Universidad de las Ciencias Informáticas
Dirección de Informatización

Sistema para la informatización del Convenio Cuba-Venezuela. Módulo ficha mixta.

Trabajo de Diploma para optar por el título de Ingeniero Informático

AUTORES

Ronny Zamora Aguilar

Alberto Tamayo Ramos

TUTOR

Ing. Yunier Saborit Ramírez

Ciudad de La Habana, Cuba
Mayo, 2007

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas (UCI) para que haga el uso que estimen pertinente con este trabajo.

Para que así conste firmo la presente a los __ días del mes de junio del 2007.

_____	_____	_____
Firma del Autor	Firma del Autor	Firma del Tutor
Alberto Tamayo Ramos	Ronny Zamora Aguilar	Ing. Yunier Saborit Ramírez

OPINIÓN DEL USUARIO DEL TRABAJO DE DIPLOMA

El Trabajo de Diploma, titulado “Sistema para la informatización del Convenio Cuba-Venezuela. Módulo ficha mixta.”, fue realizado en la Universidad de las Ciencias Informáticas (UCI) de la provincia de Ciudad Habana. Esta entidad considera que, en correspondencia con los objetivos trazados, el trabajo realizado le satisface

1. Totalmente
2. Parcialmente en un ____ %

Los resultados de este Trabajo de Diploma le reportan a esta entidad los beneficios siguientes:

Y para que así conste, se firma la presente a los ___ días del mes de junio del 2007

_____ _____
Representante de la entidad Cargo

_____ _____
Firma Cuño

OPINIÓN DEL TUTOR DEL TRABAJO DE DIPLOMA

Título: Sistema para la informatización del Convenio Cuba-Venezuela. Módulo ficha mixta.

Autores: Alberto Tamayo Ramos y Ronny Zamora Aguilar

El tutor del presente Trabajo de Diploma considera que durante su ejecución el estudiante mostró las cualidades que a continuación se detallan.

Por todo lo anteriormente expresado considero que el estudiante está apto para ejercer como Ingeniero Informático; y propongo que se le otorgue al Trabajo de Diploma la calificación de ____ .

_____ ____ de junio del 2007

Firma

Dedicatoria

A quien me forjó, me educó, siempre estará guiándome y será mi orgullo donde quiera que este,
Mi padre.
A quienes supieron darme todo el amor y educación del mundo,
Mi madre, mi abuela, mi tía.
A quien me han apoyado incondicionalmente,
Mis hermanos y tíos.
A mis amigos que sin ellos todo este sueño no se hubiera hecho realidad.

Alberto Tamayo Ramos

A quien fuese mi guía incondicional, quien durante tantos años fue madre y padre, quien siempre estará en mi corazón.
Mi abuela.
A quien junto a esta estuvo siempre a mi lado dándome apoyo, a quien siempre recordaré donde quiera que esté.
Mi tía Margot.
A sus hijos Olaisis (Pula), Odelaisis, Geoge (Chino), Osmanis y Albanis.
A mi tío Jorge a mis primos Onelkis y Karina y a mi tía Luisa por todo el apoyo que han dado.
A mi madre y padre que a pesar de todo los quiero mucho.
A mi hermana Arianna y su esposo Yudier, por todo lo que han hecho.
A mi hermano Magdiel, a su mamá Madelín a padrastro Jorge, a Yaque y Guillermo que me han apoyado mucho durante la carrera.
A toda mi familia en general.

Ronny Zamora Aguilar

Agradecimientos

Agradecemos a todos los que nos han acompañado en este largo camino de la vida y la educación dándonos fuerzas para salir adelante y realizar nuestros sueños. Y un agradecimiento especial a Eliurkis Díaz Terrero por hacer posible la realización de este trabajo.

Gracias a todos.

RESUMEN

El presente trabajo se desarrolló en la Universidad de las Ciencias Informáticas, centro en el cual se desarrollan una gran cantidad de aplicaciones Web, tanto para el desarrollo de la universidad como para la exportación.

Este trabajo se realiza con el objetivo de gestionar la ficha de los proyectos que Cuba con Venezuela. Para ello se estructura en 5 capítulos donde se recoge la fundamentación teórica del trabajo, el modelo del negocio, los requisitos del sistema, la descripción de la solución propuesta y el estudio de factibilidad.

Se propone como solución una aplicación Web, basada en tecnología PHP4 y con gestor de base de datos PostgreSQL, la cual contribuye a la reducción del tiempo en las búsquedas de información.

INDICE

INTRODUCCIÓN

Quienes llevan siempre las estadísticas de los proyectos que se firman con Venezuela tanto por la parte cubana como venezolana siempre realizan los cálculos de forma manual lo cual en ocasiones genera además de errores, ya sea por aproximaciones constantes o por errores de calculo, demora en la entrega de la información. En muchos casos esto generaba perdidas de dinero. Por lo tanto nuestro problema a resolver se basa en que en el marco del convenio Cuba-Venezuela y sobre las bases del ALBA, se hace necesario un sistema para la gestión de toda la información que se genera, comenzando por el control y las estadísticas de las fichas que describen los contratos; estas fichas contienen información muy importante para el desenvolvimiento futuro de los contratos firmados, y contienen la información inicial a gestionar. El **problema** se puede formular entonces de la siguiente manera: ¿Cómo resolver el control, la exactitud y la veracidad de toda la información y estadísticas de la ficha mixta Cuba-Venezuela?

De este trabajo no se tiene antecedente en nuestro país ni en Venezuela pues este convenio es novedoso y tienes sus peculiaridades, decir sus propias formas de accionar y controlar toda la información.

El objeto de estudio de este trabajo es la informatización del convenio Cuba-Venezuela. Mientras el campo de acción es la informatización de la ficha mixta para la gestión de los proyectos en el marco del convenio Cuba-Venezuela.

Este trabajo tiene como objetivo general el desarrollo de un sistema en software libre que sea capaz de automatizar la gestión de la información de los proyectos en el marco del convenio Cuba-Venezuela, de manera que sea capaz de gestionar la información contenida en la ficha de cada proyecto.

Como **objetivos específicos** se plantean los siguientes:

1. Obtener el diseño de una base de datos capaz de almacenar de manera organizada la información que se manipula.
2. Desarrollar el análisis y diseño del sistema.
3. Implementar el sistema con las características definidas en los procesos de análisis y diseño.

Desarrollo del módulo donde se controle y calculen todas las estadísticas e informaciones de un proyecto a partir de los datos de su ficha, y dadas estas estadísticas se genere la ficha general y se muestren los gráficos estadísticos con los totales de cada proyecto.

Para cumplir los objetivos se desarrollaron las siguientes **tareas**:

1. Realizar el análisis y diseño de la aplicación.
2. Implementar del Software haciendo uso de herramientas de código abierto como PostgreSQL 8.X como gestor de base de datos y PHP4 como lenguaje de programación.
3. Documentar la información referente al análisis, diseño e implementación del sistema, así como todo lo referente al flujo de trabajo.
4. Elaborar un documento que sirva de ayuda a los usuarios del sistema.

La presente propuesta contribuye a mejorar y a optimizar la gestión del convenio cuba-Venezuela para un mejor y más rápido intercambio de información entre los dos países. De esta forma se garantiza evitar errores estadísticos y llevar un control más eficiente de los proyectos.

El presente trabajo, estructurado en 5 capítulos, resume la siguiente información:

Capítulo 1. Fundamentación Teórica: descripción del objeto de estudio, sistemas existentes vinculados al campo de acción, tendencias y tecnologías actuales seleccionadas a emplear en el desarrollo de la propuesta y por qué su utilización.

Capítulo 2. Modelo del Negocio: descripción de los procesos, actores, trabajadores y casos de uso del negocio; y diagramas de clases del modelo de objetos del negocio.

Capítulo 3. Requisitos: definición de los requisitos funcionales y no funcionales; actores y casos de uso del sistema.

Capítulo 4. Descripción de la solución propuesta: descripción del diseño a través del diagrama de clases Web, que describen la relación entre las páginas. Se definen, además, los principios de diseño seguidos en la aplicación y el modelo de implementación mediante el diagrama de despliegue.

Capítulo 5. Estudio de factibilidad: estudio de factibilidad económica realizado para este proyecto, en el que se determina si es factible o no el desarrollo del software propuesto, analizando los diferentes criterios que influyen en el cálculo del esfuerzo, tiempo de desarrollo y costo del proyecto.

1 CAPÍTULO

Fundamentación Teórica

1.1 Introducción.

Este capítulo contiene los principales problemas que fundamentan la propuesta de solución, y los objetivos generales y específicos que se persiguen. Además de brindar un enfoque general de sistemas automatizados existentes vinculados al campo de acción y el análisis comparativo de las soluciones existentes con la propuesta dada en este trabajo. Se describen además las tecnologías actuales de desarrollo utilizadas para el análisis, diseño e implementación del sistema sobre las cuales se apoya la propuesta.

1.2 Diseño metodológico de la investigación

1.2.1 Métodos teóricos

Inductivo-deductivo: A partir del razonamiento de la ficha mixta Cuba-Venezuela se llegó a un grupo de conocimientos, lo que permitirá llegar a conclusiones rápidas y avanzadas para comenzar la implementación de dicha aplicación.

Modelación: A través de modelos representamos como pensamos que se puede desarrollar he implementar la aplicación de la ficha mixta Cuba-Venezuela, con diferentes modelos que se enmarcan en diferentes procesos y funcionalidades.

1.2.2 Metodología Informática

Metodología RUP: Es una forma disciplinada de asignar tareas, responsabilidades y de asegurar la producción de software de Calidad dentro de plazos y presupuestos predecibles.

1.3 Estado de la ficha mixta en Venezuela.

En Venezuela el estado de la ficha mixta del convenio Cuba-Venezuela se encuentra en cuanto a informatización en un nivel cero. Pues no se a trabajado para lograr una aplicación que gestione toda la información y las estadísticas que se generan a partir de estos convenios.

1.4 Estado de la ficha mixta en Cuba.

En Cuba el estado de la ficha mixta del convenio Cuba-Venezuela se encuentra en las mismas condiciones que en Venezuela.

1.5 Tendencia.

La tendencia en este trabajo es de manejar la información y las estadísticas de forma tal que sea fácil de acceder y que tenga la menor probabilidad de errores posibles para poder dar estimaciones exactas en los países que intervienen en el proyecto.

1.6 Tendencias y tecnologías actuales.

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

1.6.1 Las aplicaciones Web.

Las aplicaciones Web son una especialización y concreción de las aplicaciones cliente-servidor, o sea, su arquitectura general es la de un sistema cliente/servidor, donde tanto el cliente (el navegador) como el servidor (el servidor Web), y el protocolo mediante el que se comunican (el HTTP: HyperText Transfer Protocol) son estándar, y no han de ser creados por el desarrollador.[3]

La parte del cliente de las aplicaciones Web está formada por el código HTML (HyperText Markup Language) que forma la página Web, con opción a código ejecutable mediante los lenguajes script de los navegadores (JavaScript, VBScript, PerlScript) o mediante pequeños programas (applets) en Java. La parte del servidor está formada por un programa o script que es ejecutado por el servidor Web, y cuya salida se envía al navegador del cliente.[3]

La creciente popularidad de las aplicaciones Web se debe a sus múltiples ventajas, entre las cuales podemos citar:[6]

1. Multiplataforma: Con un solo programa, un único ejecutable, nuestras aplicaciones pueden ser utilizada a través de múltiples plataformas, tanto de hardware como de software.
2. Actualización instantánea: Debido que todos los usuarios de la aplicación hacen uso de un sólo programa que radica en el servidor, los usuarios siempre utilizarán la versión más actualizada del sistema.
3. Suave curva de aprendizaje: Los usuarios, como utilizan la aplicación a través de un navegador, hacen uso del sistema tal como si estuvieran navegando por Internet, por lo cual su acceso es más intuitivo.
4. Fácil de integrar con otros sistemas: Debido a que se basa en protocolos estándares, la información manejada por el sistema puede ser accedida con mayor facilidad por otros sistemas.
5. Acceso móvil: El usuario puede acceder a la aplicación con la única restricción de que cuente con un acceso a la red privada de la organización o a Internet, dependiendo de las políticas de dicha organización; puede hacerlo desde una computadora de escritorio, una laptop o desde una agenda electrónica; desde su oficina, hogar u otra parte del mundo.

El desarrollo de aplicaciones Web está siendo utilizado en muchas organizaciones, ésta situación va ir creciendo indefinidamente. Es por ello que día a día se requieran más programadores capacitados para desarrollos basados en el World Wide Web (WWW).

No obstante a la serie de ventajas que presenta tiene además algunas desventajas, las cuales son:

1. Acceso limitado, la necesidad de conexión permanente y rápida a Internet hacen que el acceso a estas aplicaciones no esté al alcance de todos.
2. La interactividad no se produce en tiempo real, en las aplicaciones web cada acción del usuario conlleva un tiempo de espera algunas veces excesivo hasta que se obtiene la reacción del sistema.

3. Elementos de interacción muy limitados. En comparación con el software de escritorio, las posibilidades de interacción con el usuario que ofrecen las aplicaciones web (mediante formularios principalmente) son muy escasas.
4. Diferencias de presentación entre plataformas y navegadores. La falta de estándares ampliamente soportados dificulta el desarrollo de las aplicaciones.

1.6.2 Modelo Cliente Servidor.

Se dice que la arquitectura Cliente/Servidor es la integración distribuida de un sistema en red, con los recursos, medios y aplicaciones que, definidos modularmente en los servidores, administran, ejecutan y atienden las solicitudes de los clientes; todos interrelacionados física y lógicamente, compartiendo datos, procesos e información. Se establece así un enlace de comunicación transparente entre los elementos que conforman la estructura. Entre las principales características de la arquitectura Cliente/Servidor, se pueden destacar las siguientes: [6]

1. El servidor presenta a todos sus clientes una interfaz única y bien definida.
2. El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa.
3. El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.
4. Los cambios en el servidor implican pocos o ningún cambio en el cliente.

Ventajas de la arquitectura cliente-servidor: [6]

1. El servidor no necesita potencia de procesamiento, parte del proceso se reparte con los clientes.
2. Se reduce el tráfico de red considerablemente. Idealmente, el cliente se conecta al servidor cuando es estrictamente necesario, obtiene los datos que necesita y cierra la conexión dejando la red libre.

Las arquitecturas de dos capas contienen tres componentes distribuidos en dos capas: cliente (solicitante de servicios) y servidor (proveedor de servicios). *Ver Anexo 1.*
Los tres componentes son: [6]

1. Interfaz de usuario al sistema. Tales como una sesión, entradas de texto, desplegado de menús, etc.
2. Administración de procesamiento. Tales como la ejecución de procesos, el monitoreado de los mismos y servicios de procesamiento de recursos.
3. Administración de bases de datos. Tales como los servicios de acceso a datos y archivos.

La arquitectura de software de tres capas emergió en la década de los noventas para solventar las limitaciones de la arquitectura de dos capas. La tercera capa (capa de servicios) se localiza entre la interfaz de usuarios (cliente) y el administrador de datos (servidor). Esta capa intermedia provee de servicios para la administración de procesos (tal como desarrollo, monitoreo y alimentación de procesos) que son compartidos por múltiples aplicaciones. [6]. *Ver Anexo 2.*

El servidor de la capa intermedia (también conocido como servidor de aplicaciones) centraliza la lógica de las aplicaciones, haciendo que la administración de cambios sea más sencilla. En arquitecturas más simples, cualquier cambio en la lógica, implica reescribir todas las aplicaciones que dependan de ésta. [6]

1.6.3 PHP (PHP: Hypertext Preprocessor).

El PHP, es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor. El PHP originalmente diseñado en Perl, seguidos por la escritura de un grupo de CGI binarios escritos en el lenguaje C por el programador Danés-Canadiense Rasmus Lerdorf en el año 1994 para mostrar su currículum vitae y guardar ciertos datos, como la cantidad de tráfico que su página web recibía. En los siguientes tres años, se fue convirtiendo en lo que se conoce como PHP/FI 2.0. Esta forma de programar llegó a muchos usuarios, pero el lenguaje no tomó el peso actual hasta que dos programadores israelíes de Technion, Zeev Suraski y Andi Gutmans reescribieron el analizador gramatical en el año 1997, y crearon la base del PHP 3, cambiando el nombre del lenguaje a la forma actual. Para 1999, Suraski y Gutmans reescribieron el código de PHP, produciendo lo que hoy se conoce como Zend Engine

o motor Zend. En mayo del 2000, PHP 4 fue lanzado bajo el poder del motor Zend Engine 1.0. El 13 de julio de 2004, PHP 5 fue lanzado, utilizando el motor Zend Engine II. La versión más reciente de PHP es la 5.1.4, que incluye el novedoso PDO (PHP Data Objects) y mejoras utilizando las ventajas que provee el nuevo Zend Engine 2. Según estudios, más de un millón de servidores tienen esta capacidad implementada y los números continúan creciendo. [6]

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+. [6]

Es software libre, lo que implica menos costes 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. [6]

Es multiplataforma, funciona tanto para Unix (con Apache) como para Windows (con Microsoft Internet Information Server) de forma que el código que se haya creado para una de ellas no tiene porqué 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). [6]

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. [6]

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. [6]

El funcionamiento del PHP se puede describir a través de los pasos siguientes: [6]

1. Escribir en las páginas HTML pero con el código PHP dentro.
2. Guardar la página en el servidor Web.
3. Un navegador solicita una página al servidor.
4. El servidor interpreta el código PHP.
5. El servidor envía el resultado del conjunto de código HTML y el resultado del código PHP que también es HTML.

En ningún caso se envía código PHP al navegador, por lo que todas las operaciones realizadas son transparentes al usuario, el código PHP es ejecutado en el servidor y el resultado es enviado al navegador. El resultado es normalmente una página HTML. Por lo que al usuario le parecerá que está visitando una página HTML que cualquier navegador puede interpretar.

Al ser PHP un lenguaje que se ejecuta en el servidor no es necesario que el navegador lo soporte, es independiente del navegador, pero sin embargo para que sus páginas PHP funcionen, el servidor donde están alojadas debe soportar PHP. Además se encuentra libre en el mercado y se puede acceder a él por medio de Internet.[6]

Después de seis años, y después que la comunidad ha revisado el paquete de legados que ha dejado el PHP, se han realizado cambios estructurales en el lenguaje para ofrecer innovación en el nuevo PHP 5 y solucionar muchos de los problemas encontrados en PHP 4.

Afortunadamente, lo nuevo de PHP 5 mejora muchas áreas en el lenguaje y su ejecución, como por ejemplo: [9]

1. Programación orientada a objetos (OOP).
2. MySQL.
3. XML.
4. Integración nativa con el Zend Engine.

Los diseñadores de PHP5 han realizado un cambio radical en el tratamiento de las variables objeto: en PHP5 todas las variables que nombran objetos son en realidad referencias. No hay que usar el operador '&' ni en las asignaciones, ni en el paso de parámetros que son objetos, ahorrándose con ello gran cantidad de potenciales errores. [4]

La principal novedad en las clases de PHP5 es la inclusión de modificadores de control de acceso para implementar la encapsulación --piedra angular en la programación orientada a objetos de la que adolecía PHP4--.

PHP5 introduce tres palabras clave (public, private y protected) que sustituyen a *var* en la definición de variables miembro --atributos-- de la clase, y que preceden a la definición de funciones miembro --métodos-- .

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: [6]

1. **Características multiplataformas:** Menos el ASP, que es solamente soportado por la plataforma Windows, los demás lenguajes están soportados en múltiples plataformas.
2. **Velocidad de ejecución:** la velocidad es mayor en PHP, seguidos por PERL y JSP.
3. **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.
4. **Familiaridad con el lenguaje:** En la universidad los lenguajes más utilizados por los programadores es el ASP y el PHP.

De acuerdo a estas comparaciones, el PHP resulta mucho más favorecido, por tanto pensamos que es el adecuado para implementar la propuesta de sistema de este trabajo, particularmente PHP5.

1.6.4 Servidor Web Apache.

Un servidor de páginas Web es un programa que permite acceder a páginas Web alojadas en un ordenador. Hoy en día Apache es el servidor web más utilizado del mundo, encontrándose muy por encima de sus competidores, tanto gratuitos como comerciales. Es un software de código abierto que funciona sobre cualquier plataforma. Desde su origen ha evolucionado hasta convertirse en uno de los mejores servidores en términos de eficiencia, funcionalidad y velocidad, surgió en abril de 1996 y ya en julio del 2002 era utilizado por el 57% de los sitios Web de Internet.[6]

Tiene capacidad para servir páginas tanto de contenido estático, para lo que nos serviría sencillamente un viejo ordenador 486, como de contenido dinámico a través de otras herramientas soportadas que facilitan la actualización de los contenidos mediante bases de datos, ficheros u otras fuentes de información, es muy potente y altamente configurable.

Los servidores Web suministran páginas web a los navegadores que lo solicitan. En términos más técnicos, los servidores Web soportan el Protocolo de Transferencia de Hypertexto como HTTP (HyperText Transfer Protocol), el estándar de Internet para comunicaciones web. Usando HTTP, un servidor Web envía páginas web en HTML y Common Gateway Interface (CGI), así como otros tipos de scritps a los navegadores o browsers cuando éstos los requieren. Cuando un usuario hace clic sobre un enlace a una página web, se envía una solicitud al servidor web para localizar los datos nombrados por ese enlace. El servidor web recibe esta solicitud y suministra los datos que le han sido solicitados o bien devuelve un mensaje de error.

El servidor Apache es un software que esta estructurado en módulos, es decir, está dividido en muchas porciones de código que hacen referencia a diferentes aspectos o funcionalidades del servidor web. Esta modularidad es intencionada ya que la configuración de cada módulo se hace mediante la configuración de las directivas que están contenidas dentro del módulo. Los módulos del Apache se pueden clasificar en tres categorías:

1. Módulos Base: Módulo con las funciones básicas del Apache.
2. Módulos Multiproceso: Son los responsables de la unión con los puertos de la máquina, aceptando las peticiones y enviando a los hijos a atender a las peticiones.
3. Módulos Adicionales: Cualquier otro módulo que le añada una funcionalidad al servidor.

Las funcionalidades más elementales se encuentran en el módulo base, siendo necesario un módulo multiproceso para manejar las peticiones. Se han diseñado varios módulos multiprocesos para cada uno de los sistemas operativos sobre los que se ejecuta el Apache, optimizando el rendimiento y rapidez del código.

El resto de funcionalidades del servidor se consigue por medio de módulos adicionales que se pueden cargar. Para añadir un conjunto de utilidades al servidor, simplemente hay que añadirle un módulo, de forma que no es necesario volver a instalar el software.

1.6.5 AJAX.

AJAX, acrónimo de **A**synchronous **J**avaScript **A**nd **X**ML (JavaScript y XML asíncronos), es una técnica de desarrollo web para crear aplicaciones interactivas. Estas se ejecutan en el cliente, es decir, en el navegador del usuario, y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la velocidad de interacción en la misma. [1]

AJAX no constituye una tecnología en sí, sino que es un término que engloba a un grupo de éstas que trabajan conjuntamente y que se muestra a continuación: [1]

1. XHTML (o HTML) y hojas de estilos en cascada (CSS) para el diseño que acompaña a la información.
2. Document Object Model (DOM) accedido con un lenguaje de scripting por parte del usuario, especialmente implementaciones ECMAScript como JavaScript y JScript, para mostrar e interactuar dinámicamente con la información presentada.
3. El objeto XMLHttpRequest para intercambiar datos asincrónicamente con el servidor web. En algunos frameworks y en algunas situaciones concretas, se usa un objeto IFrame en lugar del XMLHttpRequest para realizar dichos intercambios.
4. XML es el formato usado comúnmente para la transferencia de vuelta al servidor, aunque cualquier formato puede funcionar, incluyendo HTML preformateado, texto plano, JSON y hasta EBML.

Existen diferencias significativas entre las aplicaciones web tradicionales y las aplicaciones desarrolladas en AJAX (*Ver Anexo 3*) así como sus patrones de interacción sincrónica para una aplicación Web tradicional y asincrónica para una aplicación AJAX (*Ver Anexo 4*).

1.6.5.1 Ventajas del AJAX:

Interactividad

Las aplicaciones AJAX se ejecutan en la máquina del usuario, manipulando la página actual dentro de sus navegadores usando métodos de Document Object Model. Puede ser usado para multitud de tareas como actualizar o eliminar registros, expandir formularios web, devolver peticiones simples de búsqueda, o editar árboles de categorías; todo sin tener la necesidad de recargar toda la página de HTML cada vez que se realiza un cambio. Generalmente solo requiere enviar pequeñas peticiones al servidor, y se devuelven respuestas relativamente cortas. Esto permite el desarrollo de aplicaciones interactivas con más interfaces de usuario más responsivas gracias al uso de las técnicas DHTML. [1]

Portabilidad

Las aplicaciones construidas con AJAX utilizan características bien documentadas presentes en todos los navegadores importantes en la mayoría de las plataformas existentes. Aunque esta situación podría cambiar en el futuro, en este momento, los usos de AJAX son efectivos entre plataformas. Mientras que la plataforma de AJAX está más restringida que la plataforma de Java, las aplicaciones actuales de AJAX llenan con eficacia la parte de los Java Applets: ampliar el navegador con mini-aplicaciones ligeras.

1.6.5.2 Desventajas del AJAX:

Críticas de usabilidad

Una de las mayores críticas contra el uso de AJAX en aplicaciones web es que puede fácilmente acabar con el comportamiento normal del botón atrás del navegador. Las diversas expectativas entre volver a una página que se ha modificado dinámicamente y la vuelta a una página estática pueden ser sutiles. Otro problema relacionado es que las actualizaciones dinámicas hacen difícil al usuario agregar a los marcadores/favoritos un momento particular de la aplicación.

JavaScript

Aunque AJAX no necesita ningún tipo de plug-in para el navegador, requiere que los usuarios tengan el JavaScript activado. Esto se aplica a todos los navegadores que soportan esta tecnología excepto para Microsoft Internet Explorer 6 y anteriores los cuales necesitan también tener el ActiveX activado,

ya que el objeto XMLHttpRequest está implementado junto con el ActiveX en este navegador.

Como ocurre con las aplicaciones DHTML, las de AJAX deben de ser probadas rigurosamente para adaptarse a los diferentes navegadores y plataformas.

1.6.6 Patrones de Diseño.

Un patrón es un modelo que podemos seguir para realizar algo. Los patrones surgen de la experiencia de seres humanos de tratar de lograr ciertos objetivos. Los patrones capturan la experiencia existente y probada para promover buenas prácticas. [2]

Los Patrones de Diseño (Design Patterns) son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

Estos se dividen en tres grandes categorías:

Patrones Creacionales

Solucionan problemas de creación de instancias. Nos ayudan a encapsular y abstraer dicha creación.

Patrones Estructurales

Solucionan problemas de composición (agregación) de clases y objetos.

? Patrones de Comportamiento

Soluciones respecto a la interacción y responsabilidades entre clases y objetos, así como los algoritmos que encapsulan

1.6.7 Un patrón de diseño es:

1. una solución estándar para un problema común de programación
2. una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios
3. un proyecto o estructura de implementación que logra una finalidad determinada
4. un lenguaje de programación de alto nivel
5. una manera más práctica de describir ciertos aspectos de la organización de un programa

6. conexiones entre componentes de programas
7. la forma de un diagrama de objeto o de un modelo de objeto

Los patrones de diseño han contribuido a dar flexibilidad y extensibilidad a nuestros diseños. Pero en adición, han demostrado ser una forma muy útil (exitosa) de reutilizar diseño, ya que ellos no sólo nombran, abstraen e identifican aspectos claves de estructuras comunes de diseño, sino que generalmente son descritos en una forma específica documental, haciendo su comprensión y aplicación fácil para el conjunto de desarrolladores.

Podemos decir que los beneficios que un patrón produce pueden ser medidos en varios sentidos:

1. Contribuyen a reutilizar diseño, identificando aspectos claves de la estructura de un diseño que puede ser aplicado en una gran cantidad de situaciones. La importancia de la reutilización del diseño no es despreciable, ya que ésta nos provee de numerosas ventajas: reduce los esfuerzos de desarrollo y mantenimiento, mejora la seguridad, eficiencia y consistencia de nuestros diseños, y nos proporciona un considerable ahorro en la inversión.
2. Mejoran (aumentan, elevan) la flexibilidad, modularidad y extensibilidad, factores internos e íntimamente relacionados con la calidad percibida por el usuario. [13]

Incrementan nuestro vocabulario de diseño, ayudándonos a diseñar desde un mayor nivel de abstracción.

1.6.8 Modelo Vista Controlador (MVC).

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

Son muchas las empresas que deciden pasar sus aplicaciones a la arquitectura modelo vista controlador para documentar más fácilmente el código, ahorrar espacio y en caso de no disponer de diseñadores web, poder contratar los servicios de un diseñador que no sepa mucho de programación que les haga las vistas. [5]

El **Modelo** es todo acceso a datos, y las funciones que llevan lo que llaman "lógica de negocio", o sea datos y reglas de negocio. Lleva un registro de las vistas y controladores del sistema. Cada acceso a datos se pone en su función individual porque, de esta forma, si se cambia de gestor de bases de datos este cambio sólo afecta a estas funciones, no al resto de la aplicación. Tener el modelo bien delimitado permite la existencia de varias aplicaciones que compartan el mismo modelo (por ejemplo, una aplicación "tienda" y una "contabilidad" que accedan a las bases de datos de inventario, ventas, etc.). [13]

La **Vista**, en una aplicación web, es el HTML y lo necesario para convertir datos en HTML. O sea muestra la información del modelo al usuario. Tienen un registro de su controlador asociado (normalmente porque además lo instancia).

Pueden dar el servicio de "Actualización()", para que sea invocado por el controlador o por el modelo. Tener la vista separada del controlador permite cambiar la aplicación para que genere, en lugar de HTML, algo distinto (por ejemplo, WML), sin tener que tocar más que una parte completamente delimitada del código. [5]

El **Controlador** es lo que une la vista y el modelo. Por ejemplo, son las funciones que toman los valores de un formulario, consultan la base de datos (a través del modelo) y producen valores, que la vista tomará y convertirá en HTML. En resumen, gestiona las entradas del usuario. Recibe los eventos de entrada (un clic, un cambio en un campo de texto, etc.). Contiene reglas de gestión de eventos, del tipo "SI Evento Z, entonces Acción W". Estas acciones pueden suponer peticiones al modelo o a las vistas. De este modo, el código que "hace algo" está perfectamente separado del código dedicado a crear HTML, lo que ayuda a evitar el spaghetti.

Un ejemplo típico del funcionamiento de este patrón se puede observar en el *Anexo 5*. La estructura de este patrón se puede observar en el *Anexo 6*.

1.6.9 Sistemas de Gestión de Base de Datos.

Un Sistema de Gestión de Bases de Datos (SGBD) puede definirse como un paquete generalizado de software, que se ejecuta en un sistema computacional anfitrión, centralizando los accesos a los datos y actuando de interfaz entre los datos físicos y el usuario. Las principales funciones que debe cumplir un SGBD se relacionan con la creación y mantenimiento de la base de datos, el control de accesos, la manipulación de datos de acuerdo con las necesidades del usuario, el cumplimiento de las normas de tratamiento de datos, evitar redundancias e inconsistencias y mantener la integridad.

Un SGBD tiene los siguientes objetivos específicos: [6]

1. Independencia de los datos y los programas de aplicación.
2. Minimización de la redundancia.
3. Integración y sincronización de las bases de datos.
4. Integridad de los datos.
5. Seguridad y protección de los datos.
6. Facilidad de manipulación de la información.
7. Control centralizado.

La información es representada a través de tuplas, las cuales describen el fenómeno, proceso o ente de la realidad objetiva que se está analizando y se representan a través de tablas. [6]

Entre los SGBD comúnmente utilizados en el mundo tenemos Oracle, MySQL, Microsoft SQL Server, PostgreSQL, Interbase, entre otros. Todos estos presentan un enfoque relacional con un buen basamento matemático centrado en el Álgebra Relacional. [6]

Todos los sistemas mencionados anteriormente facilitan el trabajo con la base de datos y tienen características que los diferencian, por ejemplo: [6]

1. **Oracle**: requiere de una licencia para poderlo utilizar, es decir, es necesario pagar para poder utilizarlo.
2. **Microsoft SQL Server**: no es multiplataforma, solo puede ser utilizado con el sistema operativo Windows que está patrocinado por la compañía Microsoft.
3. **MySQL**: PostgreSQL soporta un subconjunto de SQL92 MAYOR que el que soporta MySQL.

Como SGBD se seleccionó el PostgreSQL por las ventajas que ofrece y por requerimientos del cliente.

1.6.9.1 PostgreSQL.

PostgreSQL es un sistema de gestión de bases de datos Objeto-Relacionales (ORDBMS) libre, liberado bajo la licencia BSD (Berkeley Software Distribution). Es una alternativa a otros sistemas de bases de datos de código abierto (como MySQL, Firebird y MaxDB), así como sistemas propietarios como Oracle y SQLServer. El mismo ha sido desarrollado de varias formas desde 1977.

En 1994, Andrew Yu y Jolly Chen añadieron un intérprete de lenguaje SQL a Postgres. Postgres95 fue lanzada a continuación a la Web para que encontrara su propio hueco en el mundo como un descendiente de dominio público y código abierto del código original Postgres de Berkeley.

En 1996, debido a un nuevo esfuerzo de código abierto y a la incrementada funcionalidad del software, *Postgres* fue renombrado a *PostgreSQL*, tras un breve periplo como *Postgres95*. El proyecto *PostgreSQL* sigue actualmente un activo proceso de desarrollo a nivel mundial gracias a un equipo de desarrolladores y contribuidores de código abierto. PostgreSQL está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo. Posee muchas

características que tradicionalmente sólo se podían ver en productos comerciales de alto calibre.

La siguiente es una breve lista de algunas de esas características, a partir de PostgreSQL 7.1.x.

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, transacciones, optimización de consultas, herencia, y arreglos.

Altamente Extensible

PostgreSQL soporta operadores, funciones métodos de acceso y tipos de datos definidos por el usuario.

Soporte SQL Comprensivo

PostgreSQL soporta la especificación SQL99 e incluye características avanzadas tales como las uniones (joins) SQL92.

Integridad Referencial

PostgreSQL soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos.

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 (Tool Command Language) como lenguaje procedural embebido.

MVCC

MVCC, o Control de Concurrencia Multi-Versión (Multi-Version Concurrency Control), es la tecnología que PostgreSQL usa para evitar bloqueos innecesarios. Los bloqueos son provocados 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.

Las principales mejoras en PostgreSQL incluyen:

1. 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.
2. Se han implementado importantes características del motor de datos, incluyendo subconsultas, valores por defecto, restricciones a valores en los campos (constraints) y disparadores (triggers).

3. 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.
4. Los tipos internos han sido mejorados, incluyendo nuevos tipos de fecha/hora de rango amplio y soporte para tipos geométricos adicionales.
5. La velocidad del código del motor de datos ha sido incrementada aproximadamente en un 20-40%, y su tiempo de arranque ha bajado el 80% desde que la versión 6.0 fue lanzada.

1.6.10 Proceso de Desarrollo.

Cada día la producción de software busca adecuarse más a las necesidades del usuario, esto trae como consecuencia que aumente en tamaño y complejidad.

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 Rational, (Rational Unified Process, de ahí las siglas RUP), fue publicado en 1998 como resultado de varios años de experiencia. [6]

RUP es un proceso de desarrollo de software, o sea, conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema de software. Es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos. [7]

Es un proceso basado en componentes, que utiliza el Lenguaje Unificado de Modelado (UML) para preparar todos los esquemas de un sistema software. No obstante, los verdaderos aspectos definitorios del Proceso Unificado se resumen en que está dirigido por casos de uso, este avanza a través de una serie de flujos de trabajo, los cuales se muestran en el A*nexo 7*, que parten de los casos de uso; centrado en la arquitectura y es iterativo e incremental. [7]

Está acompañado de una herramienta muy buena que soporta cada uno de los procesos que necesitamos: Rational Rose Enterprise Edition 2003.
Además cubre el ciclo de vida de desarrollo de un proyecto y toma en cuenta las mejores prácticas a utilizar en el modelo de desarrollo de software.

Lenguaje Unificado de Modelado (UML).

UML (Unified Modeling Language) es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema de software. [10]

Sus creadores pretendieron con este lenguaje, unificar las experiencias acumuladas sobre técnicas de modelado e incorporar las mejores prácticas en un acercamiento estándar.

El UML permite a los creadores de sistemas generar diseños que capturen sus ideas en una forma convencional y fácil de comprender para comunicarlas a otras personas que estén involucradas en el proceso de desarrollo de los sistemas, esto se lleva a cabo mediante un conjunto de símbolos y diagramas. [11]

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. Un modelo UML indica que es lo que supuestamente hará el sistema pero no como lo hará. [11]

De forma general las principales características son:

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

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 ésta 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.

1.6.11 Herramientas utilizadas.

1.6.11.1 Diseño de interfaz.

Macromedia Dreamweaver MX es uno de los editores de desarrollo Web más utilizado a nivel profesional para la creación de sitios Web. Su amplio abanico de herramientas permite crear desde la más simple página Web personal hasta el sitio Web más completo y complejo para una gran empresa y utilizar casi todos los recursos de la Web.

Este editor de HTML profesional para el diseño, codificación y desarrollo de páginas, sitios y aplicaciones Web; permite la edición visual, o sea, crear páginas rápidamente sin escribir una línea de código, así como también la codificación manual.

Dreamweaver ayuda además a construir aplicaciones Web dinámicas apoyadas en bases de datos.

Es completamente personalizable. Se pueden crear objetos y comandos propios, modificar los accesos directos de teclado, e incluso escribir código JavaScript para extender las capacidades del Dreamweaver con nuevos comportamientos. Soporta varias tecnologías del servidor para la construcción de aplicaciones Web, tales como: Macromedia ColdFusion, Microsoft ASP, Microsoft ASP.NET, Sun JavaServer Pages (JSP) y PHP.

1.6.11.2 Zend Studio.

Zend Studio es uno de los ambientes de desarrollo integrado o Integrated Development Environment (IDE) disponible para desarrolladores profesionales que agrupa todos los componentes de desarrollo necesarios para ciclo de desarrollo de aplicaciones PHP. A través de un comprensivo conjunto de herramientas de edición, depurado, análisis, optimización y bases de datos, Zend Studio acelera los ciclos de desarrollo y simplifica los proyectos complejos.

1.6.11.3 Adobe Photoshop.

Adobe Photoshop CS para el tratamiento de los gráficos. Es una herramienta muy poderosa para crear cualquier tipo de gráficos, su integración con Adobe ImageReady hacen que crear complicados gráficos para la Web sea una tarea muy fácil.

1.6.11.4 Rational Rose.

Existen herramientas CASE de trabajo visuales como el Analise, el Designe, el Rational Rose, que permiten realizar el modelado del desarrollo de los proyectos, en la actualidad la mejor y más utilizada en el mercado mundial es Rational Rose y es la que se utiliza en la modelación de este proyecto.

Rational Rose cubre todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases y entregables.

Es la herramienta CASE que comercializan los desarrolladores de UML y que soporta de forma completa la especificación del UML.

Rose es una herramienta con plataforma independiente que ayuda a la comunicación entre los miembros de equipo, a monitorear el tiempo de desarrollo y a entender el entorno de los sistemas. Una de las grandes ventajas de Rose es que utiliza la notación estándar en la arquitectura de software(UML), la cual permite a los arquitectos de software y desarrolladores visualizar el sistema completo utilizando un lenguaje común, además los diseñadores pueden modelar sus componentes e interfaces en forma individual y luego unirlos con otros componentes del proyecto.

Se decidió que se utilizaría el Rational Rose Enterprise Edition 2003, para sustentar la documentación, como modelador visual de la notación UML (Unified Modeling Language) para la confección de los diagramas que se ilustran en este documento. Esta herramienta es muy completa y ofrece amplias potencialidades.

1.6.11.5 PgAdmin

Es desarrollado por una comunidad de los expertos de PostgreSQL de varias partes del mundo y está disponible en más de una docena de idiomas. Se trata de una herramienta para la administración de bases de datos PostgreSQL. Su uso se puede extender hacia las plataformas de Linux, de FreeBSD, de Solaris, del Mac OSX y de Windows.

1.7 Conclusiones.

En este capítulo se exponen las condiciones y problemas que rodean el objeto de estudio; y a través de los conceptos y definiciones planteadas. Se evidencia la necesidad de implementar un software que permita la interoperabilidad, organización y publicación de los servicios Web en el contexto de la arquitectura SOA de la UCI. Para desarrollar el sistema se hace uso de la tecnología para la programación de páginas dinámicas el lenguaje PHP5 y con soporte de base de datos en PostgreSQL. El proceso de desarrollo es RUP, el cual está basado en la orientación a objetos y el modelamiento visual usando UML, lo cual permite incorporar al proceso de desarrollo de software un mejor control de los requerimientos y cambios.

2 CAPÍTULO

Modelo del Negocio

2.1 Introducción.

Antes de comenzar a desarrollar un sistema es necesario comprender la organización bajo estudio 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 y trabajadores del negocio y el modelo de objetos.

2.2 Modelo del negocio propuesto.

El primer paso del modelado del negocio consiste en capturar y definir los procesos de negocio de la organización bajo estudio. En el capítulo anterior se hizo una descripción general de los procesos identificados en el negocio actual, así como un análisis crítico de la ejecución de los mismos. Teniendo en cuenta las deficiencias detectadas y bajo un análisis profundo de las fuentes de problemas potenciales se ha elaborado una propuesta de negocio que mantiene invariable el flujo de los procesos pero incurre en cambios de la ejecución de los mismos. A continuación se muestra la descripción detallada del negocio propuesto, en la misma se describe la ejecución de los siguientes procesos:

2.2.1 Proceso de confeccionar ficha

Este proceso tiene como objetivo que el líder del proyecto llene la ficha con todos los datos del proyecto y así según las estadísticas que se insertan en el plan operativo, las finanzas, la adquisición de materiales y el plan de inversiones se calculan todos esos

datos mediantes formulas y variables dadas y se llega a un monto total tanto a transferir en cuba como a invertir en Venezuela que todas estas son estadísticas que se deben controlar y tener en cuenta para tener controlado todo el proceso de los negocios entre los dos países.

Fórmulas utilizadas en este proceso.

HombreMES_Total= (Consultores + Profesionales + Técnicos) * Tiempo
HombreMES_Fuera = HombreMES_Total * Porciento_Fuera

Cálculos

Honorarios = (Precio_Consultor * Consultores + Precio_Profesional * Profesionales + Precio_Técnico * Técnicos) * Tiempo

Viáticos = HombreMES_Fuera * Viáticos

Gastos Administrativos = HombreMES_Fuera * Gastos_Administrativos

Pasaje = HombreMES_Fuera * Pasaje* Precio Pasaje

2.2.2 Proceso de revisar ficha

El proceso tiene el objetivo principal la revisión de las fichas que han sido confeccionada que se lleva a cabo por el jefe general de proyecto y en caso de detectar un error este lo manda a corregir.

También el después que la ficha ha sido revisada por el jefe general de proyecto este se la envía al responsable por país el cual repite el proceso de revisar ficha para asegurarse de que todo esta en orden.

2.2.3 Proceso de corregir errores a la ficha

Una vez revisada la ficha se puede detectar algunos errores de cálculos o de sintaxis y es ahí cuando se lleva a cabo por el líder del proyecto este proceso el cual permite arreglar todos los detalles encontrados.

2.2.4 Proceso de imprimir ficha

El responsable por país, después de revisar la ficha procede a imprimirla para poder firmarla y así darles valides a toda esa información que será valida para todas las estadísticas financieras que existen en los dos países.

2.3 Reglas del negocio a considerar

1. El líder de proyecto es el encargado de confeccionar la ficha.
2. El jefe general de los proyectos es el encargado de revisar la ficha después que ha sido confeccionada por el líder del proyecto.
3. El responsable por país es el encargado de firmar y darle valides a la ficha.

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.

A continuación se muestra en la tabla 2.1, los actores del negocio y su correspondiente justificación:

Actores del negocio	Justificación
Responsable por país	Es el principal beneficiado con todo este proceso de la ficha mixta, pues es el que se

	encarga de solicitar, revisar y autorizar los presupuestos de cada proyecto para hacerlo valido

Tabla 2.1 Descripción de los actores del negocio.

2.4.1 Diagrama de casos de uso de negocio

El diagrama de casos de uso del negocio representa gráficamente los procesos del negocio y su interacción con los actores del negocio. A continuación se muestra la figura 2.1 correspondiente al diagrama de casos de uso del negocio.

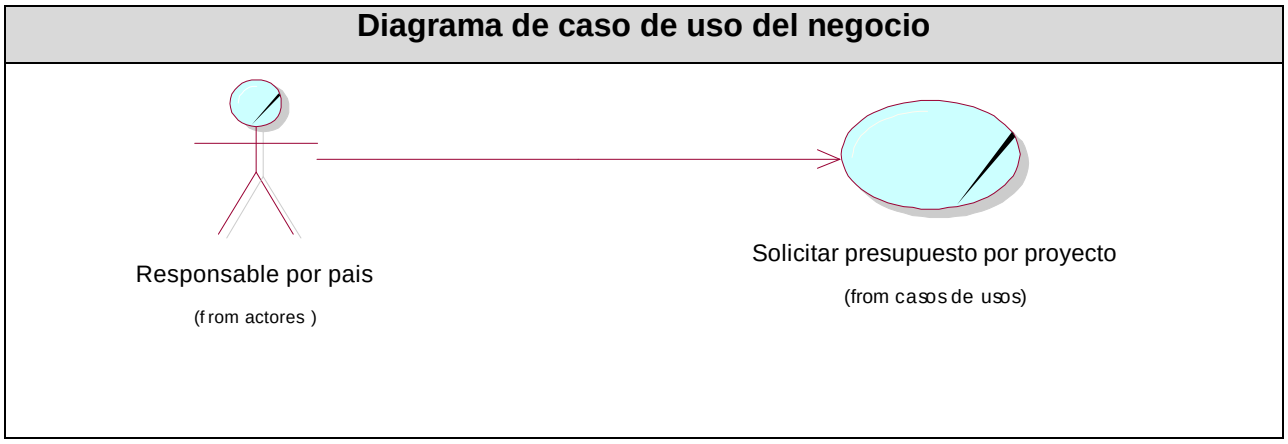


Figura 2.1 Diagrama de casos de uso del negocio

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.

A continuación se muestran en la tabla 2.2, los trabajadores del negocio y su correspondiente justificación:

Trabajadores del negocio	Justificación
Líder de proyecto	Es quien confecciona la ficha poniendo

	todas las justificaciones y los cálculos pertinentes
Jefe general de los proyectos	Es quien solicita y recibe la ficha después de haber sido confeccionada por el líder del proyecto y es el encargado de revisarla antes de enviársela al responsable por país

Tabla 2.2 Descripción de los trabajadores del negocio.

2.6 Descripción de los casos de uso del negocio.

2.6.1 Caso de uso "Solicitar presupuesto por proyecto"

Especificación textual en formato general

Caso de uso del negocio: Solicitar presupuesto por proyecto	
Actores del negocio: Responsable por país.	
Propósito: Permitir que el Responsable por país pueda obtener la ficha con el presupuesto de el proyecto.	
Resumen: El caso de uso se inicia cuando el responsable por país solicita el presupuesto por cada uno de los proyecto. Luego siguen una serie de procedimientos que pasan por el jefe general de proyectos el cual es el encardo de intermediar entre el líder de proyecto y el responsable por país hasta que es impresa y firmada la ficha por este ultimo.	
Curso normal de los eventos:	
Acción del actor	**Respuesta del proceso de negocio**
1. El responsable por país solicita el presupuesto del proyecto.	1.1. El Jefe general de los proyectos solicita la ficha al Líder de proyecto. 1.2. El líder del proyecto confecciona la ficha 1.3. Seguidamente entrega la

	ficha al Jefe general de los proyectos 1.4. El Jefe de los proyectos recibe la ficha. 1.5. Luego el Jefe de los proyectos revisa la ficha 1.6. Verificar que no exista errores en la ficha. 1.7. Entrega la ficha al responsable por país.
2. Recibe ficha	
3. Procede a imprimir ficha	
4. Firma la ficha	
Flujo alterno de los eventos:	
Acción del actor:	**Respuesta del proceso de negocio:**
	1.7. El jefe general de los proyectos informa error en la ficha al líder del proyecto 1.8. El líder de proyecto realiza la acción de corregirle los errores a la ficha. 1.9. Luego se ejecuta de nuevo al paso 1.3.
Prioridad: Alta.	
Mejoras: Gestionar a partir de la solicitud de la ficha, todo los procesos automatizados, que todo se tenga guardado en bases de datos, el rápido y fácil acceso a toda esa información, cálculos estadísticos verídicos y confiables.	
Otras secciones:	

Tabla 2.3 Especificación textual del caso de uso del negocio “Solicitar Presupuesto por Proyecto”.

2.6.1.1 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 se pueden ir 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. A continuación se muestra en la figura 2.2 el Diagrama de actividades del caso de uso del negocio “Solicitar presupuesto por proyectos”.

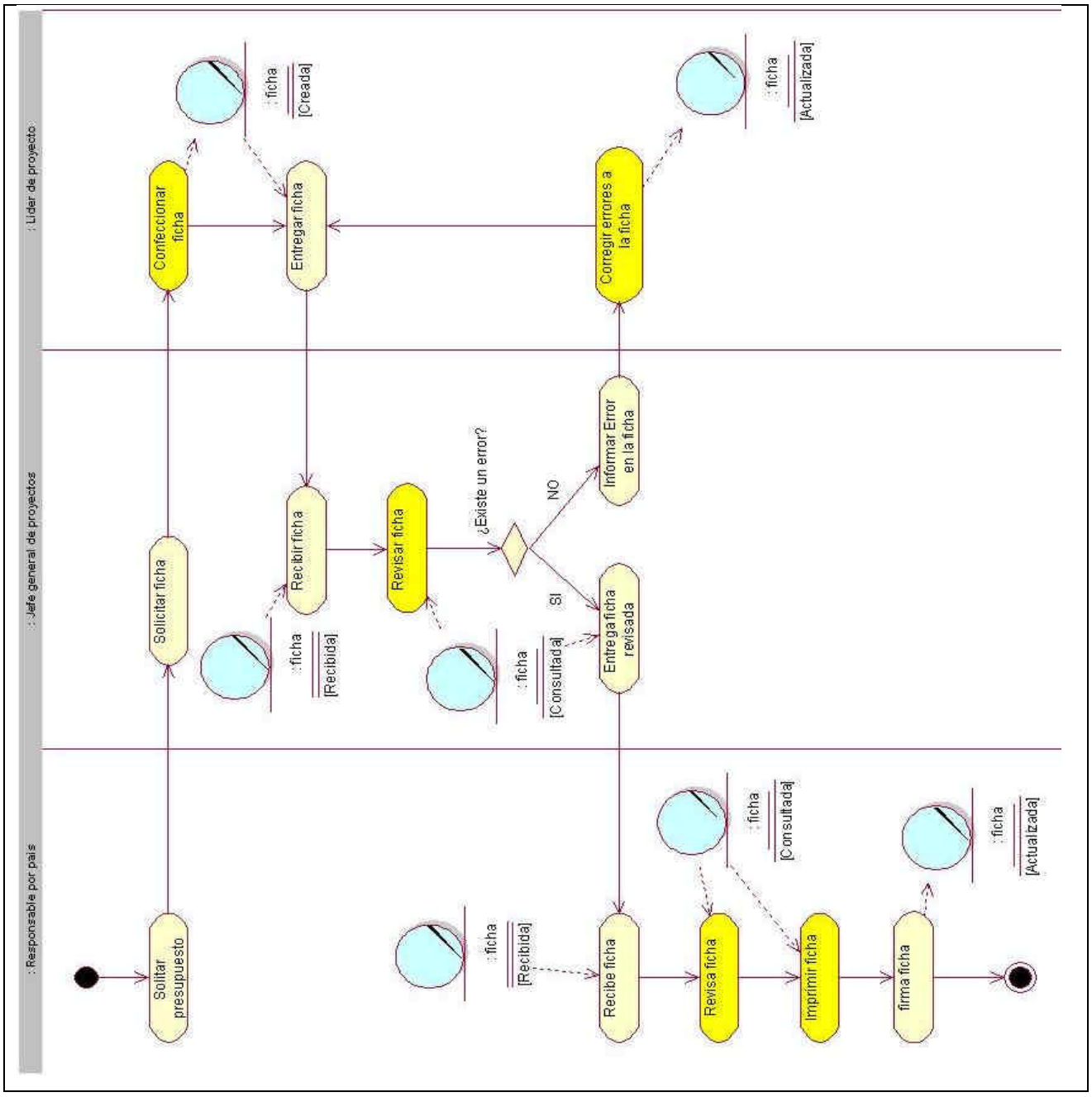


Figura 2.2 Diagrama de actividades del caso de uso del negocio “Solicitar presupuesto por proyecto”.

2.6.2.2 Diagrama de clases del modelo de objeto.

A continuación se muestra en la figura 2.5 el diagrama de clases del modelo de objetos del caso de uso del negocio “Solicitar presupuesto por proyecto”.

Diagrama de clases del modelo de objetos

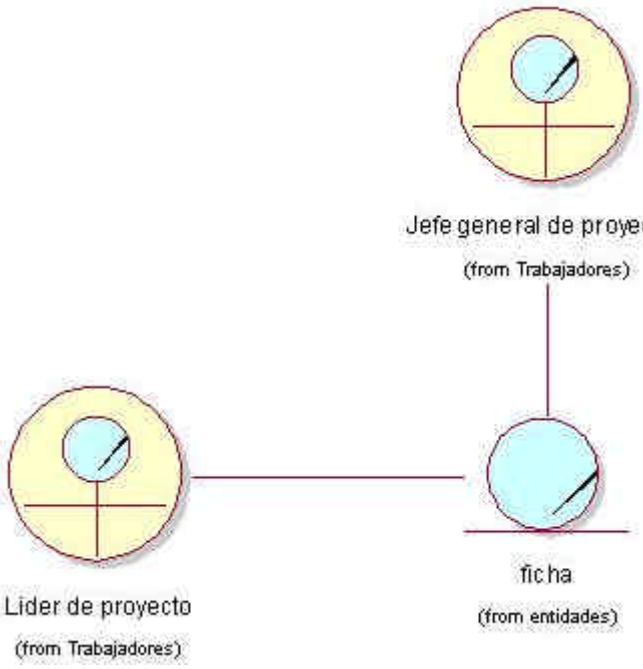


Figura 2.3 Diagrama de clases del modelo de objetos del caso de uso del negocio "Solicitar presupuesto por proyecto".

2.7 Conclusiones.

En este capítulo fue descrita la propuesta de negocio donde se detalla como deben ejecutarse los procesos que se llevan a cabo en la gestión de presupuestos en el marco del convenio Cuba-Venezuela siendo identificados, además, los roles y entidades u objetos del negocio, así como su relación en esos procesos. Esta descripción fue realizada mediante el modelo del negocio, para lo cual se elaboraron los modelos de casos de uso y objetos del negocio.

Tras realizar el modelado del negocio se pudo lograr una mejor comprensión del problema que el sistema tiene que resolver.

3 CAPÍTULO
Requisitos

3.1 Introducción.

En este capítulo se identifican 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.

3.2 Definición de los requisitos funcionales.

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

Para cumplir con los objetivos propuestos se prevé que el sistema tenga las siguientes funcionalidades:

R1. Administración

R1.1 Gestionar usuarios.

R1.1.1 Adicionar usuario.

R1.1.2 Cambiar clave usuario.

R1.1.3 Eliminar usuario.

R1.2 Gestionar proyectos.

R1.2.1 Crear proyecto.

R1.2.2 Establecer permisos a los proyectos.

R1.3 Mostrar Gráficos.

R1.3.1 Mostrar gráfico de las estadísticas generales.

R1.3.2 Mostrar gráfico de las estadísticas por ministerios.

R1.4 Actualizar Variables.

R1.5 Gestionar Responsables.

R1.5.1 Agregar Ministerio.

R1.5.2 Agregar Entidad.

R1.5.3 Agregar Persona Responsable.

R1.5.4 Actualizar Persona Responsables.

R2. Gestionar Proyecto.

R2.1 Gestionar textos del proyecto.

R2.2 Gestionar objetivos del proyecto.

R2.2.1 Adicionar o Actualizar objetivo principal del proyecto.

R2.2.2 Adicionar objetivos específicos de un proyecto.

R2.2.3 Actualizar objetivo específico de un proyecto.

R2.2.4 Ordenar los objetivos específicos de un proyecto.

R2.2.5 Eliminar objetivo específico de un proyecto.

R2.3 Gestionar plan operativo del proyecto.

R2.3.1 Adicionar actividad al plan operativo.

R2.3.2 Modificar actividad del plan operativo.

R2.3.3 Ordenar las actividades del plan operativo.

R2.3.4 Eliminar actividad del plan operativo.

R2.4 Gestionar las finanzas del proyecto.

R2.4.1 Editar las finanzas de una actividad.

R2.4.2 Recalcular las finanzas de una actividad.

R2.4.3 Registrar las compras de una actividad.

R3. Usuario

R3.1 Autenticarse.

R3.2 Cambiar contraseña.

R3.3 Accionar sobre los proyectos

R3.3.1 Listar Proyectos.

R3.3.2 Mostrar Ficha.

R3.3.3 Imprimir Ficha.

3.3 Definición de los 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.

A continuación se muestran los requerimientos no funcionales:

Apariencia o interfaz externa

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

Usabilidad.

El sistema puede ser usado por cualquier persona, comprendida en edad laboral de 18 a 60 años, que posea conocimientos básicos en el manejo de la computadora y de un ambiente Web en sentido general. Instalar el sistema trae consigo una mayor rapidez de trabajo y por consiguiente un ahorro de materiales y personal.

Rendimiento

La disponibilidad de trabajo en red contra el servidor es constante.Se garantiza que la respuesta a solicitudes de los usuarios del sistema sea en un período de tiempo breve (de segundos) para evitar la acumulación de trabajo por parte de los responsables. El sistema deberá de ser lo más estable y confiable posible.

Soporte

Se requiere que el producto reciba mantenimiento ante cualquier fallo que ocurra. El sistema es de fácil instalación.

Software

Para el funcionamiento del sistema en el servidor es necesario el S.O. Windows 98 o superior, Linux o Unix, en sus versiones de S.O. servidores. Para el funcionamiento del sistema en las terminales cliente es necesario el S.O. Windows 95 o superior, Linux o Unix.

Hardware

Se necesitan como requerimientos mínimos una PC con procesador Pentium II o superior.

Portabilidad

El producto es usado bajo los S.O. Windows, Linux y Unix.

El producto corre sobre una plataforma Web, codificada en “PHP4” y sus sistemas de bases de datos en PostgreSQL.

Seguridad

El sistema se encarga de controlar los diferentes niveles de acceso y funcionalidad de usuarios al sitio, de identificar al usuario antes de que pueda realizar cualquier acción

sobre el sistema. Garantiza que la información sea vista únicamente por quien tiene derecho a verla.

Existe un primer nivel o nivel básico donde están las funciones asociadas al usuario general o común, que requieren poca responsabilidad en este nivel solo están los usuarios que solo van a ver datos del sistema. El segundo nivel esta compuesto por funciones de mayor complejidad y que pueden destruir información relacionada a las entidades del sistema. El tercer nivel esta conformado con las funciones administrativas del sitio y del sistema por tanto es el nivel de mayor complejidad.

Se usan mecanismos de encriptación (MD5) de los datos que por cuestiones de seguridad no deben viajar al servidor en texto claro, como es el caso de las contraseñas.

Se hacen validaciones de la información tanto en el cliente como en el servidor, no obstante los usuarios acceden de manera rápida y operativa al sistema sin que los requerimientos de seguridad se conviertan en un retardo para ellos.

Confidencialidad

Toda la información está protegida del acceso no autorizado, los Lideres generales de proyectos (administradores de sistema) son los únicos que podrán transformar cualquier información, los lideres de proyectos (escritores) solo podrán teclear y modificar la información referente a sus proyectos y los responsables por países (lectores) solo podrán ver las fichas de los proyectos.

Disponibilidad

Se garantiza a los usuarios del sistema el acceso a la información solicitada en todo momento (si tiene permiso para ello).

Restricciones en el diseño y la implementación

Es una aplicación Web desarrollada con la tecnología para creación de páginas Web dinámicas PHP4 y base de datos en PostgreSQL.

Legales

El sistema se basa en un estándar que se rige por normas internacionales y cumple con las normas y leyes establecidas en nuestro país.La plataforma escogida para el desarrollo de la aplicación, está basada en la licencia GNU/GPL.

Confiabilidad

La herramienta de implementación a utilizar tiene soporte para recuperación ante fallos y errores.

Restricciones

Se utiliza UML para lograr una mejor documentación del sistema y como herramienta de apoyo Rational Rose. Se utiliza como lenguaje de programación PHP4 y el gestor de base de datos PostgreSQL.

3.4 Actores del sistema a automatizar.

Los actores del sistema pueden representar el rol que juega 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 en la tabla 3.1.

Actores	Justificación
Líder General de Proyectos	Se encarga de Registrar en el sistema los usuarios que interactúan con el mismo, así como designar sobre que proyectos van a tener acceso y que tipo de acceso. Además registra los proyectos en el sistema y genera

	los gráficos estadísticos. Hereda el rol del Responsable por País.
Líder de Proyecto	Se encarga de teclear y modificar la información referente a su proyecto o proyectos. Hereda el rol del Responsable por País.
Responsable por País	Este actor solo va a ver las fichas. Y realiza las acciones que normalmente realiza un usuario en un sistema (autenticarse, cambiar contraseña).

Tabla 3.1 Descripción de los actores del sistema

3.5 Paquetes y sus relaciones.

Un sistema grande se debe dividir en unidades más pequeñas, de modo que pueda ser entendido por las personas que necesiten consultarlo y además para que los equipos de trabajo puedan trabajar de manera independiente.

Para satisfacer los objetivos propuestos al inicio de este trabajo el sistema que se propone debe estar conformado por dos paquetes: Administración, Gestionar Proyecto, Usuario. A continuación en la figura 3.1 se representa el diagrama de paquetes y sus relaciones.

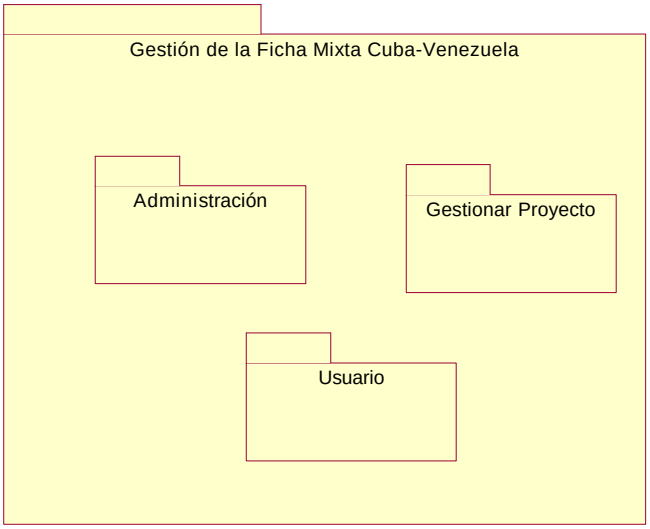


Figura 3.1 Diagrama de paquetes y sus relaciones.

3.6 Paquete "Administración"

En este paquete se recogen las funcionalidades administrativas que permiten el uso adecuado del sistema.

3.6.1 Diagrama de casos de uso.

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.

A continuación se muestra el diagrama de casos de uso del sistema correspondiente al paquete.

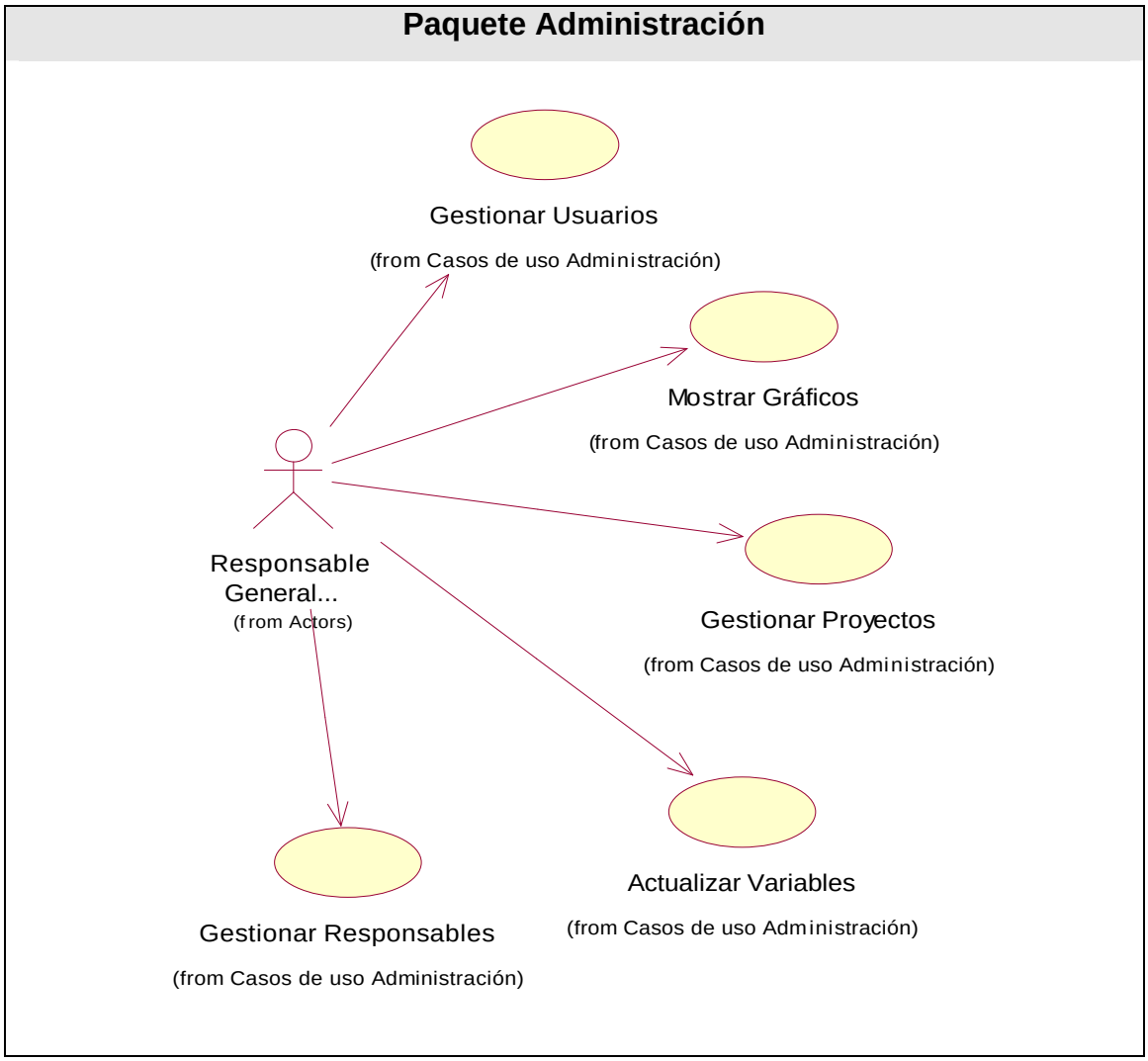


Figura 3.2 Diagrama de casos de uso del sistema del paquete Administración.

3.6.2 Descripción de los casos de uso.

3.6.2.1 Caso de uso "Gestionar Usuarios".

Nombre del Caso de Uso	**Gestionar Usuarios**
Actores	Responsable General de Proyectos
Propósito	Garantizar que los datos de cada Usuario se registren, se le cambien la clave de acceso y se eliminen en el

	sistema.
Resumen	El caso de uso se inicia cuando el Responsable General de Proyectos accede al sistema y solicita Gestionar Usuarios, inmediatamente se visualiza una pantalla donde se muestran los Usuarios existentes en el sistema, a partir de aquí el sistema brinda la posibilidad de efectuar las operaciones de agregar, cambiar la clave o eliminar uno de ellos. Finaliza una vez que el administrador realiza una de las operaciones.
Referencias	R1.1
Precondiciones	Si se desea agregar un usuario debe existir la instancia del rol que va a ejercer.
Poscondiciones	Para los siguientes procesos: Insertar: se crea una instancia de la clase usuario. Cambiar clave: se cambia la clave de la instancia seleccionada de la clase usuario. Eliminar: se elimina la instancia seleccionada de la clase usuario.
Requerimientos especiales	

Curso normal de los eventos	
Acciones del actor	**Respuestas del Sistema**
1. El Responsable General de Proyectos Responsable General de Proyectos decide administrar usuarios y	1.1. El sistema muestra una pantalla que da la posibilidad de agregar un usuario y el listado de los usuarios registrados con la posibilidad de

selecciona la opción.	eliminar y cambiar la clave a alguno de ellos.
2. El administrador selecciona una de las operaciones a realizar.	2.1. El sistema ejecuta una de las siguientes acciones: a) Si decide agregar un usuario, ir a la sección ”Agregar Usuario”. b) Si decide cambiar clave un usuario, ir a la sección “Cambiar Clave Usuario”. c) Si decide eliminar un usuario, ir a la sección “Eliminar Usuario”.
Sección “Agregar Usuario”	
3. El Responsable General de Proyectos introduce los datos del usuario.	3.1. El sistema verifica que los datos introducidos sean válidos y que los requeridos no estén vacíos. 3.2. El sistema verifica que el usuario no exista. 3.3. El usuario se registra en el sistema. 3.4. El sistema muestra un mensaje comunicando que el usuario se agregó satisfactoriamente y finaliza así el caso de uso.
Curso alternativo	
	3.2. Se emite un mensaje de error para que se llenen los campos obligatorios o se introduzcan correctamente los datos.

	3.3. Se emite un mensaje informando de la existencia del apartamento.
Sección "Cambiar Clave Usuario"	
3. El Responsable General de Proyectos selecciona dentro del listado de usuarios el que desea modificar.	3.1. El sistema da la posibilidad de cambiar la clave del usuario
4. El Responsable General de Proyectos realiza las actualizaciones deseadas.	4.1. El sistema verifica que los datos introducidos sean válidos y que los requeridos no estén vacíos. 4.2. El sistema actualiza la información. 4.3. Muestra el listado de los usuarios y finaliza el caso de uso.
Curso alternativo	
	Se emite un mensaje de error para que se llenen los campos obligatorios o se introduzcan correctamente los datos.
Sección "Eliminar Usuario"	
3. El Responsable General de Proyectos selecciona dentro del listado de usuarios el que desea eliminar.	3.1. El sistema muestra un mensaje de confirmación. 3.2. El sistema elimina el usuario y finaliza el caso de uso.
Prototipo	Ver Anexo 8.

Tabla 3.2 Descripción del caso de uso "Gestionar Usuario".

3.6.2.2 Caso de uso "Gestionar Proyectos".

Nombre del Caso de Uso	**Gestionar Proyectos**
Actores	Responsable General de Proyectos
Propósito	Agregar un nuevo proyecto al sistema y establecer quienes van a tener permiso sobre el proyecto y que tipo de permisos va a tener.
Resumen	El caso de uso se inicia cuando el administrador accede al sistema y solicita Gestionar Proyectos, inmediatamente se visualiza una pantalla donde se muestran los proyectos existentes en el sistema, a partir de aquí el sistema brinda la posibilidad de efectuar las operaciones de agregar o administrar un proyecto. Finaliza una vez que el Responsable General de Proyectos realiza una de las operaciones.
Referencias	R1.2
Precondiciones	Para agregar un nuevo proyecto deben existir las instancias Ministerio, Entidad y responsable que son los responsables del proyecto por cada país, y para establecer los permisos sobre un proyecto debe existir la instancia de dicho proyecto.
Poscondiciones	Para los siguientes procesos: Insertar: se crea una instancia de la clase Proyecto. Establecer permisos a los proyectos: se establece que usuarios van a tener permisos sobre un proyecto.
Curso normal de los eventos	
Acciones del actor	**Respuestas del Sistema**

1. El Responsable General de Proyectos decide administrar proyectos y selecciona la opción.	1.1. El sistema muestra una pantalla con el listado de los proyectos.
2. El Responsable General de Proyectos selecciona una de las operaciones a realizar.	2.1. El sistema ejecuta una de las siguientes acciones: a) Si decide agregar un proyecto, ir a la sección “Agregar proyecto”. b) Si decide establecer permisos ir a la sección ”Establecer Permisos”.
Sección “Agregar Proyecto”	
3. El Responsable General de Proyectos introduce los datos de la residencia.	3.1. El sistema verifica que los datos introducidos sean válidos y que los requeridos no estén vacíos. El proyecto se registra en el sistema. El sistema muestra un mensaje comunicando que el proyecto se agregó satisfactoriamente, dando la posibilidad de agregar otro proyecto y finaliza así el caso de uso.
Curso alternativo	
	3.2. Se emite un mensaje de error para que se llenen los campos

	obligatorios o se introduzcan correctamente los datos.
Sección "Establecer Permisos"	
3. El Responsable General de Proyectos selecciona dentro del listado de Proyectos el que desea modificar.	3.1. El sistema Muestra el listado de los usuarios y los permisos para que el responsable haga la selección.
4. El Responsable General de Proyectos selecciona el usuario y el permiso deseado.	4.1. El sistema guarda la información.
Curso alternativo	
Prototipo	Ver Anexo 9.

Tabla 3.3 Descripción del caso de uso "Gestionar Proyecto".

3.6.2.3 Caso de uso "Mostrar Gráficos".

Nombre del Caso de Uso	**Mostrar Gráficos**
Actores	Responsable General de Proyectos
Propósito	Mostrar gráficos estadísticos de la cantidad de dinero que se invierte en los proyectos.
Resumen	El caso de uso se inicia cuando el administrador accede al sistema y ver los gráficos estadísticos.
Referencias	R1.3
Precondiciones	Para ver los gráficos estadísticos debe existir al menos una instancia de proyecto.

<table>
<tr><td>Poscondiciones</td><td colspan="2">Para los siguientes procesos:
Mostrar gráfico de las estadísticas generales: se mostrará un gráfico con las estadísticas generales de todos los proyectos.
Mostrar gráfico de las estadísticas por ministerios: se mostrará un gráfico con las estadísticas generales de todos los ministerios.</td></tr>
<tr><td colspan="3">Requerimientos especiales</td></tr>
<tr><td colspan="3">Curso normal de los eventos</td></tr>
<tr><td colspan="2">Acciones del actor</td><td>Respuestas del Sistema</td></tr>
<tr><td colspan="2">1. El Responsable General de Proyectos decide mostrar los gráficos estadísticos.</td><td>1.1. El sistema da la opción de seleccionar cual de los dos gráficos se va a mostrar.</td></tr>
<tr><td colspan="2">2. El Responsable General de Proyectos selecciona una de las operaciones a realizar.</td><td>2.1. El sistema ejecuta una de las siguientes acciones:
a) Si decide Mostrar gráfico de las estadísticas generales, el sistema muestra el gráfico de las estadísticas generales.
b) Si decide mostrar gráfico de las estadísticas por ministerios, el sistema muestra el gráfico de las estadísticas por ministerios.</td></tr>
<tr><td>Prototipo</td><td colspan="2">Ver Anexo 10.</td></tr>
</table>

Tabla 3.4 Descripción del caso de uso “Mostrar Gráficos”.

3.6.2.4 Caso de uso "Actualizar Variables".

Nombre del Caso de Uso	**Actualizar Variables**
Actores	Responsable General de Proyectos
Propósito	Actualizar los valores de las variables que se usan para el cálculo de los presupuestos de los proyectos.
Resumen	El caso de uso se inicia cuando el Responsable General de Proyectos accede al sistema y solicita actualizar las variables, inmediatamente se visualiza una pantalla donde se muestran las variables existentes en el sistema, a partir de aquí el sistema brinda la posibilidad de efectuar la operación de actualizar una variable. Finaliza una vez que el Responsable General de Proyectos realiza la operación.
Referencias	R1.4
Precondiciones	Para actualizar una variable debe existir la entidad de esa variable.
Poscondiciones	En el proceso de actualizar una variable queda registrado el nuevo valor de esa variable.

Curso normal de los eventos	
Acciones del actor	**Respuestas del Sistema**
1. El Responsable General de Proyectos decide Actualizar las variables.	1.2. El sistema muestra una pantalla con el listado de las variables.
2. El Responsable General de Proyectos selecciona una de las	2.1 El sistema da la posibilidad de teclear los nuevos valores para

variables a actualizar.	la variable
3. El Responsable General de Proyectos entra los nuevos valores de la variable.	El sistema verifica que los datos son correctos. Los datos son actualizados en el sistema.
Curso alternativo	
	Se emite un mensaje de error para que se introduzcan correctamente los datos.
Prototipo	Ver Anexo 11.

Tabla 3.3 Descripción del caso de uso “Actualizar Variables”.

3.6.2.5 Caso de uso “Gestionar Responsables”.

Nombre del Caso de Uso	**Gestionar Responsables**
Actores	Responsable General de Proyectos
Propósito	Agregar y actualizar los ministerios, entidades y responsables que van a tener responsabilidades sobre los proyectos.
Resumen	El caso de uso se inicia cuando el administrador accede al sistema y solicita Gestionar Responsables, aquí va a tener la posibilidad de adicionar Ministerio, Entidad y Responsable y actualizar Responsable. Finaliza una vez que el Responsable General de Proyectos realiza una de las operaciones.
Referencias	R1.5

Precondiciones	Para agregar una entidad debe existir la instancia al ministerio a que pertenece y para agregar un responsable debe existir la instancia de la entidad a la que pertenece y para actualizar un responsable debe existir dicha la instancia a dicha entidad.
Poscondiciones	Para los siguientes procesos: Agregar Ministerio: se crea una instancia de la clase Ministerio. Agregar Entidad: se crea una instancia de la clase Entidad. Agregar Responsable: se crea una instancia de la clase Responsable. Actualizar Responsable: se actualizan los datos de la instancia Responsable.

Curso normal de los eventos	
Acciones del actor	**Respuestas del Sistema**
1. El Responsable General de Proyectos decide Gestionar los responsables.	1.3. El sistema da la posibilidad de escoger agregar Ministerio, Entidad, Persona Responsable o Actualizar los datos de una persona responsable.
2. El Responsable General de Proyectos selecciona una de las operaciones a realizar.	2.1 El sistema ejecuta una de las siguientes acciones: a) Si decide agregar un Ministerio, ir a la sección “Agregar Ministerio”. b) Si decide agregar Entidad ir la

	sección ”Agregar Entidad”. c) Si decide Agregar Responsable ir a la sección “Agregar Responsable”. d) Si decide actualizar los datos de una persona responsable ir a la sección “Actualizar Persona Responsable”.
Sección “Agregar Ministerio”	
3. El Responsable General de Proyectos introduce los datos del ministerio.	3.1 El sistema verifica que los datos introducidos sean válidos y que los requeridos no estén vacíos. 3.2 El ministerio se registra en el sistema. 3.3 El sistema muestra un mensaje comunicando que el ministerio se agregó satisfactoriamente, dando la posibilidad de agregar otro ministerio y finaliza así el caso de uso.
Curso alternativo	
	3.2 Se emite un mensaje de error para que se llenen los campos obligatorios o se introduzcan correctamente los datos.
Sección “Agregar Entidad”	

4. El Responsable General de Proyectos introduce los datos de la entidad.	El sistema verifica que los datos introducidos sean válidos. La entidad se registra en el sistema. El sistema muestra un mensaje comunicando que la entidad se agregó satisfactoriamente, dando la posibilidad de agregar otra entidad y finaliza así el caso de uso.
Curso alternativo	
	4.2 Se emite un mensaje de error para que se introduzcan correctamente los datos.
Sección "Agregar Responsable"	
5. El Responsable General de Proyectos introduce los datos del responsable.	5.1 El sistema verifica que los datos introducidos sean válidos. 5.2 Se registra el responsable en el sistema. 5.3 El sistema muestra un mensaje comunicando que la persona responsable se agregó satisfactoriamente, dando la posibilidad de agregar otra persona responsable y finaliza así el caso de uso.

Curso alternativo	
	5.2 Se emite un mensaje de error para que se introduzcan correctamente los datos.
Sección "Actualizar Persona Responsable"	
6. El Responsable General de Proyectos introduce los nuevos datos del responsable.	6.1 El sistema verifica que los datos introducidos sean válidos. 6.2 Se actualizan los datos de la persona responsable en el sistema. 6.3 El sistema muestra un mensaje comunicando que el la persona responsable actualizó satisfactoriamente, dando la posibilidad de actualizar otra persona responsable y finaliza así el caso de uso.
Curso alternativo	
	6.2 Se emite un mensaje de error para que se introduzcan correctamente los datos.
Prototipo	Ver Anexo 12

Tabla 3.3 Descripción del caso de uso "Gestionar Responsables".

3.6.3 Paquete Gestionar Proyecto.

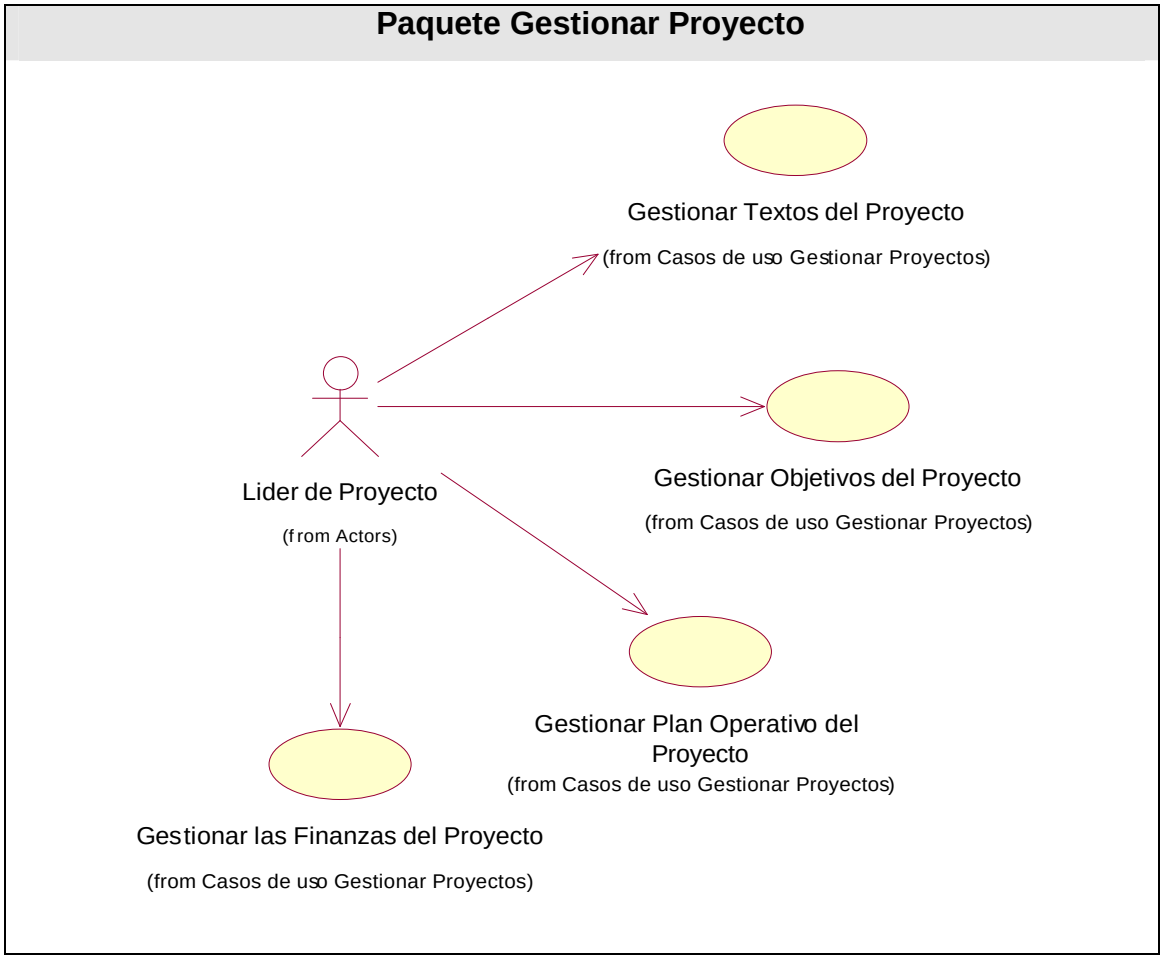


Figura 3.3 Diagrama de casos de uso del sistema “Paquete Gestionar Proyecto”.

3.6.3.1 Caso de uso “Gestionar textos del proyecto”.

Nombre del Caso de Uso	**Gestionar textos del proyecto**
Actores	Líder de Proyecto
Propósito	Registrar los textos descriptivos de los proyectos.
Resumen	El caso de uso se inicia cuando el líder de un proyecto accede al sistema y solicita gestionar los textos de su proyecto, a partir de aquí el sistema brinda la posibilidad de efectuar las operaciones introducir los textos del proyecto y actualizar los textos del proyecto. Finaliza una

	vez que el líder de proyecto realiza una de las operaciones.
Referencias	R2.1
Precondiciones	Debe existir la entidad proyecto a la que se le van a ingresar o actualizar los textos
Poscondiciones	Se registrarán o actualizarán los textos del proyecto.
Requerimientos especiales	Para registrar los textos de un proyecto debe existir la instancia del proyecto y para actualizar los textos del proyecto deben haberse registrado previamente.

Curso normal de los eventos	
Acciones del actor	**Respuestas del Sistema**
1. El líder de proyecto decide gestionar los textos del proyecto.	1.1 El sistema muestra una vista donde el se pueden teclear los textos del proyecto, de tener textos registrados este proyecto se muestra dando la posibilidad de cambiarlos.
2. El líder de proyecto teclea los textos des proyecto.	2.1 El sistema verifica si el proyecto a tiene textos registrados, si los tiene los actualiza, en caso contrario los registra. 2.2 EL sistema muestra un mensaje diciendo que la acción se ejecutó con éxito, y finaliza el caso de uso.

Prototipo	Ver Anexo 13.

Tabla 3.9 Descripción del caso de uso “Gestionar textos del proyecto”.

3.6.3.2 Caso de uso “Gestionar Objetivos del Proyecto”.

Nombre del Caso de Uso	**Gestionar Responsables**
Actores	Líder de proyecto
Propósito	Agregar y actualizar los objetivos generales y específicos de los proyectos, así como ordenar los objetivos específicos y eliminar objetivos específicos.
Resumen	El caso de uso se inicia cuando el líder de proyecto accede al sistema y decide gestionar los objetivos del proyecto.
Referencias	R2.2
Precondiciones	Para gestionar los objetivos de un proyecto debe existir la instancia del proyecto del cual se van a gestionar los objetivos.
Poscondiciones	Para los siguientes procesos: Adicionar o actualizar el objetivo principal de un proyecto se registra el objetivo del proyecto y de estar registrado el objetivo principal entonces se actualiza. Adicionar objetivos específicos del proyecto, se registran los objetivos específicos. Actualizar Objetivos específicos del proyecto, se actualizan los objetivos específicos del proyecto. Ordenar objetivos específicos del proyecto, se ordenan los objetivos específicos del proyecto. Eliminar objetivo específico del proyecto.
Curso normal de los eventos	

Acciones del actor	**Respuestas del Sistema**
1. El Líder de proyecto decide Gestionar los responsables.	1.4. El sistema Muestra una interfaz donde se pueden registrar o actualizar el objetivo del proyecto así como los objetivos específicos, ordenar y eliminar los objetivos específicos.
2. El Líder de proyecto selecciona una de las operaciones a realizar.	2.2 El sistema ejecuta una de las siguientes acciones: a) Si decide registrar o actualizar el objetivo del proyecto, ir a la sección “Adicionar o Actualizar objetivo principal del proyecto”. b) Si decide adicionar objetivo específico ir la sección ”Agregar Objetivo Específico”. c) Si decide Actualizar objetivo específico ir a la sección “Actualizar Objetivo Específico”. d) Si decide ordenar objetivos específicos ir a la sección “Ordenar Objetivos Específicos”. e) Si decide eliminar objetivo específico ir a la sección “Eliminar Objetivo Específico”.

Sección "Adicionar o Actualizar objetivo principal del proyecto"	
3. El Líder de proyecto introduce los datos del ministerio.	3.1 El sistema verifica si hay registrado objetivo del proyecto. 3.2 El ministerio se registra en el sistema y si ya existía un objetivo entonces lo actualiza. 3.3 El sistema muestra un mensaje comunicando que el objetivo se agregó o actualizó satisfactoriamente y finaliza así el caso de uso.
Curso alternativo	
	3.2 Si no existe ningún objetivo registrado del proyecto, entonces lo que intenta hacer es registrar, por tanto si el campo va vacío se emite un error indicando que se teclee el objetivo.
Sección "Agregar Objetivo Específico"	
4. El Líder de proyecto introduce los datos del objetivo específico.	4.1 El sistema verifica que los datos introducidos sean válidos. 4.2 El objetivo específico se registra en el sistema. 4.3 El sistema muestra un mensaje comunicando que el objetivo específico se agregó

	satisfactoriamente, dando la posibilidad de agregar otro objetivo específico y finaliza así el caso de uso.
Curso alternativo	
	4.2 Se emite un mensaje de error para que se introduzcan correctamente los datos.
Sección "Actualizar Objetivo Específico"	
5. El Líder de proyecto introduce los nuevos datos del objetivo específico.	5.1 El sistema verifica que los datos introducidos sean válidos. 5.2 Se actualiza el objetivo específico en el sistema. 5.3 El sistema muestra un mensaje comunicando que el objetivo específico se actualizó satisfactoriamente y finaliza así el caso de uso.
Curso alternativo	
	5.2 Se emite un mensaje de error para que se introduzcan correctamente los datos.
Sección "Ordenar Objetivos Específicos"	
6. El líder de proyecto indica la	6.1 El sistema actualiza el orden de

variación en el orden de un objetivo específico.	los objetivos específicos y refresca la interfaz. 6.3 El sistema da la posibilidad de seguir ordenando los objetivos específicos y finaliza así el caso de uso.
Sección "Eliminar Objetivo Específico"	
7. El líder de proyecto selecciona el objetivo específico a eliminar.	7.1 El sistema elimina el objetivo específico. 7.2 Muestra un mensaje informando que el objetivo específico fue eliminado y refresca la interfaz. Finaliza así el caso de uso.
Prototipo	Ver Anexo 14.

Tabla 3.3 Descripción del caso de uso "Gestionar Objetivos del Proyecto".

3.6.3.3 Caso de uso "Gestionar plan operativo del proyecto".

Nombre del Caso de Uso	**Gestionar plan operativo del proyecto**
Actores	Líder de proyecto
Propósito	Adicionar, actualizar, ordenar y eliminar las actividades definidas para el plan operativo del proyecto.
Resumen	El caso de uso se inicia cuando el líder de proyecto accede al sistema y decide gestionar el plan operativo del proyecto.
Referencias	R2.3
Precondiciones	Para gestionar el plan operativo de un proyecto debe

	existir la instancia del proyecto del cual se va a gestionar el plan operativo.
Poscondiciones	Para los siguientes procesos: Adicionar una actividad al plan operativo de un proyecto se registra la actividad en el sistema. Modificar una actividad del plan operativo, se actualiza en el sistema la actividad seleccionada. Ordenar las actividades del plan operativo, se ordenan las actividades del plan operativo. Eliminar actividad del plan operativo, se elimina la actividad seleccionada.

Curso normal de los eventos	
Acciones del actor	**Respuestas del Sistema**
1. El Líder de proyecto decide Gestionar el plan operativo del proyecto.	1.1 El sistema Muestra una interfaz donde se puede adicionar, actualizar, ordenar y eliminar las actividades del plan operativo.
2. El Líder de proyecto selecciona una de las operaciones a realizar.	2.1 El sistema ejecuta una de las siguientes acciones: a) Si decide registrar una actividad del plan operativo, ir a la sección “Adicionar Actividad al Plan Operativo”. b) Si decide modificar una actividad del plan operativo ir la sección ”Modificar Actividad del Plan Operativo”.

	c) Si decide ordenar las actividades del plan operativo ir a la sección "Ordenar Actividades del Plan Operativo". d) Si decide Eliminar actividad del plan operativo ir a la sección "Eliminar Actividad del Plan Operativo".
Sección "Adicionar Actividad al Plan Operativo"	
3. El Líder de proyecto introduce los datos de la actividad.	3.1 El sistema verifica los datos. 3.2 El registra la actividad. 3.3 El sistema muestra un mensaje comunicando que la actividad se agregó satisfactoriamente y finaliza así el caso de uso.
Curso alternativo	
	3.2 Se emite un mensaje de error para que se introduzcan correctamente los datos.
Sección "Modificar Actividad del Plan Operativo"	
4. El Líder de proyecto introduce los datos de la actividad.	4.2 El sistema verifica que los datos introducidos sean válidos. 4.3 La actividad se registra en el sistema. 4.4 El sistema muestra un mensaje comunicando que la actividad se

	agregó satisfactoriamente, finaliza así el caso de uso.
Curso alternativo	
	4.2 Se emite un mensaje de error para que se introduzcan correctamente los datos.
Sección "Ordenar Actividades del Plan Operativo"	
5. El Líder de proyecto indica el orden a variar por una actividad.	5.1 El sistema actualiza el orden de las actividades y refresca la interfaz.
Sección "Eliminar Actividad del Plan Operativo"	
6. El líder de proyecto selecciona la actividad a eliminar.	6.1 El sistema elimina la actividad seleccionada. 6.2 El sistema muestra un mensaje informando que la actividad ha sido eliminada y finaliza así el caso de uso.
Prototipo	Ver Anexo 15.

Tabla 3.3 Descripción del caso de uso "Gestionar plan operativo del proyecto".

3.6.3.4 Caso de uso "Gestionar las finanzas del proyecto".

Nombre del Caso de Uso	**Gestionar las finanzas del proyecto**
Actores	Líder de proyecto
Propósito	Editar manualmente las finanzas del proyecto y hacer que el sistema recalcule las finanzas.
Resumen	El caso de uso se inicia cuando el líder de proyecto accede al sistema y decide gestionar las finanzas del

	proyecto.
Referencias	R2.4
Precondiciones	Para gestionar las finanzas de un proyecto debe existir la instancia del proyecto del cual se van a gestionar las finanzas.
Poscondiciones	Para los siguientes procesos: Editar las finanzas se registran en el sistemas valores diferentes a los que este calcula. Recalcular las finanzas de un proyecto, se desechan los datos que no coinciden con lo que calcula el sistema y se retoman los calculados. Entrar las compras de una actividad, registra en el sistema los materiales comprados para la realización de la actividad en caso de que requiera compra de materiales.

Curso normal de los eventos	
Acciones del actor	**Respuestas del Sistema**
1. El Líder de proyecto decide Gestionar las finanzas del proyecto.	1.1 El sistema Muestra una interfaz donde se muestra el calculo que realiza el sistema para determinar las finanzas, desglosado por actividades y país donde se realizan. Dando la posibilidad de variar los datos calculados por el sistema tecleando otros, retomar los datos que calcula el sistema y registrar las compras que

	requiere una actividad.
2. El Líder de proyecto selecciona una de las operaciones a realizar.	2.2 El sistema ejecuta una de las siguientes acciones: a) Si decide editar las finanzas de una actividad, ir a la sección “Editar Finanzas”. b) Si decide recalcular las finanzas de una actividad, ir la sección ”Recalcular”. c) Si decide registrar compras de una actividad ir a la sección “Registrar Compra”.
Sección “Editar Finanzas”	
3. El Líder de proyecto introduce nuevos datos de las finanzas de una actividad.	3.1 El sistema verifica los datos. 3.2 Se registran lo datos de las finanzas de la actividad. 3.3 El sistema muestra un mensaje comunicando que se han guardado los datos satisfactoriamente y finaliza así el caso de uso.
Curso alternativo	
	3.2 Se emite un mensaje de error para que se introduzcan correctamente los datos.
Sección “Recalcular”	

4. El Líder de proyecto selecciona la actividad de la que desea se recalculen sus finanzas.	4.2 El sistema recalcula las finanzas de la actividad y refresca la interfaz.
Sección "Registrar Compra"	
5. El Líder de proyecto introduce los datos de una compra de la actividad.	5.1 El sistema verifica los datos. 5.2 El sistema registra la compra de materiales. 5.3 El sistema muestra un mensaje comunicando que se han guardado los datos satisfactoriamente y finaliza así el caso de uso.
Curso alternativo	
	5.2 Se emite un mensaje de error para que se introduzcan correctamente los datos.
Prototipo	Ver Anexo 16.

Tabla 3.3 Descripción del caso de uso "Gestionar las finanzas del proyecto".

3.6.4 Paquete Usuario.

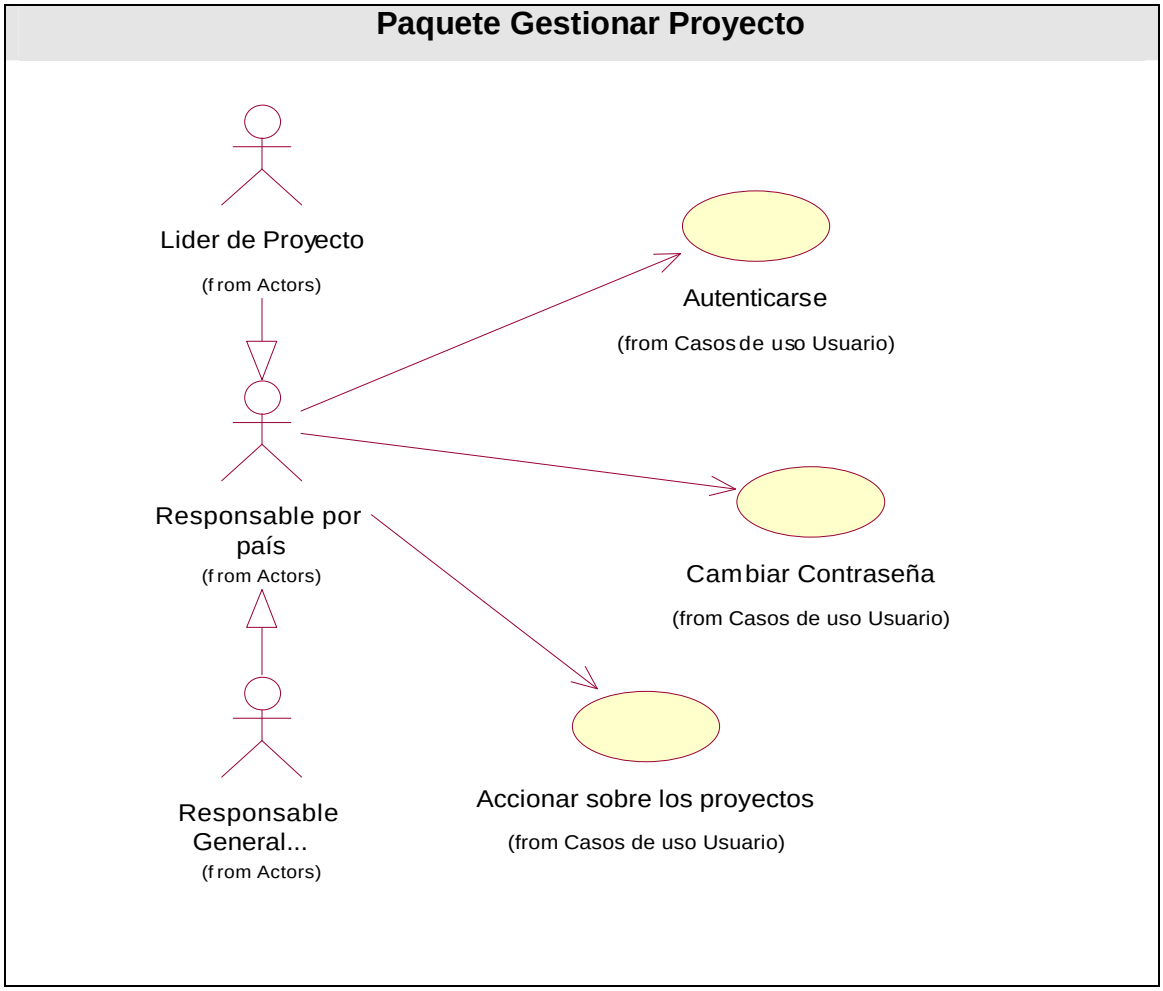


Figura 3.3 Diagrama de casos de uso del sistema “Paquete Usuario”.

3.6.4.1 Caso de uso “Autenticarse”.

Nombre del Caso de Uso	**Autenticarse**.
Actores	Responsable por País
Propósito	Autenticarse en el sistema para poder operar en el.
Resumen	El caso de uso inicia cuando el Responsable por país escribe su usuario y su contraseña para acceder al sistema, este verifica el usuario y la contraseña y le permite acceder a los vínculos a los que tiene permiso.

Referencias	R3.1
Precondiciones	Para autenticarse en el sistema y acceder al mismo debe existir la instancia del usuario.
Poscondiciones	Para los siguientes procesos: Autenticarse en el sistema, el usuario tiene acceso en el sistema a los vínculos para los cuales tiene permiso.
Curso normal de los eventos	
Acciones del actor	**Respuestas del Sistema**
3. El Responsable por País introduce el usuario y la contraseña.	1.1 el sistema verifica los datos introducidos. 1.2 El sistema le muestra el listado de los proyectos y los vínculos que podrá accionar y finaliza el caso de uso.
Curso Alternativo	
	1.2 El sistema muestra un mensaje indicando que el usuario y la contraseña no son correctos.
Prototipo	Ver Anexo 17.

Tabla 3.3 Descripción del caso de uso “Autenticarse”.

3.6.4.2 Caso de uso “Cambiar Contraseña”.

Nombre del Caso de Uso	**Cambiar Contraseña**.
Actores	Responsable por País

Propósito	Cambiar la contraseña.
Resumen	El caso de uso inicia cuando el Responsable por país entra los datos para cambiar su contraseña.
Referencias	R3.2
Precondiciones	Para cambiar la contraseña debe existir la instancia del usuario.
Poscondiciones	Se cambia la contraseña.

Curso normal de los eventos	
Acciones del actor	**Respuestas del Sistema**
1. El Responsable por País introduce la contraseña actual y la nueva para que sea cambiada.	1.3 El sistema verifica los datos introducidos. 1.4 El sistema muestra un mensaje de que la contraseña ha sido cambiada y finaliza el caso de uso.
Curso Alternativo	
	1.2 El sistema muestra un mensaje indicando que hay error en los datos.

Prototipo	Ver Anexo 18.

Tabla 3.3 Descripción del caso de uso “Cambiar Contraseña”.

3.6.4.3 Caso de uso "Accionar Sobre los proyectos".

Nombre del Caso de Uso	**Accionar Sobre los proyectos**
Actores	Responsable por País
Propósito	Ver las fichas de los proyectos registrados e imprimir las mismas.
Resumen	El caso de uso inicia cuando el Responsable por país accede al sistema y decide ver la ficha de un proyecto.
Referencias	R3.3
Precondiciones	Para ver la ficha de un proyecto debe existir la instancia del mismo.
Poscondiciones	Para los siguientes procesos: Listar los Proyectos, el sistema muestra un listado con todos los proyectos a sobre los que tiene permiso. Mostrar Ficha, el sistema muestra una tabla con el contenido de la ficha de un proyecto. Imprimir ficha, se imprime la ficha mediante una impresora acoplada a la PC.

Curso normal de los eventos	
Acciones del actor	**Respuestas del Sistema**
4. El Responsable por País decide revisar los proyectos.	1.1 El sistema muestra el listado de los proyectos sobre los que el Responsable por País tiene permiso.
5. El Responsable por País selecciona una de las operaciones a realizar.	2.1 El sistema ejecuta una de las siguientes acciones:

	a) Si decide mostrar la ficha, ir a la sección “Mostrar Ficha”. b) Si decide Imprimir la ficha, ir la sección ”Imprimir Ficha”. c) Si decide listar los proyectos ir a la sección “Listar Proyectos”.
Sección “Mostrar Ficha”	
3. El Responsable por País selecciona el proyecto del cual quiere ver la ficha.	3.1 El sistema recopila los datos necesarios. 3.2 El sistema muestra la ficha del proyecto y finaliza así el caso de uso.
Sección “Imprimir Ficha”	
4. El Responsable por País indica al sistema imprimir la ficha que está en pantalla.	4.2 El sistema imprime la ficha y termina el caso de uso.
Prototipo	Ver Anexo 19.

Tabla 3.3 Descripción del caso de uso “Accionar Sobre los proyectos”.

3.7 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 representaron los diagramas de casos de uso del sistema, y finalmente se describieron las acciones de los actores del sistema con los casos de uso con los que interactúan. Gracias a esto ahora se puede empezar a construir el sistema tratando de que se cumplan todos los requerimientos y las funciones que han sido consideradas necesarias en este capítulo.

4 CAPÍTULO
Descripción de la solución propuesta

4.1 Introducción.

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 diseño del sistema propuesto y cómo se implementa éste. Así, se presentan los diagramas de clases Web que detallan la interacción de las distintas páginas; se estructura la información que se desea persista a través del diseño de la base de datos; se describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. Son también descritos los estándares de diseño y programación seguidos.

4.2 Clases Base.

Los casos de uso del sistema definidos en el capítulo anterior, deben encajar en la arquitectura cuando se llevan a cabo, mientras que la arquitectura debe permitir el desarrollo de los casos de uso requeridos ahora y en el futuro. De esa manera, la arquitectura debe diseñarse para permitir que el sistema evolucione, que los desarrolladores puedan progresar hasta obtener una visión común, que se organice el desarrollo del software y que se fomente la reutilización. A partir de esto se definen explícitamente interfaces, haciendo posible una buena comunicación entre los desarrolladores. También se consideran las posibilidades de reutilización de las partes del sistema parecidas o de productos software generales. Los subsistemas, interfaces u otros elementos del diseño se añaden posteriormente.

Teniendo en cuenta las posibilidades que ofrece el patrón Modelo Vista Controlador, descrito en el Capítulo 1 se decide hacer uso del mismo para definir el sistema de clases base de la aplicación, pues se persigue facilidad de mantenimiento, escalabilidad, rapidez de desarrollo e independencia entre las distintas capas del sistema para facilitar su futura evolución.

4.2.1 Diagrama de Clases.

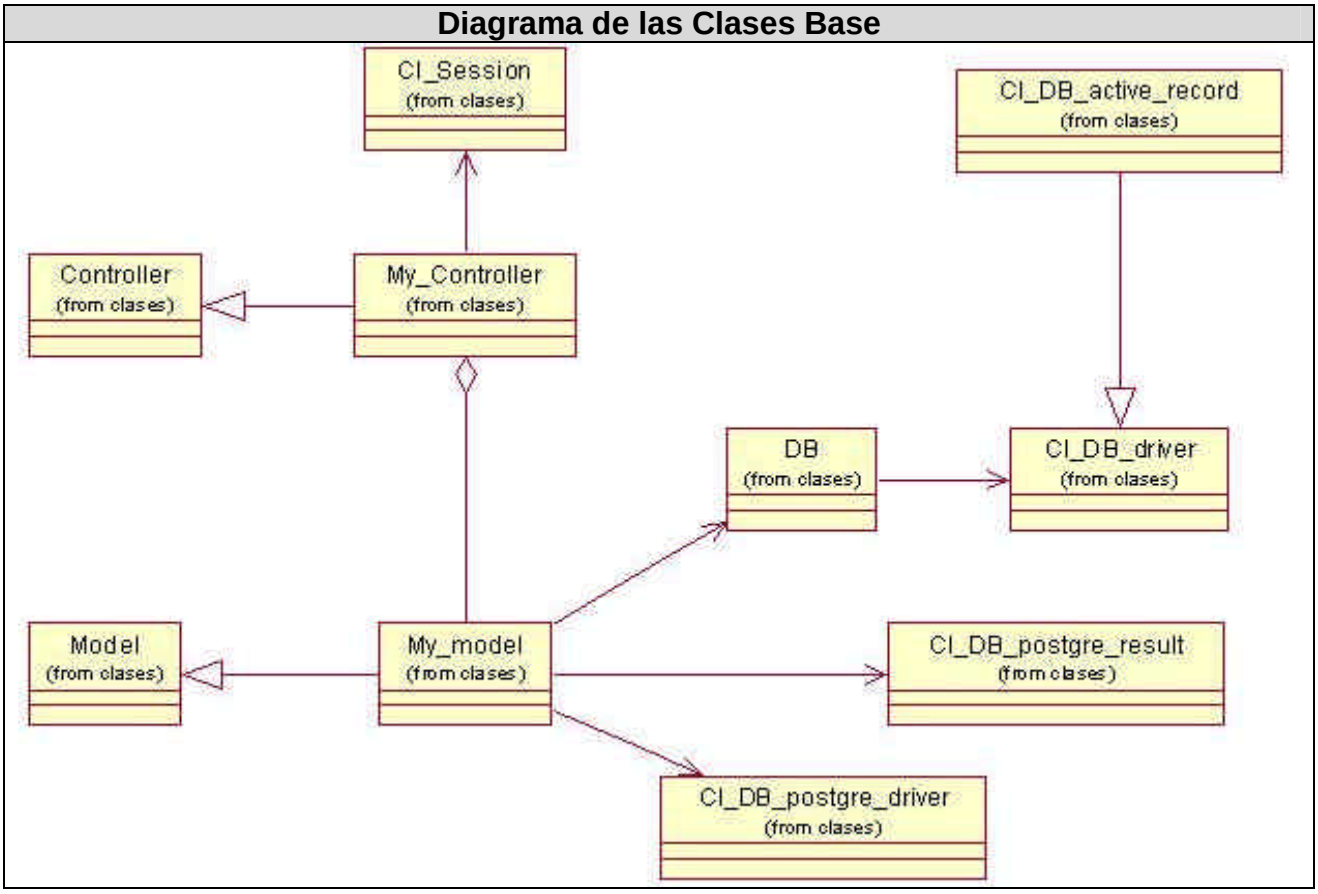


Figura 4.1 Diagrama de las Clases Base.

4.2.2 Descripción de las clases.

Controller	**Súper clase que se encarga de servir de base funcional de las clases controladoras.**
Métodos	
Controller	Constructor de la clase que se encarga de inicializar los elementos.
_ci_initialize	Asigna todas las clases bases cargadas por el inicio de la Controller por variables en esta clase. También se encarga de llamar al autoload de la aplicación.

Tabla 4.1 Descripción de la clase Controller

Model	**Súper clase que se encarga de servir de base funcional de las clases modelos.**
Métodos	
Model	Constructor de la clase que se encarga de inicializar los elementos.
_assign_libraries	Crea referencia locales a todos los objetos instanciados concurrentemente y permite que cualquier sintaxis se legalmente usado en la las controladoras o entre modelos.

MY_Controller	**Clase que se encarga de servir de base añadiendo nuevos métodos funcionales.**
Métodos	
MY_Controller	Constructor de la clase que se encarga de inicializar los elementos.
render	Renderiza la página con el template adjunto
isLogin	Determina si existe un usuario logueado en el sistema
Permission	Comprueba que exista un determinado permiso
isPermission	Comprueba si tiene un permiso determinado (este método no redirecciona)
verificar_proyecto	Verifica la existencia de un proyecto activo en edición
verificar_permiso_proyecto	Verificar el permiso (ROL) dentro del proyecto
PermissionProyecto	Verifica el privilegio de una persona en el proyecto
isPostBack	Verifica si existe el envío por POST

MY_Model	**Clase que se encarga de servir de base añadiendo nuevos métodos funcionales.**
Métodos	
MY_Model	Constructor de la clase que se encarga de inicializar los elementos.
get_table_fields	Obtiene el listado de campos de una tabla

	determinada en la DB.
get_elements	Extrae de un arreglo de datos la porción que coincida con los valores de otro arreglo.
insertar	Inserta datos en una tabla y devuelve el ID generado.
actualizar	Actualiza una tabla de la base de datos.
get_table_data	Obtiene los datos de una tabla de la base de datos con los valores formateados.
isset_proyecto	Dado un ID de proyecto verifica si el proyecto existe.
obtener_entes_by_paises	Muestra las entidades pertenecientes al país dado.
obtener_idministerio_by_entidad	Muestra todos los id y nombre de ministerios dado una entidad.
obtener_ministerio	Selecciona un ministerio que se quiera buscar.
obtener_entidad	Selecciona una entidad que se quiera buscar.
obtener_rol_proyecto	Verifica si el usuario es creador y el privilegio que tiene sobre ese proyecto.

Session	**Clase que se encarga de tratar todos los métodos y funcionalidades para el trabajo con sesión**
Métodos	
Session	Constructor de la clase que se encarga de inicializar los elementos.
regenerate_id	Genera el id de la sesión
destroy	Destruye la sesión y borra todo lo esta guardado en la sesion
get	Devuelve lo guardado en la session
exist	Verifica si existe un atributo en la sesión
set	Guarda en atributos en la sesión
delete	Borra los atributos de las sesiones
_session_id_expired	Chequea si la sesión ha expirado

CI_DB_postgre_driver	**Clase que se encarga de proveer métodos y funcionalidades importante a la hora del manejo de datos**
Métodos	
CI_DB_postgre_driver	Constructor de la clase que se encarga de inicializar los elementos.
db_connect	Conexión no persistente de a la base de datos.
db_pconnect	Conexión persistente a la base de datos.
_version	Muestra la versión.
_execute	Ejecuta un query.
trans_begin	Inicia transacciones.
trans_commit	Termina transacciones.
trans_rollback	Vuelve a atrás las transacciones.
insert_id	Inserta un id.
count_all	Cuenta todas las consultas.
_list_tables	Muestra la consulta de una tabla.
_error_message	Muestra mensages de error.
_insert	Dado los datos forma una consulta Insert.
_update	Dado los datos forma una consulta Update.
_delete	Dado los datos forma una consulta Delete.
_close	Cierra la conexión a la base de datos.

CI_DB_postgre_result	**Clase que se encarga de proveer métodos y funcionalidades importante a la hora del manejo de datos**
Métodos	
CI_DB_postgre_result	Constructor de la clase que se encarga de inicializar los elementos.

num_rows	Muestra el números de filas del resultado enviado
num_fields	Muestra el números de campos del resultado enviado
list_fields	Genera un arreglo de columnas con sus nombres
field_data	Obtiene los datos del campo

DB	**Clase de gestionar el acceso a métodos de clases para el acceso a datos.**
Métodos	
DB	Constructor de la clase que se encarga de inicializar los elementos. Ademas carga todas las clase instanciadas a el.

CI_DB_driver	**Clase que se encarga de proveer métodos y funcionalidades importante a la hora del manejo de datos**
Métodos	
CI_DB_driver	Constructor de la clase que se encarga de inicializar los elementos.
initialize	Inicializa archivos de la base de datos.
platform	Devuelve el nombre de la plataforma de base de datos que esta en uso.
version	Devuelve la versión de la base de datos
query	Ejecuta una consulta dada.
trans_off	Desabilita transacciones
trans_start	Habilita transacciones
trans_complete	Transacciones completadas
trans_status	Estado de las transacciones
is_write_type	Comprueba si la consulta es del tipo escritura
total_queries	Retorna el numero total de consulta
last_query	Returna el ultimo query que ha sido ejecutado

list_tables	Retorna un arreglo con nombres de las tablas
table_exists	Determina si una tabla dada existe
field_exists	Determina si un campo dada existe
field_data	Retorna un objeto con los datos de los campos
insert_string	Genera una consulta Insert
update_string	Genera una consulta Update
display_error	Muestra un error en pantalla

CI_DB_active_record	**Clase que se encarga de proveer métodos y funcionalidades importante a la hora del manejo de datos**
Métodos	
CI_DB_active_record	Constructor de la clase que se encarga de inicializar los elementos.
select	Genera la porción del select de la consulta
from	Genera la porción del from de la consulta
join	Genera la porción del join de la consulta
where	Genera la porción del where de la consulta
like	Genera la porción del %like% de la consulta
groupby	Genera la porción del groupby de la consulta
having	Genera la porción del having de la consulta
orderby	Genera la porción del orderby de la consulta
limit	Genera la porción del limit de la consulta
insert	Genera la porción del insert de la consulta
update	Genera la porción del update de la consulta
delete	Genera la porción del delete de la consulta
order_by	Genera la porción del order_by de la consulta

4.2.3 Diagramas de secuencia.

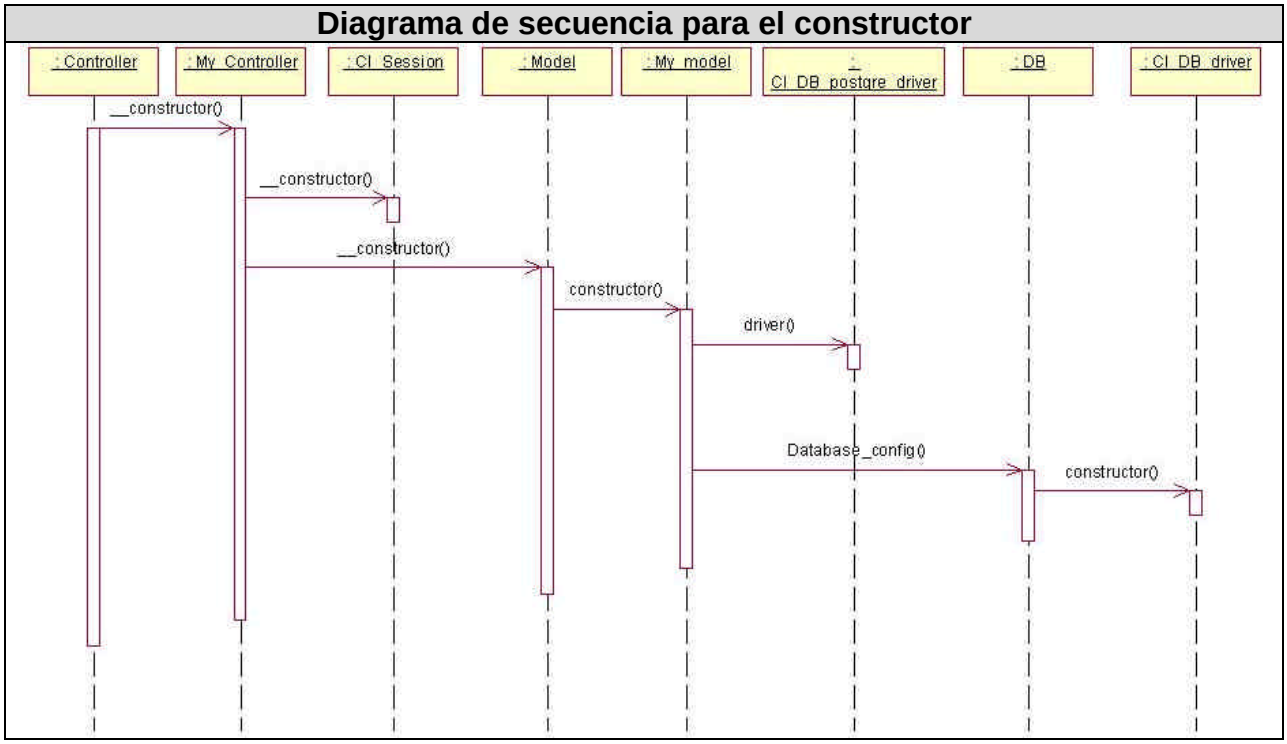


Figura 4.2 Diagrama de secuencia para el Constructor.

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.

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 paquetes.

4.3.1 Paquete "Administración"

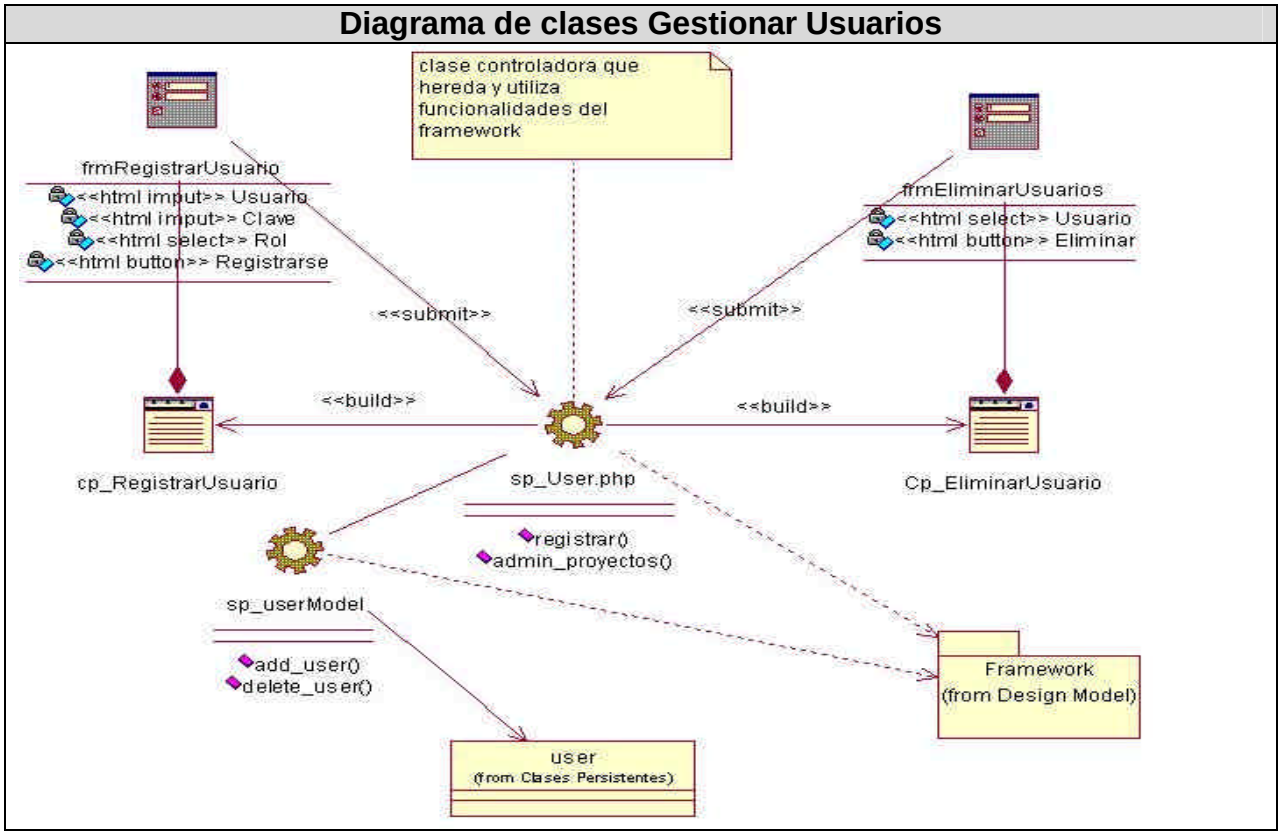


Figura 4.3 Diagrama de clases Gestionar Usuarios.

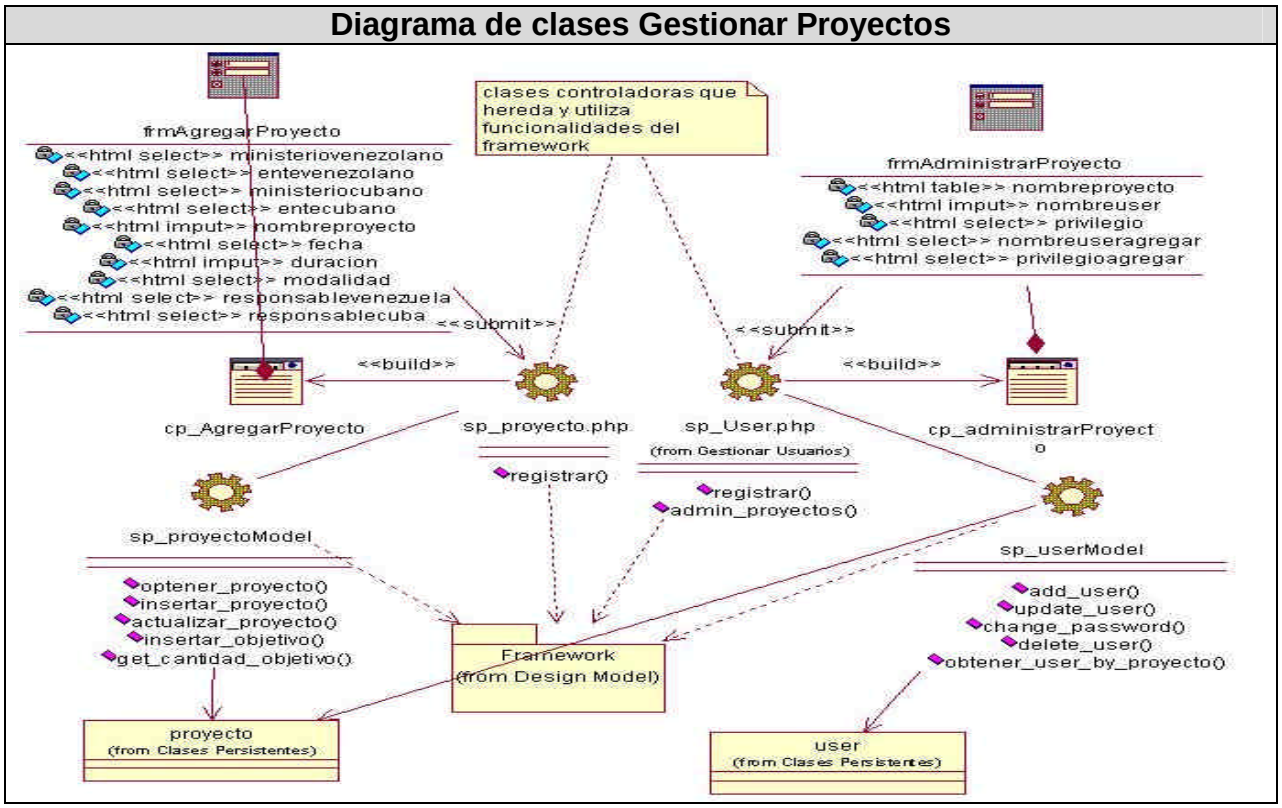


Figura 4.4 Diagrama de clases Gestionar Proyectos.

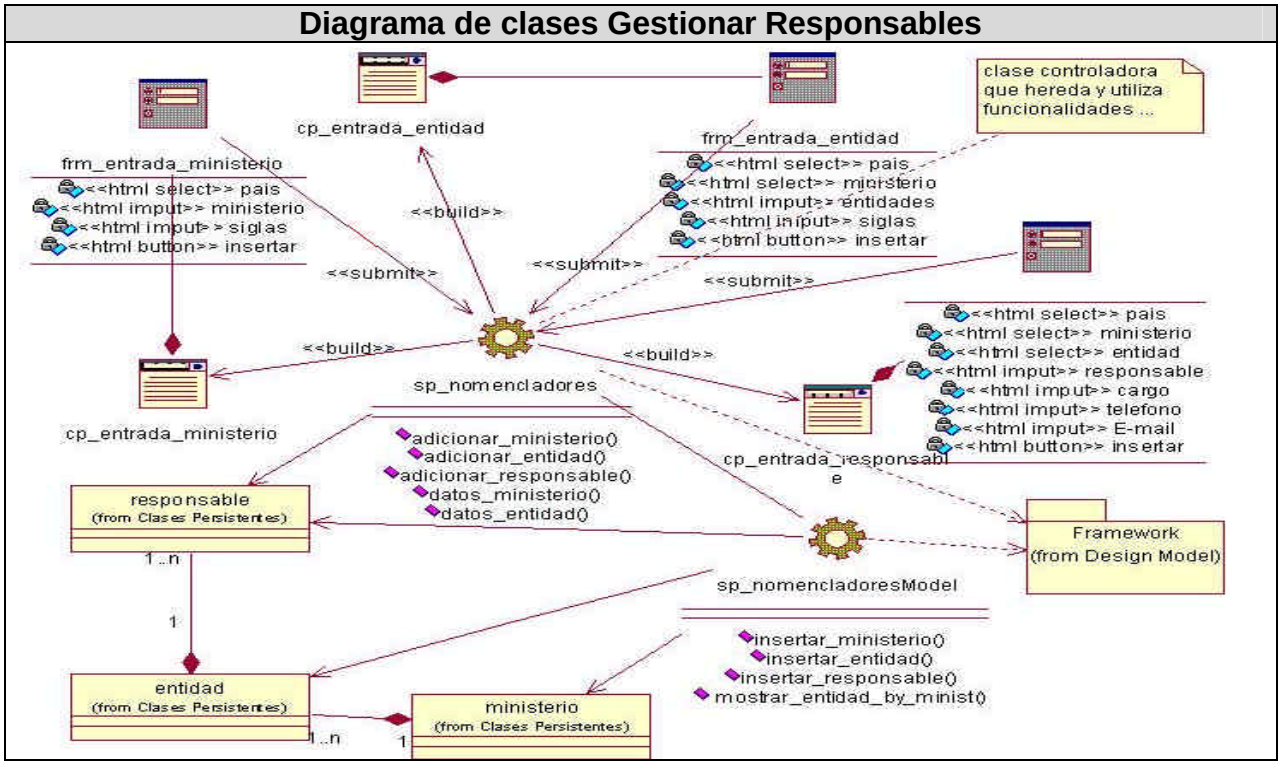


Figura 4.5 Diagrama de clases Gestionar Responsables.

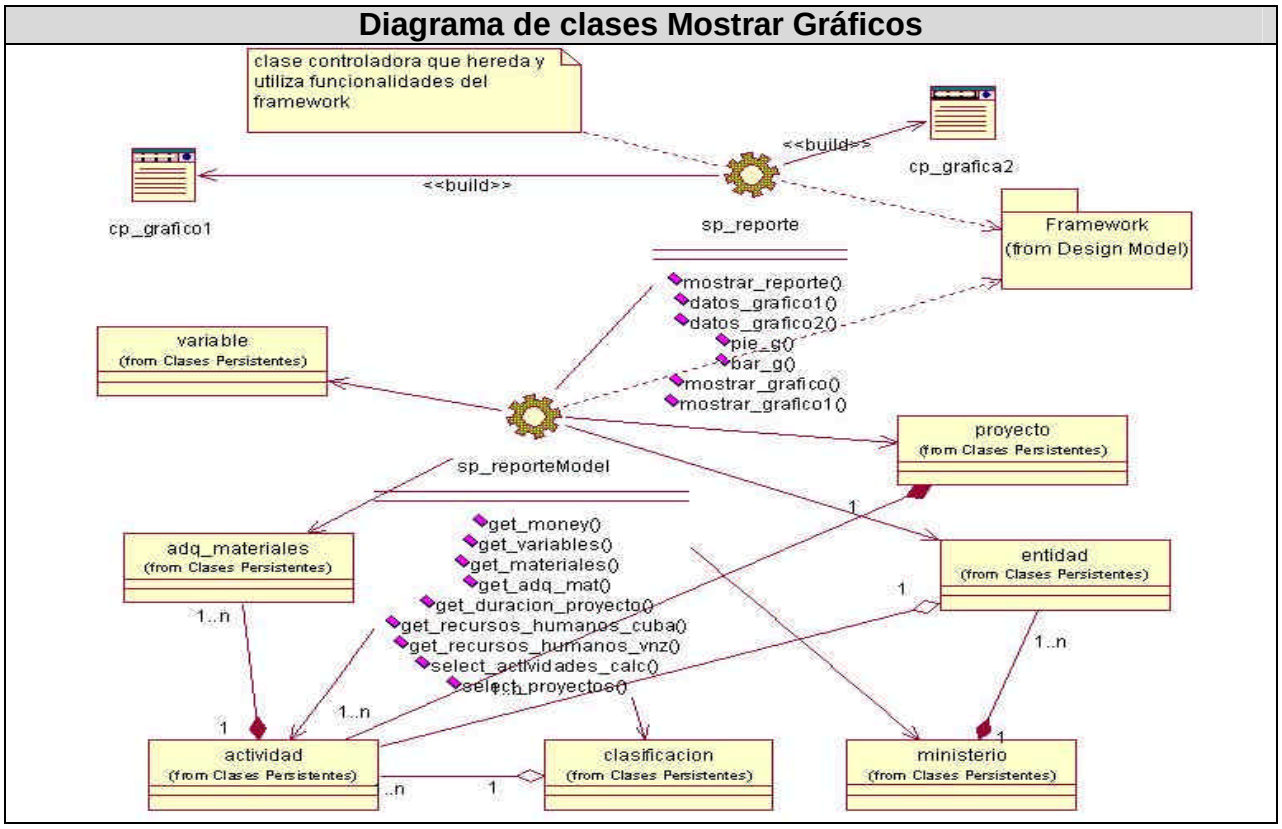


Figura 4.6 Diagrama de clases Mostrar Graficos.

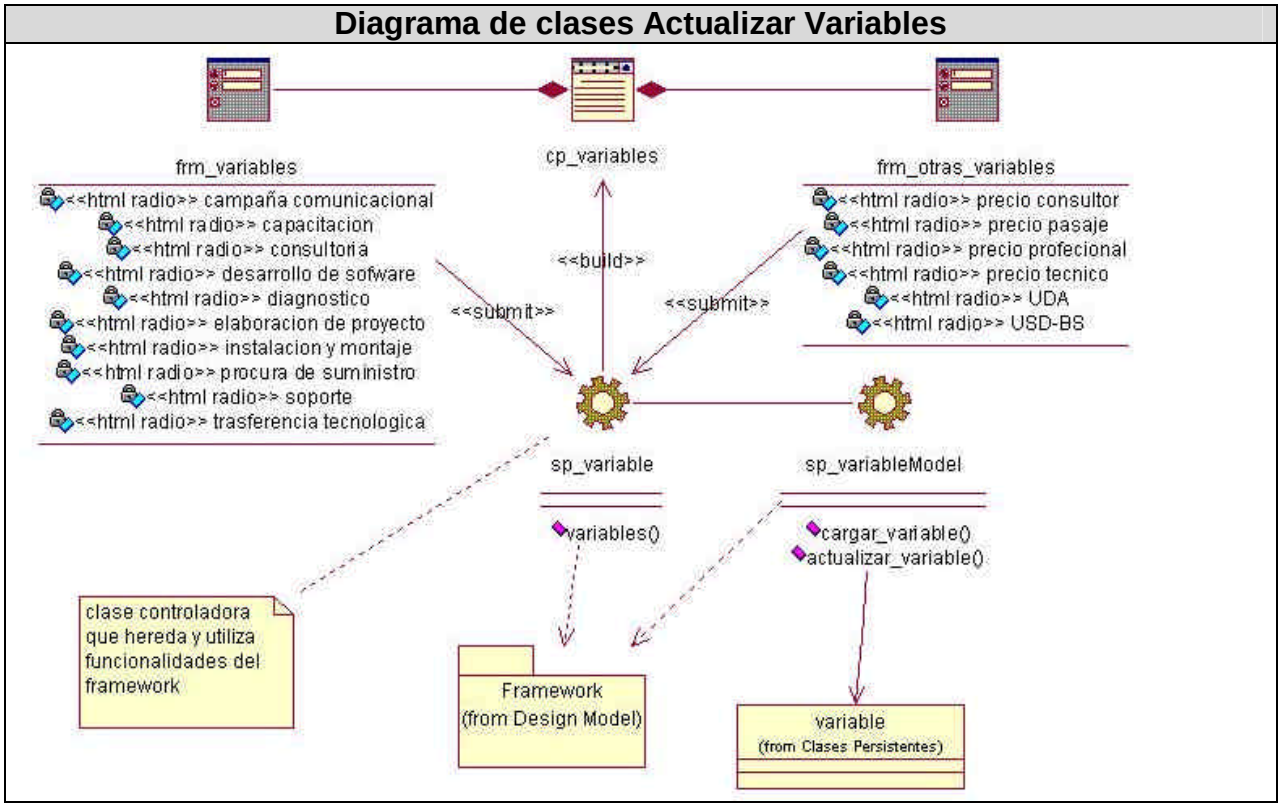


Figura 4.6 Diagrama de clases Actualizar Variables.

4.3.1 Paquete "Gestionar proyectos".

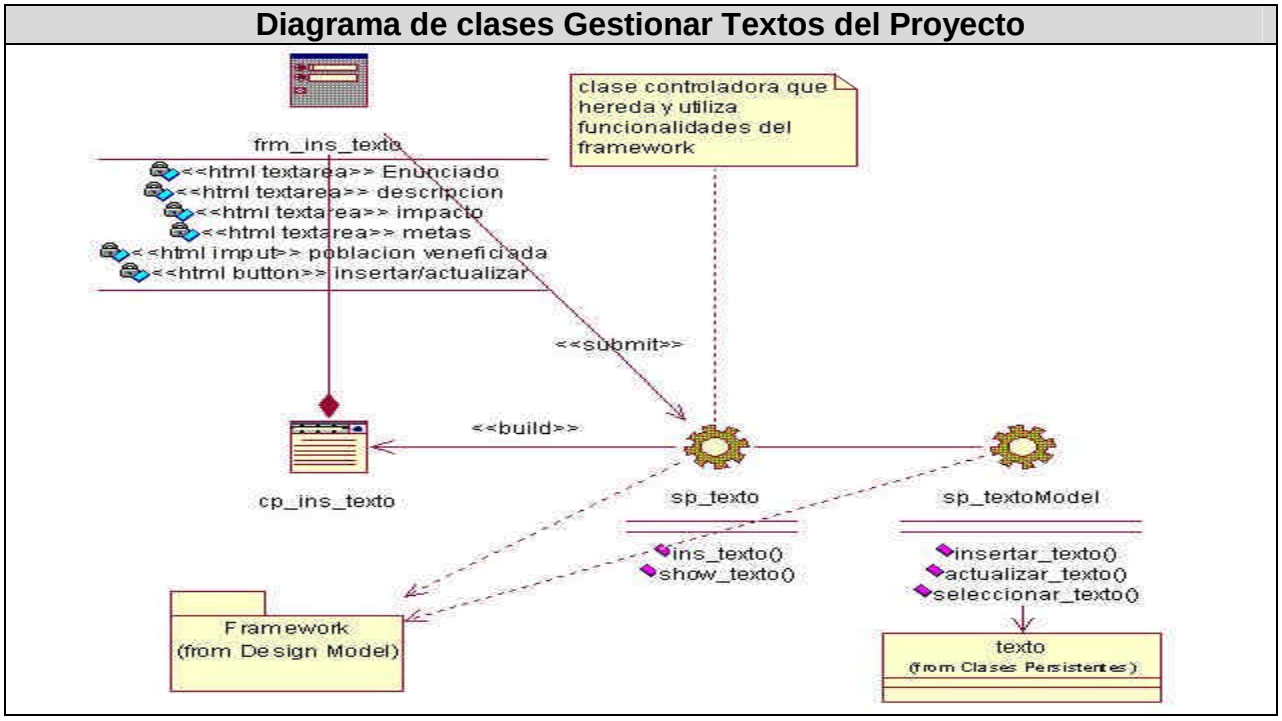


Figura 4.7 Diagrama de clases Gestionar Textos del Proyecto.

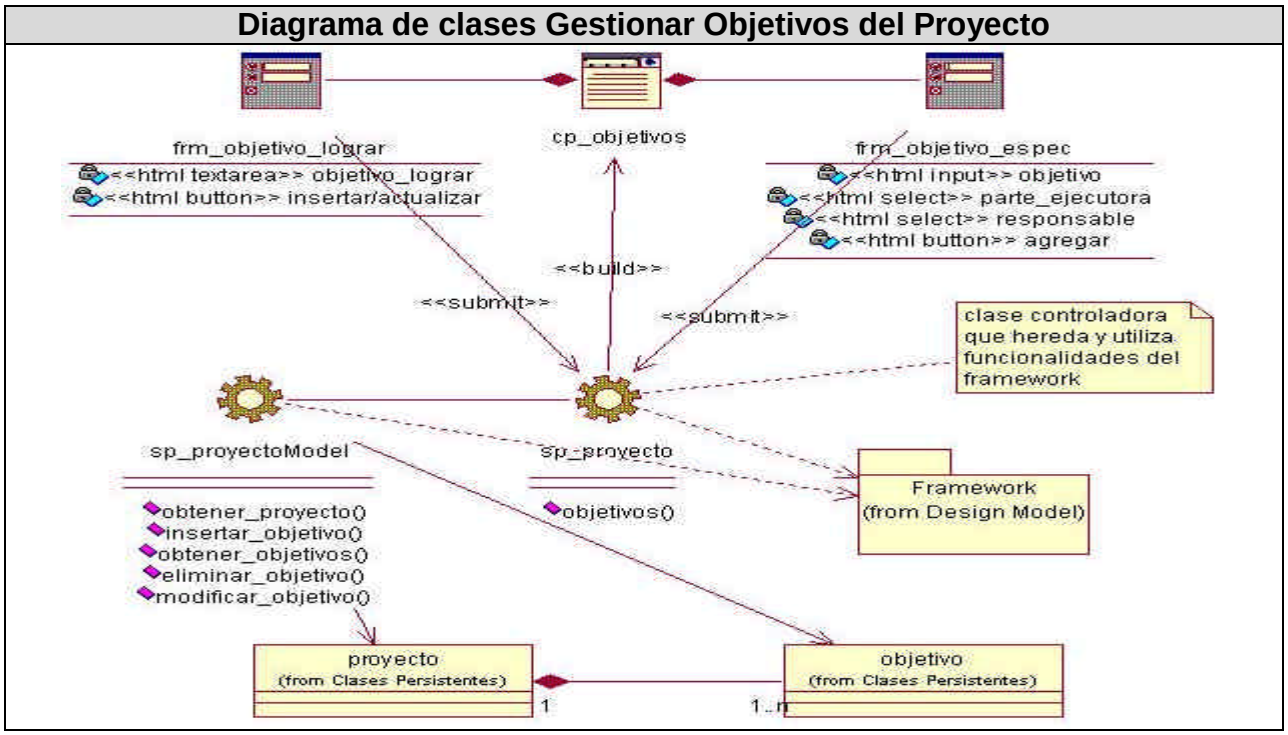


Figura 4.8 Diagrama de clases Gestionar Objetivos del Proyecto.

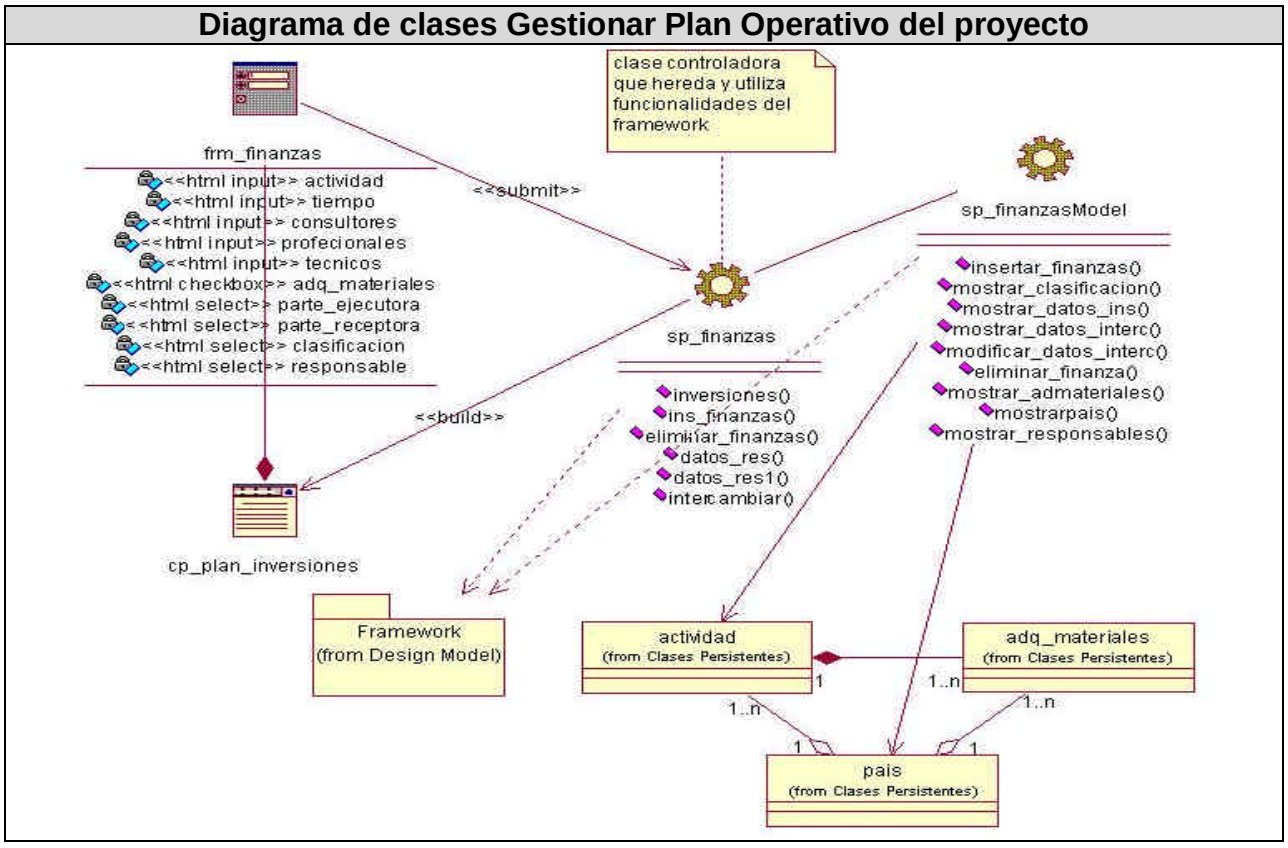


Figura 4.9 Diagrama de clases Gestionar Plan Operativo del Proyecto.

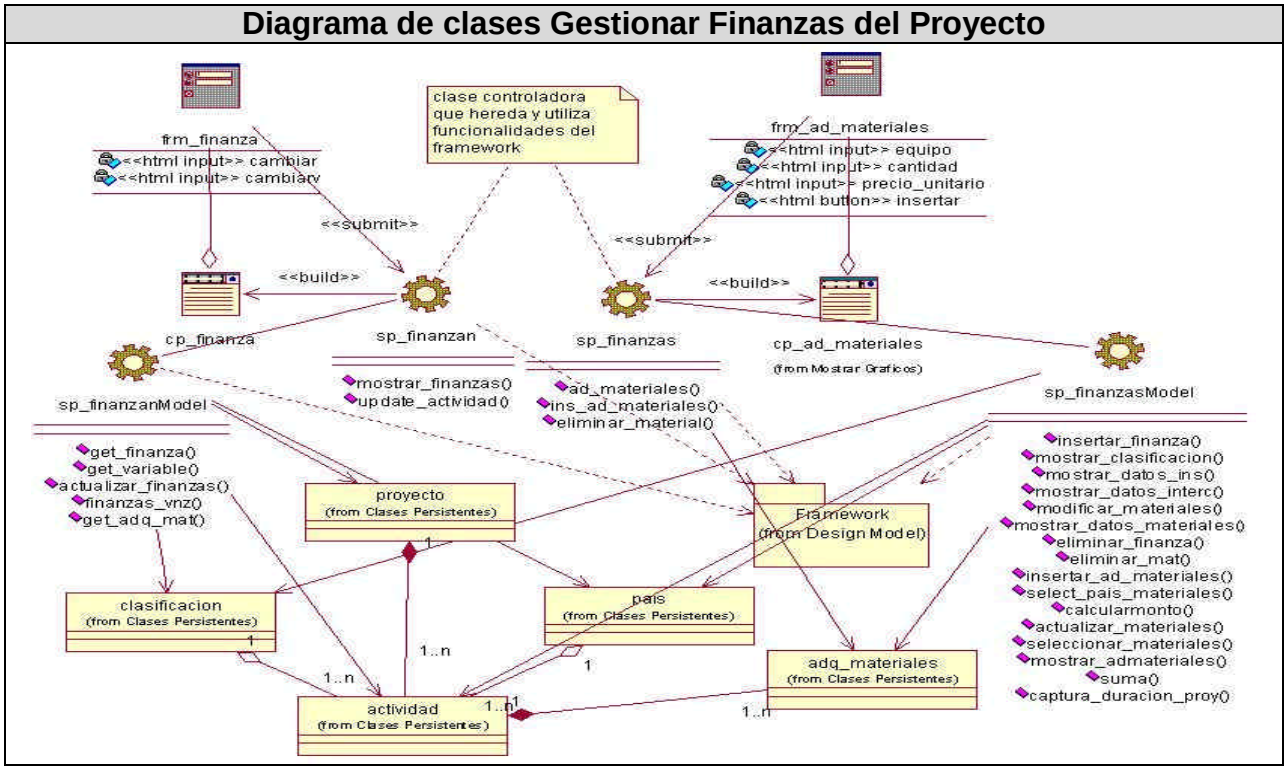


Figura 4.10 Diagrama de clases Gestionar Finanzas del Proyecto.

4.3.1 Paquete "Usuarios".

Diagrama de clases Autenticarse

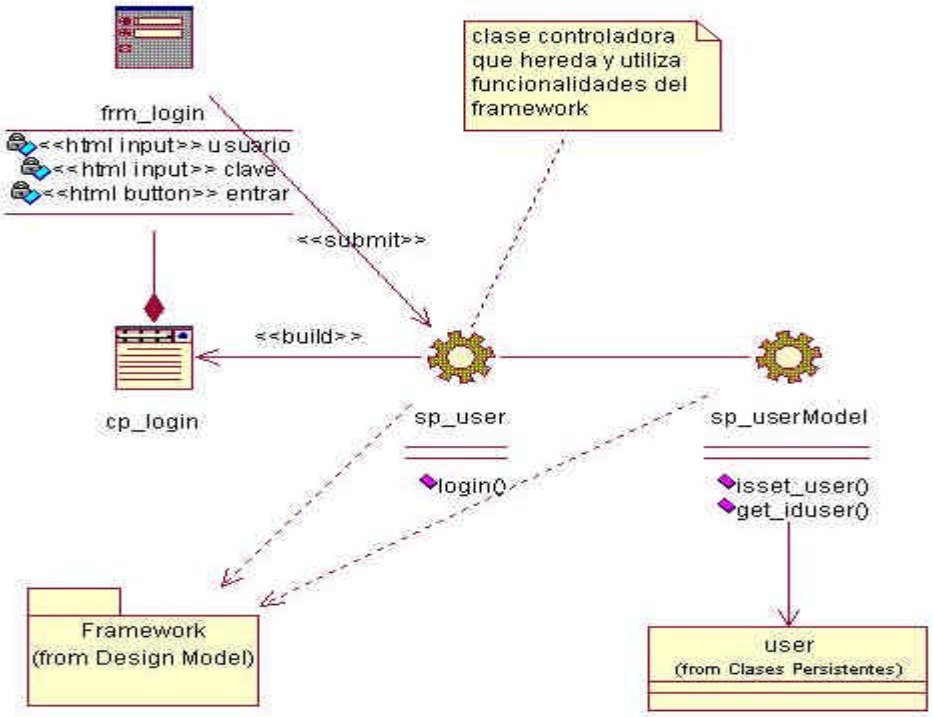


Figura 4.11 Diagrama de clases Autenticarse.

Diagrama de clases Cambiar Contraseña

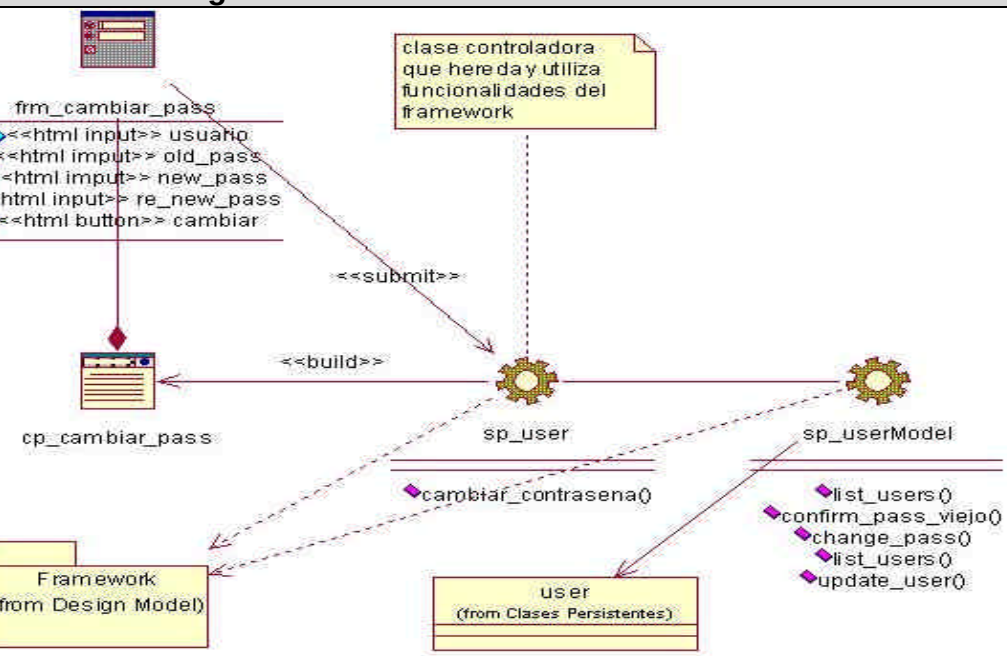


Figura 4.12 Diagrama de clases Cambiar Contraseña.

Diagrama de clases Accionar sobre los Proyectos

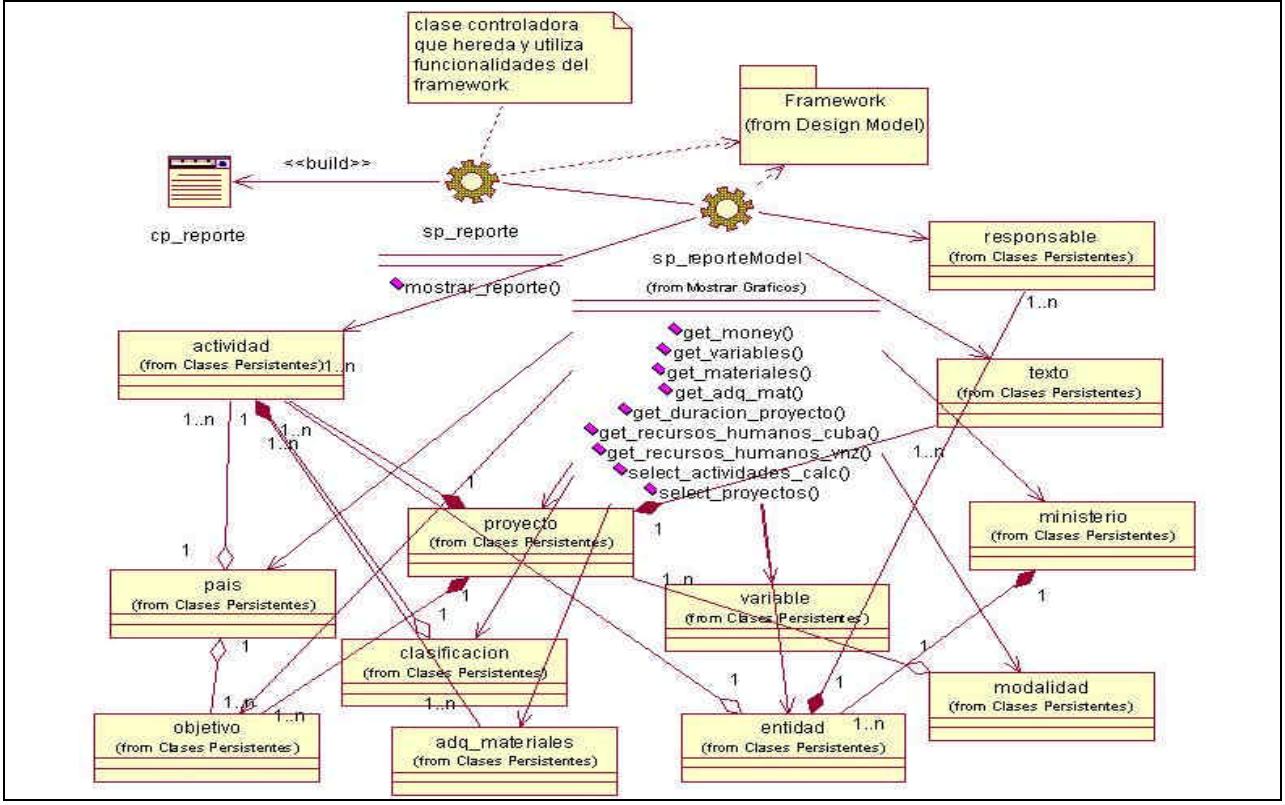


Figura 4.13 Diagrama de clases Accionar sobre los Proyectos.

4.4 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.

4.4.1 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”.

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 tuvo en cuenta los siguientes aspectos:

1). Reducir la carga a la memoria.
2). Atajos a Usuarios expertos.
3). Obtener información de retroalimentación.
4). Diseño de diálogos que conducen a una conclusión.
5). Previsión de errores y manejo de errores simples.
6). Lograr deshacer acciones fácilmente.
7). Que el usuario sintiera la sensación de control.

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

1) Proporcionan un ambiente amigable.
2) Conducen a un aprendizaje más natural.
3) Establecen un “sentimiento” (sobre todo en la uniformidad del ambiente) al usuario que enriquece su experiencia en el uso de la aplicación.
4) Además de estos principios, se tuvieron en cuenta las siguientes características:
5) Utilizar una misma tipografía, forma y estilo en todas las páginas.
6) La facilidad del usuario de poder navegar desde cualquier punto a otro dentro de la aplicación.
7) Se tuvo presente siempre el ancho de banda y por ello se utilizaron formato de imágenes de compresión favorables.
8) La simplicidad y consistencia, favoreciendo la usabilidad de la aplicación.
9) Navegación simple en todas las páginas de la aplicación, de forma tal que siempre sea accesible por el usuario.
10) Estabilidad y uniformidad del diseño, para así poder ubicar al usuario dentro del mismo y hacerlo sentir parte de él.

Se utilizó 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.

Los formularios de entradas ocupan el centro superior y las entradas organizadas por importancia. Se incluye una breve explicación del objetivo del formulario, y alguna especificación con respecto a las entradas.

Se realizan múltiples operaciones en cada página, de forma que el usuario no tenga que moverse tanto dentro de la aplicación, para completar una operación.

4.4.2 Formato de salida de los reportes.

El resumen de resultados de la ficha, se han concebido en una ventana de la aplicación, utilizando letra legible y colores claros, de fondo, para no recargar la página y lograr calidad y nitidez en la impresión de la información. La ficha tiene un encabezado que le identifica y describe brevemente, luego se muestra la información obtenida de manera legible y organizada.

4.4.3 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.

4.4 Diseño de la base de datos.

La base de datos es el sistema utilizado para el almacenamiento de datos y acceso controlado a los datos almacenados. En este epígrafe se muestra el diseño de la base de datos del sistema propuesto a través del diagrama de clases persistentes y el esquema de la base de datos generados a partir de este, el modelo de datos.

4.4.1 Diagrama de clases persistentes.

Las clases persistentes son las clases que necesitan ser capaz de guardar su estado en un medio permanente, la necesidad de guardar su estado esta dado por al almacenamiento físico permanente de la información de la clase, para la copia de seguridad en caso del fracaso del sistema, o para el intercambio de información. A continuación se muestra el diagrama de clases persistentes.

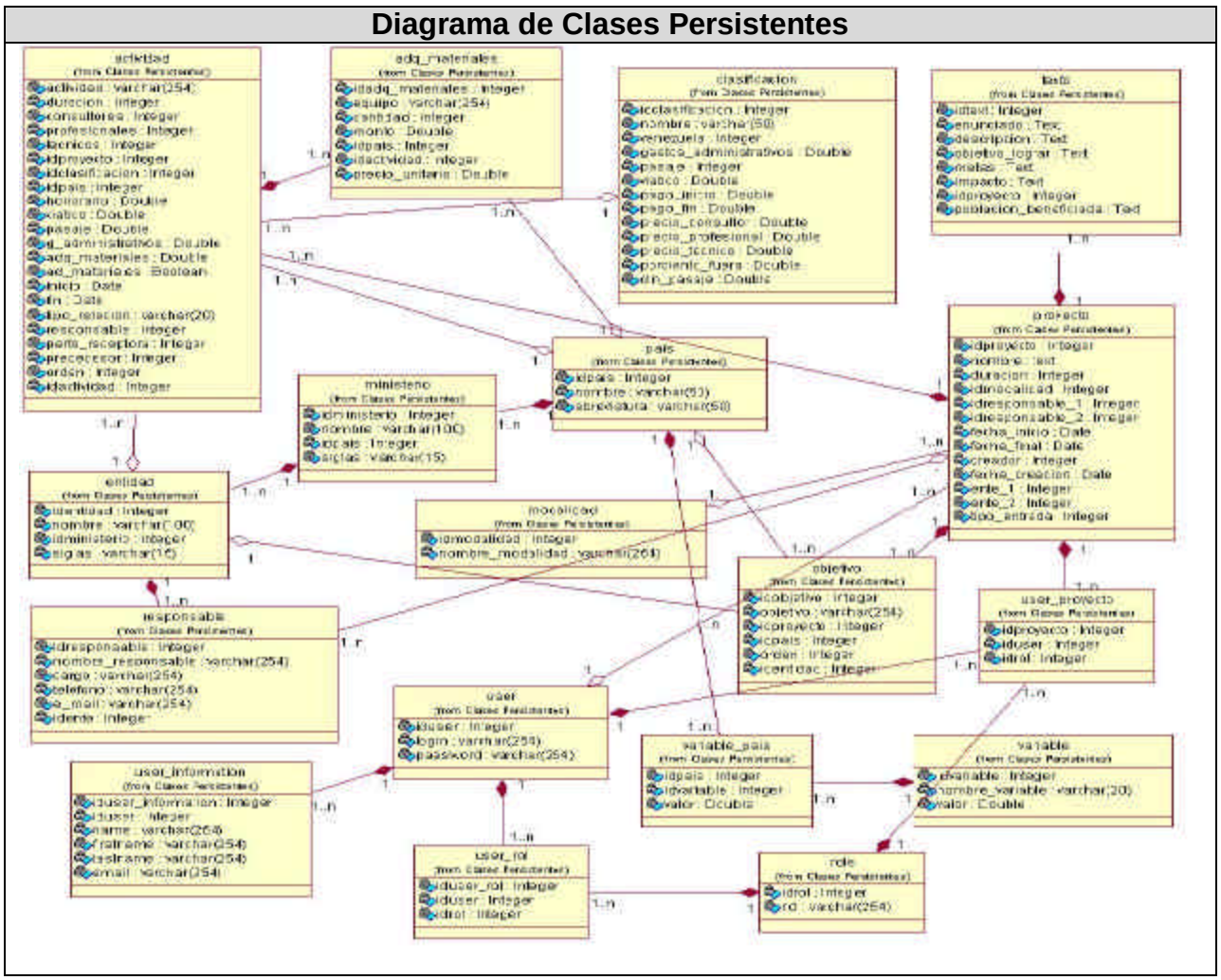
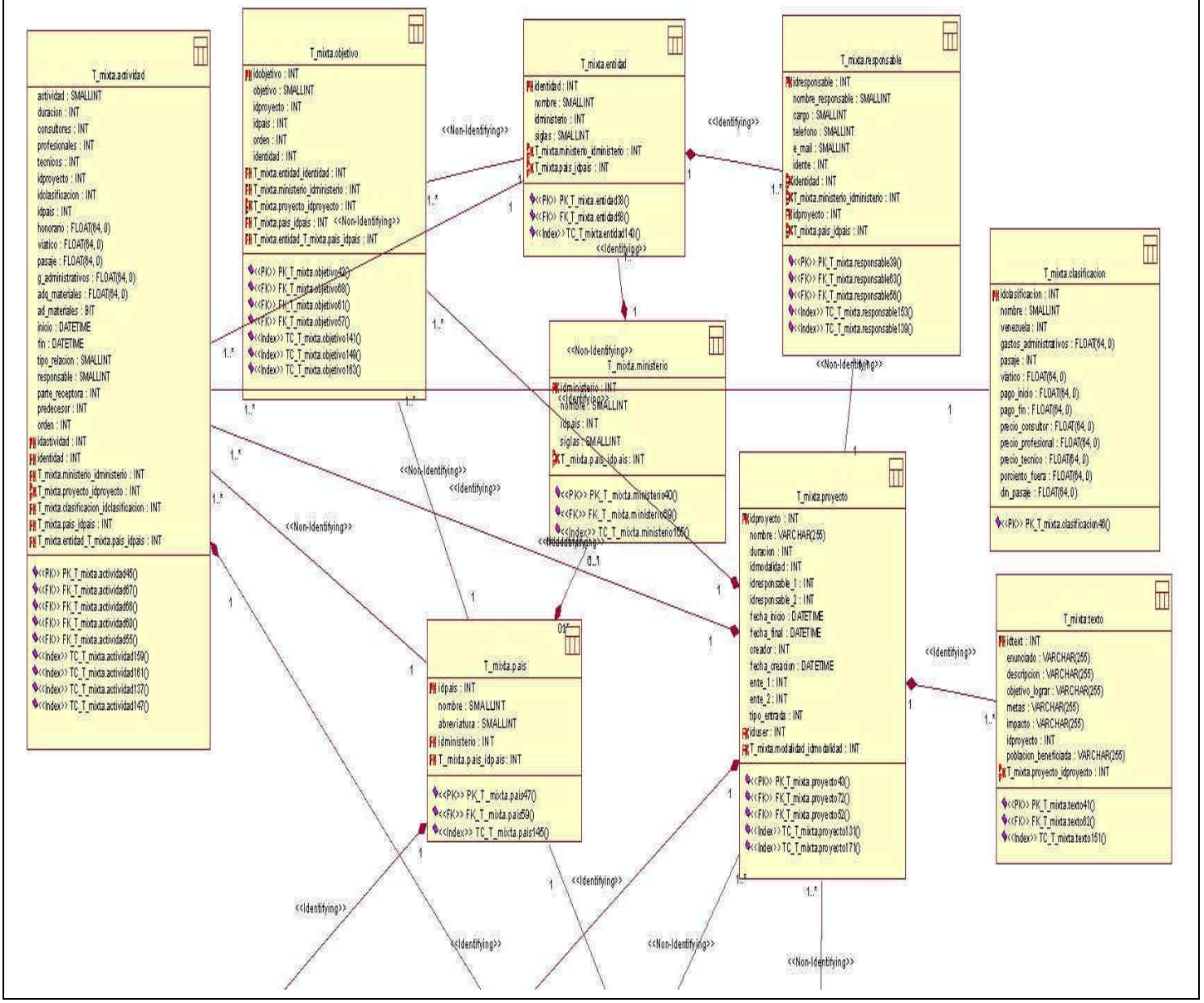


Figura 4.14 Diagrama de clases Persistentes.

4.4.2 Modelo de datos.

El modelo de los datos describe la representación lógica y física de datos persistentes en el sistema. A continuación se muestra el modelo de datos.

Diagrama del Modelo de Datos



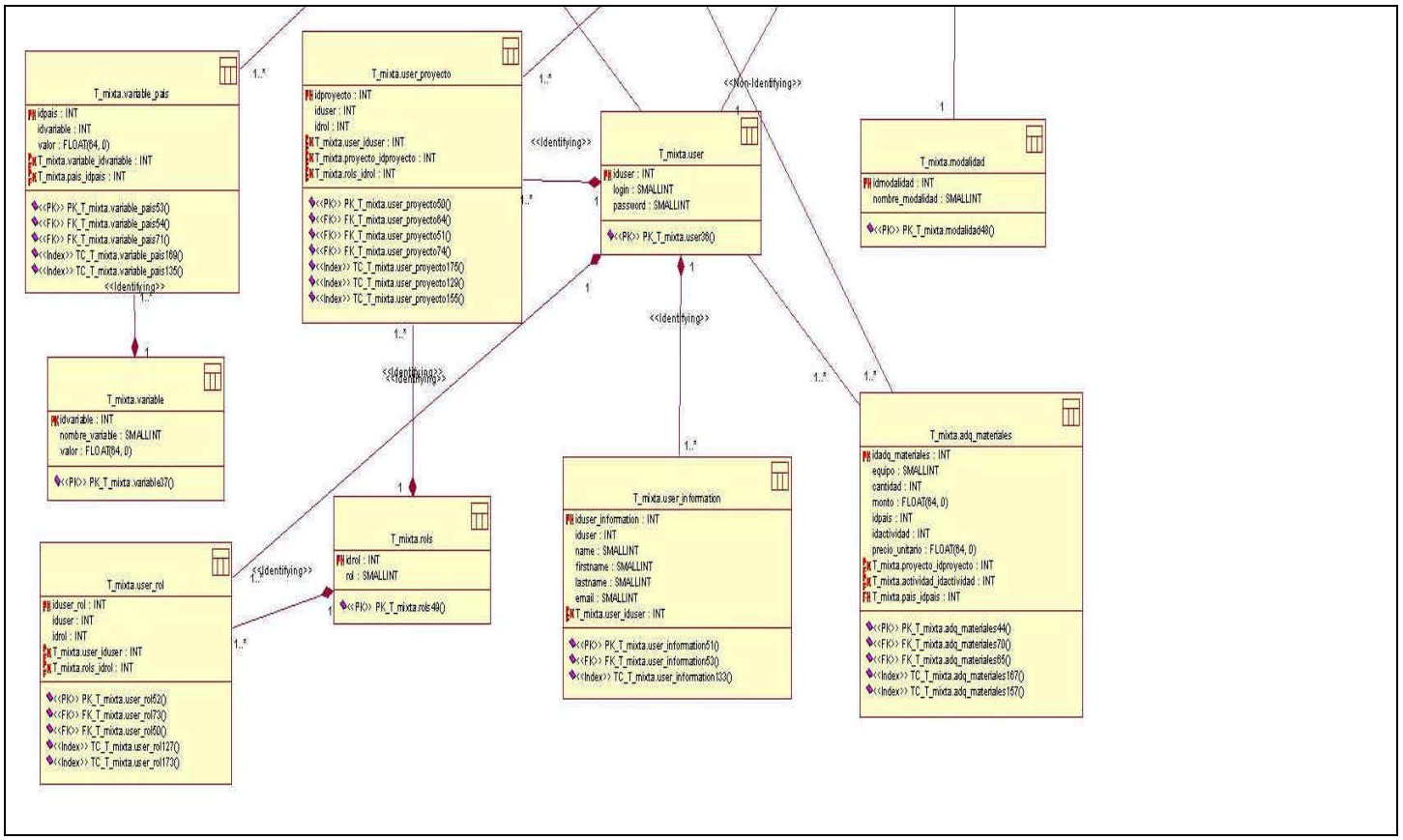


Figura 4.15 Diagrama del Modelo de Datos.

4.6 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.

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.

El diagrama de despliegue muestra la topología del hardware sobre el que se ejecuta el sistema. Ver Tabla 4.2 Diagrama de despliegue.

Diagrama de Despliegue

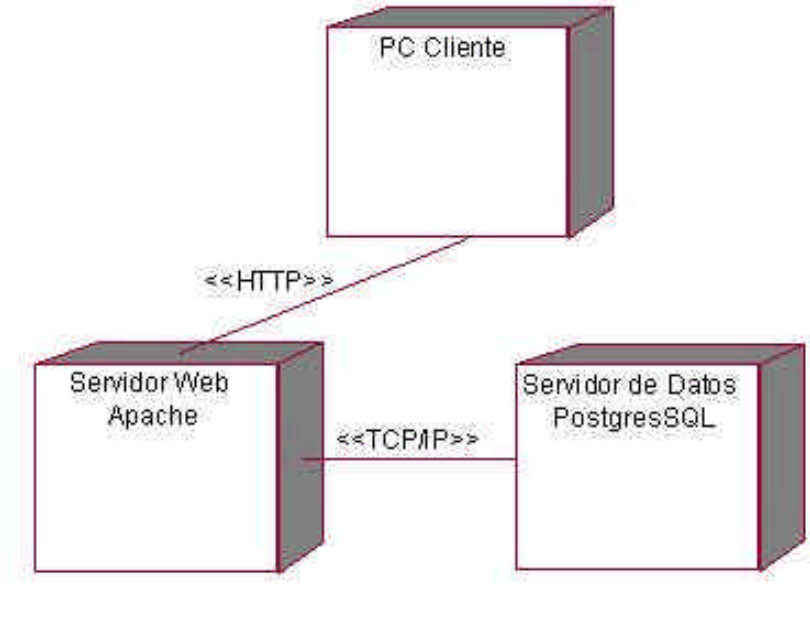


Figura 4.16 Diagrama de Despliegue.

4.7 Conclusiones.

En el presente capítulo se desarrollaron los diagramas de clases de la aplicación y el diseño de la base de datos del sistema. Se describieron, además, los principios de diseño seguidos, específicamente, los temas de estándares de la interfaz, concepción del tratamiento de errores y principios de codificación.

5 CAPÍTULO
Estudio de Factibilidad

5.1 Introducción.

Para la realización de un proyecto es de suma importancia el análisis del costo y los beneficios que reportará. Como resultado de este análisis se obtiene el tiempo de desarrollo en meses, costo y la cantidad de personas que se necesitan para desarrollar el proyecto.

En este capítulo se describe la estimación de costos del sistema propuesto y sus beneficios.

5.2 Planificación basada en casos de uso.

Paso 1. Cálculo de los Puntos de casos de uso Desajustados.

$$UUCP = UAW + UUCW$$

Donde:

UUCP: Puntos de casos de uso sin ajustar.

UAW: Factor de peso de los actores sin ajustar.

UUCW: Factor de peso de los casos de uso sin ajustar.

Tipo de actor	Descripción	Factor de peso	Actores	Total
Simple	Sistema con sistema a través de interfaz de programación.	1	0	0
Medio	Sistema con sistema mediante protocolo de interfaz basada en texto.	2	0	0
Complejo	Persona que interactúa con el sistema mediante interfaz gráfica.	3	3	9
	Total		3	9

Tabla 5.1 Factor de peso de los actores sin ajustar.

$$UAW = \sum cant\ actores * peso$$

$$UAW = 9$$

Tipo de CU	Descripción	Peso	Cantidad de CU	Total
Simple	El caso de uso tiene de 1 a 3 transacciones.	5	6	30
Medio	El caso de uso tiene de 4 a 7 transacciones.	10	2	20
Complejo	El caso de uso tiene más de 8 transacciones.	15	4	60
Total			12	110

Tabla 5.2 Factor de peso de los casos de uso sin ajustar.

$$UUCW = \sum cant\,CU * Pesc$$

$$UUCW = 110$$

$$UUCP = 9 + 110$$

$$UUCP = 119$$

Paso 2. Cálculo de los Puntos de casos de uso ajustados.

$$UCP = UUCP * TCF * EF$$

Donde:

UCP: Puntos de casos de uso ajustados.

UUCP: Puntos de casos de uso sin ajustar.

TCF: Factor de complejidad técnica.

EF: Factor de ambiente.

El factor de complejidad técnica (TCF) se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada factor se cuantifica en un valor desde 0 (aporte irrelevante) hasta 5 (aporte muy relevante).

Factor	Descripción	Peso	Valor asignado	Total
T1	Sistema distribuido	2	0	0
T2	Tiempo de respuesta	1	4	4
T3	Eficiencia del usuario final	1	1	1
T4	Funcionamiento Interno complejo	1	3	3
T5	El código debe ser reutilizable	1	3	3
T6	Facilidad de instalación	0,5	4	2
T7	Facilidad de uso	0,5	5	2,5
T8	Portabilidad	2	5	10
T9	Facilidad de cambio	1	3	3
T10	Concurrencia	1	4	4
T11	Incluye objetivos especiales de seguridad	1	3	3
T12	Provee acceso directo a terceras partes	1	4	4
T13	Se requieren facilidades especiales de entrenamiento de usuarios	1	1	1
	Total			40.5

Tabla 5.3 Factor de complejidad técnica.

$$TCF = 0.6 + 0.01 * \sum (peso * valor\ asignado)$$

TCF = 0.6 + 0.01 * 40.5

TCF = 0.6 + 0.46

$TCF = 1.005$

El factor de ambiente (EF) está relacionado con las habilidades y entrenamiento del grupo de desarrollo que realiza el sistema. Cada factor se cuantifica con un valor desde 0 (aporte irrelevante) hasta 5 (aporte muy relevante).

Factor	Descripción	Peso	Valor asignado	Total
E1	Familiaridad con el modelo de proyecto utilizado	1,5	4	6
E2	Experiencia en la aplicación	0,5	4	2
E3	Experiencia en la orientación a objetivos.	1	5	5
E4	Capacidad del analista líder.	0,5	4	2
E5	Motivación.	1	5	5
E6	Estabilidad de requerimientos	2	5	10
E7	Personal Part–Time	-1	0	0
E8	Dificultad del lenguaje de programación	-1	2	-2
Total				28

Tabla 5.4 Factor de ambiente.

$$EF = 1.4 - 0.03 * \sum (peso * valor\ asignado)$$

EF = 1.4 – 0,03 * 28

EF = 1.4 – 0.84

$EF = 0.56$

UCP = UUCP * TCF * EF

UCP = 119 *1.06 * 0.56

$UCP = 66.9732$

Paso 3. Estimación de esfuerzo a través de los puntos de casos de uso.

$$E = UCP * CF$$

Donde:

E: Esfuerzo estimado en horas hombres.

UCP: Punto de casos de usos ajustados.

CF: Factor de conversión.

Para obtener el factor de conversión (CF) se cuentan cuantos valores de los que afectan el factor ambiente (E1...E6) están por debajo de la media (3), y los que están por arriba de la media para los restantes (E7, E8). Si el total es 2 o menos se utiliza el factor de conversión 20 Horas-Hombre / Punto de Casos de uso. Si el total es 3 o 4 se utiliza el factor de conversión 28 Horas-Hombre / Punto de Casos de uso. Si el total es mayor o igual que 5 se recomienda efectuar cambios en el proyecto ya que se considera que el riesgo de fracaso del mismo es demasiado alto.

En este caso se puede decir que:

CF = 20 Horas-Hombre / Punto de Casos de uso.

$$E = 66.9732 * 20$$

$$E = 1339.464 \text{ Horas-Hombre}$$

Paso 4. Calcular esfuerzo de todo el proyecto.

Actividad	Porcentaje %	Horas-Hombres
Análisis	10	334.866
Diseño	20	669.732
Implementación	40	1339.464
Pruebas	15	502.299
Sobrecarga (otras actividades)	15	502.299
Total	100	3348.66

Tabla 5.5 Esfuerzo del proyecto.

Si **E_T = 3348.66 horas-hombre** y se estima que cada mes tiene como promedio 192 horas laborables, eso daría un **E_T = 17.4409375 mes-hombre.**
Esto quiere decir que 1 persona puede realizar el problema analizado en 17 meses y medio aproximadamente.

-Costo del Proyecto.

Se asume como salario promedio mensual $50.00

CH: Cantidad de hombres.

Tiempo: Tiempo total del proyecto.

CH = 3 hombres

CHM = 3 * Salario Promedio

CHM = 150.00 $/mes

Costo = CHM * E_T / CH

Costo = 150.00 *17.4409375 / 3

Costo = $ 872.046875 ˜ $872.04

Tiempo = E_T / CH

Tiempo = 17.4409375 / 3

Tiempo = 5.813645 ˜ 5.81 meses

De los resultados obtenidos se interpreta que con 3 hombres trabajando en el proyecto el mismo se desarrolla en 5,81 meses y su costo total se estima que sea $872.04.

5.3 Beneficios tangibles e intangibles.

El Sistema para la informatización del Convenio Cuba-Venezuela. Módulo ficha mixta no es un producto con fines comerciales, su principal objetivo es resolver los problemas que existen durante la gestión de la ficha de cada uno de los proyectos que se firman en el marco del convenio.

El beneficio fundamental del sistema es contar con una aplicación Web flexible, dinámica y de interfaz agradable que le permita registrar, actualizar y conocer de una forma más precisa y en el menor tiempo posible la información referente a los proyectos y generar de forma automática la ficha de los mismos.

Por tanto, los beneficios inmediatos son generalmente intangibles:

- Disminución del tiempo y esfuerzo que se invierte en esta tarea que se realiza, hasta ahora, de forma manual.

- Disminución de la acumulación de materiales impresos relacionados con los procesos de gestión de la información referente a los proyectos.

- Disminución de los gastos pues resulta menos costoso crear y procesar información digital que copias duras.

- Fácil detección de errores.

- Fácil y rápido acceso y publicación de la información actualizada.

- Fácil procesamiento de la información y obtención, dinámica, de reportes de la situación de ejecución del presupuesto de los proyectos en cualquier momento.

5.4 Análisis de costos y beneficios.

Desarrollar un producto informático cuesta. Justificar entonces su desarrollo depende de los beneficios que reportarían su implantación y utilización. Los beneficios pueden ser económicos y de orden social, estos últimos son de tanta importancia como los primeros. El sistema que se propone está dirigido a la Universidad de las Ciencias Informáticas, para la gestión de los proyectos que exporta, por tanto su mayor beneficio es de orden económico.

Una vez implantado el sistema éste contribuirá a aumentar la eficiencia en la gestión de la información y la generación de la ficha referente a los proyectos que se desarrollan en el marco de la convención Mixta Cuba-Venezuela, al disminuir el tiempo necesario a emplear en el registro, consulta y actualización de la diversa información; y generar informes de resultados de los procesos que se desarrollan con mayor rapidez y certeza.

La tecnología utilizada para el desarrollo del sistema es totalmente libre, por tanto no hay que incurrir en gastos en el pago de licencias de uso. El sistema es portable por lo que un cambio de plataforma para la implantación del mismo es viable y factible, y no

hay que incurrir en muchos cambios; debido a la estructuración en capas de los procesos del negocio que se diseñaron.

Analizando el costo del proyecto, los numerosos beneficios que reporta, detallados con anterioridad, se puede concluir que su implementación es realmente factible.

5.5 Conclusiones.

En este capítulo se describió el estudio de factibilidad realizado correspondiente al sistema propuesto, teniendo en cuenta el costo estimado y los beneficios que reportará al ser implantado.

La herramienta propuesta reportará beneficios significativos e importantes para la gestión de los proyectos que se desarrollan en la UCI, al contribuir a mejorar los procesos que se realizan aquí en función de controlar la información de los mismos, lo que indica que es factible implementar la herramienta propuesta.

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.

Se alcanzó, satisfactoriamente, el objetivo propuesto: desarrollar una solución robusta, flexible y única de software que dé soporte a los procesos de gestión de al información de los proyectos que se desarrollan en el marco de la Convención Mixta Cuba-Venezuela; reafirmando así la utilidad y validez de emplear las tecnologías informáticas para apoyar la labores que se desarrollan en cualquier tipo de esfera. Se obtuvieron además los siguientes resultados:

1. Se ha demostrado la eficacia de los lenguajes y tecnologías utilizadas para el desarrollo del sistema.

2. Se realizó una base de datos, donde se almacena toda la información necesaria de los proyectos, para de esta forma garantizar la veracidad y centralización de la misma.

3. Se realizó el análisis, diseño e implementación del sistema.

4. La solución propuesta ha sido acertada, los requerimientos soportan al sistema y los casos de uso satisfacen las necesidades funcionales.

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

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

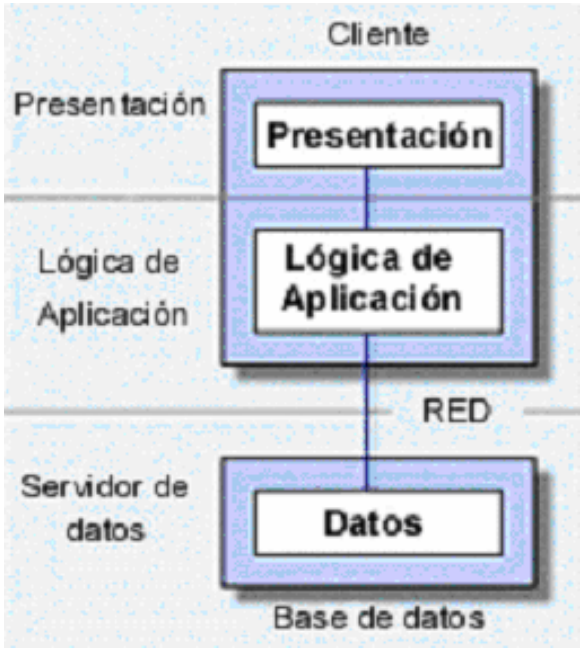
RECOMENDACIONES

Se recomienda:

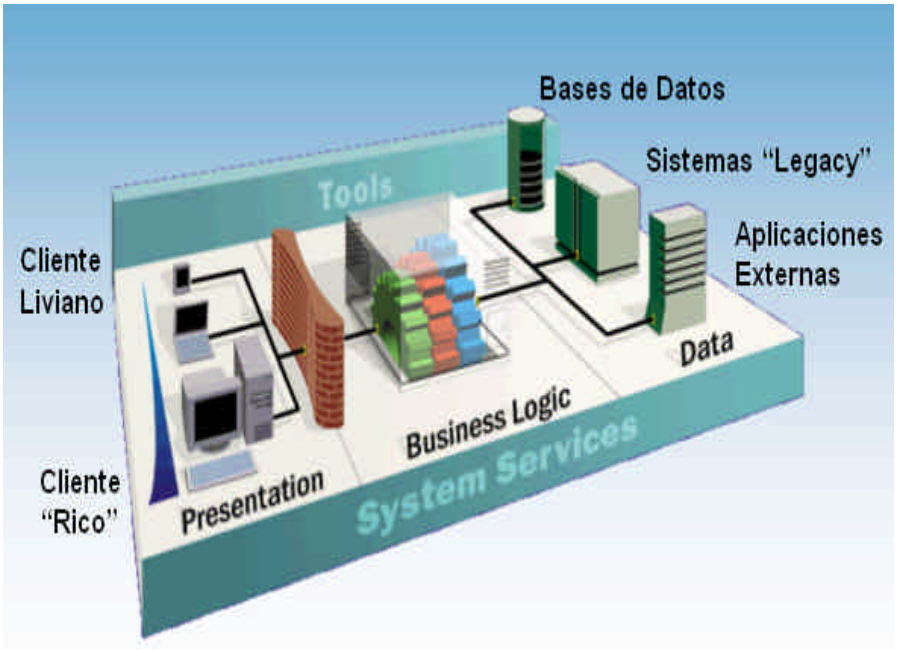
1. Poner a prueba el sistema durante un período de tiempo significativo, para comprobar su desempeño y que las funcionalidades del sistema se correspondan con la realidad de lo que se desea.
2. Continuar el estudio con el objetivo de añadir nuevas funcionalidades.
3. Proponer, tras corroborar un desempeño exitoso, la utilización y generalización de este sistema en la UCI para la gestión de los proyectos que la universidad desarrolla con Venezuela.

ANEXOS

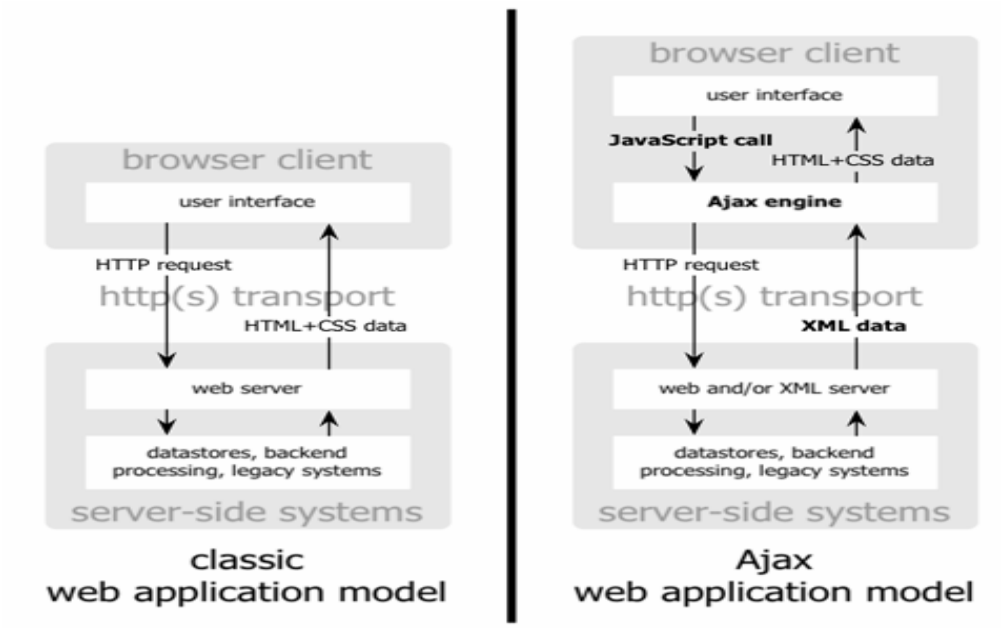
Anexo 1: Modelo Cliente – Servidor de dos capas.



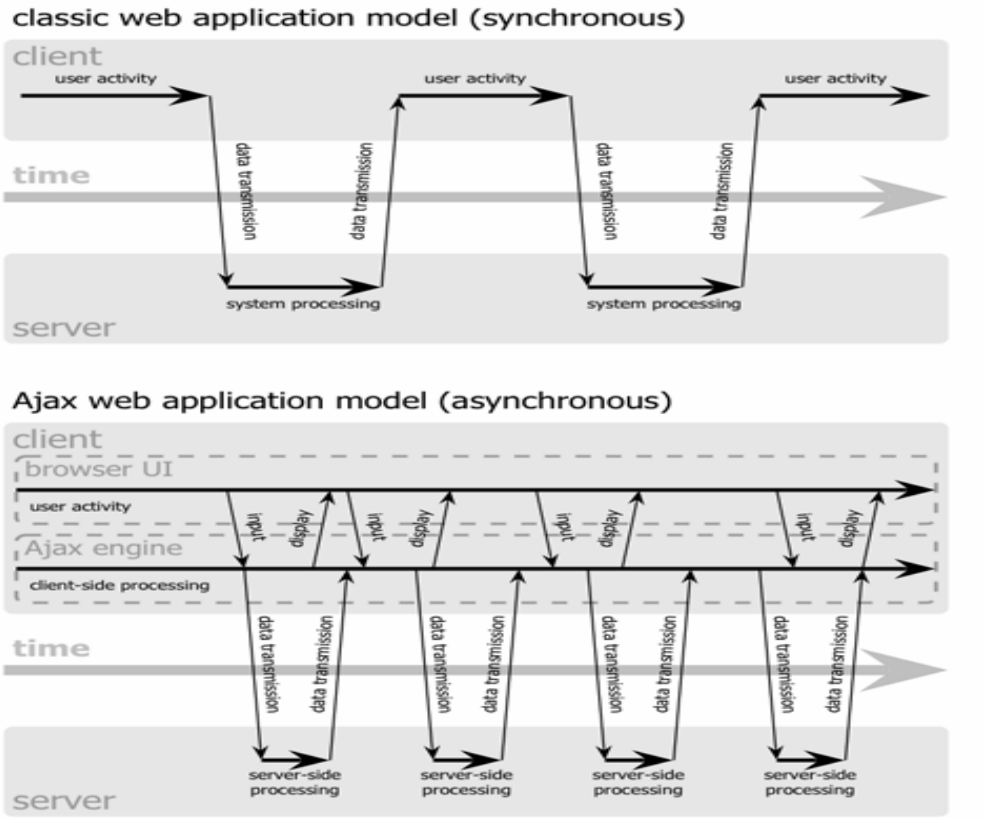
Anexo 2: Modelo Cliente – Servidor de tres capas.



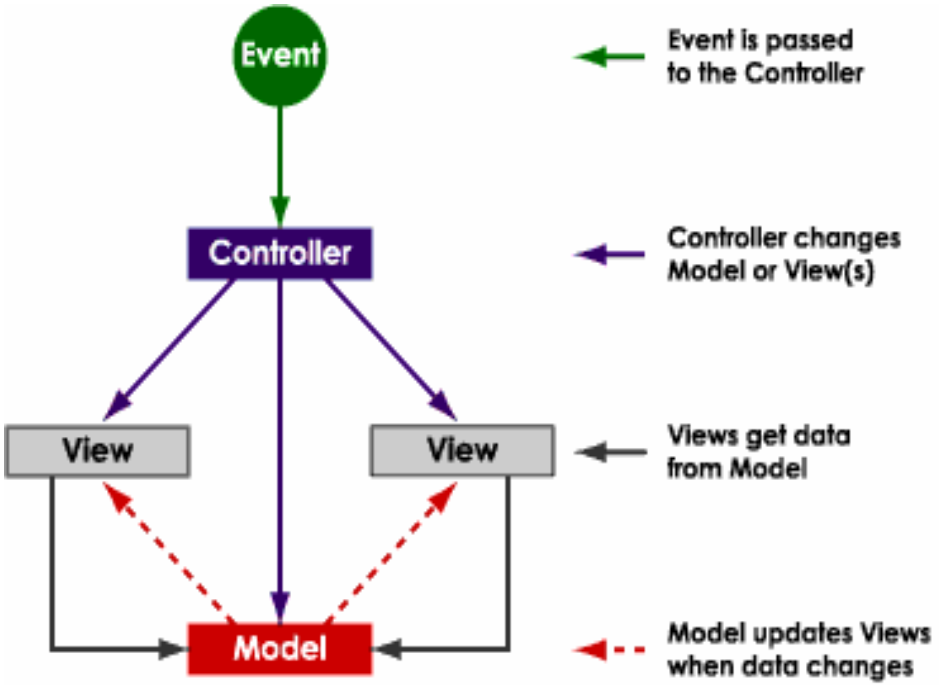
Anexo3: Se muestra el modelo tradicional para las aplicaciones Web (*a la izquierda*), comparado con el modelo de AJAX (*a la derecha*).



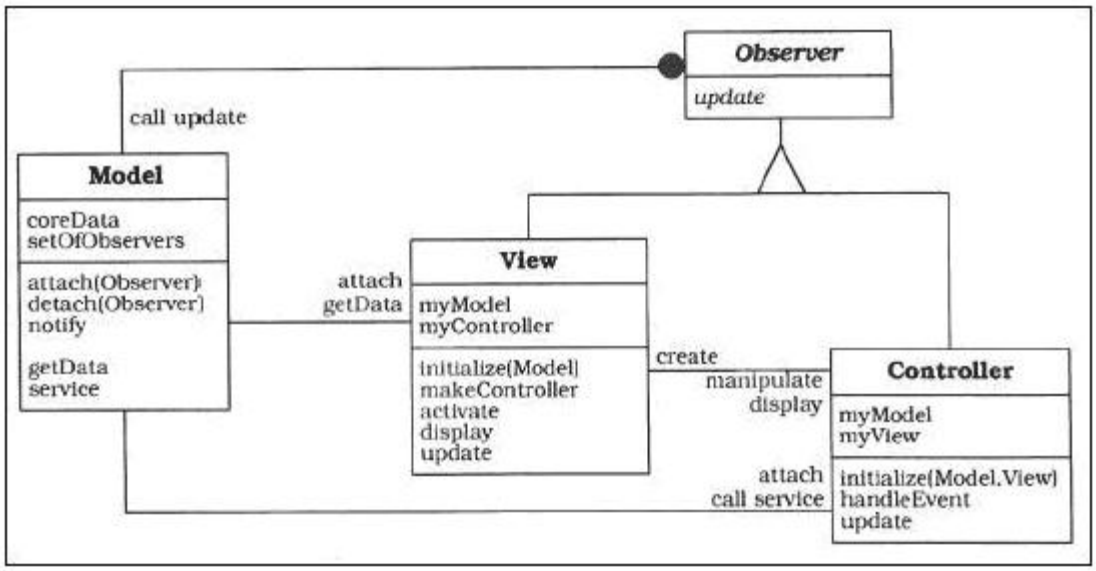
Anexo 4: Muestra el patrón de interacción sincrónica de una aplicación Web tradicional (*arriba*) comparada con el patrón asincrónico de una aplicación AJAX (*abajo*).



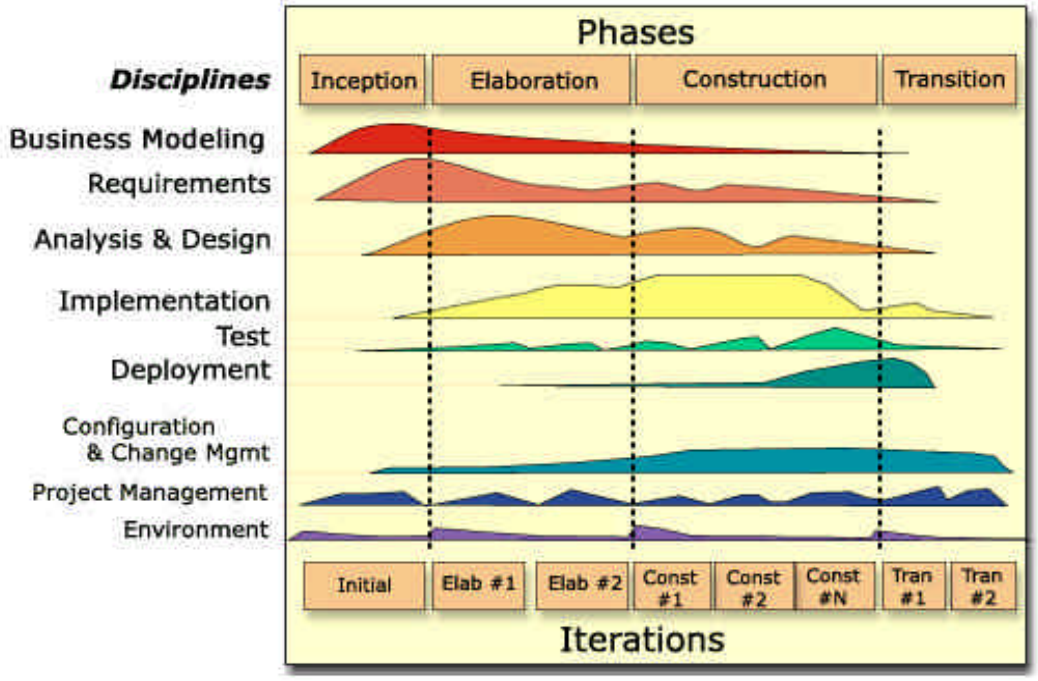
Anexo 5: Funcionamiento del patrón MVC.



Anexo 6: Estructura del patrón MVC.



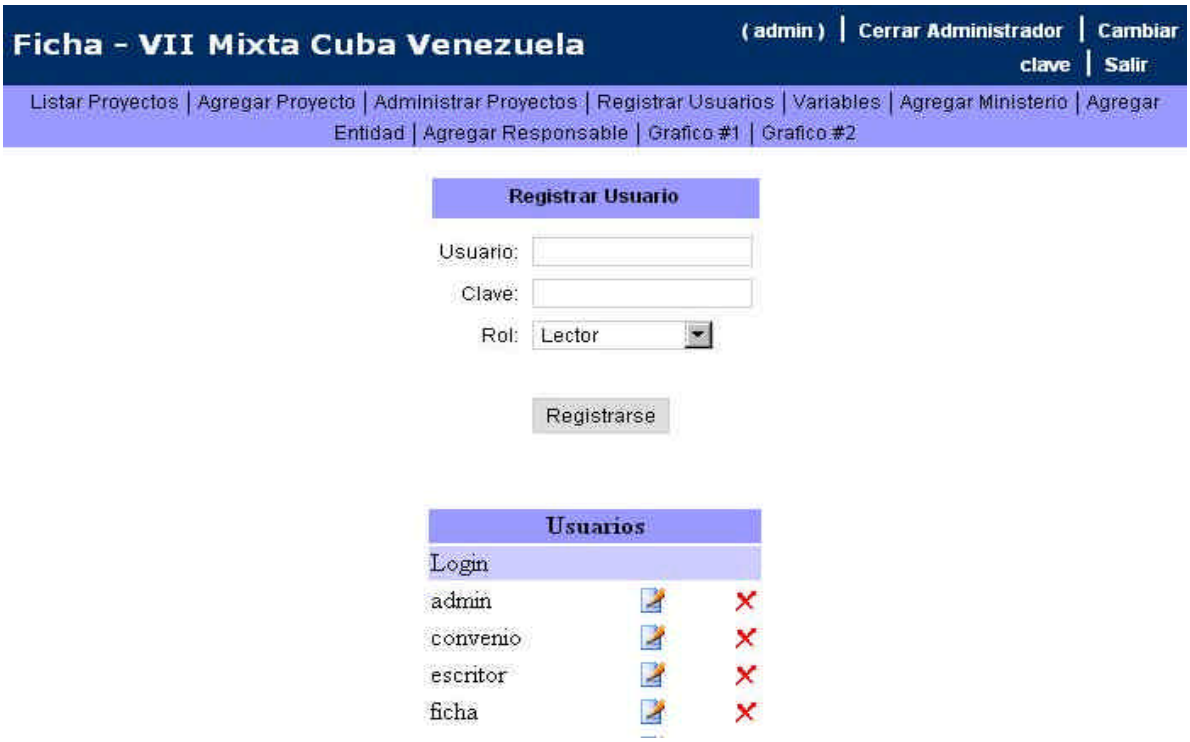
Anexo 7: Flujos de trabajo de RUP.



Flujos de trabajo:

1. **Modelamiento del negocio.**
2. **Requerimientos.**
3. **Análisis y diseño.**
4. **Implementación.**
5. **Prueba (Testeo).**
6. **Instalación.**
7. **Administración del proyecto.**
8. **Administración de configuración y cambios.**
9. **Ambiente.**

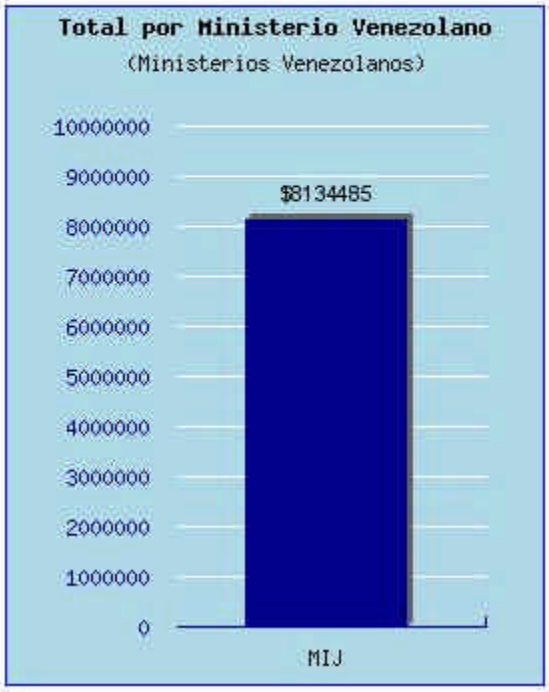
Anexo 8: Vista del caso de uso "Gestionar Usuarios".

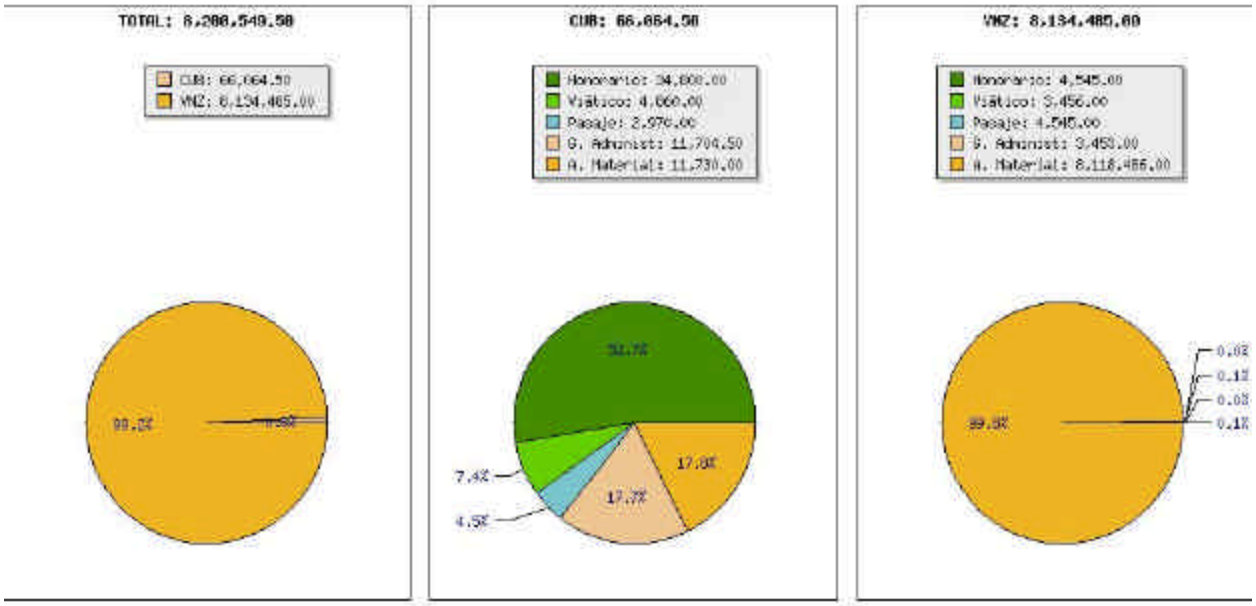


Anexo 9: Vista del caso de uso "Gestionar Proyectos".

Anexo 10: Vista del caso de uso "Gestionar Proyectos".

Ministerio	Total
MIJ	$ 8,134,485.00





Total a Transferir a Cuba: 66,064.50
Total a Ejecutar por Entes Venezolanos: 8,134,485.00

Anexo 11: Vista del caso de uso "Actualizar Variables".

Variables						
	Clasificación	Venezuela	Gastos Admin x mes($)	Pasaje	Viatico x mes($)	Editar
	Campana Comunicacional	30 %	2167.5	2	900	
	Capacitacion	100 %	2407.5	1	900	
	Consultoria	100 %	2167.5	3	530	
	Desarrollo Sofware	30 %	2167.5	3	900	
	Diagnostico	100 %	3039.6	1	530	
	Elaboracion de Proyecto	100 %	2107.5	1	900	
	Instalacion y Montaje	100 %	3039.6	1	530	
	Procura de Suministro	10 %	2167.5	1	900	
	Soporte	100 %	3039.6	2	530	
	Tranferencia de Tecnologia	100 %	2167.5	3	900	

(Otras variables)			
	Variable	Valor	Editar
	Precio Consultor	3000	
	Precio Pasaje	550	
	Precio Profesional	2000	
	Precio Tecnico	1200	
	UDA	12.5 %	
	USD-BS	2150	

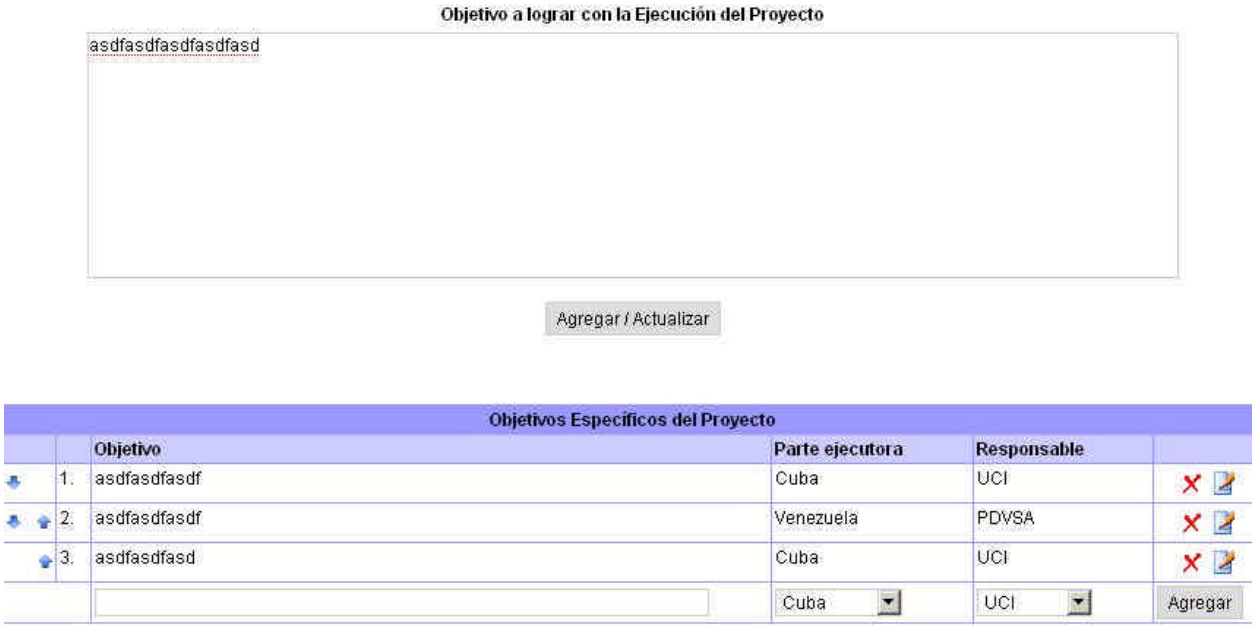
Anexo 12: Vista del caso de uso "Gestionar Responsables".


Ficha - VII Mixta Cuba Venezuela
(admin) | Cerrar Administrador | Cambiar clave | Salir
Listar Proyectos | Agregar Proyecto | Administrar Proyectos | Registrar Usuarios | Variables | Agregar Ministerio | Agregar Entidad | Agregar Responsable | Grafico #1 | Grafico #2
Entrada de Entidades
Pais
Cuba
Ministerio
Ministerio de la Informatica y las Comunicaciones
Entidades
Siglas
insertar


Ficha - VII Mixta Cuba Venezuela
(admin) | Cerrar Administrador | Cambiar clave | Salir
Listar Proyectos | Agregar Proyecto | Administrar Proyectos | Registrar Usuarios | Variables | Agregar Ministerio | Agregar Entidad | Agregar Responsable | Grafico #1 | Grafico #2
Entrada de ministerios
Pais
---Seleccione---
Ministerio
Siglas
insertar

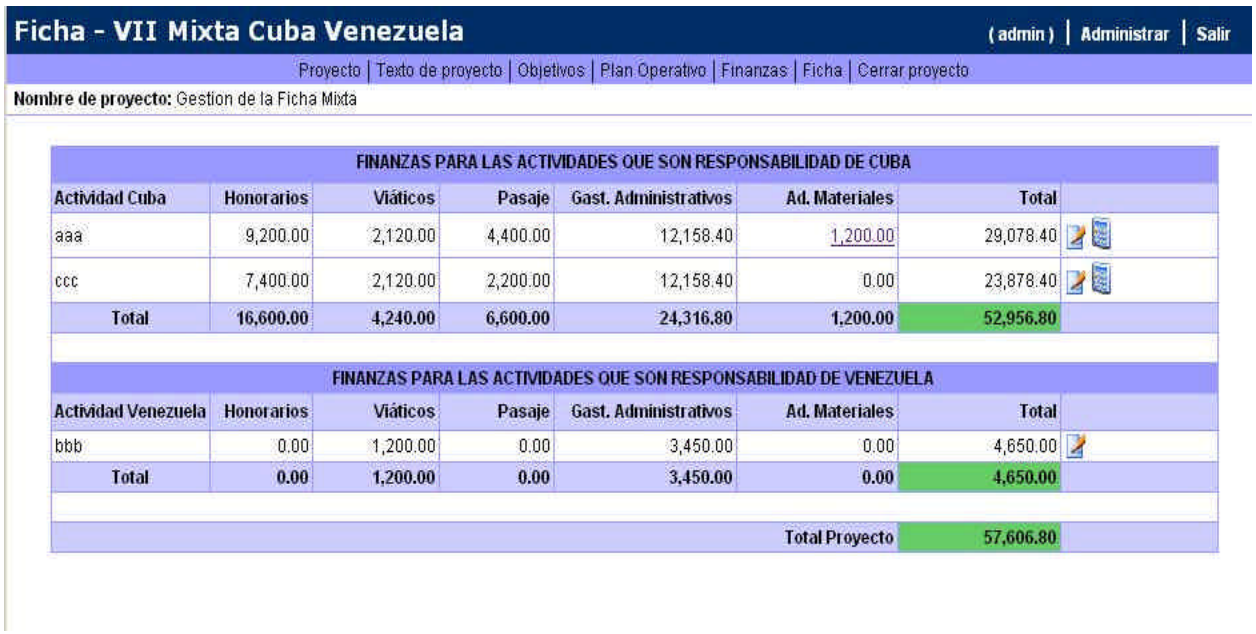
Anexo 13: Vista del caso de uso "Gestionar textos del proyecto".

Anexo 14: Vista del caso de uso "Gestionar Objetivos del Proyecto".

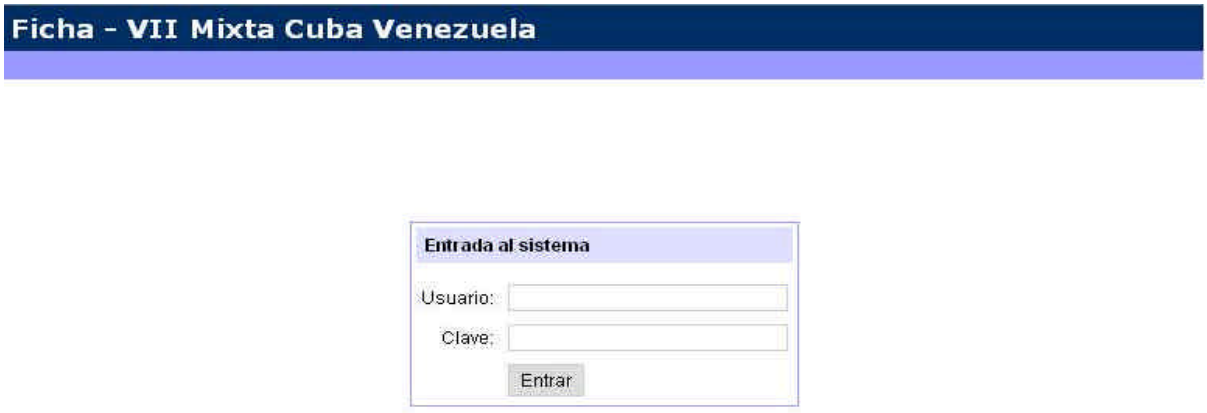


Anexo 15: Vista del caso de uso "Gestionar plan operativo del proyecto".

Anexo 16: Vista del caso de uso "Gestionar las finanzas del proyecto".

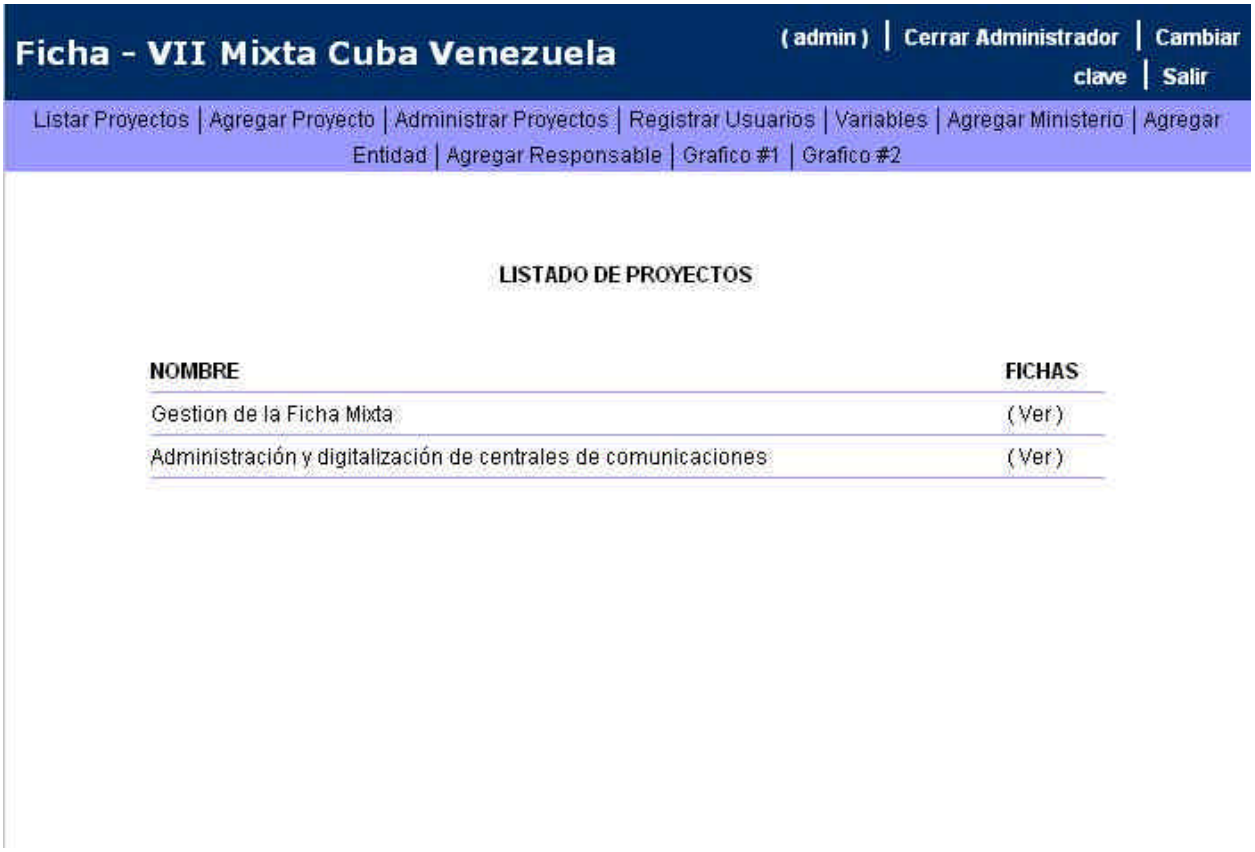


Anexo 17: Vista del caso de uso "Autenticarse".



Anexo 18: Vista del caso de uso "Cambiar Contraseña".

Anexo 19: Vista del caso de uso "Accionar Sobre los proyectos".



REFERENCIAS BIBLIOGRÁFICAS

[1] *AJAX*, 2006.

[2] *Conferencia 7 de Ingeniería del Software Patrones de Diseño*, 2005.

[3] *Programación Web.*

[4] J. Cantero, *Un vistazo a PHP5*

[5] J. C. P. y. A. Cid, *Herramienta de autor para la creación y gestión de objetos de aprendizaje*, 2006.

[6] C. G. Hernandez, *Módulo Alojamiento del Sistema Automatizado para la Gestión de Información de la Misión Milagro*, Universidad de las Ciencias Informticas Ciudad de la Habana 2006.

[7] B. Jacobson, Rumbaugh y otros, *El Proceso Unificado de Desarrollo de software.Addison-Wesley*, (2000).

[8] Macromedia, *Macromedia Dreamweaver MX* 2004.

[9] Mandrake, *Revisión Rápida de PHP5 integrado con Zend*, 2004.

[10] OMG, *OMG Unified Modeling Language Specification*, 2003.

[11] Schmulle, *J. Aprendiendo UML en 24 horas.Prentice Hall*.

[12] E. D. Terrero, *Módulo Flujo Médico del Sistema Automatizado para la Gestión de Información de la Misión Milagro.*, Universidad de las Ciencias Informatica Ciudad de la Habana 2006.

[13] L. Welicki, *Patrones y Antipatrones: una Introducción –Parte II*.

GLOSARIO DE TÉRMINOS Y SIGLAS

1. Administrador: es la persona que tiene privilegios para determinadas funcionalidades del sistema.
2. APACHE: es un servidor HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etcétera), Windows y otras, que implementa el protocolo HTTP/1.1.
3. AJAX: Asynchronous JavaScript And XML.
4. Arquitectura Cliente/Servidor: es un modelo para el desarrollo de sistemas de información, en el que las transacciones se dividen en elementos independientes que cooperan entre sí para intercambiar información, servicios o recursos.
5. CASE: *Computer Aided Software Engineering.*
6. CGI: *Common Gateway Interface.*
7. CEIS: *Centro de Estudio de Ingeniería de Sistemas*.
8. DHTML: *Dynamic HTML.*
9. 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.
10. HTTP: *HyperText Transfer Protocol.* Protocolo de Transferencia de Hipertextos. Modo de comunicación para solicitar páginas Web.
11. Herramientas CASE: Herramientas utilizadas para el desarrollo de proyectos de Ingeniería de Software.
12. Hardware: Componentes electrónicos, tarjetas, periféricos y equipo que conforman un sistema de computación; se distinguen de los programas (software) porque son tangibles.
13. Internet: Sistema de redes de computación ligadas entre sí, con alcance mundial, que facilita servicios de comunicación de datos como registro remoto, transferencia de archivos, correo electrónico y grupos de noticias. Internet es

una forma de conectar las redes de computación existentes que amplía en gran medida el alcance de cada sistema participante.

14. JSP: *Java Server Pages*. Es la tecnología para generar páginas web de forma dinámica en el servidor, desarrollado por Sun Microsystems, basado en scripts que utilizan una variante del lenguaje java. La tecnología JSP, o de JavaServer Pages, es una tecnología Java que permite a los programadores generar dinámicamente HTML, XML o algún otro tipo de página web. Esta tecnología permite al código Java y a algunas acciones predefinidas ser embebidas en el contenido estático.
15. Linux: Es el nombre de un núcleo, pero se suele denominar con este nombre a un sistema operativo de libre distribución software libre (y de código abierto), donde el código fuente está disponible públicamente y cualquier persona, con los conocimientos informáticos adecuados, puede libremente estudiarlo, usarlo, modificarlo y redistribuirlo.
16. Macromedia Dreamweaver MX: Herramienta para el desarrollo de aplicaciones Web de Macromedia. Combina en un único entorno de desarrollo accesible y potente las reconocidas herramientas de presentación visual de Dreamweaver, las características de rápido desarrollo de aplicaciones Web de Dreamweaver UltraDev y ColdFusion Studio, y el extenso soporte de edición de código de HomeSite. Ofrece una completa solución abierta para las tecnologías Web y estándares de hoy, incluyendo la accesibilidad y servicios Web.
17. Microsoft: Compañía que manufactura los sistemas de operación DOS y Windows.
18. 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.
19. MVC: *Modelo Vista Controlador.*
20. PC : *Personal Computer.*
21. PHP: *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 scripts. Con el objetivo de crear aplicaciones potentes.

22. PostgreSQL: es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (ORDBMS) libre.
23. Perl: *Practical Extraction and Report Language.* Es un lenguaje de programación desarrollado por Larry Wall inspirado en otras herramientas de UNIX como son: sed, grep, awk, c-shell.
24. PDO: *PHP Data Objects.*
25. RUP: *Rational Unified Process* (Proceso Unificado de desarrollo). Metodología para el desarrollo de Software.
26. Software: Programas de sistema, utilerías o aplicaciones expresados en un lenguaje de máquina.
27. 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.
28. Sitio Web: Es un conjunto de páginas web, típicamente comunes a un dominio de Internet o subdominio en la World Wide Web en Internet.
29. 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.
30. UCI: *Universidad de las Ciencias Informáticas*.
31. 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.
32. WEB (WWW): Red de documentos HTML intercomunicados y distribuidos entre servidores del mundo entero.
33. WML: *Website Meta Language*.

34. 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.

BIBLIOGRAFÍA

1. *AJAX un nuevo acercamiento a aplicaciones Web*, mayo 28 del 2005. Disponible en: <http://www.uberbin.net/archivos/internet/ajax-un-nuevo-acercamiento-a-aplicaciones-web.php> [Fecha de consulta 31 mayo 2006].
2. *Clases de Ingienería del Software I*, curso 2005-2006, UCI.
3. *Introducción a php*. Disponible en: <www.ciberteca.net/webmaster/php> [Fecha de consulta 24 marzo 2006].
4. Larman, Craig. *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* Primera Edición por Prentice Hall, Hispanoamericana S.A. 1999.
5. Larman, Craig. *UML y Patrones. Introducción al análisis y diseño orientado a objetos y al proceso unificado.* Segunda Edición por Prentice Hall.
6. MARRERO, D. *Modelado de aplicaciones Web con UML.* En: Conferencia de Ingeniería de Software, Diciembre 2002, ISPJAE (CEIS).
7. Matos, Rosa María. *Introducción al trabajo con Base de Datos*. Asignatura de Sistemas de Gestión de Base de Datos.
8. *PostgreSQL 8.1.x*. Disponible en: <http://www.postgresql.cl/> [Fecha de consulta 26 marzo 2006].
9. Peralta, Mario. *Estimación del esfuerzo basada en casos de uso*. Centro de Ingienería del Software e Ingienería del Conocimiento, Buenos Aires, Argentina.
10. Quatrani, Terry. *Visual Modeling with Rational Rose 2000 and UML,* Publisher Addison Wesley, Second Edition October 19, 1999
11. *Tutorial de PostgreSQL*. Disponible en: <http://es.tldp.org/Postgresql-es/web/navegable/tutorial/tutorial.html> [Fecha de consulta 26 marzo 2006].