

Para este caso las vistas serán del tipo JSP. Las JSP's son las encargadas de generar las páginas HTML que constituyen la visualización de los resultados generados por la petición realizada por el usuario. Para recibir la información que deben mostrar acceden a objetos generados por el Modelo. La recepción de la información se puede hacer de varias maneras, aunque las más habituales son:

- Accediendo a objetos almacenados dentro del Modelo como propiedades.
- El Modelo almacena los resultados como atributos de la petición.
- El Modelo escribe los resultados en la "sesión".

Cada vista tiene un controlador asociado. Los controladores reciben entradas en forma de eventos que responden a mandos realizados por el usuario a través del ratón o del teclado. El control traduce estos eventos a peticiones a la vista o al modelo.

Cada petición se identifica mediante un parámetro. En base a esta identificación, el Controlador decide qué objeto u objetos de negocio (Modelo) debe ejecutar para resolver la petición.

Tras la ejecución de los objetos de negocio, y en función del resultado devuelto por estos, el Controlador determina qué JSP se usará para visualizar el resultado, generando una redirección que concluirá con la generación del código HTML de la página [10].

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 [11].

- 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 [11].

1.8 CONCLUSIONES

Propuesta de tendencias y tecnologías a utilizar:

Después de haber mostrado las tecnologías y herramientas que hoy enriquecen y abarcan a la informática, se llego a la conclusión que se utilizará como metodología para el desarrollo del software RUP debido a que presenta una forma muy disciplinada de asignar tareas y responsabilidades, además de implementar las mejores prácticas de ingeniería como son desarrollo iterativo, administración de requisitos, control de cambio, y verificación de la calidad del software. RUP se caracteriza por ser iterativo e Incremental, centrado en la arquitectura y guiado por casos de uso lo que permite una mayor organización del trabajo. Para la modelación se propone UML por ser un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software que tiene en cuenta características como son: lenguaje unificado para la modelación de sistemas, tecnología orientada a objetos, el cliente participa en todas las etapas del proyecto, corrección de errores viables en todas las etapas, aplicable para tratar asuntos de escala inherentes a sistemas complejos de misión crítica, tiempo real y cliente/servidor, desarrollando dicha modelación en el Rational Rose que es la Herramienta CASE que comercializan los desarrolladores de UML y que soporta de forma completa la especificación del UML.

Como lenguaje de programación por el lado del cliente: JavaScript. Este conocido lenguaje basado en su sencillez y pensado para hacer las cosas con rapidez, Dhtml, y algo que ha surgido nuevo que se encuentra de moda aunque aun no se ha catalogado como una tecnología pero si contiene combinaciones de ellas y de lenguajes como javascript, Ajax, y para la presentación de los datos en la capa de interfaz se hará uso de CSS. Como lenguaje del lado del servidor se utiliza el lenguaje PHP por su sencillez, rapidez e interacción, además porque el mismo soporta una gran cantidad de gestores de bases de datos. Entre las que se pueden mencionar InterBase, mSQL, MySQL, Oracle, Informix, PostgreSQL, entre otras, facilitando que se pueda migrar la base de datos en dependencia de lo que se necesite. PHP también ofrece la integración con varias bibliotecas externas, que dan al desarrollador la posibilidad de realizar cualquier tarea, desde generar documentos en pdf (Portable Document Format) hasta analizar código XML. Como arquitectura se hará uso del patrón Modelo –Vista – Controlador (MVC) por su integridad, facilidad de mantenimiento del código y ventajas que presenta. En cuanto al almacenamiento de datos se utilizará el potente gestor de base de datos Postgre por ser de gran escalabilidad y adaptarse fácilmente a la cantidad de procesadores en que se trabaje, implementa el uso de rollbacks, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz. Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos, y por parte de los servidores Web utilizaremos el Apache, que esta disponible para una gran multitud de plataformas entre ellas citar FreeBSD, NetBSD, OpenBSD, GNU/Linux, Mac OS Mac OS X Server y Windows (R).

CAPÍTULO 2
Modelo del Negocio

2.1 INTRODUCCION

Antes de comenzar a desarrollar un sistema o una aplicación es necesario comprender la organización y los procesos que en ella tienen lugar, a fin de lograr una mejor comprensión del problema a resolver y el común entendimiento entre clientes y desarrolladores, para lo cual se realiza la modelación del negocio.

El modelo del negocio, posibilita obtener una visión más clara del proceso en cuestión, por ello en este capítulo se exponen las políticas y condiciones que deben cumplirse, entendidas como reglas del negocio asociadas al campo de acción. Se describen los actores, trabajadores del negocio y el modelo de objetos.

2.2 Modelo del negocio actual

La Planificadora elabora un Excel con la estructura general del horario, este es revisado por la vicedecana docente para ver si hay inconvenientes de no ser así, el mismo es enviado a los profesores y estudiantes que vean sus turnos de clases y la ubicación donde les corresponde impartir y recibir las clases respectivamente, en caso del profesor estar inconforme o ver que los turnos le coinciden con alguna otra actividad solicitan a la planificadora un cambio de turno esta busca donde puede ubicarlo mientras no afecte a nadie más y luego el profesor o la planificadora le envían un correo a los estudiantes a los cuales este profesor les imparte clases.

2.3 Reglas del negocio a considerar

- ✓ No planificar nada el día de EF en esa misma sesión.
- ✓ Restricciones de algunas asignaturas (Ej. IP, Conf. y LAB).
- ✓ Planificar PP en sección contraria.
- ✓ Un Profesor puede impartirle clases a varios años.
- ✓ No planificar nada los martes a 4^{to} turno (afectación de la facultad).
- ✓ El profesor que de clase a 3^{er} turno no puede dar clase a 4^{to} turno.
- ✓ Los profesores externos no pueden impartir clases a 6^{to} turno.

- ✓ La asignatura de EF no se debe planificar en 3er y 4to turno.
- ✓ Planificar los turnos de FEU después de los miércoles.
- ✓ Dejar un día intermedio de descanso para los profesores.
- ✓ Planificar alternamente la sección a los grupos en cada semestre.

2.4 Actores del negocio

Un actor del negocio es cualquier individuo, grupo, organización, máquina o sistema de información externo que interactúa con el negocio. El término actor significa el rol que algo o alguien juega cuando interactúa con el negocio para beneficiarse de sus resultados. De acuerdo con esta idea un actor del negocio representa un tipo particular de usuario del negocio más que un usuario físico, ya que varios usuarios físicos pueden realizar el mismo papel en relación al negocio, o sea, ser instancias de un mismo actor.

Actores del Negocio	Descripción
Profesor	Persona que interviene en el proceso modificar horario, es responsable de solicitar un cambio en el horario e interactúa con este proceso para evitar contradicciones en el horario.
Tiempo	Encargado de comenzar el proceso de crear horario, ya que es el que inicia este proceso a la llegada de un nuevo semestre.
Directivo	Persona (Decano, vicedecano), que interviene en el proceso modificar horario, siendo un profesor aparte de ser directivo, es responsable de solicitar un cambio en el horario y interactúa con este proceso para evitar contradicciones en el horario.
Alumno Ayudante	Estudiante con la característica similar a un profesor y es que imparte clases, interviene en el proceso modificar horario, es responsable de solicitar un cambio en el horario y interactúa con este proceso para evitar contradicciones en el horario.

2.5 Trabajadores del negocio

Un trabajador define el comportamiento y las responsabilidades de un individuo que actúa en el negocio realizando una o varias actividades, interactuando con otros trabajadores del negocio y manipulando entidades del negocio.

Trabajadores del Negocio	Descripción
Planificadora	Persona que interviene en los procesos de crear horario, modificar horario y enviar por correo el horario en caso de realizar algún proceso anteriormente mencionado.

2.6 Entidades del negocio

P1, Horario, Afectaciones, Local

2.7 Casos de uso del negocio

Un caso de uso del negocio representa a un proceso de negocio, por lo que se corresponde con una secuencia de acciones que producen un resultado observable para ciertos actores del negocio. Desde la perspectiva de un actor individual, define un flujo de trabajo completo que produce resultados deseables.

2.7.1 Casos de uso del negocio a tratar

- Modificar Horario
- Crear Horario
- Enviar Horario

2.7.2 Descripción de los casos de uso del negocio

- **Descripción del caso de uso Modificar Horario**

Nombre del caso de uso del negocio:	Modificar Horario
Actores del negocio:	Directivo
Propósito:	Modificar horario cuando exista alguna inconveniencia por parte de algún profesor, directivo o alumno ayudante
Resumen: El caso de uso se inicia cuando un Profesor , Alumno ayudante o directivo se presenta en secretaría docente para solicitar el cambio de algún turno porque le coincide con alguna actividad o por algún otro interés. Este le informa a la planificadora su problema , la misma busca un turno donde pueda ponerlo teniendo en cuenta sus afectaciones y las de los demás , viendo que no altere a una tercera persona.	

Casos de uso asociados:	-
Flujo de trabajo	
Acción del actor	**Respuesta del negocio**
1- El profesor, alumno ayudante o directivo solicita cambiar su horario por razones porque le coincide con alguna afectación. **4-** El profesor, alumno ayudante o directivo informa su motivo. **7-** Este escucha el cambio de turno propuesto por la planificadora **8-** Si no le coincide con alguna afectación, le informa a la planificadora que sí, que está de acuerdo con la variación. **11-** Este escucha que su turno fue cambiado y se retira	**2-** La planificadora escucha su solicitud de cambiar algún turno. **3-** La panificadora pregunta porqué desea modificar el horario del turno. **5-** La planificadora busca otro turno para efectuar la modificación. **6-** La planificadora informa para el turno que se lo puede cambiar. **9-** Escucha la respuesta del mismo y le cambia el turno para donde le propuso la misma **10-** La panificadora le informa que su turno fue modificado y se remite al **caso de uso Enviar Horario**
Prioridad:	Alto
Mejoras:	Se automatizará el proceso de cambiar algún turno de un profesor
Cursos alternos:	
Acción del actor	
8.1- Si le coincide con alguna afectación este informa a la planificadora que le coincide con alguna otra actividad.	**10-** Escucha su problema, se remite al paso **5**

- **Descripción del caso de uso Crear Horario**

Nombre del caso de uso del negocio:	Crear Horario
Actores del negocio:	Tiempo
Propósito:	Elaborar los horarios con una mayor rapidez para hacer llegar dicha información a los Usuarios (Profesores, Estudiantes, Alumnos ayudantes, Directivos).
Resumen: El caso de uso se inicia cuando el tiempo le solicita a la planificadora crear el horario para el nuevo semestre que está por comenzar, la planificadora al ver que llega este momento, comienza con la planificación del mismo.	
Casos de uso asociados:	-
Flujo de trabajo	
Acción del actor	**Respuesta del negocio**
1- El tiempo informa que es momento de realizar la planificación del horario por la llegada del nuevo semestre.	**2-** La planificadora al ver que se acerca el nuevo semestre se prepara para dicha solicitud **3-** La Panificadora consulta los P1, las afectaciones de cada departamento y los locales disponibles. **4-** Teniendo en cuenta esto planifica el horario, creando un nuevo horario para ese semestre. **5-** Es creado el horario y se remite al **caso de uso Enviar Horario**
Prioridad:	Alta
Mejoras:	Facilita el trabajo de la Planificadora de la facultad y una mayor rapidez para los usuarios a la hora de interesarse por sus horarios de clases.
Cursos alternos:	

- **Descripción del caso de uso Enviar Horario**

Nombre del caso de uso del negocio:	Enviar Horario
Actores del negocio:	Directivo y Tiempo
Propósito:	Enviar el horario una vez que el mismo es creado o es modificado.
Resumen: El caso de uso se inicia cuando un directivo o el tiempo solicitan modificar o crear el horario respectivamente, una vez realizada la operación , se les envía un correo a los estudiantes con el nuevo horario.	
Casos de uso asociados:	-
Flujo de trabajo	
Acción del actor	**Respuesta del negocio**

	1- Una vez creado el horario o modificado, la planificadora recuerda enviarlo con los últimos ajustes. **2-** Si lo que realizó fue una modificación busca el horario del grupo afectado y le envía un correo con el nuevo.
Prioridad:	Alta
Mejoras:	Facilita a los estudiantes y profesores obtener el horario en tiempo y forma para poder planificarse y estar informados.
Cursos alternos:	
	2.1- Si lo que hizo fue crear el horario le envía el mismo a todos los grupos

2.7.3 Modelo de casos de uso del negocio

El modelo de Casos de Uso del Negocio es un modelo que describe los procesos de un negocio (casos de uso del negocio) y su interacción con elementos externos (actores), tales como socios y clientes, es decir, describe las funciones que el negocio pretende realizar y su objetivo básico es describir cómo el negocio es utilizado por sus clientes y socios.

- **Diagrama de casos de uso del negocio**

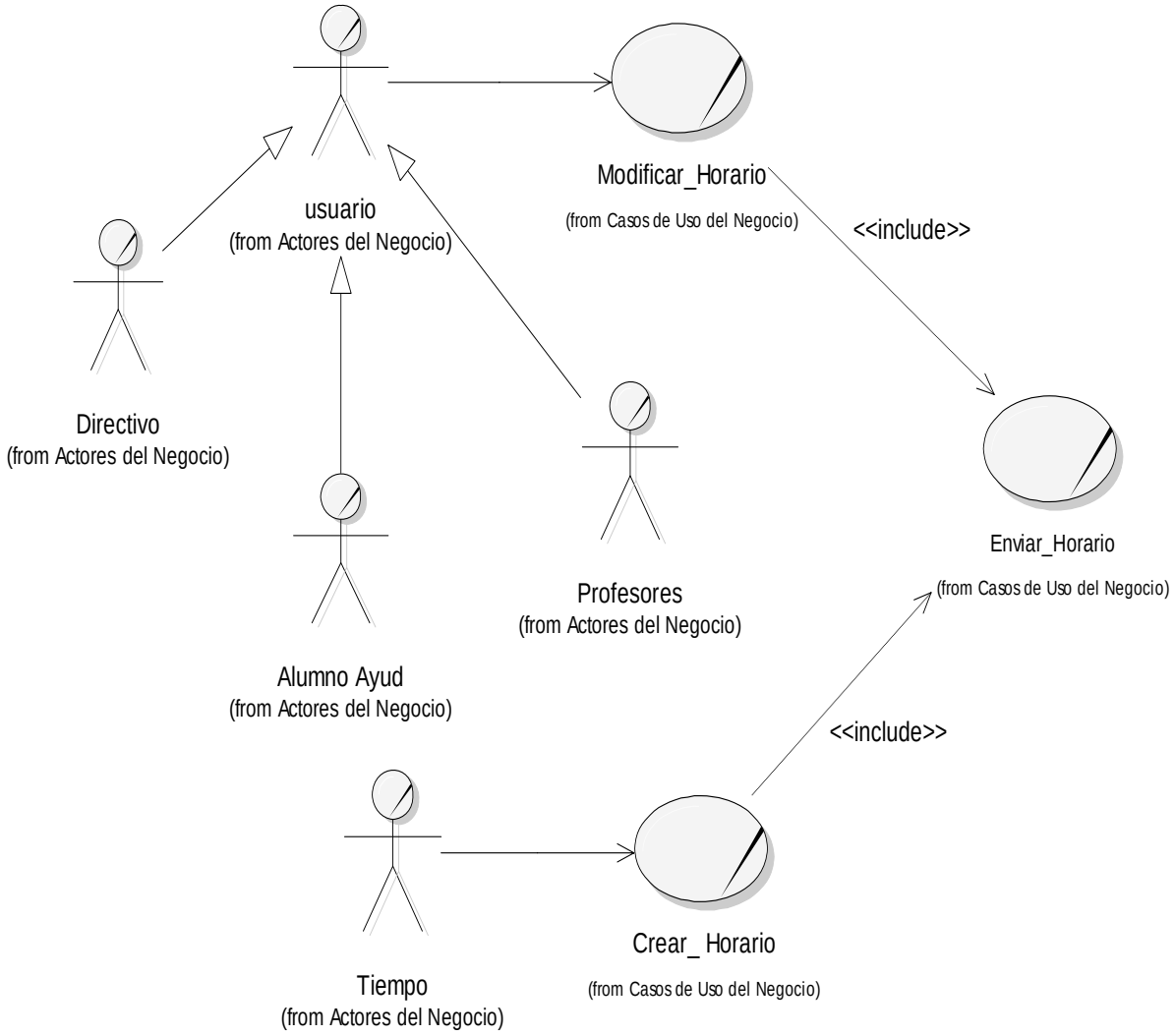
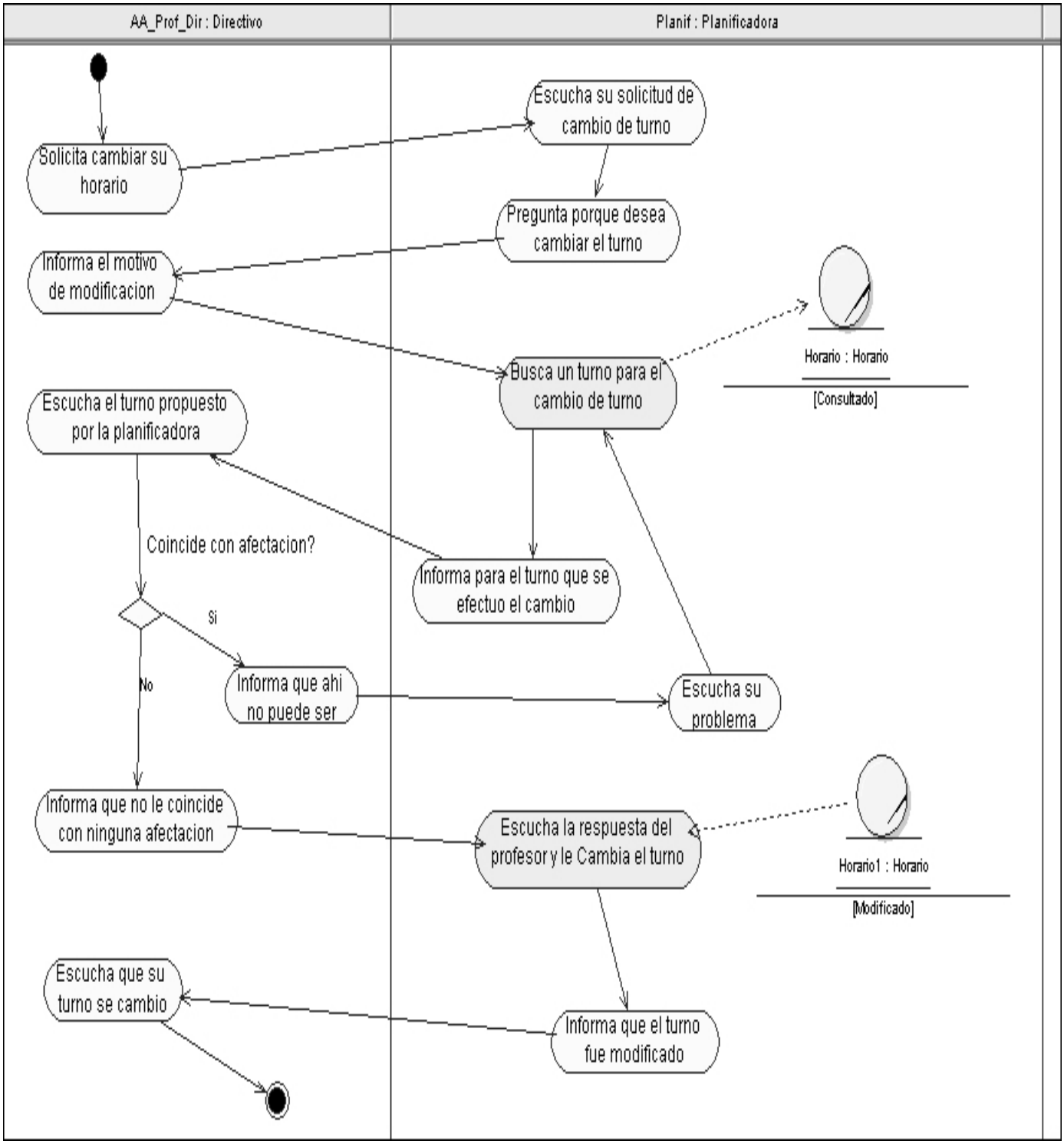


Figura 2.1: Diagrama de caso de uso del negocio

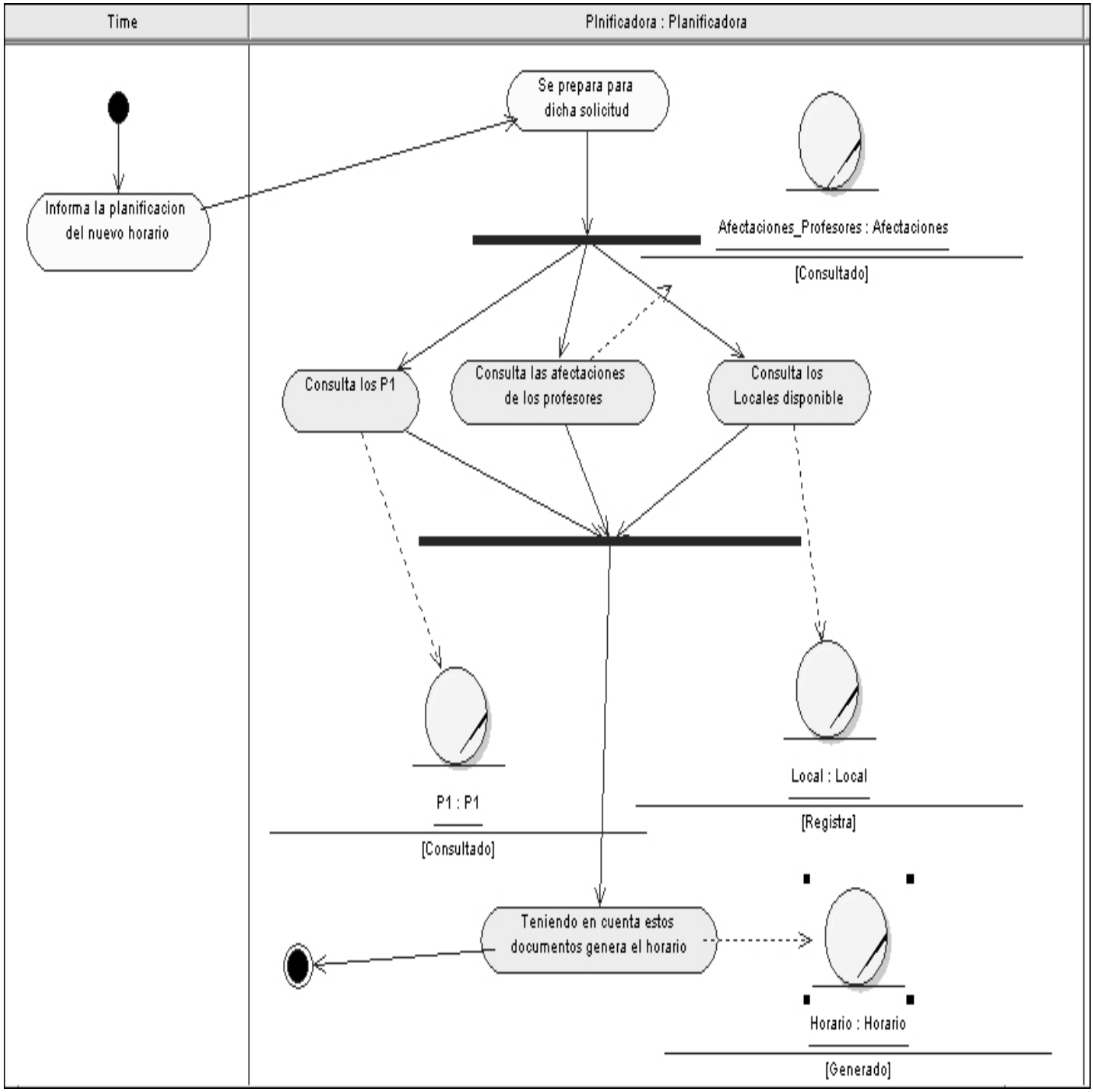
2.8 Diagrama de Actividades

Un diagrama de actividad demuestra la serie de actividades que deben ser realizadas en un proceso del negocio, así como las distintas rutas que pueden irse desencadenando. Este es dividido en canales, donde cada canal representa el actor que está llevando a cabo la actividad y muestra cómo se utilizan las entidades del negocio.

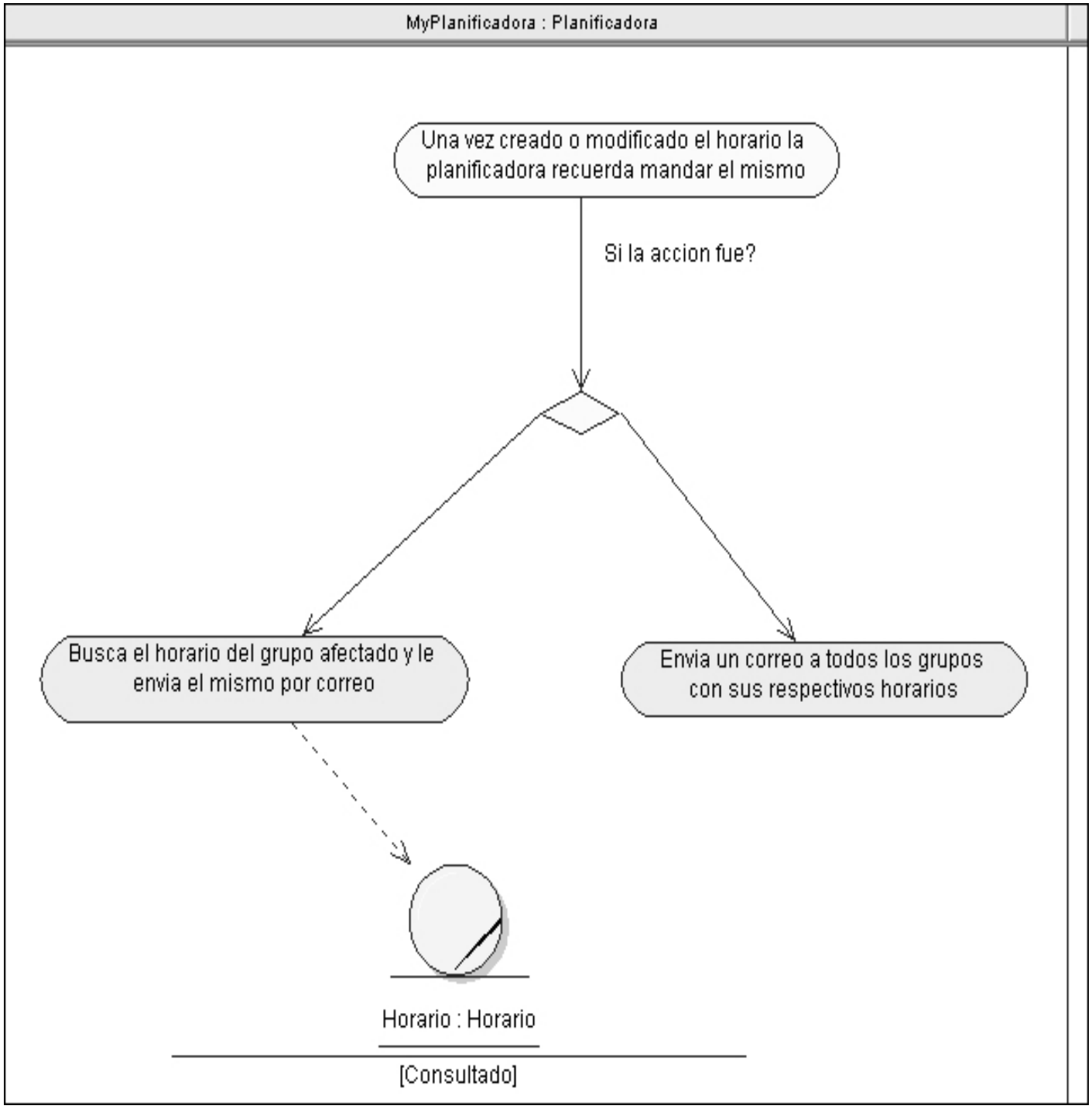
2.8.1 Diagrama de actividades del caso de uso Modificar Horario



2.8.2 Diagrama de actividades del caso de uso Crear Horario



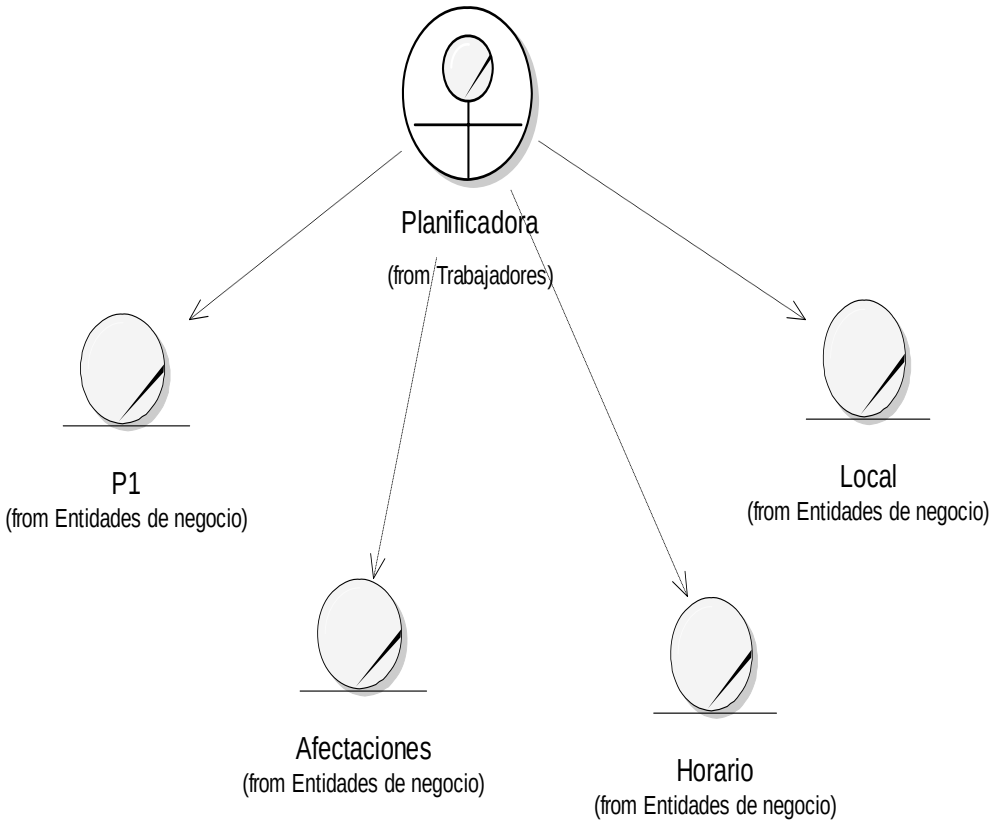
2.8.3 Diagrama de actividades del caso de uso Enviar Horario



2.9 Modelo de objeto del Negocio

Un modelo de objetos del negocio es un modelo interno a un negocio. Describe como cada caso de uso del negocio es llevado a cabo por parte de un conjunto de trabajadores que utilizan un conjunto de entidades del negocio y unidades de trabajo. [9]
A continuación se muestra en el Diagrama de clases del modelo de objetos del negocio.

2.9.1 Diagrama de clases del modelo de objeto



2.10 CONCLUSIONES

En este capítulo fueron identificados, los actores del proceso en cuestión, es decir, personas beneficiadas con la creación del sistema, entidades u objetos del negocio, así como su relación en esos procesos. Se realizó una descripción de los mismos con la finalidad de obtener una mejor comprensión de la problemática que el sistema pretende resolver. Además se describieron los casos de uso involucrados en el negocio para una mejor comprensión a la hora de pasar para el siguiente capítulo.

CAPÍTULO 3 Requisitos

3.1 INTRODUCCION

En este capítulo se especifican los requisitos funcionales y no funcionales del sistema que dará solución al problema planteado, quiénes interactuarán con él (actores del sistema) y las distintas funcionalidades que ofrecerá a cada uno de los actores. Además de describir con grandes medidas de detalladamente lo que el sistema debe hacer.

3.2 Definición de los requisitos funcionales

Los requisitos funcionales son aquellos 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. Para cumplir con los objetivos propuestos se prevé que el sistema tenga las siguientes funcionalidades:

RF 1 Gestionar horarios

1.1 Buscar horario (Estudiantes, Profesores y Locales).

1.2 Modificar horario (Estudiantes, Profesores y Locales).

1.3 Crear horario (Estudiantes, Profesores y Locales).

1.4 Enviar Horario (Estudiantes, Profesores).

1.5 Generar Reporte Horarios (Locales, Profesores y Estudiantes).

RF 2 Gestionar Afectaciones

2.1 Registrar afectaciones

2.2 Buscar afectaciones

2.3 Modificar afectaciones

2.4 Eliminar Afectaciones

RF 3 Gestionar P1

3.1 Registrar P1.

3.2 Modificar P1.

RF 4 Gestionar Local

4.1Registrar Local

RF 5 Mostrar Horario

RF 6 Mostrar P1

RF 7 Mostrar Afectaciones

RF 8 Mostrar Horario Local

RF 9 Autenticar Usuario

3.3 Requisitos 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.

RNF 1: Apariencia o interfaz interna

1.1 Uso de los colores que distinguen a la UCI y a la Facultad8.

1.2 Debe ser interactivo con el usuario.

RNF 2: Usabilidad

2.1 El sistema podrá ser usado por cualquier persona que acceda a él y que tenga algún conocimiento básico de computación y trabajo en Web.

2.2 El texto deberá tener un tamaño no pequeño para que pueda ser observado a un metro o más de distancia además por especificaciones del cliente.

RNF 3: Rendimiento

3.1 El sistema deberá ser capaz de gestionar toda la información referente a la planificación docente y dar respuesta a las solicitudes lo más rápido.

3.2 Debe ser eficiente a la hora de gestionar las solicitudes logrando que sin mucha navegación por el sitio se obtenga los resultados deseados.

3.3 Debe estar disponible las 24 horas del día.

RNF 4: Soporte

4.1 Debe ser lo más extensible posible. Puede ser usado en otras áreas docentes.

4.2 Fácil para el mantenimiento, de configuración sencilla y asequible para los clientes.

RNF 5: Portabilidad

5.1 Multiplataforma. El sistema se podrá montar sobre Unix, Linux, Windows, etc. Así mismo podrá usar una serie de SGBD como PostgreSQL, MySQL, Oracle, SQLite, entre otros, aunque preferiblemente se desea la portabilidad sobre tecnología o software libre.

RNF 6: Seguridad

6.1 La seguridad esta bien definida, solamente tendrá acceso la planificadora y tendrá que autenticarse antes de realizar su trabajo.

RNF 7: Confiabilidad

7.1 Disponible en todo momento debido a su importancia para planificar las clases.

7.2 Capaz de restaurarse de las fallas rápidamente.

RNF 8: Software

8.1 Para el servidor Web: Servidor Web Apache v1.x o superior.

8.2 SGDB: PostgreSQL preferiblemente v8.x en adelante.

8.3 Para clientes: Navegador Internet Explorer v4.0 o superior.

3.4 Actores del sistema a automatizar

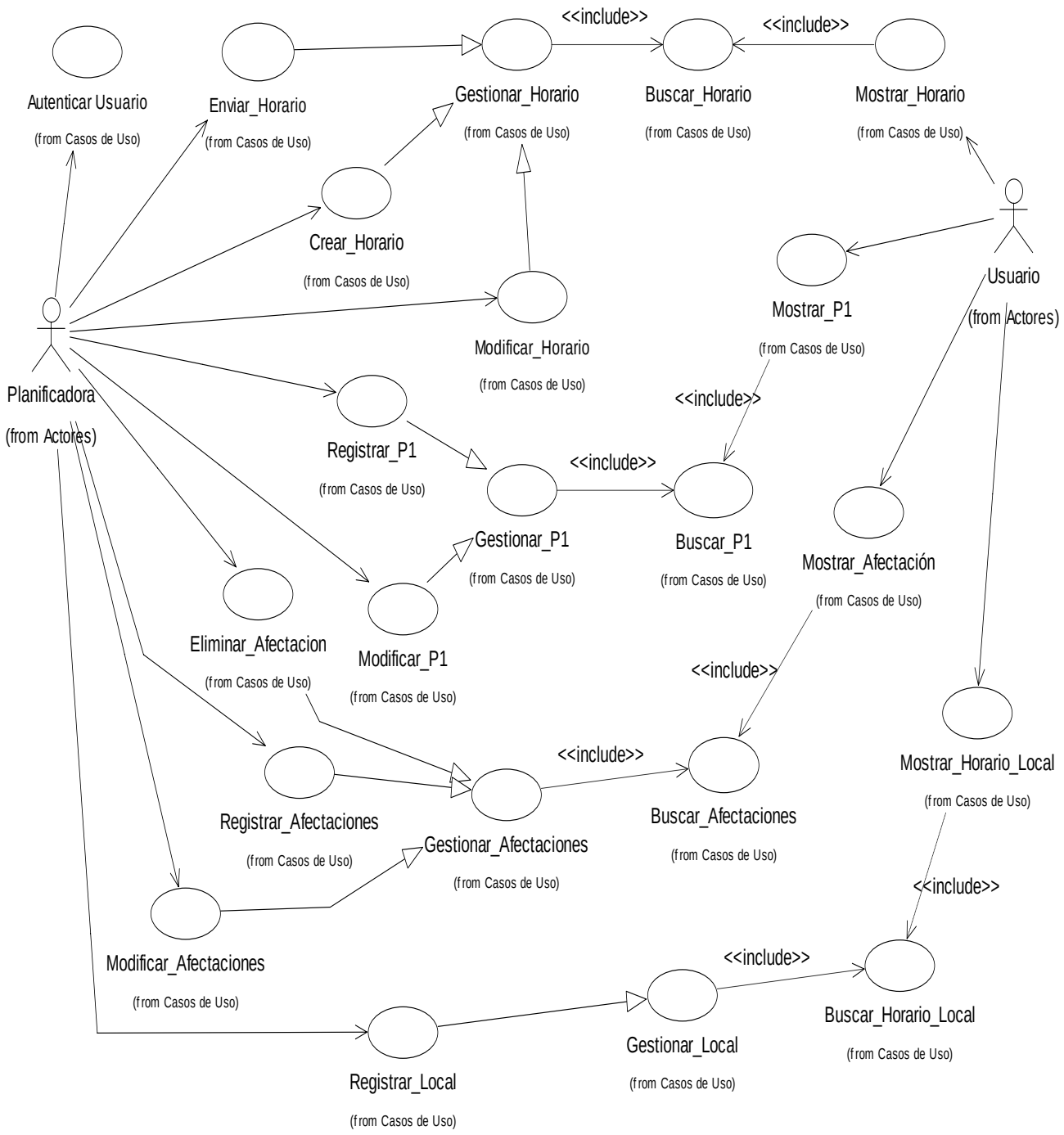
Los actores del sistema pueden representar el rol que juegan una o varias personas, un equipo o un sistema automatizado, son parte del sistema, y pueden intercambiar información con él o ser recipientes pasivos de información. En este caso los actores que interactúan con el sistema se definen a continuación:

Directivos – Estudiantes – Profesores – Alumnos Ayudantes – Planificadora

3.5 Diagrama de casos de uso del sistema

Los Casos de Uso son “fragmentos” de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores. De manera más precisa, un Caso de Uso especifica una secuencia de acciones que el sistema puede llevar a cabo interactuando con sus actores, incluyendo alternativas dentro de la secuencia.

2.5.1 Diagrama de Casos de Uso del Sistema



3.6 Descripción de los casos de uso del sistema

3.6.1 Descripción del Caso de Uso del Sistema Gestionar Horario Estudiantes

Caso de uso:	***Gestionar Horario***
Actores :	Planificadora(inicia)
Propósito:	Permite a la planificadora gestionar (Crear, Modificar) los horarios o uno en específico.

Resumen:
El CUS se inicia cuando la planificadora selecciona la opción de Gestionar Horario, luego selecciona el tipo de gestión a realizar, introduce los datos necesarios, el sistema realiza la acción seleccionada por la planificadora y termina el CUS.

Referencias:
RF1, 1.2,1.3

Precondiciones:

Poscondiciones:
Toda la información necesaria para realizar los horario este guardada en la Base de datos.

Curso normal de los eventos:

Acción del actor:		***Respuesta del proceso del Sistema:***	
1	La Planificadora selecciona la opción de Gestionar Horario.	**1.1**	El sistema muestra las opciones: Modificar Horario , Crear Horario
Escenario 1: Crear Horario			
1	La planificadora selecciona la opción de crear el horario.	**1.1**	El sistema muestra una interfaz con el horario en blanco
2	La planificadora registra los turnos, el tipo de actividad y el local que corresponde.	**2.1**	El sistema muestra como queda el horario registrado y termina el caso de uso.

Curso alterno de los eventos:

Escenario 2: Modificar Horario			
1	La planificadora selecciona la opción de Modificar horario	**1.1**	El sistema muestra una interfaz para buscar los horarios que va ha modificar y se remite al caso de uso buscar Horario.
2	La planificadora mira el resultado de la Búsqueda, y selecciona el horario ha modificar.	**2.1**	El sistema automáticamente le muestra ese horario que ella quería en una interfaz automáticamente...
3	La planificadora selecciona el turno para donde desea planificarlo.	**3.1**	Si el turno o los turnos modificados son correctos el sistema actualiza los datos correctamente.

Curso alterno de los eventos:

Acción 3.1:	Si los turnos introducidos por La planificadora son incorrectos el sistema muestra un mensaje de error indicando donde está el turno e indica al usuario retornar a la acción 3.
Prioridad :	critico

3.6.2 Descripción del Caso de Uso del Sistema Enviar Correo Estudiantes

Caso de uso:	***Enviar Correo EST***
Actores :	Planificadora (inicia).
Propósito:	Para que los estudiantes se informen de las clases que deben recibir.

Resumen:
El CUS se inicia cuando la planificadora selecciona la opción enviar correo a los grupos que desee para que los estudiantes estén informado de las clases que reciben, el sistema realiza la acción y termina el CUS.

Referencias:
RF 1.5

Precondiciones:

Poscondiciones:
Los horarios tienen que estar generados.

Curso normal de los eventos:

Acción del actor:		***Respuesta del proceso del Sistema:***	
1	La Planificadora selecciona la opción de enviar Horario.	**1.1**	El sistema muestra una interfaz para la que envié el horario...
2	La planificadora selecciona los grupos al que desee mandarle el correo, y presiona el botón enviar ...	**2.1**	El sistema muestra un cartel diciéndole que los correos fueron enviados correctamente y termina el caso de uso.

Curso alterno de los eventos:

Prioridad :	secundario

3.6.3 Descripción del Caso de Uso del Sistema Gestionar Afectaciones

Caso de uso:	***Gestionar Afectaciones***
Actores :	Planificadora(inicia)
Propósito:	Permite a la planificadora gestionar (Modificar, Registrar y Eliminar) las Afectaciones o uno en específico.

Resumen:
El CUS se inicia cuando la planificadora selecciona la opción de Gestionar Afectaciones, luego selecciona el tipo de gestión a realizar, introduce los datos necesarios, el sistema realiza la acción seleccionada por la planificadora y termina el CUS.

Referencias:
RF 2,2.1,2.3,2.4

Precondiciones:

Poscondiciones:
Todas las afectaciones tienen que estar guardadas en la Base de datos.

Curso normal de los eventos:

Acción del actor:		***Respuesta del proceso del Sistema:***	
1	La Planificadora selecciona la opción de Gestionar Afectaciones	**1.1**	El sistema muestra las opciones: Modificar Afectaciones, Registrar Afectaciones y Eliminar afectaciones.
Escenario 1: Modificar Afectaciones			
1	La planificadora selecciona la opción de Modificar Afectaciones	**1.1**	El sistema muestra una interfaz para buscar Las afectaciones a ser modificada y se remite al caso de uso buscar Afectación.
2	La planificadora mira el resultado de la búsqueda selecciona la afectación que desea modificar. La panificadora modifica los datos deseados.	**2.1**	El sistema le muestra un cartel con los datos de la afectación seleccionada El sistema le muestra un cartel diciéndole que los datos fueron cambiados y se termina el caso de uso.

Curso alterno de los eventos:

Escenario 2: Registrar Afectaciones			
1	La planificadora selecciona la opción de Registrar Afectaciones	**1.1**	El sistema muestra una interfaz con los datos a introducir.
2	La planificadora llena los datos	**2.1**	Si los datos están bien se registran las afectaciones.
3	La planificadora selecciona el turno que desea modificar en el horario	**3.1**	El sistema actualiza las afectaciones automáticamente.

Curso alterno de los eventos:

Accion1.1:	Si existe algún error en el llenado de los datos el sistema muestra un cartel de error y lo remite a la accion2.

Escenario 3: Eliminar Afectaciones			
1	La planificadora selecciona la opción de Eliminar Afectaciones	**1.1**	El sistema muestra una interfaz para buscar Las afectaciones ha eliminar y se remite al caso de uso buscar Afectación.
2	La planificadora Ve la búsqueda de todas las afectaciones y selecciona la afectación a eliminar	**2.1**	El sistema elimina la afectación.
		2.2	El sistema muestra un cartel diciéndole que la afectación o afectaciones fueron eliminadas correctamente.

Curso alterno de los eventos:

Prioridad : secundario

3.6.4 Descripción del Caso de Uso del Sistema Enviar Correo Profesor

Caso de uso:	***Enviar Correo Profesor.***
Actores :	Planificadora (inicia).
Propósito:	Para que los profesores se informen de las clases que deben impartir.

Resumen:
El CUS se inicia cuando la planificadora selecciona la opción enviar correo a los profesores para estén informado de las clases que tiene que impartir, el sistema realiza la acción y termina el CUS.

Referencias:
RF 1.5

Precondiciones:

Poscondiciones:
Los horarios tienen que estar generados.

Curso normal de los eventos:

Acción del actor:		***Respuesta del proceso del Sistema:***	
1	La Planificadora selecciona la opción de enviar Horario	**1.1**	El sistema muestra una interfaz para la que envié el horario...
2	La planificadora selecciona los profesores para enviarle el horario y presiona el botón enviar.	**2.1**	El sistema muestra un cartel diciéndole que los correos fueron enviados correctamente y termina el caso de uso.

Curso alterno de los eventos:

Prioridad : secundario

3.6.5 Descripción del Caso de Uso del Sistema Gestionar P1

Caso de uso:	***Gestionar P1***
Actores :	Planificadora(inicia)
Propósito:	Permite a la planificadora gestionar (Modificar y Registrar) los P1

Resumen:
El CUS se inicia cuando la planificadora selecciona la opción de Gestionar P1, luego selecciona el tipo de gestión a realizar, introduce los datos necesarios, el sistema realiza la acción seleccionada por la planificadora y termina el CUS.

Referencias:
RF 3,3.1,3.2

Precondiciones:

Poscondiciones:
Los P1 ya fueron enviados por la universidad.

Curso normal de los eventos:

Acción del actor:		***Respuesta del proceso del Sistema:***	
1	La Planificadora selecciona la opción de Gestionar P1	**1.1**	El sistema muestra las opciones: Modificar P1 y Registrar P1.
Escenario 1: Registrar P1			
1	La planificadora selecciona la opción de Registrar P1	**1.1**	El sistema muestra una interfaz solicitando los datos del P1, donde estos son insertados por semanas.
2	La planificadora inserta los datos que tienen los documentos físicos, P1 que le mandan de la universidad.	**2.1**	Si no existen errores en los datos, el sistema le muestra un cartel diciéndole que la semana del P1 fue registrado satisfactoriamente.

Curso alterno de los eventos:

Acción 2.1	Si existen errores se remite al paso 1.1

Escenario 2:Modificar P1			
1	La planificadora selecciona la opción de Modificar P1.	**1.1**	El sistema muestra una interfaz para buscar el P1 a modificar y se remite al caso de uso buscar horario local.
2	La planificadora ve el resultado de la búsqueda y escoge un P1.	**2.1**	El sistema muestra los datos del P1 seleccionado.
3	La planificadora escoge los datos a modificar del P1 seleccionado	**3.1**	El sistema modifica los datos seleccionados y muestra un cartel diciéndole que sus datos fueron cambiados satisfactoriamente.

Curso alterno de los eventos:

Prioridad :	secundario

3.6.6 Descripción del Caso de Uso del Sistema Gestionar Local

Caso de uso:	***Gestionar Local***
Actores :	Planificadora(inicia)
Propósito:	Permite a la planificadora gestionar (Modificar) los locales o uno en específico.
Resumen:	
El CUS se inicia cuando la planificadora selecciona la opción de Gestionar Local, luego	

selecciona el tipo de gestión a realizar, introduce los datos necesarios, el sistema realiza la acción seleccionada por la planificadora y termina el CUS.

Referencias:
RF4,4.1,4.3

Precondiciones:

Poscondiciones:
Toda los locales tienen que estar registrados en la base de datos

Curso normal de los eventos:

Acción del actor:		***Respuesta del proceso del Sistema:***	
1	La Planificadora selecciona la opción de Gestionar Local.	**1.1**	El sistema muestra las opciones: Modificar local.
Escenario 2: Registrar Local			
1	La planificadora selecciona la opción de Registrar Local	**1.1**	El sistema muestra una interfaz para que los entre y ser guardados en la BD.
2	La planificadora inserta todos los locales	**2.1**	El sistema le muestra un cartel diciéndole que los datos han sido insertados correctamente si no existen errores.

Curso alterno de los eventos:	
Acción 2.1	En caso de existir algún error se remite al paso 2
Prioridad:	Secundario

3.6.7 Descripción del caso de uso Buscar Horario

Caso de uso:	***Buscar Horario***
Actores :	Planificadora (inicia).
Propósito:	Para realizar alguna modificación o revisarlo.

Resumen:
El CUS se inicia cuando la planificadora selecciona la opción buscar horario del grupo que desee para modificarlo, el sistema realiza la acción y termina el CUS.

Referencias:
RF 1.1

Precondiciones:

Poscondiciones:
Los horarios tienen que estar creados.

Curso normal de los eventos:

Acción del actor:		***Respuesta del proceso del Sistema:***	
1	La Planificadora selecciona la opción de buscar Horario	**1.1**	El sistema muestra una interfaz para la búsqueda del horario...

2	La planificadora selecciona el botón buscar horario.	**2.1**	El sistema muestra una interfaz con todos los horarios buscados y se remite al caso de uso gestionar horario

Curso alterno de los eventos:	
Acción 2.1	En caso de ser un usuario El sistema muestra una interfaz con todos los horarios buscados y se remite al caso de uso Mostrar Horario
Prioridad:	crítico

3.6.8 Descripción del caso de uso Buscar P1

Caso de uso:	***Buscar P1***
Actores :	Planificadora (inicia).
Propósito:	Para realizar alguna modificación.
Resumen:	

El CUS se inicia cuando la planificadora selecciona la opción buscar P1 para modificarlo, o el usuario desee ver algún p1 que desee, el sistema realiza la acción y termina el CUS.

Referencias:

RF 3.3.

Precondiciones:

Poscondiciones:

Los P1 deben estar introducidos en la base de datos.

Curso normal de los eventos:

Acción del actor:		***Respuesta del proceso del Sistema:***	
1	La Planificadora selecciona la opción de buscar P1	**1.1**	El sistema muestra una interfaz para la búsqueda de los P1.
2	La planificadora selecciona el botón buscar P1	**2.1**	El sistema muestra una interfaz con todos los P1 Buscado y se remite al caso de uso gestionar P1.

Curso alterno de los eventos:	
Acción 2.1	En caso de ser un usuario El sistema muestra una interfaz con todos los P1 buscados y se remite al caso de uso Mostrar P1
Prioridad:	Crítico

3.6.9 Descripción del caso de uso Buscar Afectaciones

Caso de uso:	***Buscar Afectaciones***
Actores :	Planificadora (inicia).
Propósito:	Para realizar alguna modificación en las afectaciones.
Resumen:	

El CUS se inicia cuando la planificadora selecciona la opción buscar afectaciones para ser

modificada, el sistema realiza la acción y termina el CUS.

Referencias:
RF 2.2

Precondiciones:

Poscondiciones:
Las afectaciones deben de ser entregadas por los directivos y estos introducidos en la base de datos.

Curso normal de los eventos:

Acción del actor:		***Respuesta del proceso del Sistema:***	
1	La Planificadora selecciona la opción de buscar afectaciones.	**1.1**	El sistema muestra una interfaz para la búsqueda de las afectaciones.
2	La planificadora selecciona el botón de Buscar afectación de un profesor.	**2.1**	El sistema muestra una interfaz con la búsqueda de todas las afectaciones del profesor y se remite al caso de uso gestionar afectación.

Curso alterno de los eventos:	
Acción 2.1	En caso de ser un usuario El sistema muestra una interfaz con todas las Afectaciones buscados y se remite al caso de uso Mostrar Afectación
Prioridad:	crítico

3.8 CONCLUSIONES

En este capítulo se comenzó a desarrollar la propuesta de solución, obteniéndose a partir del análisis de los procesos del negocio, un listado con las principales funcionalidades que debe tener el sistema y los requisitos adicionales. Se presentaron los Diagramas de Casos de Uso del Sistema, y finalmente se describieron las acciones de los actores del mismo con los casos de uso con los que interactúan.

CAPÍTULO 4
Análisis Y Diseño

4.1 INTRODUCCION

Tras la definición y descripción, en el anterior capítulo, de las funcionalidades deseadas y necesarias del sistema propuesto; se hace necesario definir cómo se desarrollará.
Este capítulo tiene el objetivo de plantear la concepción general del análisis y diseño del sistema propuesto y cómo se implementa éste. Así, se presentan los diagramas de clases de análisis y diagramas de diseño Web que detallan la interacción de las distintas páginas.

4.2 ANALISIS

4.2.1 Modelo de clases de análisis

Un **modelo de objetos** describe la realización de los CU, y cuales sirven como abstracción del artefacto: Modelo de diseño. El modelo de análisis contiene los resultados del análisis de los CU, instancias del artefacto: clases del análisis. [15]

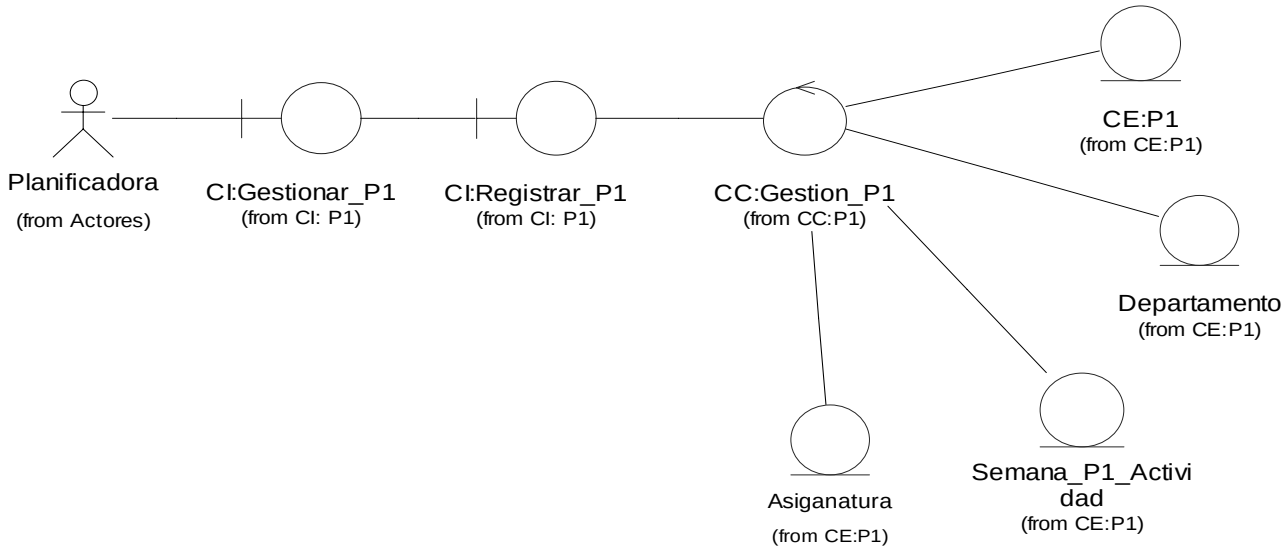
Propósito: El modelo de análisis contiene las clases del análisis y cualquier artefacto asociado. Puede ser un artefacto temporal, como es en el caso donde evoluciona al modelo de diseño, o puede continuar su vida a través de alguno o todos los proyectos, y quizás sirva de vista conceptual del sistema. [15]

4.2.2 Diagrama de clases del análisis

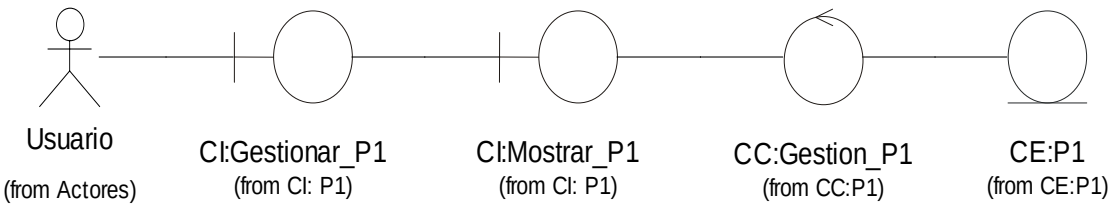
Un **Diagrama de clases del análisis** es un artefacto en el que se representan los conceptos en un dominio del problema. Representa las cosas del mundo real, no de la implementación automatizada de estas cosas. [15]

4.2.2 Diagramas de clases de análisis de Gestión de P1

- **Diagrama de clases de análisis del Caso de uso Registrar P1**



- **Diagrama de clases de análisis del Caso de uso Mostrar P1**

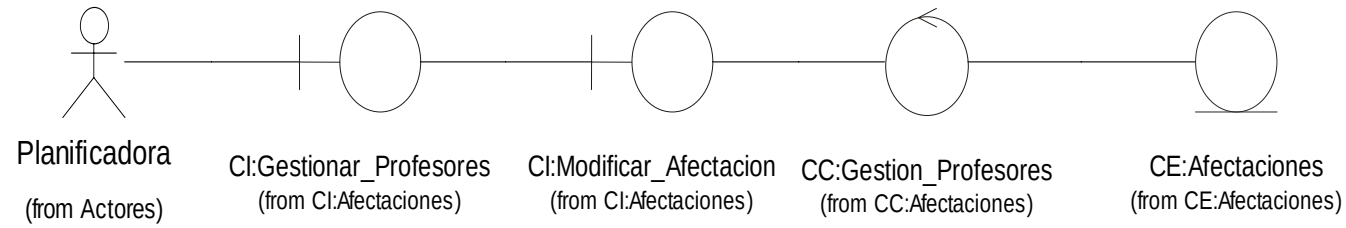


4.2.3 Diagramas de clases de análisis de Gestión de Profesores

- **Diagrama de clases de análisis del caso de uso Registrar Afectaciones**

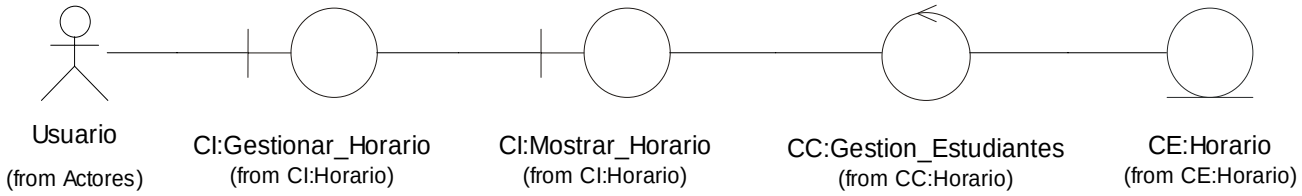
- **Diagrama de clases de análisis del caso de uso Mostrar Afectaciones**

- **Diagrama de clases de análisis del caso de uso Modificar Afectaciones**

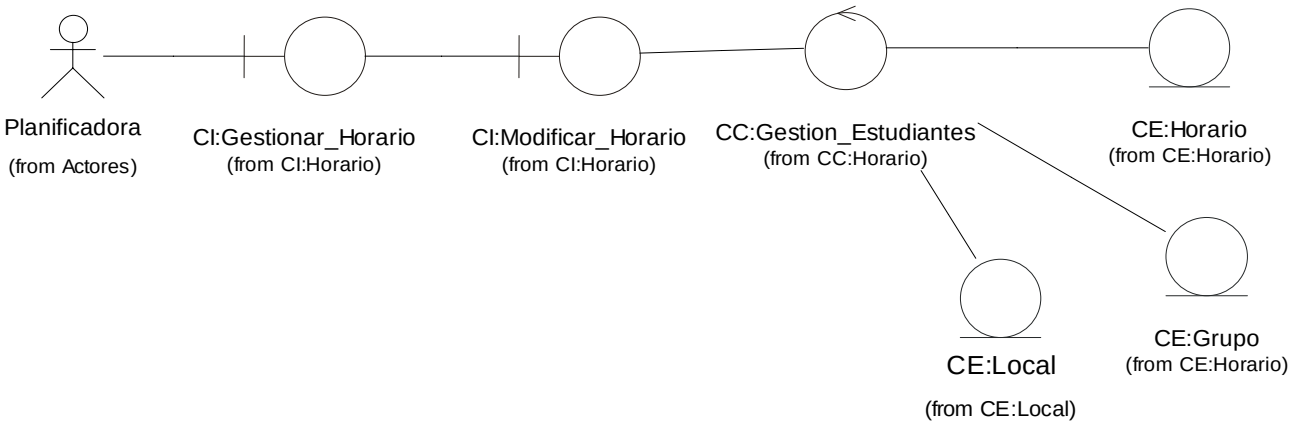


4.2.4 Diagramas de clases de análisis de Gestión de Estudiantes

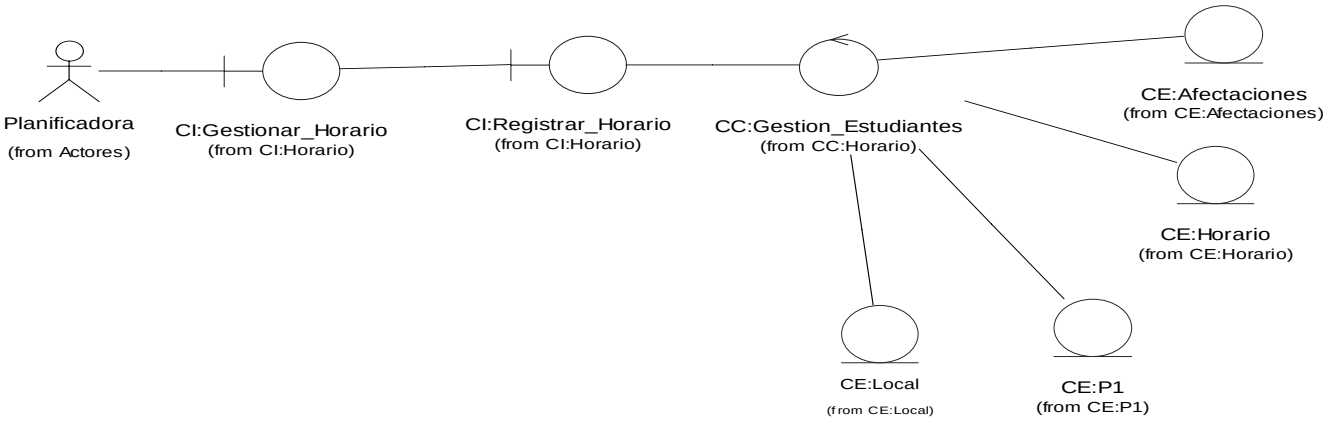
- **Diagramas de clases de análisis del caso de uso Mostrar Horario**



- **Diagramas de clases de análisis del caso de uso Modificar Horario**

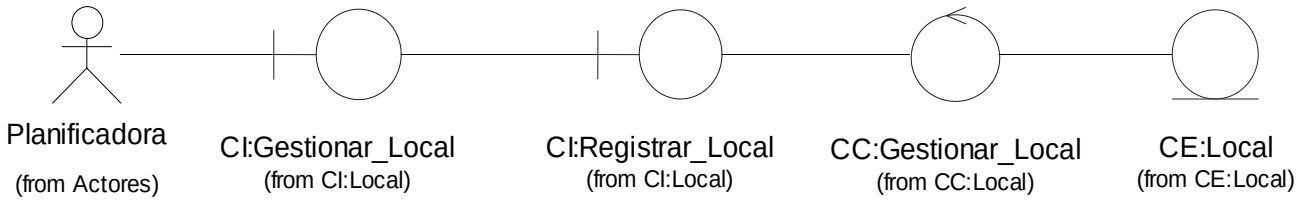


- **Diagramas de clases de análisis del caso de uso Crear Horario**

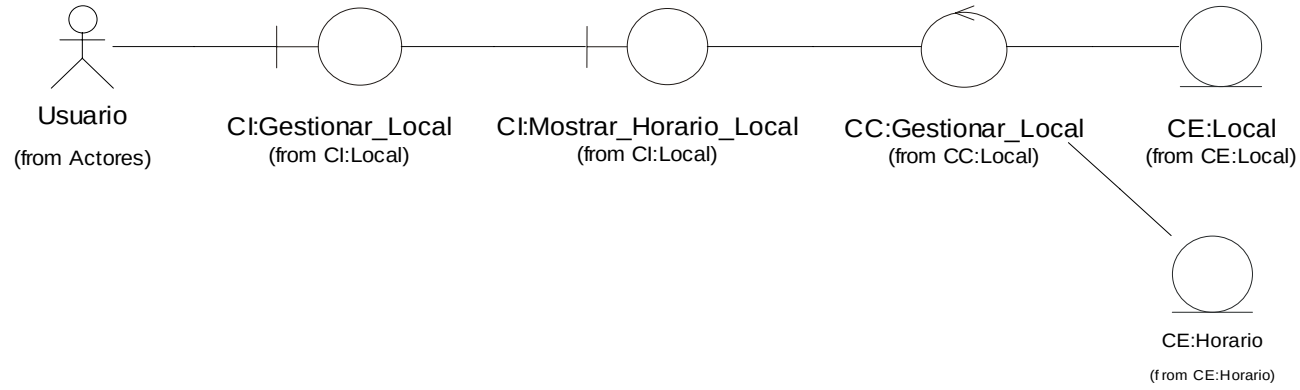


4.2.5 Diagramas de clases de análisis de Gestión de Locales

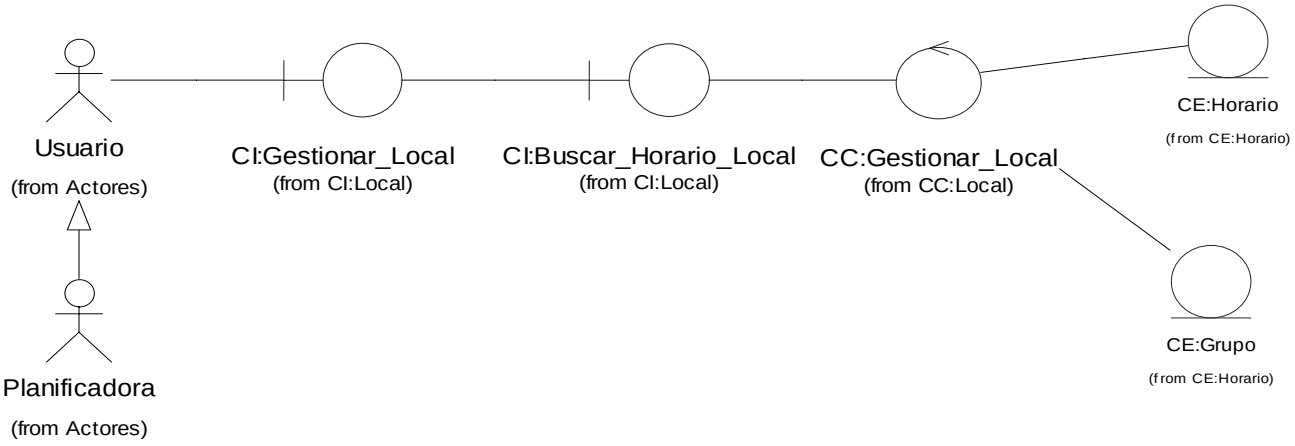
- **Diagramas de clases de análisis del caso de uso Registrar Local**



- **Diagramas de clases de análisis del caso de uso Mostrar Local**



- **Diagramas de clases de análisis del caso de uso Buscar Horario Local**



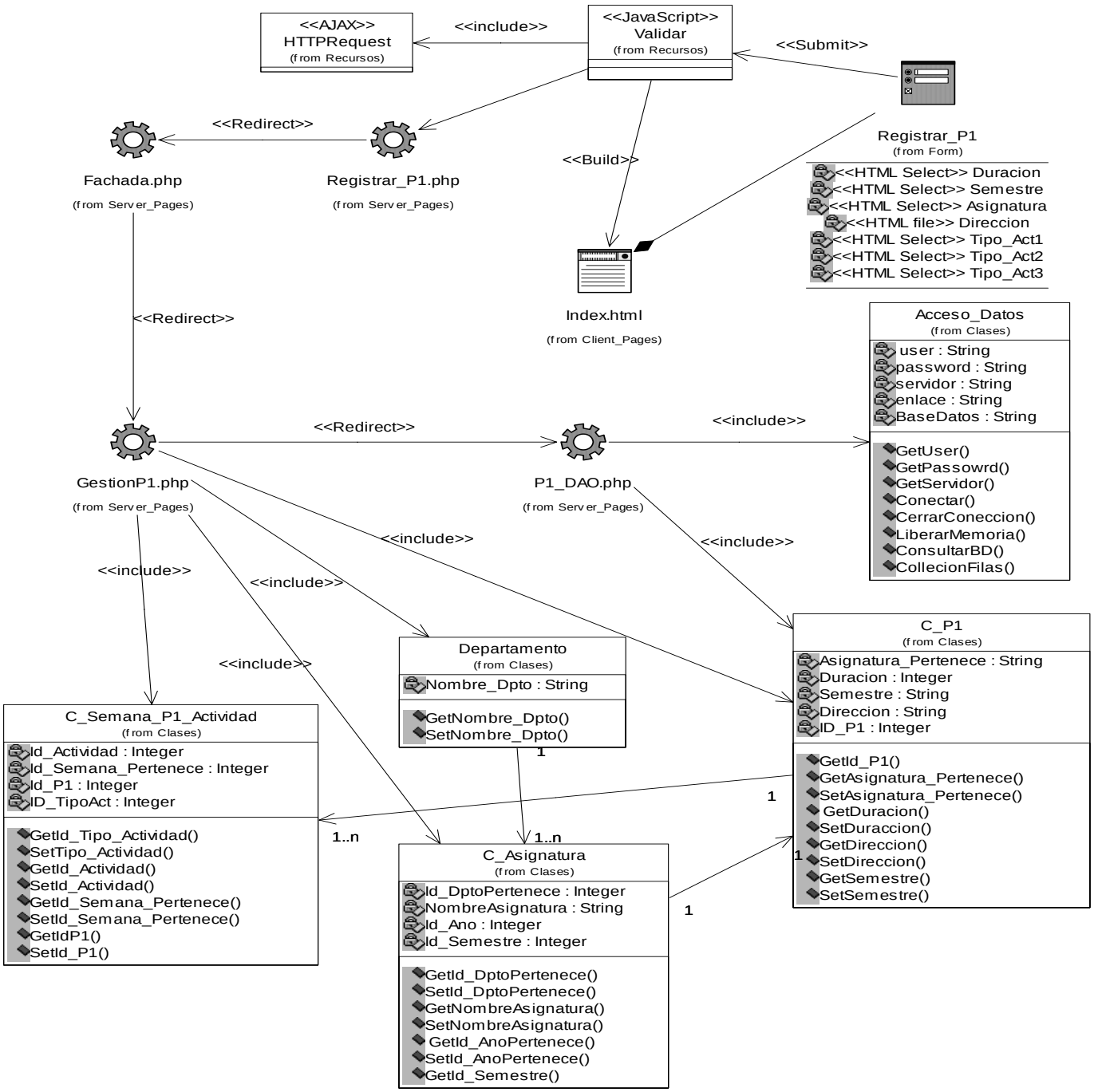
4.3 Diagrama de clases del diseño

Un diagrama de clases presenta las clases del sistema con sus relaciones estructurales y de herencia, todo el código que irá creando las páginas, así como el contenido dinámico de estas una vez que estén en el navegador del cliente. En el caso de las aplicaciones Web, el diagrama de clases representa las colaboraciones que ocurren entre las páginas, donde cada página lógica puede ser representada como una clase, es muy importante pues estos son los artefactos que se necesitan modelar para que el desarrollador los implemente y obtener así el producto final con la calidad requerida. Al tratar de utilizar el diagrama de clases tradicional para modelar aplicaciones Web surgen varios problemas, por lo cual los especialistas del Rational plantearon la creación de una extensión al modelo de análisis y diseño que permitiera representar el nivel de abstracción adecuado y la relación con los restantes artefactos de UML. [15]

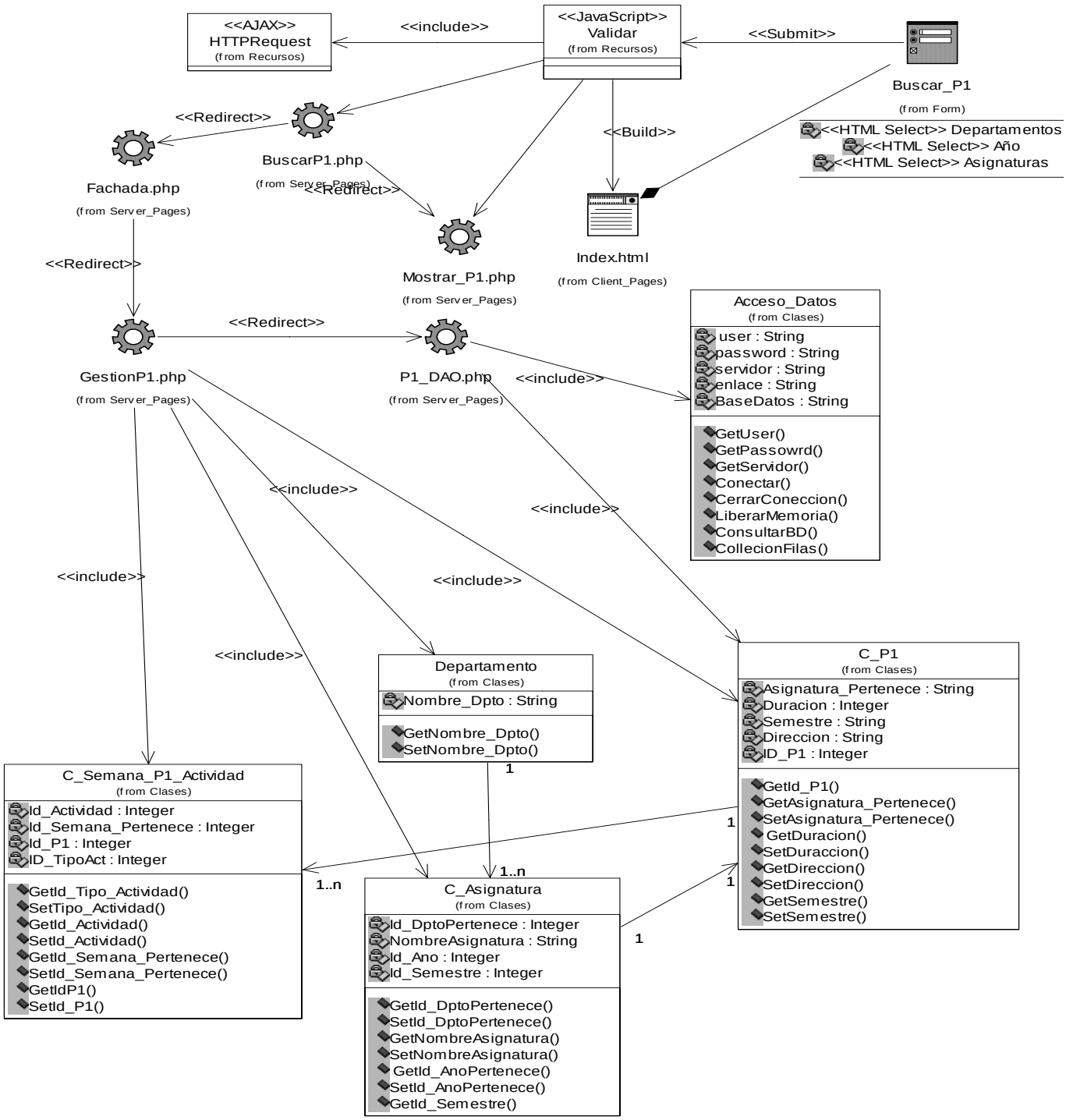
El diagrama de clases Web, fue definido, a partir de los diferentes casos de uso del sistema y empleando las extensiones de UML para Web, a continuación se muestran los diagramas de clases para los distintos Casos de Uso.

4.3.1 Diagramas de clases Web de Gestión de P1

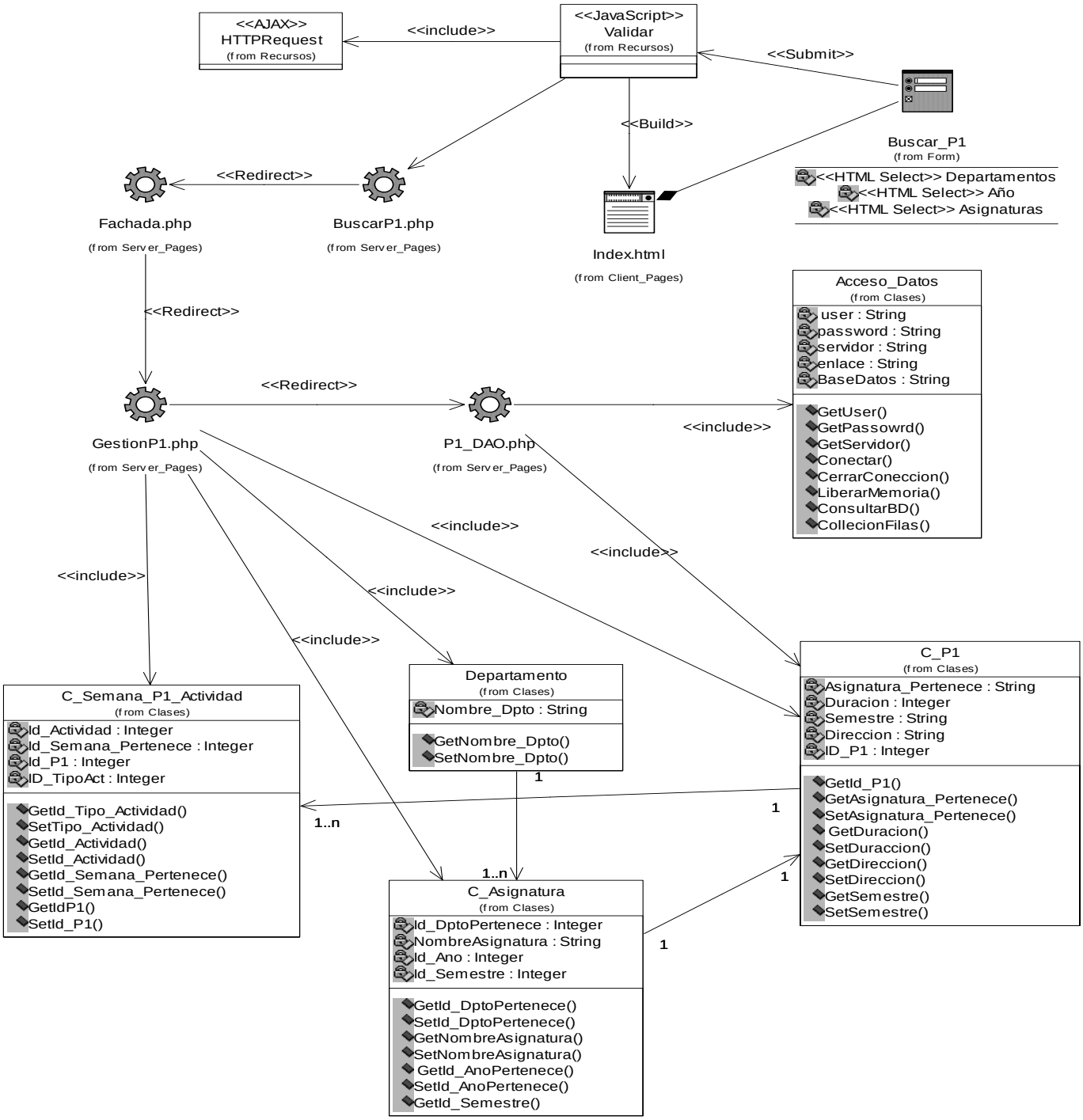
- **Diagrama de clases Web del Caso de Uso de Registrar P1**



- **Diagrama de clases Web del Caso de Uso de Mostrar P1**

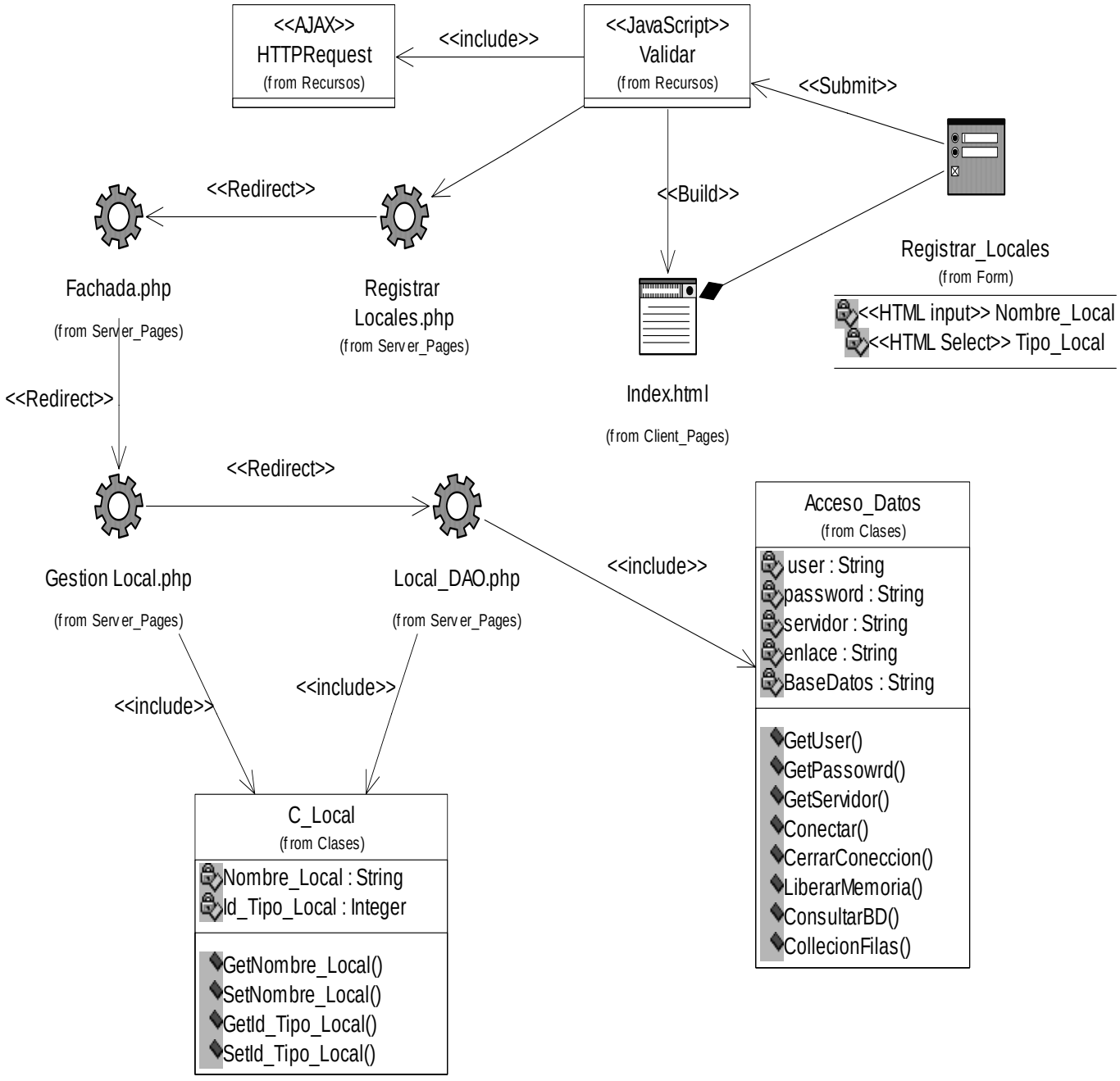


- **Diagrama de clases Web del Caso de Uso de Buscar P1**

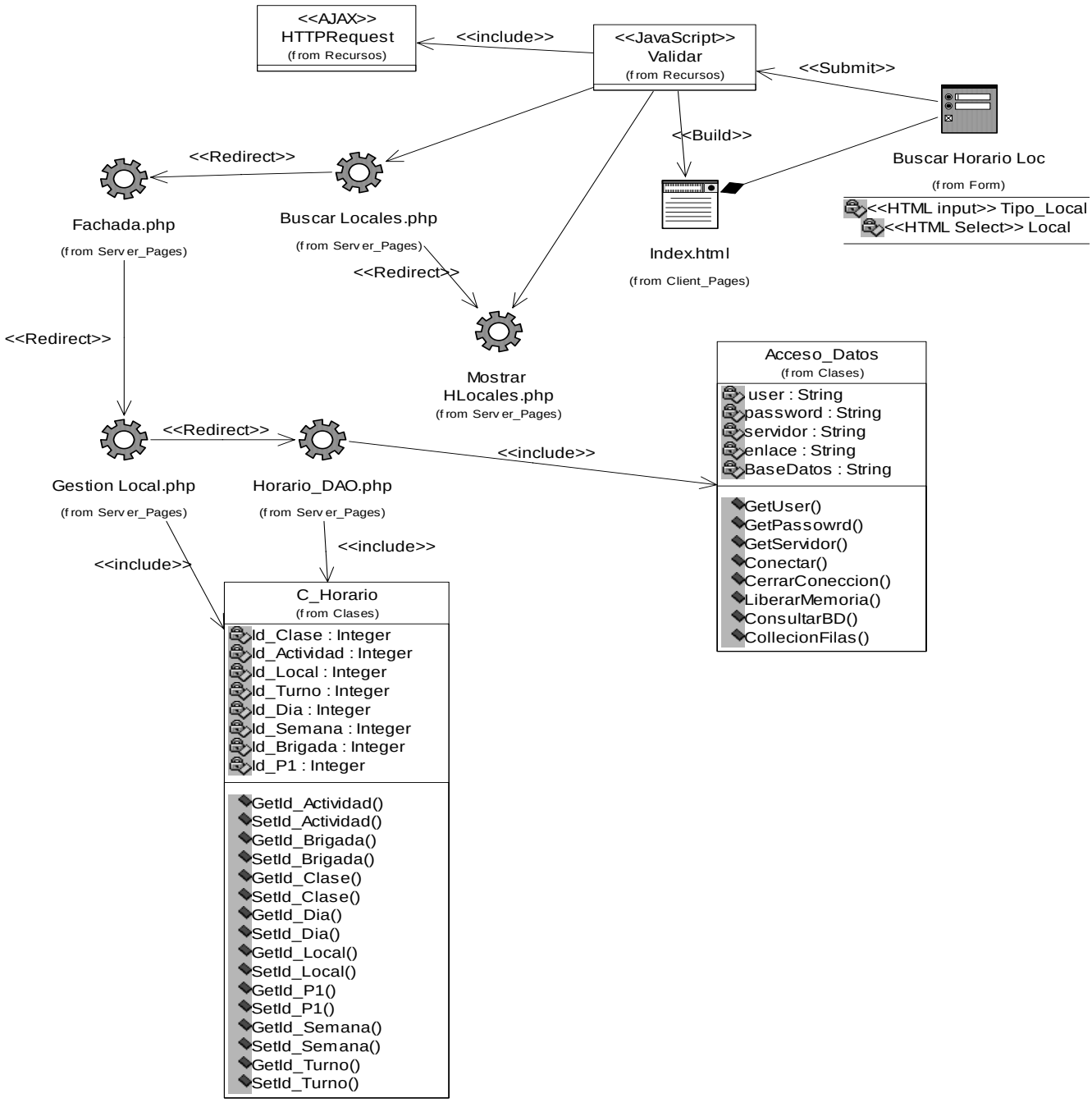


4.3.2 Diagrama de clases Web de Gestión de Locales

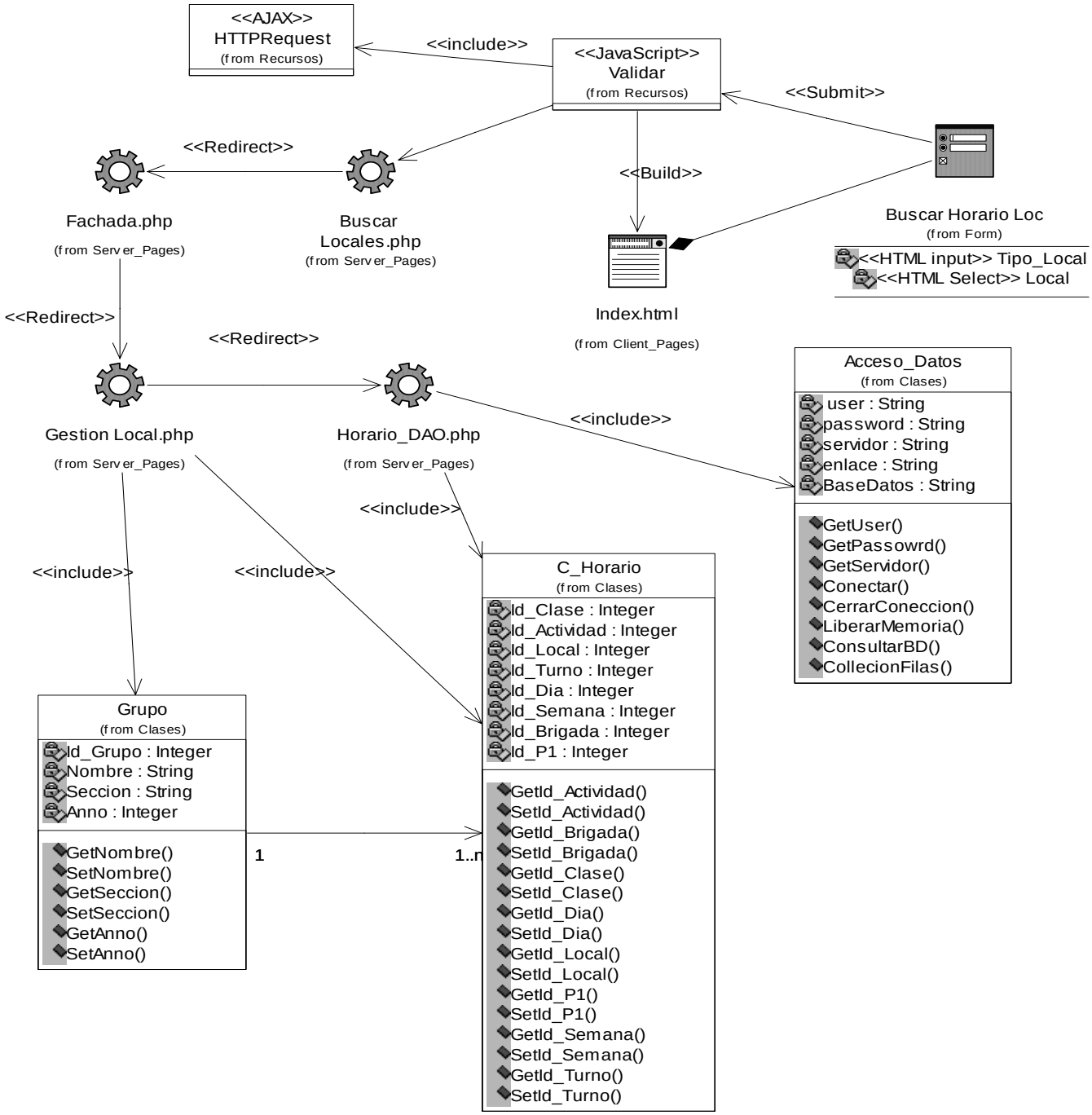
- **Diagrama de clases Web del Caso de Uso Registrar Locales**



- **Diagrama de clases Web del Caso de Uso Mostrar Horario Locales**

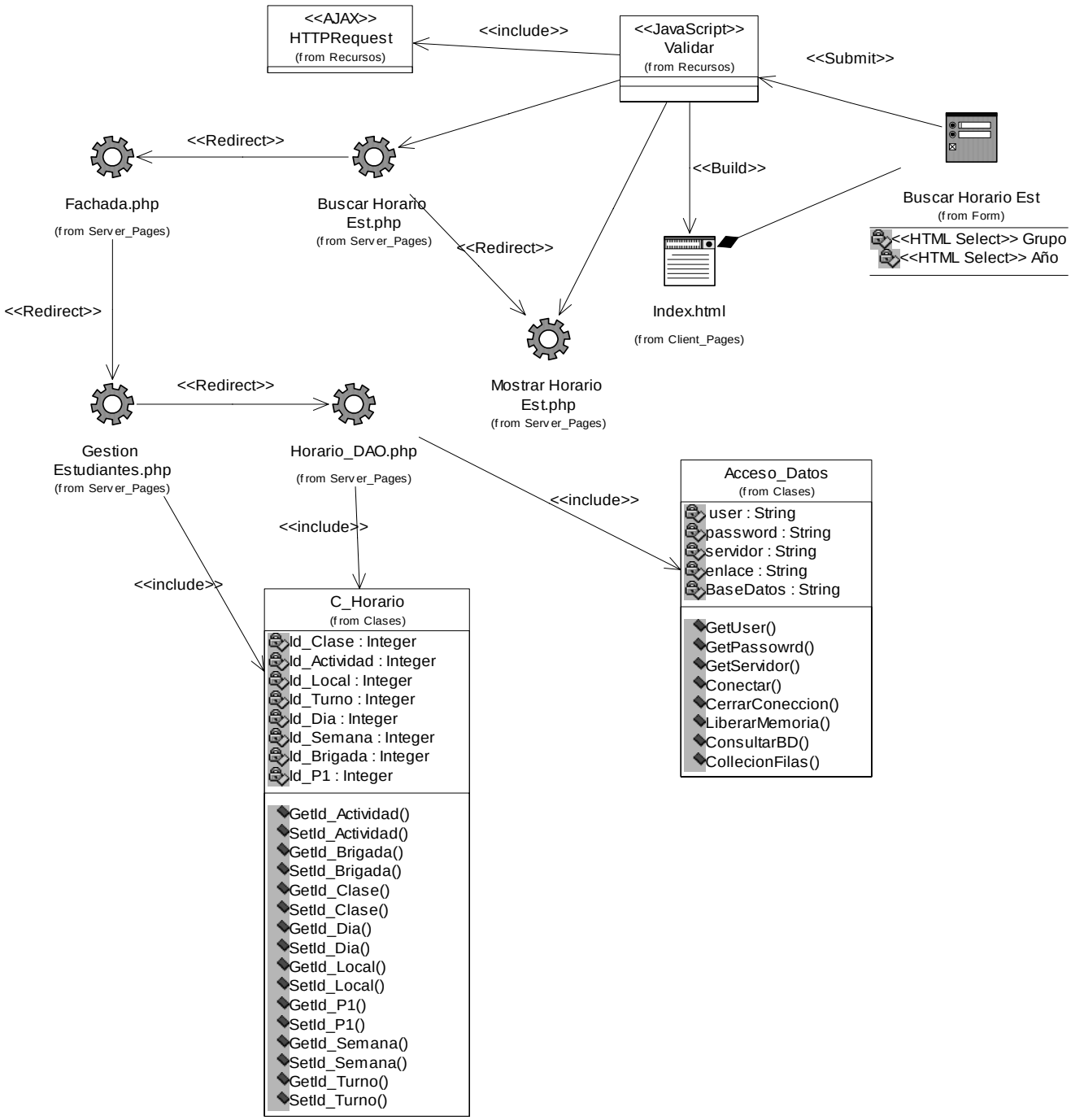


- **Diagrama de clases Web del Caso de Uso Buscar Horario Locales**

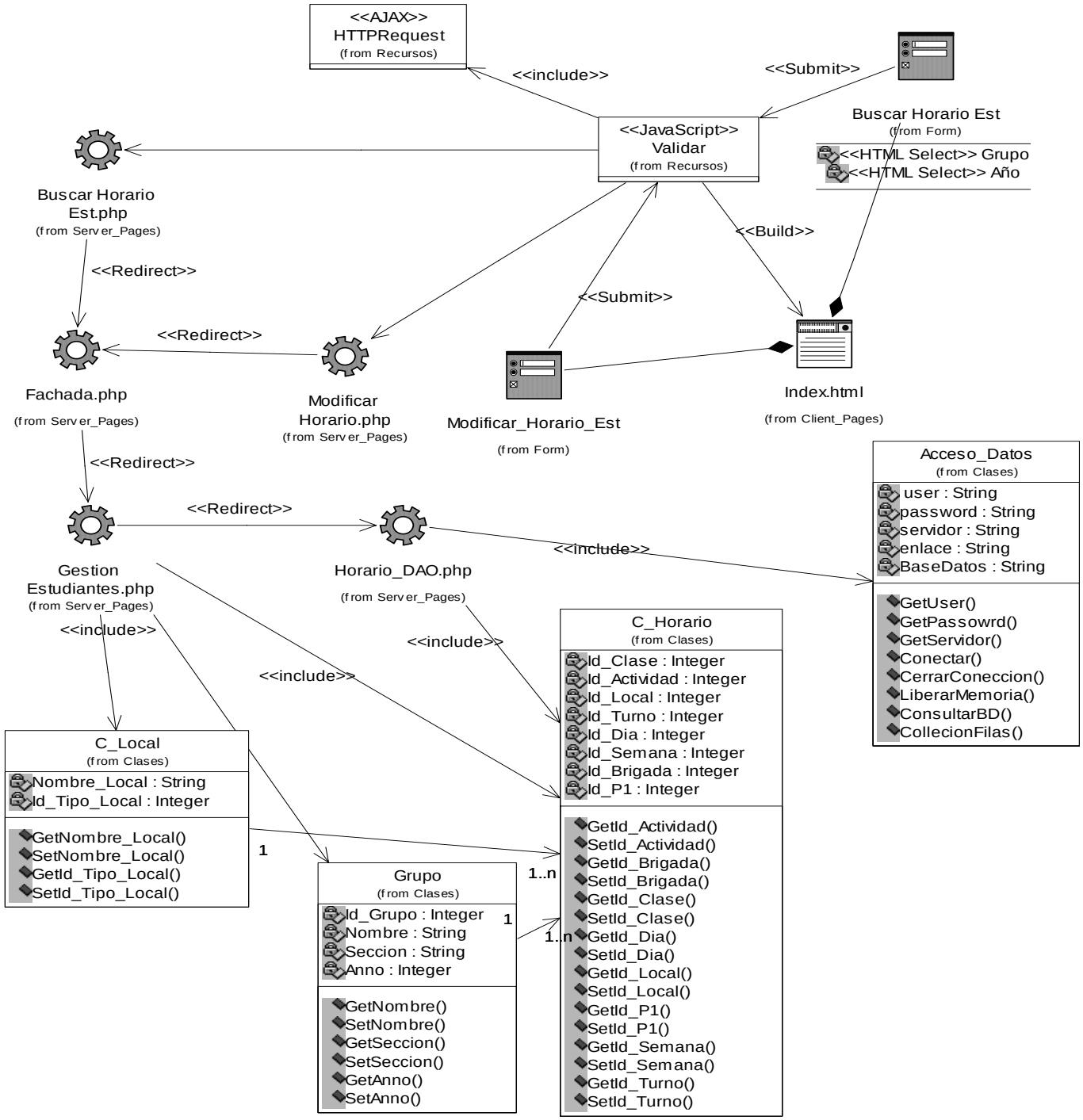


4.3.3 Diagrama de clases Web de Gestión de Estudiantes

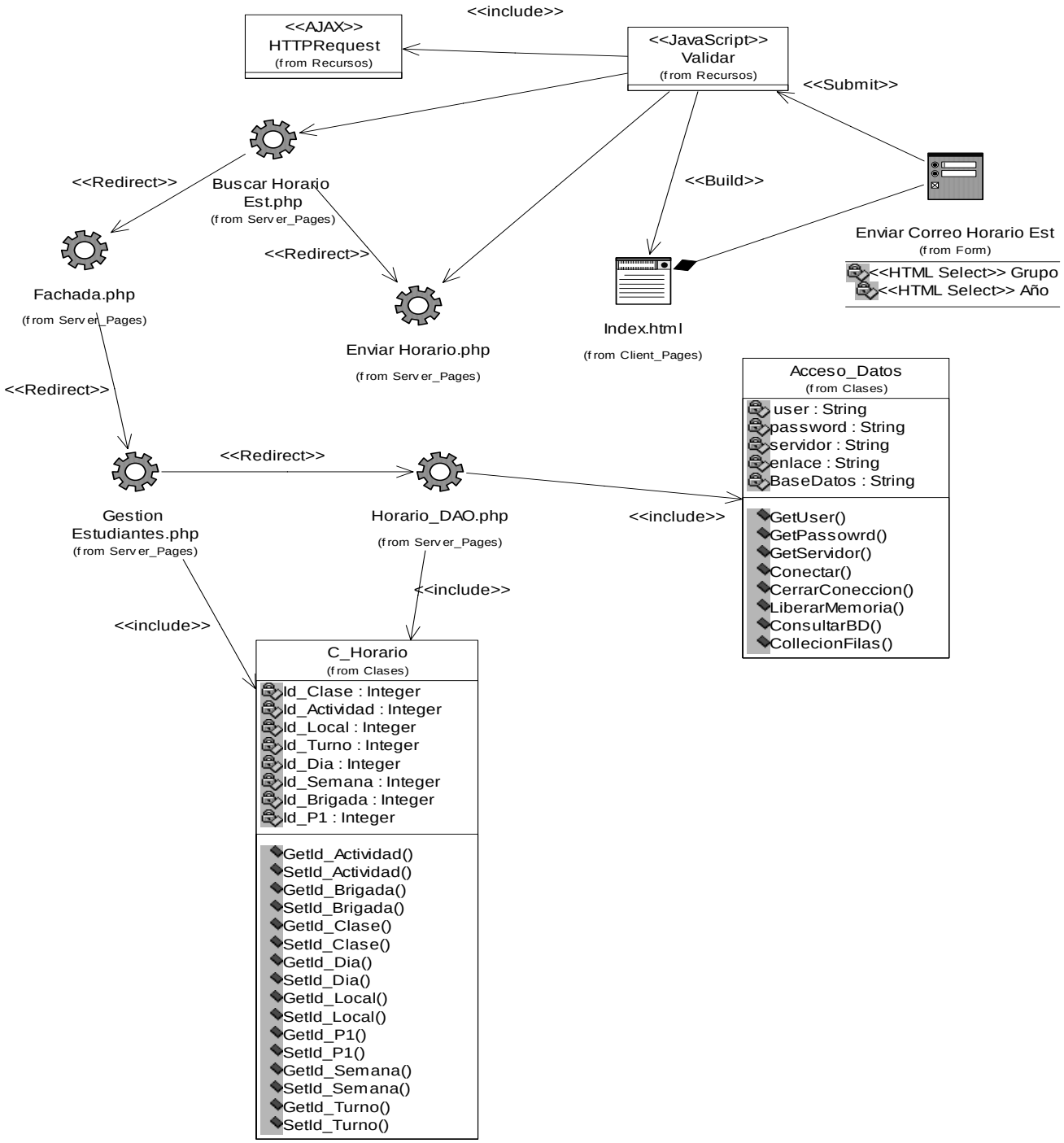
- **Diagrama de clases Web del Caso de Uso Mostrar Horario Estudiantes**



- **Diagrama de clases Web del Caso de Uso Modificar Horario Estudiantes**

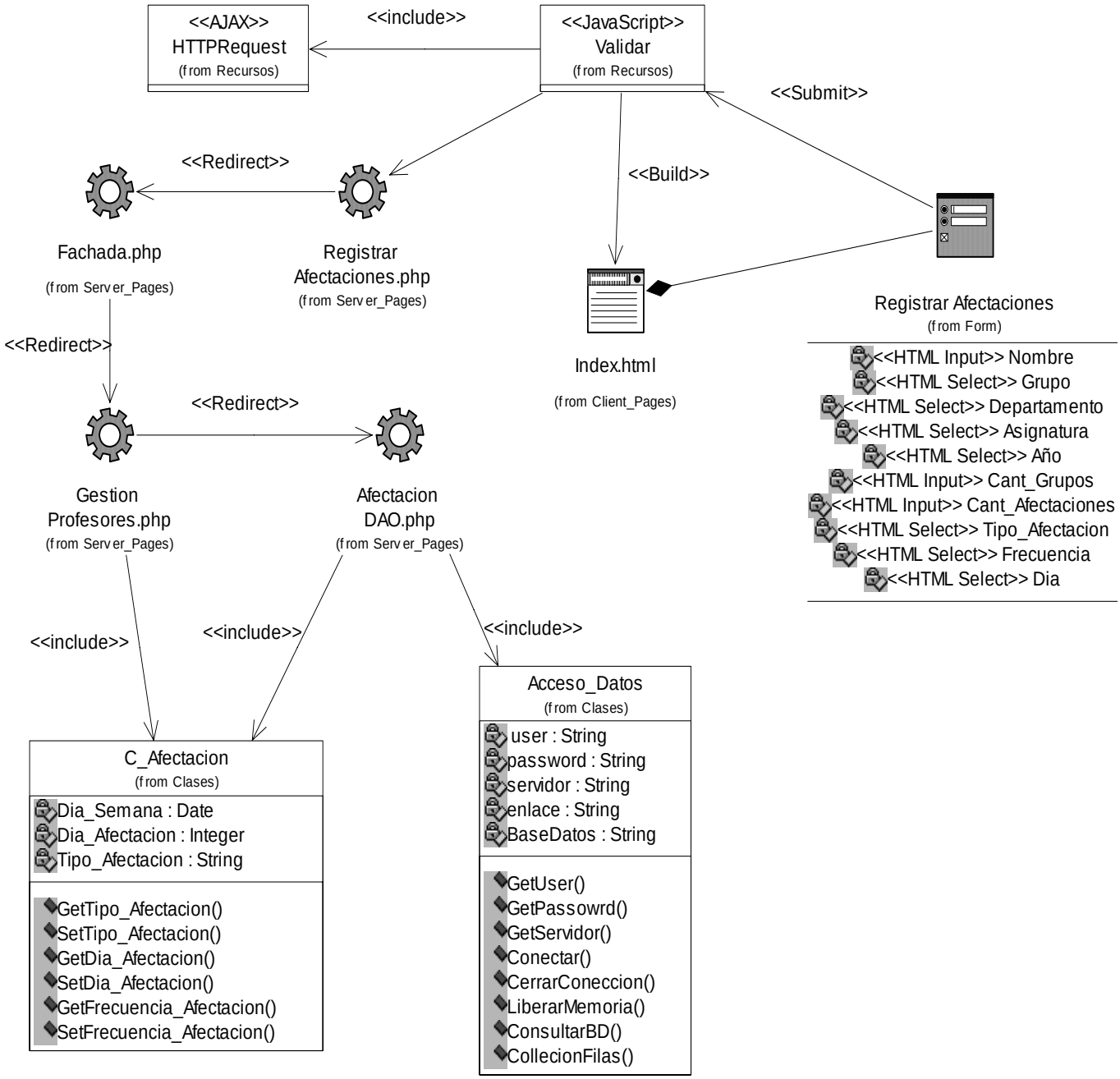


- **Diagrama de clases Web del Caso de Uso Enviar Horario Estudiantes**

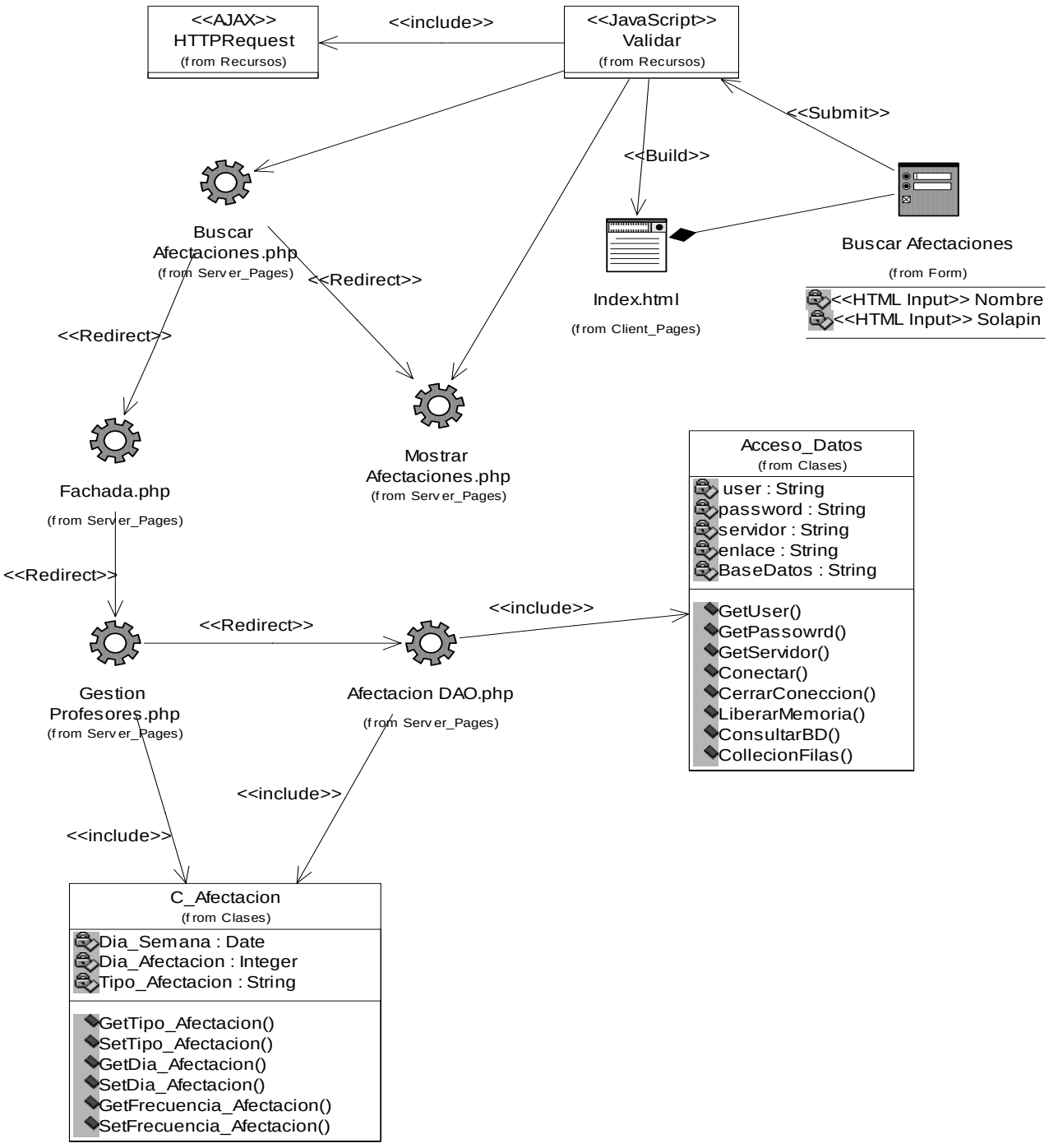


4.3.4 Diagrama de clases Web de Gestión de Profesores

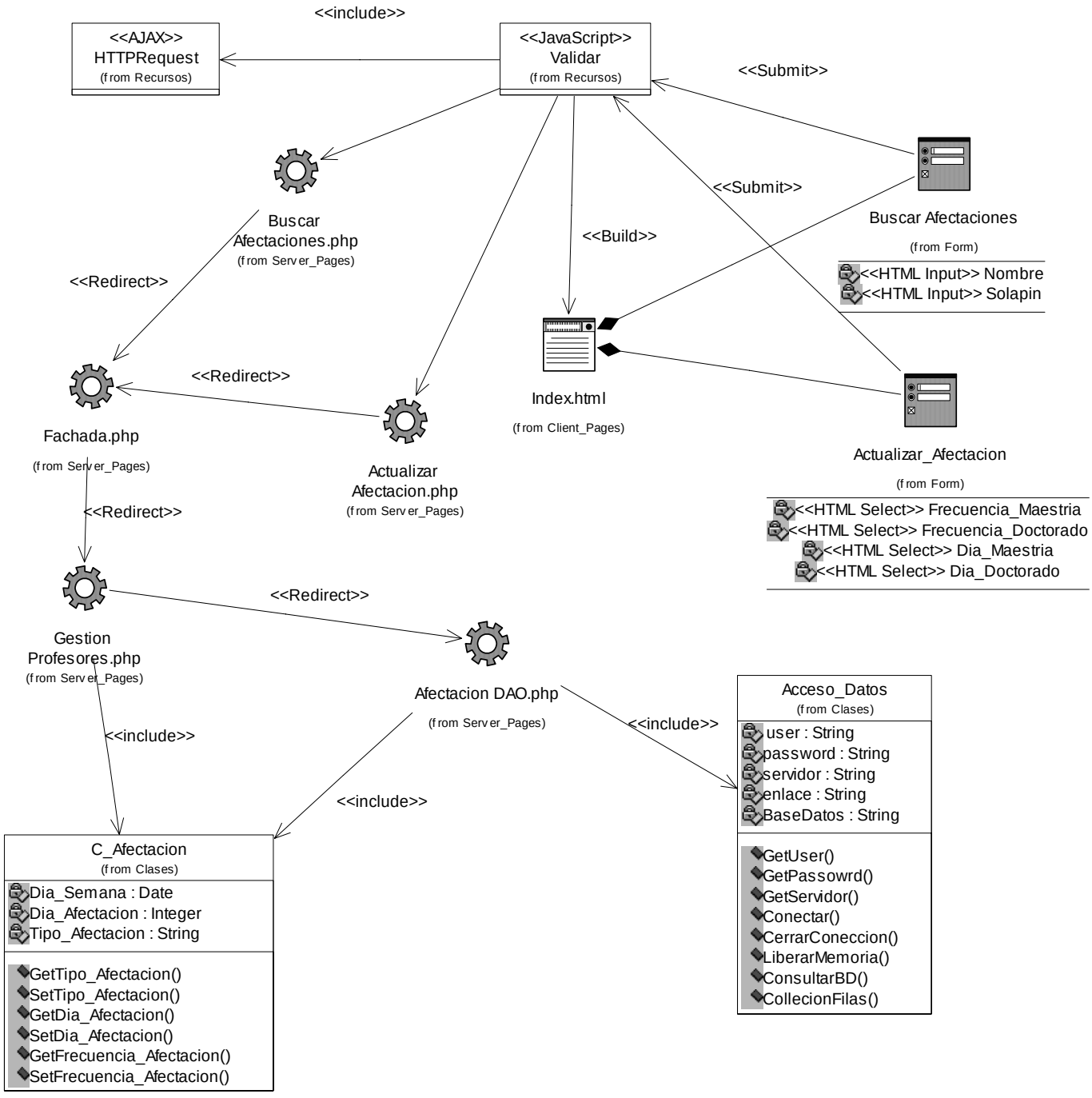
- **Diagrama de clases Web del Caso de Uso Registrar Afectaciones**



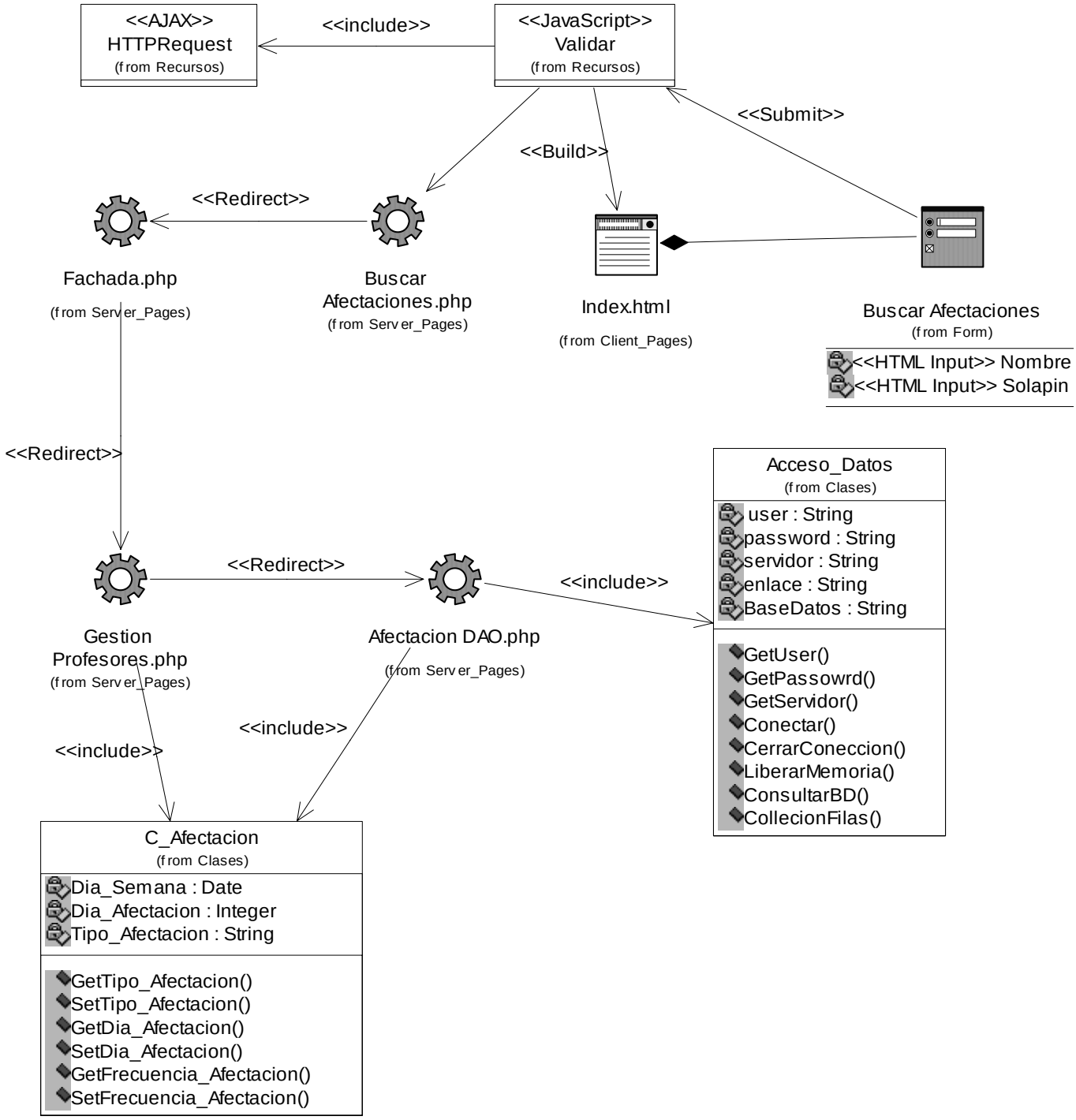
- **Diagrama de clases Web del Caso de Uso Mostrar Afectaciones**



- **Diagrama de clases Web del Caso de Uso Modificar Afectaciones:**



- **Diagrama de clases Web del Caso de Uso Buscar Afectaciones**



4.4 Descripción de Las Clases

4.4.1 Descripción de la Clase Fachada

Nombre: Fachada	
Tipo de clase : Controladora	
Atributo	**Tipo**
ObjGestiónP1	GestiónP1
ObjGestiónLocales	GestiónLocal
objGestiónEstudiantes	GestiónEstudiantes
ObjGestiónProfesores	GestiónProfesores
Para cada responsabilidad:	
Nombre:	Registrar_P1 (Asignatura, DirDoc, Duración, Semestre). ExisteP1 (IdAsignatura). ListarDptos (). Registrar_Semana_P1_Actividad (Id_Actividad, IdTipo_Actividad, Id_Semana_Pertenece, Id_P1). Buscar_P1 (Asignatura, Año, Dpto). Mostrar_Asignaturas (Dpto, Año). MostrarTodosP1 (). ActualizarDatosP1 (Asignatura, DireccionDoc, Durac, Semest, ID_P1). Actualizar_Actividad_Semana_P1(Id_P1,id_Semana,Id_Actividad,Id_TipoActividad) MostrarTodosDptos (). Buscar_Actividades_P1 (Id_Semana_Pertenece, Id_P1). Buscar_Asignatura_Dpto_Ano (Id_Año, Id_Dpto). ExisteLocal (Id_tipo_local, Nombre_local). RegistrarLocal (Id_tipo_local, Nombre_local). Buscar_Local (IdTipoLocal). Buscar_Horario_Local (Id_Local). Existe_Asignatura (Nombre_Asignatura, Id_Semestre, Id_Año, Id_Dpto). Insertar_Asignatura (Nombre_Asignatura, Id_Semestre, Id_Año, Id_Dpto). Buscar_Grupos (Id_Año). MostrarGrupos_Año(NombreG,SeccionG,AñoCursaG) Buscar_Asignatura(Id_Ano) Buscar_Actuvidades_Libres_Asignatura(Id_Asigantura) Buscar_Afectacion_Asignatura_Grupo(Id_Asignatura,Id_Grupo) Buscar_Locales_disponibles(Id_Semana,Id_Día,Id_turno) Insertar_Horario(ID_Actividad,Id_Local,Id_Turno,Id_Día,Id_Semanal,Id_Brigada,id_P1) Buscar_SemanaP1_Asignatura_Semana(Id_Asigantura,Id_Semana) Buscar_Horario_Estudiantes(Id_Brigada) Actualizar_Horario_estudiantes(ID_Actividad,Id_Local,Id_Turno,Id_Día,Id_Semana,Id_Brigada,id_P1,id_Clase) Eliminar_Turno_Horario_Estudiantes(ID_Clase) Buscar_Correo_Estudiantes_Grupo(Id_Brigada) Insertar_Tiene_Afectación(Id_Afectación,Identificador_Persona,Día_Semana,Id_Frecuencia)

	Insertar_ImparteClase(Brigada,Identificador_Persona) Insertar_Imparte_Asignatura(Id_Asignatura,Identificador_Persona) Insertar_Docente(Id_Dpto,Identificador_Persona) Buscar_Docente(PNombre,SNombre,PApellido,SApellido) Buscar_Horario_Profesor(Id_Profesor) Buscar_Afectaciones_Profesor(Id_Docente) Buscar_Profesor(PNombre,SNombre,PApellido,SApellido,Solapín,Foto,Categoría_Docente,Categoria_Científica)
Descripción:	Es la clase principal que realiza todas generalidades del Módulo de Planificación.

4.4.2 Descripción de las clases Controladoras de Gestión

Nombre: GestiónEstudiantes	
Tipo de clase : Controladora	
Atributo	**Tipo**
OBJ_DAOAsigantura	Asignatura_DAO
OBJ_DAOGrupo	Grupo_DAO
OBJ_DAOAfectación	Afectación_DAO
OBJ_DAOLocal	Local_DAO
OBJ_DAOHorario	Horario_DAO
OBJ_DAOEstudiante	Estudiante_DAO
OBJ_DAOSemana_P1_Actividad	Semana_P1_Actividad_DAO
Para cada responsabilidad:	
Nombre:	Existe_Asignatura(Nombre_Asignatura,Id_Semestre,Id_Año,Id_Dpto) Insertar_Asignatura(Nombre_Asignatura,Id_Semestre,Id_Año,Id_Dpto) Buscar_Grupos_Año(NombreG,SeccionG,AnoCursaG) Buscar_Actuvidades_Libres_Asignatura(Id_Asigantura) Buscar_Asignatura(Id_Año) Buscar_Grupos(Id_Año) Buscar_Afectación_Asignatura_Grupo(Id_Asignatura,Id_Grupo) Buscar_Locales_disponibles(Id_Semana,Id_Día,Id_turno) Insertar_Horario(Id_Actividad, Id_Local, Id_Turno, Id_Día, Id_Semana, Id_Brigada, id_P1) Buscar_SemanaP1_Asignatura_Semana(Id_Asigantura,Id_Semana) Buscar_Horario_Estudiantes(Id_Brigada) Buscar_Horario_Local(Id_Local) Actualizar_Horario_estudiantes(ID_Actividad,Id_Local,Id_Turno,Id_Día,Id_Semana,Id_Brigada,id_P1,id_Clase) Eliminar_Turno_Horario_Estudiantes(ID_Clase) Buscar_Correo_Estudiantes_Grupo(Id_Brigada)
Descripción:	Esta clase me permite gestionar todo lo que tiene que ver con los estudiantes en el Módulo de planificación Docente.

Nombre: GestiónLocales	
Tipo de clase : Controladora	
Atributo	**Tipo**
Obj_DAOLocal	Local_DAO
Para cada responsabilidad:	
Nombre:	Existe_Local(MId_Tipo_Local,Nombre_Local) RegistrarLocal(Id_Tipo_Local,Nombre_Local) Buscar_Local(IdTipoLocal)
Descripción:	Esta clase me permite gestionar todo lo que tiene que ver con los Locales en el Módulo de planificación Docente.

Nombre: GestiónP1	
Tipo de clase : Controladora	
Atributo	**Tipo**
Obj_DAOP1	P1_DAO
Obj_DAOSemana_P1	Semana_P!_DAO
Obj_DAOAsignatura	Asignatura_DAO
Obj_DAODepartamento	Departamento_DAO
Obj_DAOSemana_P1_Actividad	Semana_P1_Actividad_DAO
Para cada responsabilidad:	
Nombre:	InsertarP1(Asignatura,DirDoc,Duración,Semestre) ExisteP1(IdAsignatura) ListarDptos() Insertar_Semana_P1_Actividad(Id_Actividad,Id_Tipo_Actividad,Id_Semana_Pertenece,Id_P1) Buscar_P1(Asignatura,año,Dpto) Mostrar_Asigaturas(Dpto,Año) Mostrar_TodosP1() Buscar_Actividad_P1(Id_Semana_Pertenece,Id_P1) ActualizarDatosP1(Asignatura,DirecciónDoc,Durac,Semest,ID_P1) Buscar_Asignatura_Dpto_Año(Id_Año,Id_Dpto) Actualizar_Actividad_Semana_P1(Id_P1,id_Semana,Id_Actividad,Id_TipoActividad) BuscarTodosDptos()
Descripción:	Esta clase me permite gestionar todo lo que tiene que ver con los P1 en el Módulo de planificación Docente.

Nombre: GestiónProfesores	
Tipo de clase : Controladora	
Atributo	**Tipo**
Obj_DAOAfectación	Afectación_DAO
ObjDAOTiene_Afectación	Tiene_Afectación_DAO
ObjDAOImparteClase	ImparteClase_DAO
ObjDAOImparte_Asignatura	Imparte_Asignatura_DAO

ObjDAODocente	Docente_DAO
ObjDAOHorario	Horario_DAO
Para cada responsabilidad:	
Nombre:	Insertar_Afectación(TipoAfectación,frecuencia,Día) Buscar_Profesores_Nombre(PNombre,SNombre,PApellido,SApellido,Solapín,Foto,Categoría_Docente,Categoria_Científica) Insertar_Tiene_Afectación(Id_Afectación,Identificador_Persona,Día_Semana,Id_Frecuencia) Insertar_ImparteClase(BrigadaIdentificador_Persona) Insertar_Imparte_Asignatura(Id_Asignatura,Identificador_Persona) Insertar_Docente(Id_Dpto,Identificador_Persona) Buscar_Docente(PNombre,SNombre,PApellido,SApellido) Buscar_Afectaciones_Profesor(Id_Docente) Buscar_Horario_Profesor(Id_Profesor)
Descripción:	Esta clase me permite gestionar todo lo que tiene que ver con los Profesores en el Módulo de planificación Docente.

4.4.3 Descripción de las clases Controladoras DAO

Nombre: Afectación_DAO	
Tipo de clase : Controladora	
Atributo	**Tipo**
ObjAfectación	Afectación
ObjBD	PG_Connection
Para cada responsabilidad:	
Nombre:	Insertar_Afectación() Buscar_Afectación_Asignatura_Grupo(Id_Asignatura,Id_Grupo) Eliminar_Afectación()
Descripción:	Esta clase permite Acceder a la base de datos para retornar las consultas para obtener los valores.

Nombre: Asignatura_DAO	
Tipo de clase: Controladora	
Atributo	**Tipo**
ObjAsignatura	Asignatura
ObjBD	PG_Connection
Para cada responsabilidad:	
Nombre:	Existe_Asigantura() Insertar_Asignatura() Buscar_Asignatura() Buscar_Asignatura_Dpto_Ano()
Descripción:	Esta clase permite Acceder a la base de datos para retornar las consultas para obtener los valores.

Nombre: Departamento_DAO	
Tipo de clase: Controladora	
Atributo	**Tipo**
ObjDpto	Departamento
ObjBD	PG_Connection
Para cada responsabilidad:	
Nombre:	ListarDptos()
Descripción:	Esta clase permite Acceder a la base de datos para retornar las consultas para obtener los valores.

Nombre: Docente_DAO	
Tipo de clase :controladora	
Atributo	**Tipo**
ObjDocen	Docente
ObjBD	PG_Connection
Para cada responsabilidad:	
Nombre:	Insertar_Docente() Buscar_Docente(PNombre,SNombre,PApellido,SApellido)
Descripción:	Esta clase permite Acceder a la base de datos para retornar las consultas para obtener los valores.

Nombre: Estudiante_DAO	
Tipo de clase :controladora	
Atributo	**Tipo**
ObjEstudiante	Estudiante
ObjBD	PG_Connection
Para cada responsabilidad:	
Nombre:	Buscar_Correo_Estudiantes_Grupo() Actualizar_Grupo() Eliminar_Grupo()
Descripción:	Esta clase permite Acceder a la base de datos para retornar las consultas para obtener los valores

Nombre: Grupo_DAO	
Tipo de clase :controladora	
Atributo	**Tipo**
ObjGrupo	Grupo
ObjBD	PG_Connection
Para cada responsabilidad:	
Nombre:	Insertar_Grupo() Buscar_Todos_Grupos() Buscar_Grupos() Actualizar_Grupo() Eliminar_Grupo()

Descripción:	Esta clase permite Acceder a la base de datos para retornar las consultas para obtener los valores

Nombre: Horario_DAO		
Tipo de clase :controladora		
Atributo		**Tipo**
ObjHorario		Horario
ObjBD		PG_Connection
Para cada responsabilidad:		
Nombre:	Insertar_Horario() Buscar_Horario_Estudiantes() Actualizar_Horario_estudiantes() Eliminar_Turno_Horario_Estudiantes() Buscar_Horario_Local() Buscar_Horario_Profesor(Id_Profesor)	
Descripción:	Esta clase permite Acceder a la base de datos para retornar las consultas para obtener los valores	

Nombre: Imparte_Asignatura_DAO		
Tipo de clase :controladora		
Atributo		**Tipo**
ObjImparte_Asignatura		Imparte_Asignatura
ObjBD		PG_Connection
Para cada responsabilidad:		
Nombre:	Insertar_Imparte_Asignatura()	
Descripción:	Esta clase permite Acceder a la base de datos para retornar las consultas para obtener los valores	

Nombre: ImparteClase_DAO		
Tipo de clase :controladora		
Atributo		**Tipo**
ObjImparteC		Imparte_Clase
ObjBD		PG_Connection
Para cada responsabilidad:		
Nombre:	Insertar_ImparteClase()	
Descripción:	Esta clase permite Acceder a la base de datos para retornar las consultas para obtener los valores	

Nombre: Local_DAO		
Tipo de clase :controladora		
Atributo		**Tipo**
ObjLocal		Local
ObjBD		PG_Connection
Para cada responsabilidad:		
Nombre:	ExisteLocal()	

	RegistrarLocal() Buscar_Local() Buscar_Locales_disponibles(Id_Semana,Id_Día,Id_turno)
Descripción:	Esta clase permite Acceder a la base de datos para retornar las consultas para obtener los valores

Nombre: P1_DAO	
Tipo de clase :controladora	
Atributo	**Tipo**
ObjP1	P1
ObjBD	PG_Connection
Para cada responsabilidad:	
Nombre:	Buscar_P1(año,Dpto) Actualizar_P1() Buscar_Actividades_P1() Eliminar_P1() ListarDptos() ExisteP1() Insertar_P1() Listar_Todos_P1()
Descripción:	Esta clase permite Acceder a la base de datos para retornar las consultas para obtener los valores

Nombre: Semana_P1_Actividad_DAO	
Tipo de clase :controladora	
Atributo	**Tipo**
ObjSemP1_Actividad	Semana_P1_Actividad
ObjBD	PG_Connection
Para cada responsabilidad:	
Nombre:	Insertar_Actividad_Semana_P1() Buscar_Actividad_Semana_P1() Buscar_Actuvidades_Libres_Asignatura(Id_Asigantura) Actualizar_Actividad_Semana_P1() Buscar_SemanaP1_Asignatura_Semana(Id_Asigantura,Id_Semana)
Descripción:	Esta clase permite Acceder a la base de datos para retornar las consultas para obtener los valores

Nombre: Tiene_Afectación_DAO	
Tipo de clase :controladora	
Atributo	**Tipo**
ObjTiene_Afect	Tiene_Afectación
ObjBD	PG_Connection
Para cada responsabilidad:	
Nombre:	Insertar_Tiene_Afectación() Buscar_Afectaciones_Profesor()
Descripción:	Esta clase permite Acceder a la base de datos para retornar las consultas

	para obtener los valores

Nombre: Afectación		
Tipo de clase :Entidad		
Atributo		**Tipo**
Tipo_Afectación		string
Frecuencia_Afectación		string
Día_Afectación		integer
Para cada responsabilidad:		
Nombre:	GetTipo_Afectación() SetTipo_Afectación(New_Tipo) GetDia_Afectación() SetDia_Afectación(New_Día) GetFrecuencia_Afectación() SetFrecuencia_Afectación(New_Afectación)	
Descripción:	Esta clase es una clase entidad que nos permite obtener sus atributos.	

Nombre: Asignatura		
Tipo de clase : Entidad		
Atributo		**Tipo**
Id_DptoPertenece		integer
NombreAsignatura		string
Id_Ano		integer
Id_Semestre		integer
Para cada responsabilidad:		
Nombre:	GetId_DptoPertenece() Actualizar_P1() SetId_DptoPertenece(New_NomDptoPertenece) GetNombreAsignatura() SetNombreAsignatura(New_NombreAsignatura) GetId_AnoPertenece() SetId_AnoPertenece(New_Ano) GetId_Semestre()	
Descripción:	Esta clase es una clase entidad que nos permite obtener sus atributos.	

Nombre: Estudiante		
Tipo de clase : Entidad		
Atributo		**Tipo**
Id_Brigada		interger
Id_Año		interger
Identificador		interger
Para cada responsabilidad:		
Nombre:	GetId_Brigada() SetId_Brigada(NewId_Brigada) GetId_Año() SetId_Año(NewId_Año)	

	GetIdentificador() SetIdentificador(NewIdentificador)
Descripción:	Esta clase es una clase entidad que nos permite obtener sus atributos.

Nombre: Grupo	
Tipo de clase : Entidad	
Atributo	**Tipo**
Nombre_Grupo	string
Sección	string
AñoPet	interger
Para cada responsabilidad:	
Nombre:	GetAño_Pertenece() SetAño_Pertenece(New_Año_Pertenece) GetNombre_Grupo() SetNombre_Grupo(New_NameG) GetSección() SetSección(New_Seccion)
Descripción:	Esta clase es una clase entidad que nos permite obtener sus atributos.

Nombre: Horario	
Tipo de clase : Entidad	
Atributo	**Tipo**
Id_Clase	interger
Id_Actividad	interger
Id_Local	interger
Id_Turno	interger
Id_Día	interger
Id_Semana	interger
Id_Brigada	interger
Id_P1	interger
Para cada responsabilidad:	
Nombre:	GetId_Actividad() SetId_Actividad(New_Id_Actividad) GetId_Brigada() SetId_Brigada(New_Id_Brigada) GetId_Clase() SetId_Clase(New_Id_Clase) GetId_Día() SetId_Día(New_Id_Dia) GetId_Local() SetId_Local(New_Id_Local) GetId_P1()

	SetId_P1(New_Id_P1) GetId_Semana() SetId_Semana(New_Id_Semana) GetId_Turno() SetId_Turno(New_Id_Turno)
Descripción:	Esta clase es una clase entidad que nos permite obtener sus atributos.

Nombre: Imparte_Asignatura		
Tipo de clase : Entidad		
Atributo		**Tipo**
IdAsignat		inerger
IdDocente		interger
Para cada responsabilidad:		
Nombre:	SetId_Asignatura(New_Id_Asignatura) GetId_AsiganturaPertenece() GetId_Docente() SetId_Docente(New_Id_Docente)	
Descripción:	Esta clase es una clase entidad que nos permite obtener sus atributos.	

Nombre: ImparteClase		
Tipo de clase : Entidad		
Atributo		**Tipo**
IdBrigada		inerger
IdDocent		inerger
Para cada responsabilidad:		
Nombre:	GetId_Brigada() SetId_Brigada(New_Id) GetId_Docente() SetIdDocente(NewId_Docente)	
Descripción:	Esta clase es una clase entidad que nos permite obtener sus atributos.	

Nombre: Local		
Tipo de clase : Entidad		
Atributo		**Tipo**
Nombre_Local		string
Id_Tipo_Local		interger
Para cada responsabilidad:		
Nombre:	GetNombre_Local() SetNombre_Local(New_Nombre) GetId_Tipo_Local() SetId_Tipo_Local(NewIdTipo_Local)	
Descripción:	Esta clase es una clase entidad que nos permite obtener sus atributos.	

Nombre: P1	
Tipo de clase : Entidad	
Atributo	**Tipo**
Asignatura_Pertenece	string
Duración	string
Semestre	string
ID_P1	interger
Para cada responsabilidad:	
Nombre:	GetId_P1() GetAsignatura_Pertenece() SetAsignatura_Pertenece(PAsignatura_Pertenece) GetDuración() SetDuración(PDurac) GetDirección() SetDirección(PDircc) GetSemestre() SetSemestre(PSemest)
Descripción:	Esta clase es una clase entidad que nos permite obtener sus atributos.

Nombre: Semana_P1_Actividad	
Tipo de clase : Entidad	
Atributo	**Tipo**
Id_Actividad	interger
Id_Semana_Pertenece	interger
Id_P1	interger
ID_TipoAct	interger
Para cada responsabilidad:	
Nombre:	GetId_Tipo_Actividad() SetTipo_Actividad(New_Tipo) GetId_Actividad() SetId_Actividad(New_Id_Actividad) GetId_Semana_Pertenece() SetId_Semana_Pertenece(New_Id_Semana_Pertenece) GetIdP1() SetId_P1(NewP1)
Descripción:	Esta clase es una clase entidad que nos permite obtener sus atributos.

Nombre: Persona	
Tipo de clase : Entidad	
Atributo	**Tipo**
PNombre	string
SNombre	string
PApellido	string
SApellido	string
Solapín	interger
Correo	string
Foto	string

Para cada responsabilidad:	
Nombre:	GetPNombre_Persona() SetPNombre_Persona(NewPNombre) GetSNombre() SetSNombre_Persona(NewSNombre) GetP_Apellido() SetP_Apellido(NewP_Apellido) GetS_Apellido() SetS_Apellido(NewS_Apellido) GetSolapín() SetSolapín(New_Solapin) GetFoto() SetFoto(New_Foto) GetCorreo() SetCorreo(New_Correo)
Descripción:	Esta clase es una clase entidad que nos permite obtener sus atributos.

Nombre: Tiene_Afectación	
Tipo de clase : Entidad	
Atributo	**Tipo**
IdAfect	interger
IdentificadorPersona	interger
Día_Semana	interger
Id_Frecuencia	interger
Para cada responsabilidad:	
Nombre:	GetId_Afectación() SetId_Afectación(NewId_Afectación) GetIdentificadorPersona() GetDía_Semana() GetId_Frecuencia()
Descripción:	Esta clase es una clase entidad que nos permite obtener sus atributos.

Nombre: PG_Connection	
Tipo de clase : Entidad	
Atributo	**Tipo**
user	string
password	string
servidor	string
enlace	string
BaseDatos	interger
Para cada responsabilidad:	
Nombre:	GetUser() GetPassowrd() GetServidor() Conectar()

	CerrarConección() LiberarMemoria() ConsultarBD(consulta) CollecionFilas()
Descripción:	Esta clase es una clase entidad que nos permite obtener sus atributos.

4.4.5 Descripción de las clases Interfaz para la Gestión de Estudiantes

Nombre: Registrar Asignatura	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Año	<<HTML Select>>
Nombre	<<HTML Input>>
Departamento	<<HTML Select>>
Semestre	<<HTML Select>>
Para cada responsabilidad:	
Nombre:	Registrar Asignatura
Descripción:	Esta clase me muestra una Interfaz con los datos para registrar asignatura

Nombre: Modificar Horario	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Año	<<HTML Select>>
Grupo	<<HTML Select>>
Para cada responsabilidad:	
Nombre:	Mostrar Horario()
Descripción:	Esta clase me muestra una Interfaz para Modificar los datos.

Nombre: Crear Horario	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Año	<<HTML Select>>
Grupo	<<HTML Select>>
Para cada responsabilidad:	
Nombre:	**Crear Horario()**
Descripción:	Esta clase muestra una interfaz para registrar los Horarios de los estudiantes

Nombre: Buscar Horario_Est	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Año	<<HTML Select>>

Grupo	<<HTML Select>>
Para cada responsabilidad:	
Nombre:	**Buscar Horario Est()**
Descripción:	Esta clase muestra una interfaz para Buscar los Horarios de los estudiantes.

4.4.6 Descripción de las clases Interfaz para la Gestión de P1

Nombre: RegistrarP1	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Departamento	<<HTML Select>>
Año	<<HTML Select>>
Asignatura	<<HTML Select>>
Documento	<<HTML Input>>
Dirección	<<HTML Select>>
Semestre	<<HTML Select>>
Para cada responsabilidad:	
Nombre:	**RegistrarP1()**
Descripción:	Esta clase muestra una interfaz para Registrar los P1.

Nombre: BuscarP1	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Departamento	<<HTML Select>>
Año	<<HTML Select>>
Asignatura	<<HTML Select>>
Para cada responsabilidad:	
Nombre:	**BuscarP1()**
Descripción:	Esta clase muestra una interfaz para Buscar Los P1.

Nombre: ActualizarP1	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Departamento	<<HTML Select>>
Año	<<HTML Select>>
Asignatura	<<HTML Select>>
Para cada responsabilidad:	
Nombre:	**ActualizarP1()**
Descripción:	Esta clase muestra una interfaz para Actualizar Los P1.

4.4.7 Descripción de las clases Interfaz para la Gestión de Locales

Nombre: Registrar_Local	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Nombre Local	<<HTML Input>>
Tipo Local	<<HTML Select>>
Para cada responsabilidad:	
Nombre:	**Registrar_Local()**
Descripción:	Esta clase muestra una interfaz para Registrar Los Locales.

Nombre: Buscar_Local	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Tipo	<<HTML Select>>
Local	<<HTML Select>>
Para cada responsabilidad:	
Nombre:	**Buscar_Local**
Descripción:	Esta clase muestra una interfaz para Buscar Los Locales.

Nombre: Actualizar Horario Local	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Tipo	<<HTML Select>>
Local	<<HTML Select>>
Para cada responsabilidad:	
Nombre:	**Actualizar Horario Local**
Descripción:	Esta clase muestra una interfaz para Actualizar Los Locales.

4.4.8 Descripción de las clases Interfaz para la Gestión de Profesores

Nombre: RegistrarAfectaciones	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Nombre	<<HTML Input>>
Para cada responsabilidad:	
Nombre:	**RegistrarAfectaciones()**
Descripción:	Esta clase muestra una interfaz para Registrar Las afectaciones de los profesores.

Nombre: Buscar_Horario Prof	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Nombre	<<HTML Input>>
Solapín	<<HTML Input>>
Para cada responsabilidad:	

Nombre:	**Buscar_Horario()**
Descripción:	Esta clase muestra una interfaz para Buscar el Horario a los profesores.
Nombre: Buscar_Afectación	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Nombre	<<HTML Input>>
Solapín	<<HTML Input>>
Para cada responsabilidad:	
Nombre:	**Buscar_Afectación()**
Descripción:	Esta clase muestra una interfaz para Buscar las Afectaciones a los profesores.

Nombre: Actializar_Afectación	
Tipo de clase : Interfaz	
Atributo	**Tipo**
Nombre	<<HTML Input>>
Solapín	<<HTML Input>>
Para cada responsabilidad:	
Nombre:	**Actializar_Afectacion()**
Descripción:	Esta clase muestra una interfaz para Actualizar las Afectaciones a los profesores.

4.4.9 Descripción de las clases controladoras para la Gestión de Estudiantes

Nombre: Crear Horario Est	
Tipo de clase : Controladora	
Atributo	**Tipo**
Para cada responsabilidad:	
Nombre:	**Crear Horario Est()**
Descripción:	Esta clase crea el objeto fachada y obtiene los valores por el POST

Nombre: ActualizarHorarioEst	
Tipo de clase : Controladora	
Atributo	**Tipo**
Para cada responsabilidad:	
Nombre:	**ActualizarHorarioEst()**
Descripción:	Esta clase crea el objeto fachada y obtiene los valores por el POST

Nombre: Modificar Horario Est

Tipo de clase : Controladora	
Atributo	**Tipo**
Para cada responsabilidad:	
Nombre:	**Modificar Horario Est()**
Descripción:	Esta clase crea el objeto fachada y obtiene los valores por el POST

4.5 Principios de diseño

El diseño de la interfaz de una aplicación, el formato de los reportes, la concepción de la ayuda y el tratamiento de excepciones tiene gran influencia en el éxito o fracaso de una aplicación. A continuación se describen los principios de diseño seguidos para el desarrollo del sistema en cuestión.

4.5.1 Interfaz de usuario

El diseño de interfaces de usuario es una tarea que ha adquirido relevancia en el desarrollo de un sistema, se puede definir como: “el conjunto de trabajos y pasos que seguirá el usuario, durante todo el tiempo que se relacione con el programa, detallando lo que verá y escuchará en cada momento, y las acciones que realizará, así como las respuestas que el sistema dará”. [5]
La calidad de la interfaz de usuario puede ser uno de los motivos que conduzca a un sistema al éxito o al fracaso, es por eso que uno de los aspectos más relevantes de la usabilidad de un sistema es la consistencia de su interfaz de usuario.
Para el desarrollo de la interfaz se tiene en cuenta los siguientes aspectos:

- Reducir la carga a la memoria.
- Atajos a Usuarios expertos.
- Obtener información de retroalimentación.
- Diseño de diálogos que conducen a una conclusión.
- Previsión de errores y manejo de errores simples.
- Lograr deshacer acciones fácilmente.
- Que el usuario sintiera la sensación de control.

Una de las premisas fundamentales de la aplicación es la ventaja que proporciona las interfaces Web sobre las interfaces de comando. Ya que las interfaces Web:

- Proporcionan un ambiente amigable.

- Conducen a un aprendizaje más natural.
- Establecen un “sentimiento” (sobre todo en la uniformidad del ambiente) al usuario que enriquece su experiencia en el uso de la aplicación.

Además de estos principios, se tuvieron en cuenta para el Modulo de Planificación Docente las siguientes características:

- Utilizar una misma tipografía, forma y estilo en todas las páginas.
- La facilidad del usuario de poder navegar desde cualquier punto a otro dentro de la aplicación.
- Se tuvo presente siempre el ancho de banda y por ello se utilizaron formato de imágenes de compresión favorables.
- La simplicidad y consistencia, favoreciendo la usabilidad de la aplicación.
- Navegación simple en todas las páginas de la aplicación, de forma tal que siempre sea accesible por el usuario.
- Estabilidad y uniformidad del diseño, para así poder ubicar al usuario dentro del mismo y hacerlo sentir parte de él.

Se debe utilizar una hoja de estilos para guardar la configuración del diseño para todas las páginas, para los botones y las líneas se utilizaron estos estilos, eliminando así el número de imágenes que demoren la presentación de la página. Se deben realizar múltiples operaciones en cada página, de forma que el usuario no tenga que moverse tanto dentro de la aplicación, para completar una operación.

4.5.2 Ayuda

La ayuda estará accesible como parte del menú en todas las páginas de la aplicación, y con el fin de que el usuario vea solo la información que necesita en ese momento, cada página mostrará como realizar solo aquellas operaciones que se estén realizando en el momento, además se aportan los conceptos que se manejan en la aplicación, para que el usuario se familiarice con algunas entradas.

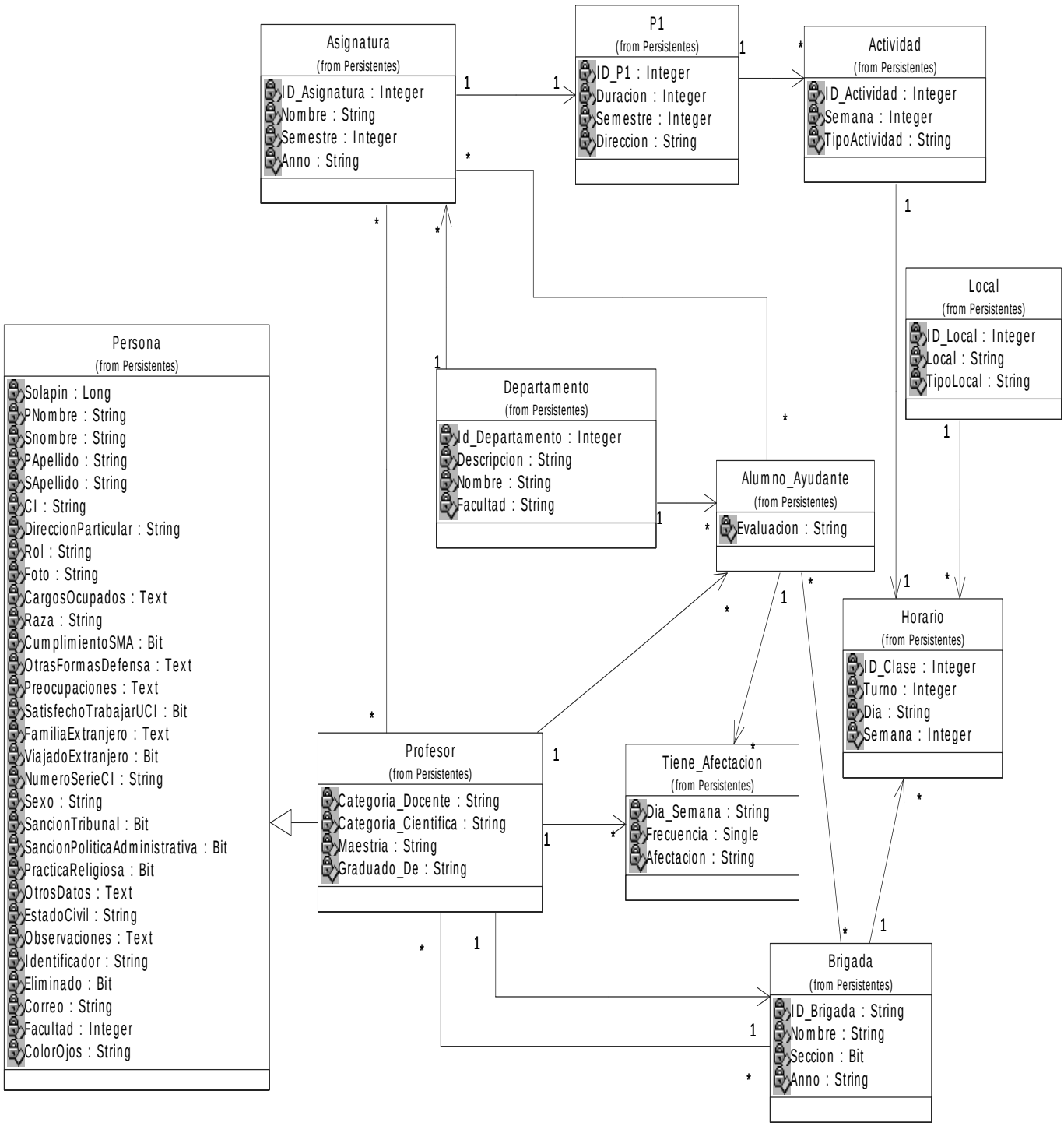
La ayuda constará en gran parte de la explicación funcional del sistema aunque abarcará algunos temas teóricos para mayor comprensión. Esto tiene el objetivo de que el usuario no solo tenga la explicación funcional del sistema sino que también pueda entender en que

consiste el mismo y tenga mayor información en caso de decidir posteriormente en su mantenimiento.

4.5.3 Tratamiento de errores

En el sistema propuesto se evitan, minimizan y tratan los posibles errores, con el fin de garantizar la integridad y confiabilidad de la información que en este se registra y muestra. Los errores se tratan en una página especial que incluye el fichero de configuración general, y está preparada para recoger el número del error y presentar la pantalla con el error que le corresponde a ese código. Los mensajes de error que emite el sistema se muestran en un lenguaje de fácil comprensión para los usuarios. Cuando se introduce información en un formulario y faltan datos, sale un cuadro de alerta indicando el campo o dato que falta. Similar ocurre cuando se introduce información errónea en un campo numérico, e-mail o moneda. [5].

4.6 Diagrama de clases Persistentes



4.8 CONCLUSIONES

Se finaliza la etapa de análisis del sistema, habiéndose hecho el modelo de clases de Análisis. También se concluye así con la etapa de diseño del sistema. Se graficaron los Diagramas de diseño Web del sistema y se hizo referencia al tratamiento de errores generales del sistema y se graficaron además los diagramas de clases persistentes y modelo de datos que representa una mayor claridad a la hora de implementar. Con todos estos elementos, se tiene una idea más precisa sobre los elementos constitutivos del sistema que propone. Al finalizar la etapa correspondiente al diseño del sistema, se arriba a las siguientes

• Los Patrones de diseño agilizan en gran medida el trabajo de los programadores pues proponen soluciones a problemas concretos, o re-soluciones técnicas basadas en la Programación Orientada a Objetos (POO). Ya que se basan en la experiencia Acumulada para resolver problemas reiterativos. Ayudando a construir el software a partir de la reutilización, y construcción de clases reutilizables.

CAPÍTULO 5
Implementación y Prueba

5.1 INTRODUCCION

En el capítulo anterior modelamos los diagramas de clases de análisis y los diagramas de clases Web .En este capítulo se pasa a la implementación del Software en el lenguaje de programación escogido anteriormente, se desarrolla el código de una manera certificada, definiendo estándares de programación, codificando y se realizan pruebas unitarias del producto y se modelaran los diagramas de Despliegues y Componentes para una mayor vista para la implementación del producto.

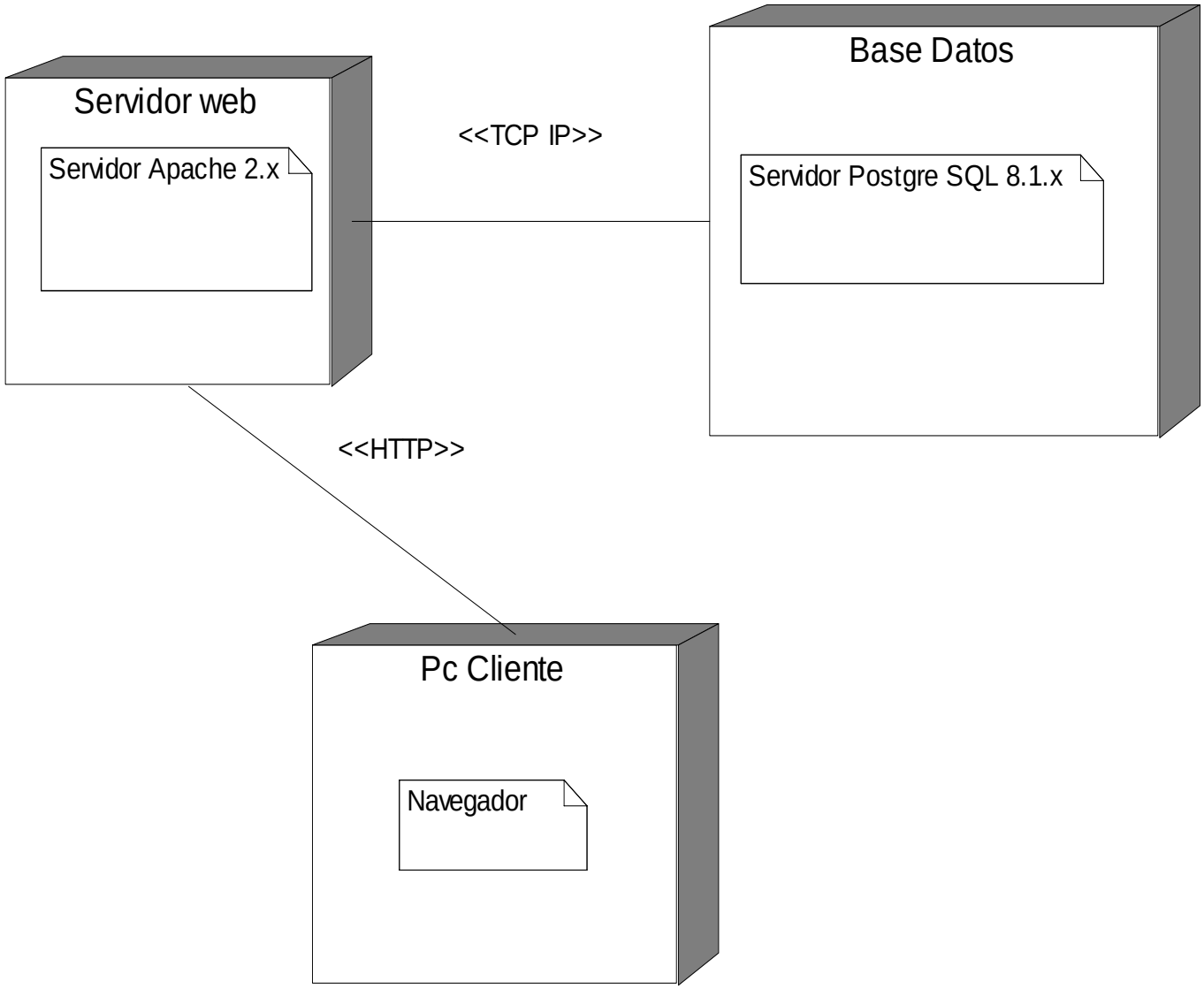
5.2 Diagrama de despliegue

El diagrama de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. Es una colección de nodos y arcos; donde cada nodo representa un recurso de cómputo, normalmente un procesador o un dispositivo de hardware similar. [6]

Muestra la configuración de los componentes hardware, los procesos, los elementos de procesamiento en tiempo de ejecución y los objetos que existen en tiempo de ejecución. En este tipo de diagramas intervienen nodos, asociaciones de comunicación, componentes dentro de los nodos y objetos que se encuentran a su vez dentro de los componentes. Un nodo es un objeto físico en tiempo de ejecución, es decir una máquina que se compone habitualmente de, por lo menos, memoria y capacidad de procesamiento, a su vez puede estar formada por otros componentes. [12]

El diagrama de despliegue muestra la topología del hardware sobre el que se ejecuta en el Módulo de Planificación docente.

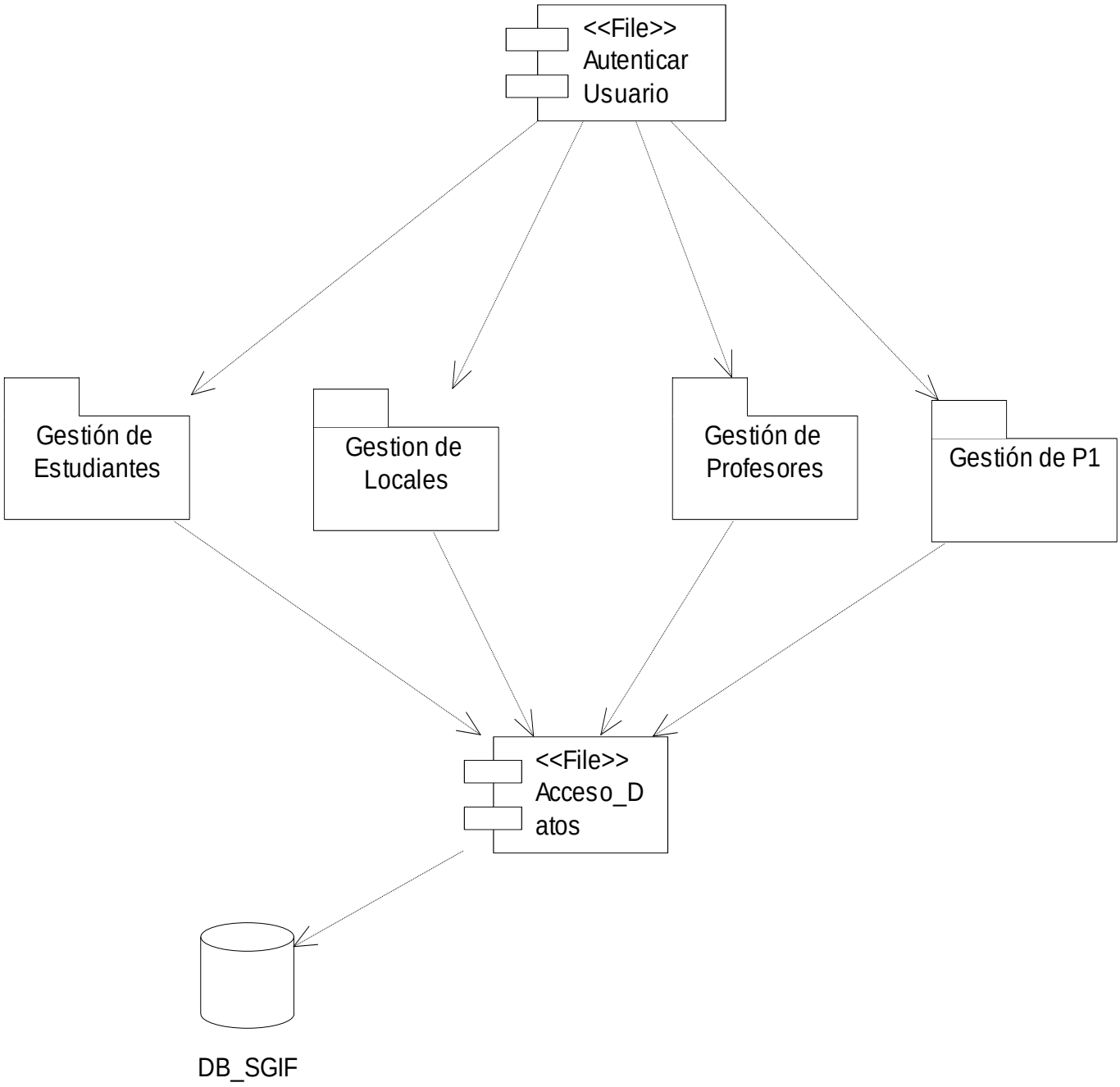
5.2.1 Diagrama de despliegue



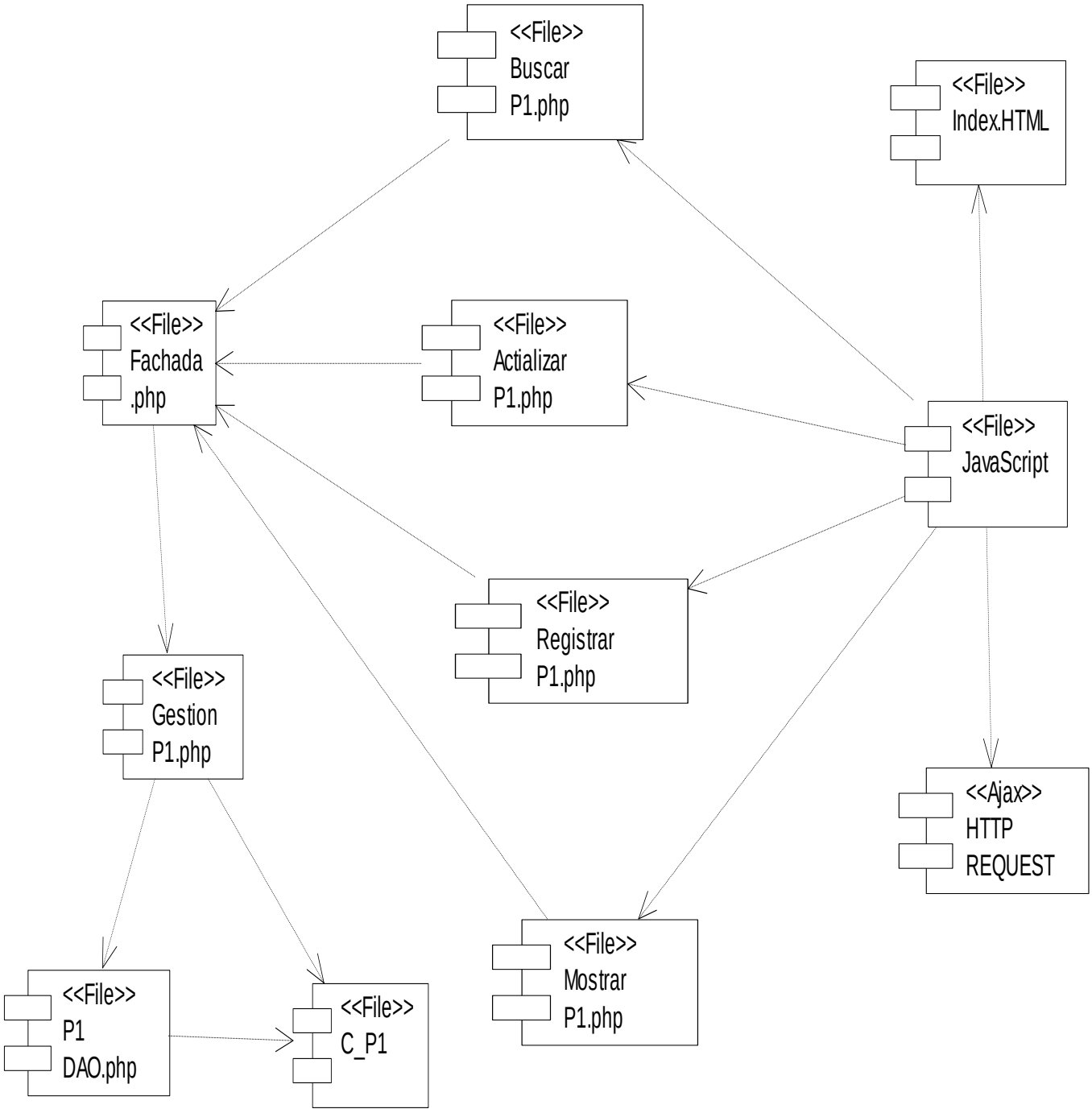
5.3 Modelo de Implementación

El **Modelo de implementación** representa la composición física de la implementación en términos de subsistemas de implementación y elementos de implementación (carpetas y ficheros, incluyendo código fuente, datos y ejecutables). [12]

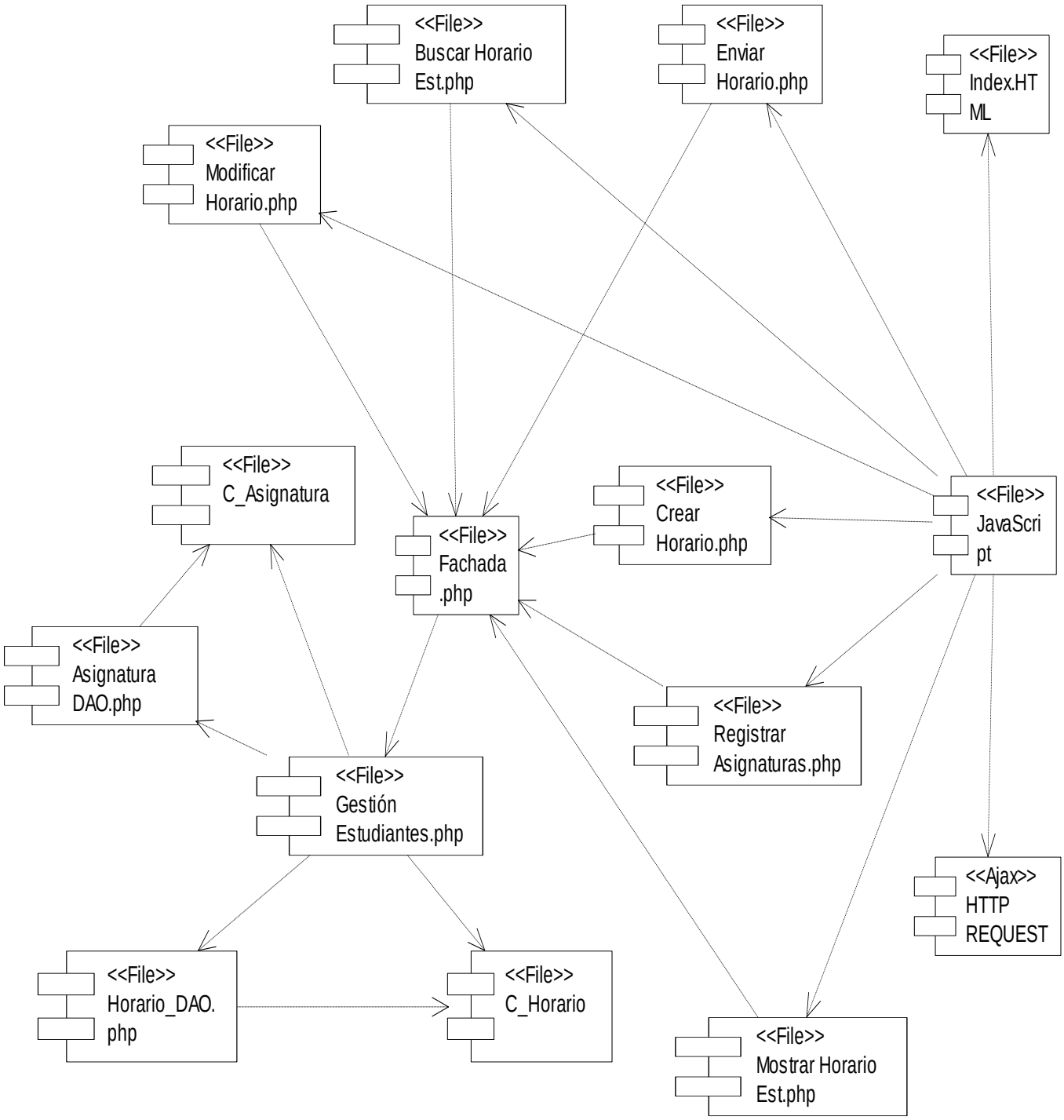
5.3.1 Diagrama de Componentes



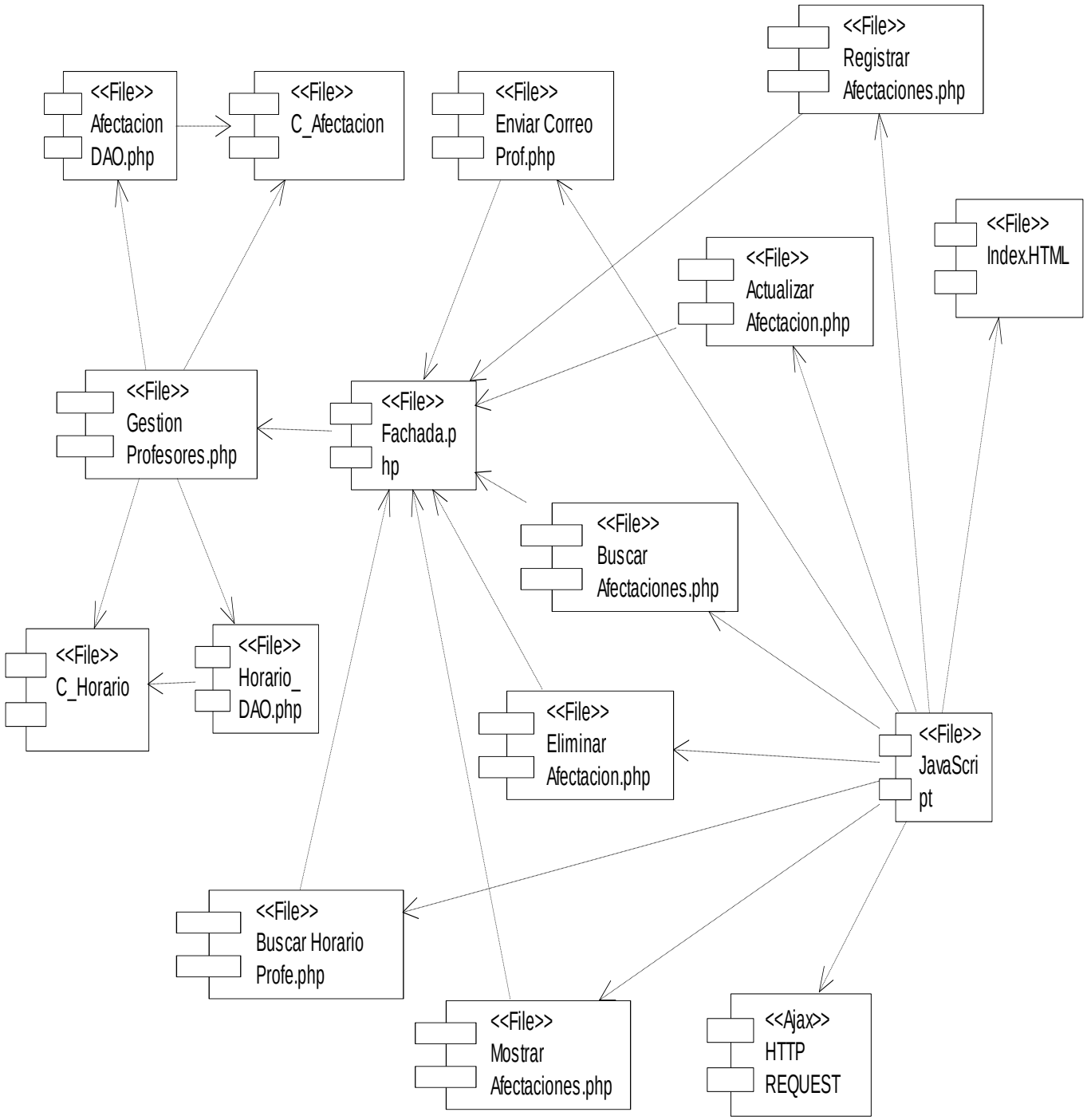
5.3.2 Diagrama de Componentes del Paquete Gestión de P1



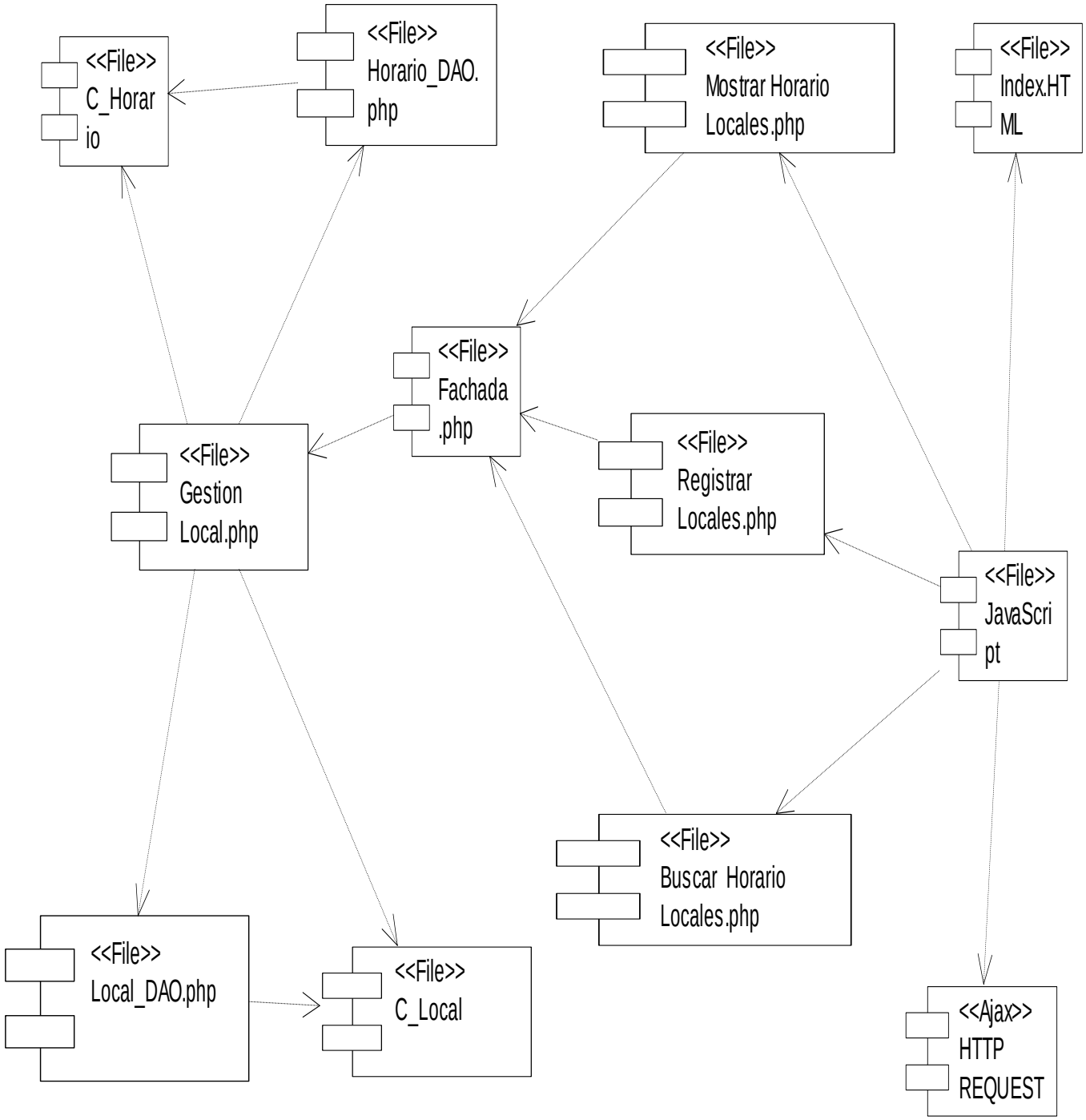
5.3.3 Diagrama de Componentes del Paquete de Gestión Estudiantes



5.3.4 Diagrama de Componentes del Paquete de Gestión de Profesores



5.3.5 Diagrama de Componentes del Paquete de Gestión de Locales



5.4 PRUEBAS

5.4.1 Casos de Prueba para el Caso de Uso Registrar P1

Entrada	Casos Válidos	Casos no Válidos
Departamento	Seleccionar un departamento de los que aparece en el select...	No seleccionar ningún departamento
Año	Seleccionar un año de los que aparece en el select...	No seleccionar ningún año
Asignatura	Seleccionar una asignatura de los que aparece en el select...	No seleccionar ninguna asignatura
Dirección del documento	Seleccionar un P1	Dejar el campo vacío.
Duración	Seleccionar la duración	No seleccionar ninguna duración
Semestre	Seleccionar el semestre	No seleccionar el semestre
Tipo actividad 1.	Seleccionar el tipo	No seleccionar el tipo
Tipo actividad 2.	Seleccionar el tipo	No seleccionar el tipo
Tipo actividad 3.	Seleccionar el tipo	No seleccionar el tipo

Caso de uso: Registrar P1.
Caso de prueba: Registrar los P1 entrando bien todos los datos
Entrada: El usuario introduce correctamente los datos necesarios para registrar el P1. departamento: Humanidades Año: Primero Asignatura: IE2 Dirección del documento: C:\Documents and Settings\yhurtado\Escritorio\pd.jpg Duración:14 Semestre: Primero Tipo actividad 1: Conf. Tipo actividad 2: CP. Tipo actividad 3: LAB.
Resultado: Se insertan los datos del P1 en la base de datos (si es adicionado).
Condiciones: Los datos para Registrar los P1 tienen que ser validos.

Caso de uso: Registrar P1.
Caso de prueba: Registrar los P1 entrando algún dato erróneo.
Entrada: El usuario introduce alguno de los datos no indicados para registrar el P1. departamento: "Campo Vació" Año: Primero Asignatura: IE2 Dirección del documento: "Campo Vació" Duración:14 Semestre: Primero Tipo actividad 1: Conf. Tipo actividad 2: "Campo Vació" Tipo actividad 3: LAB.
Resultado: El sistema muestra un cartel, señalando que los debe llenar los campos.
Condiciones:

5.4.4 Casos de Prueba para el Caso de Uso Registrar Locales

Entrada	Casos Válidos	Casos no Válidos
Nombre	Secuencia de números de tres cifras	Dejar el campo vació
Tipo Local	Seleccionar un tipo en el select.	No seleccionar ningún tipo.

Caso de uso: Registrar Local.
Caso de prueba: Registrar locales entrando bien todos los datos
Entrada: El usuario introduce correctamente los datos necesarios para Registrar Locales. Nombre: 103 Tipo Local: aula.
Resultado: Se Registro correctamente ese local.

Caso de uso: Registrar Locales.
Caso de prueba: Registrar locales entrando algún dato erróneo.

Entrada: Nombre:" Campo Vació" Tipo Local: aula.
Resultado: El sistema muestra un cartel diciendo que hay campos vacíos, y en caso de que se repitan el nombre muestra en color rojo que ese local existe. .
Condiciones:

5.4.6 Casos de Prueba para el Caso de Uso Buscar Horario Estudiantes

Entrada	Casos Válidos	Casos no Válidos
Año	Seleccionar un Año en el select.	No seleccionar ningún año
Grupo	Seleccionar un Grupo en el select.	No seleccionar ningún grupo.

Caso de uso: Buscar Horario EST.
Caso de prueba: Buscar Horario EST entrando bien todos los datos
Entrada: El usuario introduce correctamente los datos necesarios para Buscar Horario de EST. año: Primero Grupo: 8105.
Resultado: Se muestra el Horario de ese grupo.

Caso de uso: Buscar Horario EST.
Caso de prueba: Buscar Horario EST entrando algún dato erróneo.
Entrada: año: Primero Grupo: "Campo Vació".
Resultado: El sistema No hace ninguna funcionalidad.

Condiciones:

5.4.9 Casos de Prueba para el Caso de Uso Buscar Horario Profesor

Entrada	Casos Válidos	Casos no Válidos
Nombre	Secuencia de de letras	Dejar el campo Vació
Elemento por paginas	Secuencia de Números de un digito	Dejar el campo Vació

Caso de uso: Buscar Horario Prof
Caso de prueba: Buscar Horario Prof entrando bien todos los datos
Entrada: El usuario introduce correctamente los datos necesarios para Buscar Horario Prof Nombre: Yariel Elemento por paginas : 1
Resultado: Se muestra el horario de ese profesor.

Caso de uso: Buscar Horario Prof
Caso de prueba: Buscar Horario Prof entrando algún dato erróneo.
Entrada: Nombre: “Campo Vació” Elemento por paginas : 1
Resultado: El sistema muestra un cartel diciendo que debe de llenar los campos vacíos
Condiciones:

5.4.9 Casos de Prueba para el Caso de Uso Buscar Afectación

Entrada	Casos Válidos	Casos no Válidos
Nombre	Secuencia de de letras	Dejar el campo Vació
Elemento por paginas	Secuencia de Números de un digito	Dejar el campo Vació

Caso de uso: Buscar Afectación
Caso de prueba: Buscar Afectación entrando bien todos los datos
Entrada: El usuario introduce correctamente los datos necesarios para Buscar Afectación Nombre: Yariel Elemento por paginas : 1
Resultado: Se muestra Las Afectaciones de ese profesor.

Caso de uso: Buscar Afectación
Caso de prueba: Buscar Afectación entrando algún dato erróneo.
Entrada: Nombre: "Campo Vació" Elemento por paginas : 1
Resultado: El sistema muestra un cartel diciendo que debe de llenar los campos vacíos
Condiciones:

5.5 CONCLUSIONES

Se finaliza la etapa de Implementación del sistema, habiéndose hecho el modelo de despliegues y los diagramas de componentes... También se concluye así con la etapa de prueba del sistema. Se describieron los caso de pruebas del sistema con datos de entradas que retornaban valores, que en gran medida representan una mayor visión de lo que hace el sistema. Con todos estos elementos, se tiene una idea más precisa sobre los elementos constitutivos del sistema que propone.

CONCLUSIONES

Llegado este punto se espera que el documento haya servido para la comprensión teórica de la situación problémica existente y su solución, así como el desarrollo de las diferentes etapas de la aplicación usando la metodología RUP.

El desarrollo de este trabajo de tesis está orientado a la concepción de una herramienta informática para el Módulo de Planificación Docente de la Facultad 8. El valor fundamental de esta herramienta se expresa en la contribución a simplificar el trabajo y la demora que produce el procesamiento manual de la información y mejorar la gestión de las actividades que se realizan en este lugar.

Se alcanzó, satisfactoriamente, el objetivo propuesto: el análisis y diseño de un sistema Web que permita agilizar y humanizar la gestión de la información docente dentro de la facultad, reafirmando así la utilidad y validez de emplear las tecnologías informáticas para apoyar las labores que se desarrollan en cualquier tipo de esfera.

Se ha demostrado la eficacia de los lenguajes y tecnologías utilizadas para el desarrollo del sistema. La solución propuesta ha sido acertada, los requerimientos soportan al sistema y los casos de uso satisfacen las necesidades funcionales.

Se han seguido los principios básicos de diseño descritos para el desarrollo del sistema.

Se logra una seguridad y protección de los datos consecuente con el nivel de seguridad requerido.

RECOMENDACIONES

Se recomienda:

- Continuar el estudio con el objetivo de añadir nuevas funcionalidades.
- Proponer, tras corroborar un desempeño exitoso, la utilización y generalización de este sistema en las diferentes facultades de la UCI.
- Continuar trabajando en el Modulo de Planificación docente para seguir perfeccionándolo.
- Proponer que se lleve a todas las universidades y escuelas del país para mejor la calidad del trabajo de los planificadores en nuestro país.

BIBLIOGRAFIA

[0]. Sistema Generador de Horarios. Disponible
En: <http://lcc.ens.uabc.mx/sh2007_1/index.jsp> [Fecha de consulta 30 enero 2007].

[1]. Generador de Horarios para Centros Docentes 7.0. Disponible en:
<http://generador-de-horarios-para-centros-docentes.softonic.com/ie/22084> [Fecha de consulta 31 enero 2007].

[2].Generador de Horarios para Centros de Enseñanza. Disponible
En: <http://www.penalara.com/> [Fecha de consulta 31 enero 2007].

[3].Refusiones para el Generador de Horarios y Generador de Grupos Milenio-6. Disponible en: < http://www.adossis.es/REFUGHM6.htm> [Fecha de consulta 30 enero 2007].

[4]. Cronos Disponible en: < http://cronos.sisvenca.com/Cronos.html> [Fecha de consulta 25 enero 2007].

[5].Principios de Diseño. Disponible en
<http://www.sidar./recur/desdi/traduc/es/visitable/quees/dcu.htm > [Fecha de consulta 23 mayo 2007].

[6].Características de un Diagrama de Despliegue. Disponible
En: < http://www-gris.det.uvigo.es/~avilas/UML/node50.html > [Fecha de consulta 20 mayo 2007].

[7].Índice del manual de lenguajes del Web. Disponible
En: <http://www.desarrolloweb.com/articulos/709.php > [Fecha de consulta 10 abril 2007].

[8]. Qué es AJAX. Disponible en: <http://www.marciobarrios.com/ajax> [Fecha de consulta 13 marzo 2007].

[9]. El servidor Web Apache 1.3. Disponible en:
<http://www.augcyl.org/glol/old/N_1/apache.html > [Fecha de consulta 20 enero 2007].

[10]. Patrón Modelo-Vista-Controlador. Disponible en: < http://www.proactiva-calidad.com/java/patrones/mvc.html> [Fecha de consulta 4 marzo 2007].

[11]. Patrón Modelo-Vista-Controlador. Disponible
En:<http://www.microsoft.com/spanish/msdn/comunidad/mtj.net/voices/MTJ_3317.as> [Fecha de consulta 3 marzo 2007].

[12].Metodologías De Desarrollo De Software. Disponible en:
<http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html > [Fecha de consulta 20 marzo 2007].
[12]. Clases de Ingeniería del Software I, curso 2005-2006, UCI.

[13]. Desarrollo Metodología EXTREME PROGRAMING. Disponible
En:<http://www.peruserver.com/des_metodologia.php > [Fecha de consulta 3 abril 2007].

[14]. Tutorial de UML. Disponible en:
< http://www.dcc.uchile.cl/~psalinas/uml/introduccion.html > [Fecha de consulta 10 febrero 2007].

[15]. RATIONAL ROSE: PROCEDIMIENTOS BÁSICOS PARA DESARROLLAR UN PROYECTO CON UML. Disponible
En:<http://www.vico.org/TallerRationalRose.pdf> [Fecha de consulta 15 febrero 2007].

[15].Clases de Ingeniería del Software I, curso 2005-2006, UCI.

[16].Sistema Gestor de base de datos SGBD. Disponible
En:<http://www.error500.net/garbagecollector/bases_de_datos/sistema_gestor_de_base_de_dato.html> [Fecha de consulta 15 marzo 2007].

[17]. PostgreSQL 8.1.x. Disponible en: <http://www.postgresql.cl/> [Fecha de consulta 20 marzo 2007].

[18]. Tutorial de PostgreSQL. Disponible en:
<http://es.tldp.org/Postgresql-es/web/navegable/tutorial/tutorial.html> [Fecha de
Consulta 22 marzo 2007].
[19].MySQL 5.0 Reference Manual. Disponible
En:< http://dev.mysql.com/doc/refman/5.0/es/index.html > [Fecha de consulta 25 marzo 2007].
[20].Programación en Capas. Disponible
En:< http://es.wikipedia.org/wiki/Programaci%C3%B3n_por_capas> [Fecha de consulta 20 mayo 2007].

GLOSARIO DE TERMINOS

A:

Autónoma: Que trabaja por cuenta propia

Artefactos: Todo lo que se realiza para un mayor conocimiento del trabajo

Anteproyecto: Algo que se realiza antes de un proyecto.

Accesibilidad: indica la facilidad con la que algo puede ser usado, visitado o accedido en general por todas las personas.

C:

CUS: Caso de uso del sistema.

CUN: Caso de uso del negocio.

Ciclos: Son los periodos de enseñanza en los que se estructuran las titulaciones

Cúmulo: Montón, agrupaciones.

CMS: Significa sistema de administración de contenido (en inglés Content Management System), un sistema de este tipo funciona para la creación y administración de contenido, actualmente se usa para denominar a los sistema que sirven para administrar el contenido de páginas Web.

CASE: Computer Aided Software Engineering.

E:

Expertos: Simplemente conoce sobre un campo delimitado del saber.

Estandarizado: Ajustado a un tipo, modelo o norma

H:

HTML: Hypertext Markup Language. 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.

Hito*:* Punto de control de objetivo intermedio antes de que el proyecto finalice

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

Herramientas CASE: Herramientas utilizadas para el desarrollo de proyectos de Ingeniería de Software.

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

I:

Iterativo: Algo repetitivo.

Metodología*:* Metodología se refiere a los métodos de investigación en una ciencia. Aun cuando el término puede ser aplicado a las artes cuando es necesario efectuar una observación o análisis más riguroso o explicar una forma interpretar la obra de arte.

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.

Online: Si esta conectado a una red.

Óptimos: Algo que se realiza lo mejor posible

P:

PHP: Hypertext Preprocessor. Es un ambiente script del lado del servidor que permite crear y ejecutar aplicaciones Web dinámicas e interactivas. Con PHP se pueden combinar páginas HTML y script*s*. Con el objetivo de crear aplicaciones potentes.

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

R:

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

Roles: En los ciclos de publicación, es necesaria la definición y manejo de roles como por ejemplo editor, autorizador, publicador, administrador, etc. Un administrador de contenido maneja esta funcionalidad como base para el sistema de flujo de documentos.

S:

Software: Programas de sistema, utilerías o aplicaciones expresados en un lenguaje de máquina.

SQL: Structured Query Language. Es un lenguaje declarativo de acceso a bases de datos que permite especificar diversos tipos de operaciones sobre las mismas. Aúna características del álgebra y el cálculo relacional permitiendo lanzar consultas con el fin de recuperar información de interés de una base de datos.

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.

T:

Trazabilidad: Término que se utiliza en los procesos industriales o actividades económicas que requieren la certificación de su buena práctica. Por ejemplo en los sistemas de gestión de calidad, de gestión medioambiental y sistemas de control conocidos como cadena de custodia.

U:

UML: Unified Modeling Language. 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.

UCI: Universidad de las Ciencias Informáticas.

X:

XML: Extensible Markup Language. Es un lenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium. Orientado principalmente al almacenamiento, procesamiento y transmisión de mensajes.