Universidad Central "Marta Abreu" de las Villas
Facultad de Matemática, Física y Computación
Licenciatura en Ciencias de la Computación

Trabajo de Diploma

Software para la planificación y el control de inversiones

Autor:
Dayron Ramón González Ramírez

Tutores:
Lic. Yuri Denis Conrrado
Dr. Leonides González Castellanos

Consultante
Norge Teruel Fernández

Julio 2009

Hacemos constar que el presente Trabajo de Diploma ha sido realizado en la facultad de Matemática, Física y Computación de la Universidad Central “Marta Abreu” de Las Villas (UCLV) como parte de la culminación de los estudios de Licenciatura en Ciencia de la Computación, autorizando a que el mismo sea utilizado por la institución para los fines que estime conveniente, tanto de forma total como parcial.

_____

Firma del Autor

Los abajo firmantes, certificamos que el presente trabajo ha sido realizado según acuerdo de la dirección de nuestro centro y que el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____ _____

Firma del Tutor Firma del Tutor

_____

Jefe del Seminario de
Programación

La confianza en sí mismo es el primer secreto del éxito.

Ralph Waldo Emerson.

A mi mamá, por el amor, el esfuerzo, la preocupación y la entrega

de tantos años. Gracias por confiar en mí.

A mi papá, que quiero mucho por su preocupación y ayuda incondicional.

A mi abuela, por su apoyo y cariño infinito.

A mis hermanas.

A mi tío Michel.

A mis compañeros estudio, Yusnel, Yoandy, Ernesto, Anailys, el Curra por su comprensión y ayuda en todos los momentos.

Gracias por Todo.

A mis tutores Yuri y Leonides, por su apoyo y paciencia sin límite.

Por estar presentes a pesar de la distancia.

A mi familia, que de una manera u otra forman parte de este logro.

A mis amigos y profesores, por el tiempo compartido.

Muchas Gracias a todos

Dayron

RESUMEN

Este trabajo presenta un software denominado PlaCoIn para la planificación y control de inversiones, específicamente inversiones del proyecto endógeno Cuba-Venezuela (PENDOCUVE). El mismo permite a expertos no especialistas en computación la posibilidad de realizar una planificación de inversiones dentro de proyectos y subproyectos, además del control de la misma. Así, utilizando las tecnologías de desarrollo Web el sistema presenta características como, multiplataforma, multiusuario, enlaces con servidores de bases de datos, cubre los problemas comunes de diseño Web, permitiéndose así la visualización desde cualquier navegador Web.

El proceso de la planificación y el control de inversiones generalmente se desarrollan en aplicaciones de propósito general para la gestión de proyectos de, como el Microsoft Project. PlaCoIn ofrece una planificación y control estrictamente para PENDOCUVE, cubriendo las necesidades locales de ese proyecto.

ABSTRACT

This investigation present a software named PlaCoIn for the plans making and the control of investment s, specifically those of the endogenous project Cuba – Venezuela (PENDOCUVE). It allows experts who are not specialists in computers to perform an investment plan for the projects and sub projects as well as controlling the operation. Besides that and with the help of web development technologies, the system present the following features: multiplatform, multi users and link with servers of data bases. It also covers the common web design troubles so it makes it possible to be visualized from any web browser.

The planning process and the control of the investment is generally developed in applications of general propose for projects management, for example, Microsoft Project. PlaCoIn provides the plan making and control needed strictly for PENDOCUVE and it covers all of its demands.

ÍNDICE

INTRODUCCIÓN

PENDOCUVE, Proyecto Endógeno Cuba – Venezuela, es un proyecto para el desarrollo de la producción agropecuaria. Hasta el año 2006 se reportan 149 Núcleos de Desarrollo Endógeno activados, los cuales son un referente importante de cómo fomentar espacios territoriales en los que se materialice el desarrollo sostenible en sus dimensiones ecológica, económica y social. (Taily Galindo n.d.)

La gran mayoría de estos proyectos endógenos constan de tres módulos fundamentales para su desarrollo; módulo de inversión, módulo estadístico de producción y un módulo de capacitación.

En la actualidad estos proyectos difieren en el uso de software para llevar a cabo el control de su desarrollo, ejemplos de software que son utilizados son Microsoft Project, SPSS, Microsoft Excel, etc. Existe un software en el MINAGRI (Ministerio de la Agricultura) llamado 4C elaborado sobre la plataforma MSDOS que se usa a nivel nacional en la agricultura para controlar la estadística de producción del MINAGRI, este software es usado también para el desarrollo de Proyectos Endógenos en otras áreas del país con el propósito de controlar el módulo de estadística y producción.

Para el control del módulo de inversión se usan modelos Excel (.xls) con grandes volúmenes de datos, complejizándose en la medida que crecen los proyectos, lo cual hace ineficiente la búsqueda y el control de la información.

Existen otros procedimientos que son empleados para el desarrollo del Proyecto Endógeno los cuales no cuentan con un software capaz de llevar a cabo esa tarea, y su realización tiene que ser manualmente sobre el papel.

La mayoría de los software antes mencionados se rigen por una licencia de carácter privativo, teniendo en cuenta la política gubernamental existente, que trata sobre la migración hacia el software libre y la diversidad de software con diferentes propósitos que son usados para poder llevar a cabo el desarrollo del Proyecto Endógeno, se pretende en este trabajo implementar un software que esté enfocado a cubrir todas las necesidades del proyecto dentro del módulo de inversión para controlar y procesar la información que genera el Proyecto Endógeno, así como la generación de reportes para la toma de decisiones en cada uno de sus módulos.

Teniendo en cuenta lo antes expuesto se formulan las siguientes preguntas de investigación:

1. ¿Qué aspectos deben ser automatizados para el procesamiento de la información del módulo de inversión del proyecto Binacional Cuba – Venezuela para el desarrollo endógeno de la producción agropecuaria?
2. ¿Cómo se procesaría la información de forma automatizada que genera el proyecto Binacional Cuba – Venezuela para el desarrollo endógeno de la producción agropecuaria?

Objetivo general

Elaborar un software para la planificación y el control de inversiones del proyecto Binacional Cuba – Venezuela para el desarrollo endógeno de la producción agropecuaria.

Objetivos específicos

1. Realizar un diagnostico de la información que se requiere para la planificación y el control de inversiones del proyecto PENDOCUVE.
2. Realizar un estudio de tecnologías “libres” disponibles para la elaboración de un software, específicamente orientado al desarrollo Web.
3. Diseñar los módulos de software que automatizan la planificación y el control de inversiones.
4. Implementar los módulos de software que automatizan la planificación y el control de inversiones.

Justificación de la investigación

La principal dificultad del control del proyecto endógeno es que no existe un software que satisfaga todas la necesidades para el control del proyecto, es decir no esta centralizado el procesamiento de la información. Hasta el momento se ha empleado el uso de aplicaciones con fines especificas para el procesamiento de la información, lo cual ha llevado el uso de varios formatos de almacenamiento de la información debido a que cada software implementa un formato diferente. Con el sistema se dispondrá de toda la información centralizada, así como el procesamiento de la misma. Además, la aplicación contará con una interfaz gráfica la cual permitirá al usuario final interactuar eficazmente con el software.

El trabajo está estructurado de la siguiente manera:

El Capítulo 1 está dedicado a la investigación de software para el control de la información en las inversiones, las causas que han llevado su desarrollo y lo propuesto para cubrir esta necesidad. Además se aborda el estado actual de algunas de las tecnologías imprescindibles en la solución del problema.

El Capítulo 2 posee los detalles de diseño e implementación de los módulos del software para la planificación y el control de inversiones, además menciona los softwares utilizados y algunas tecnologías importantes para el diseño e implementación.

El Capítulo 3 está reservado para la documentación y escenarios de uso de los módulos del software.

CAPÍTULO I. BASES PARA DESARROLLO DEL SOFTWARE PARA LA PLANIFICACIÓN Y EL CONTROL DE INVERSIONES.

Este Capítulo aborda softwares para la planificación y el control de inversiones, la necesidad de esta investigación así como algunas tecnologías utilizadas en la implementación de la solución propuesta.

1.1 Proyectos Endógenos de la Producción Agropecuaria.

La entidad cubana, contra parte oficial del proyecto, el Poder Popular Municipal, se relaciona con los beneficiarios, a partir, en primer lugar, de los convenios gubernamentales Cuba – Venezuela y el intercambio que ello genera entre sus poblaciones. Los beneficiarios locales del proyecto son las comunidades urbanas, periurbanas y rurales del municipio que comprenden una población residente de 164 749 habitantes distribuida en 19 Consejos Populares. Los escenarios del proyecto benefician directamente a una población rural de 7 886 habitantes y periférica estimada en otros 30000 habitantes, a través de la mejora de su entorno y la potenciación de su actividad económica y productiva.

1.2 Necesidad de un Software para la planificación y el control de inversiones

En la actualidad, los proyectos endógenos desarrollados en nuestro país no cuentan con un software que integre como un todo, las acciones necesarias para el control de toda la información que genera dicho proyecto; esta información es controlada por una diversidad de software con diferentes fines. Debido a esto, la búsqueda de información del proyecto puede no ser eficiente en determinado momento. Tener la información centralizada y bien organizada puede llevar a una buena organización y confiabilidad.

1.3 Solución a la necesidad

Como solución a esta necesidad se presenta la creación de una herramienta que suministre múltiples funcionalidades a los usuarios con el fin de planificar y controlar las inversiones haciendo posible un mejor control sobre los datos del proyecto.

1.4 Lenguaje Unificado de Modelado

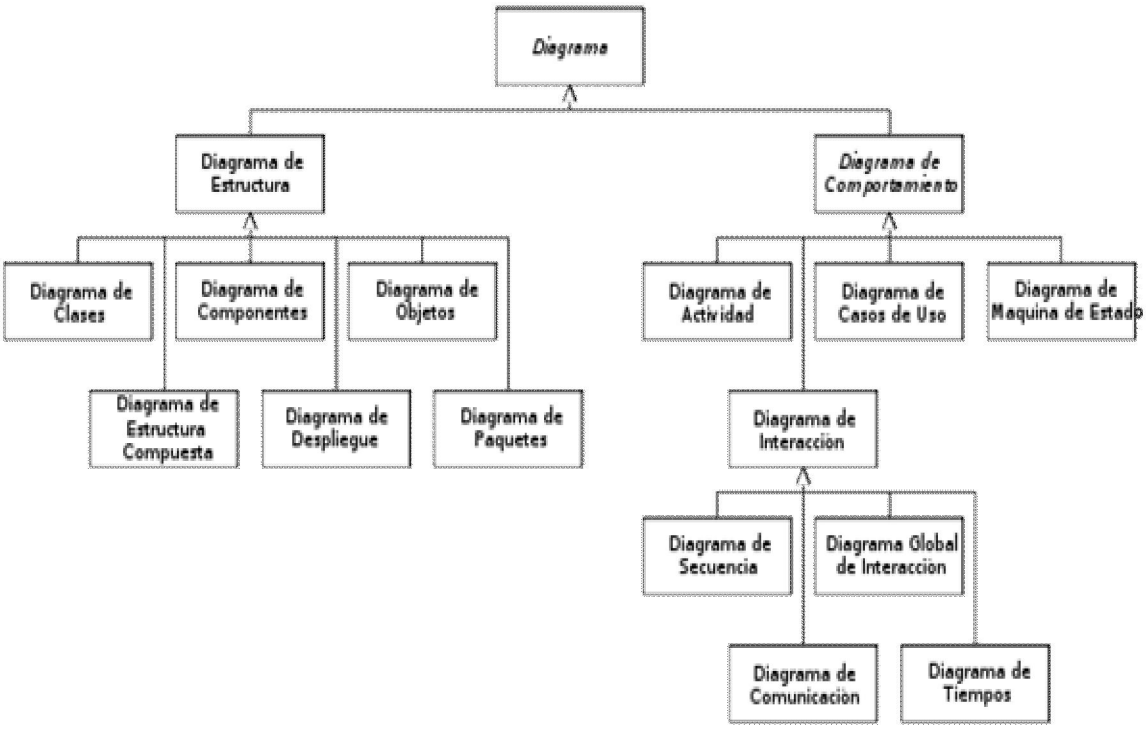
El Lenguaje Unificado de Modelado (UML, Unified Modeling Language) es un lenguaje gráfico para visualizar, especificar, construir y documentar los artefactos de un sistema con gran cantidad de software. UML proporciona una forma estándar de escribir los planos de un sistema, cubriendo tanto las cosas conceptuales, tales como procesos del negocio y funciones del sistema, como las cosas concretas, tales como las clases escritas en un lenguaje de programación específico, esquemas de bases de datos y componentes de

softwares reutilizables.(Grady BOOCH et al. n.d.)

UML no puede compararse con la programación estructurada, pues UML significa Lenguaje Unificado de Modelado, no es programación, solo se diagrama la realidad de una utilización en un requerimiento. Mientras que, programación estructurada, es una forma de programar como lo es la orientación a objetos, sin embargo, la programación orientada a objetos viene siendo un complemento perfecto de UML, pero no por eso se toma UML sólo para lenguajes orientados a objetos.(actores varios n.d.)

UML cuenta con varios tipos de diagramas, los cuales muestran diferentes aspectos de las entidades representadas.

En UML 2.0 hay 13 tipos diferentes de diagramas. Para comprenderlos de manera concreta, a veces es útil categorizarlos jerárquicamente, como se muestra en la figura siguiente.



Tipos de diagramas UML

1.5 Framework

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

Un framework representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio. (Yenier Cruz Brito 2007)

Un framework Web provee una infraestructura de programación para tus aplicaciones, para que puedas focalizarte en escribir código limpio y de fácil mantenimiento.(Adrian Holovaty & Jacob Kaplan-Moss 2008)

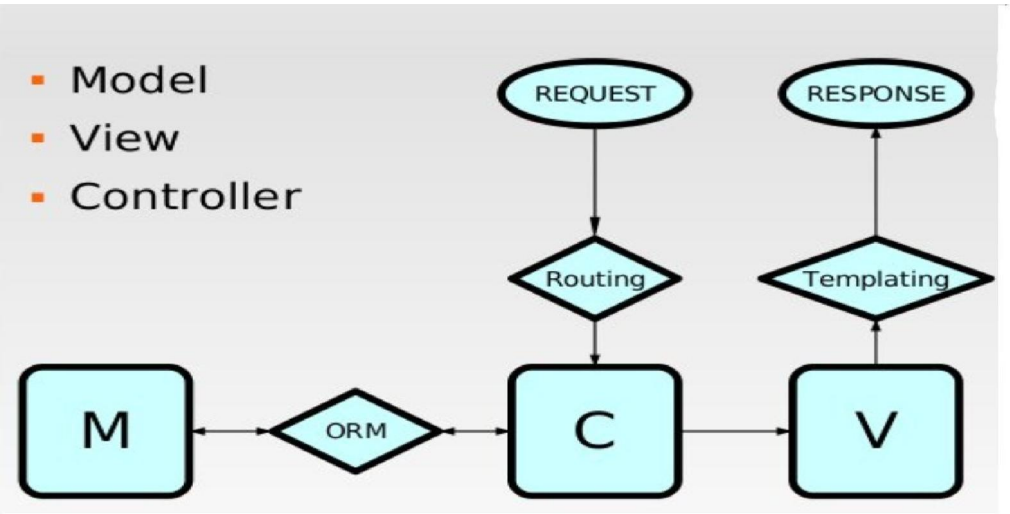
Son diseñados con el intento de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software que tratando con los tediosos detalles de bajo nivel de proveer un sistema funcional.

1.5.1 Arquitectura

Este aspecto se basa generalmente en el modelo MVC (Controlador => Modelo => Vista) ya que fragmenta la programación.

El MVC define una forma de desarrollar software en la que el código para definir y acceder a los datos (el modelo) está separado del pedido lógico de asignación de ruta (el controlador), que a su vez está separado de la interfaz del usuario (la vista).

- **Controlador**: Controla el acceso (incluso todo) a nuestra aplicación esto pueden ser: archivos, scripts o programas; cualquier tipo de información que permita la interfaz. Así, podremos diversificar nuestro contenido de forma dinámica, y estática (a la vez); pues, sólo debemos controlar ciertos aspectos (como se ha mencionado antes).
- **Modelo**: Este miembro del controlador maneja las operaciones lógicas, y de manejo de información (previamente enviada por su ancestro). Cada miembro debe ser meticulosamente llamado, en su correcto nombre y en principio, con su verdadera naturaleza: el manejo de información, su complementación directa.
- **Vista**: Al final, a este miembro de la familia le corresponde dibujar, o expresar la última forma de los datos: la interfaz gráfica que interactúa con el usuario final del programa. Después de todo, a este miembro le toca evidenciar la información obtenida hasta hacerla llegar con el controlador.(actores varios n.d.)



Modelo Vista Controlador

Una ventaja clave de este enfoque es que los componentes son "loosely coupled". Eso significa que cada pieza de la aplicación (por ejemplo una aplicación django) tiene un único propósito clave que puede ser modificado independientemente sin afectar las otras piezas.

Por ejemplo, un desarrollador puede cambiar la URL de cierta parte de la aplicación sin afectar la implementación subyacente. Un diseñador puede cambiar el HTML de una página sin tener que tocar el código (ejemplo Python) que la renderiza. Un administrador de base de datos puede renombrar una tabla de la base de datos y especificar el cambio en un único lugar, en lugar de tener que buscar y reemplazar en varios archivos.

1.6 Lenguajes de programación.

Los lenguajes de programación no existen en el vacío, ellos son herramientas para escribir software. Existen múltiples lenguajes de programación para el desarrollo Web, todos con características particulares pero que giran sobre la misma idea de desarrollo. Durante el proceso de desarrollo de un software los lenguajes de programación tienen un roll importante, por lo que sería conveniente un análisis de sus principales características antes de usarlo.

1.6.1 PHP (acrónimo de Hypertext Preprocessor)

Es un lenguaje de secuencia de comandos de servidor diseñado específicamente para la Web. Dentro de una página Web puede incrustar código PHP que se ejecutará cada vez que se visite una página. El código PHP es interpretado en el servidor Web y genera código HTML y otros contenidos.(Luke Welling & Laura Thomson n.d)

Entre sus características fundamentales están:

- Gratuito. A1 tratarse de software libre, puede descargarse y utilizarse en cualquier aplicación, personal o profesional, de manera completamente libre.
- Gran popularidad.
- Integración con múltiples bases de datos.
- Versatilidad. PHP puede usarse con la mayoría de sistemas operativos, ya sea basados en UNIX (Linux, Solares, FreeBSD ...), como con Windows, el sistema operativo de Microsoft.
- Gran número de funciones predefinidas.

(José Antonio Gallego Vásquez n.d)

1.6.2 Python

Python es un lenguaje de programación interpretado creado por Guido van Rossum en el año 1991.

Python permite dividir el programa en módulos reutilizables desde otros programas Python. Viene con una gran colección de módulos estándar que se pueden utilizar como base de los programas (o como ejemplos para empezar a aprender Python). También hay módulos incluidos que proporcionan E/S de ficheros, llamadas al sistema, sockets y hasta interfaces a GUI (interfaz gráfica con el usuario) como TK, GTK, QT entre otros.

Python se utiliza como lenguaje de programación interpretado, lo que ahorra un tiempo considerable en el desarrollo del programa, pues no es necesario compilar ni enlazar. El intérprete se puede utilizar de modo interactivo, lo que facilita experimentar con características del lenguaje, escribir programas desechables o probar funciones durante el desarrollo del programa. (autores varios n.d.)

Python presenta una serie de ventajas que lo hacen muy atractivo para su uso profesional:

- El Python es un lenguaje de programación potente.
- El ambiente de Python es completamente abierto y permite la integración con herramientas externas.
- Presenta una completa caja de herramientas y módulos con lotes de funciones y clases que pueden ser contenidos en un único archivo.
- La transferencia de funciones como argumentos de funciones es sencillo.
- Fácil construcción y uso de estructuras de datos heterogéneas.
- Soporta programación orienta a objetos.
- Permite la integración con otros lenguajes de programación como C, C++, Fortran, etc.
- Es libre (open source) y es multiplataforma.

(Hans Petter Langtangen n.d.)

1.7 El Framework de Python (Django)

Django es un Framework Web escrito en Python de código abierto, con un diseño pensado en aplicaciones manejadores de contenidos, que satisface cualquier tipo de aplicación Web que se quiera construir.(Simon Willison 2005)

Con Django se ahorra tiempo en el desarrollo Web y hace que este desarrollo sea eficaz. Permite crear y mantener aplicaciones Web de alta calidad con mínimo esfuerzo. En su mejor estado, el desarrollo web es un acto entretenido y creativo; en su peor estado, puede ser una molestia repetitiva y frustrante. Django te permite enfocarte en la parte creativa la esencia de tus aplicaciones Web, al mismo tiempo que mitiga el esfuerzo de las partes repetitivas.

De esta forma, provee un alto nivel de abstracción de patrones comunes en el desarrollo Web, atajos para tareas frecuentes de programación y convenciones claras sobre como solucionar problemas. Al mismo tiempo, Django intenta no entrometerse, dejándote trabajar fuera del ámbito del framework según sea necesario.(Adrian Holovaty 2007)

Otras características de Django son:

- Un mapeador objeto – relacional.
- Aplicaciones "enchufables" que pueden instalarse en cualquier página gestionada con Django.
- Una API de bases de datos robusta.
- Un sistema incorporado de "vistas genéricas" que ahorra tener que escribir la lógica de ciertas tareas comunes.
- Un sistema extensible de plantillas basado en etiquetas, con herencia de plantillas.
- Un despachador de URLs basado en expresiones regulares.
- Un sistema "middleware" para desarrollar características adicionales; por ejemplo, la distribución principal de Django incluye componentes middleware que proporcionan cacheo, compresión de la salida, normalización de URLs, protección CSRF y soporte de sesiones.
- Soporte de internacionalización, incluyendo traducciones incorporadas de la interfaz de administración.

- Documentación incorporada accesible a través de la aplicación administrativa (incluyendo documentación generada automáticamente de los modelos y las librerías de plantillas añadidas por las aplicaciones).

1.8 Servidores de bases de datos.

Las aplicaciones en red son cada día más numerosas y versátiles. En muchos casos, el esquema básico de operación es una serie de scripts que rigen el comportamiento de una base de datos.
Debido a la diversidad de lenguajes y de bases de datos existentes, la manera de comunicar entre unos y otras sería realmente complicada a gestionar de no ser por la existencia de estándares que nos permiten el realizar las operaciones básicas de una forma universal.

Las bases de datos contribuyen a que las aplicaciones Web sean más dinámicas y le den una mejor atención a los usuarios en cuanto a informaciones propias de cada uno de ellos, facilidad de uso y visualización de reportes.

Un Sistema de Gestión de Bases de Datos (SGBD) es un conjunto de programas de propósito general que permite controlar el acceso y la utilización de la base de datos por los usuarios, para incluir, modificar o recuperar información de acuerdo a las relaciones definidas al crear la base de datos. Luego, este software controla la organización, almacenamiento, recuperación, seguridad e integridad de los datos de una base de datos.

SQL es un lenguaje estandarizado que permite el fácil almacenamiento, actualización y acceso a la información.

Es de eso de lo que trata el Structured Query Language que no es más que un lenguaje estándar de comunicación con bases de datos. Hablamos por tanto de un lenguaje normalizado que nos permite trabajar con cualquier tipo de lenguaje (Python o PHP) en combinación con cualquier tipo de base de datos (MS Access, SQL Server, MySQL, Postgres...).

El hecho de que sea estándar no quiere decir que sea idéntico para cada base de datos. En efecto, determinadas bases de datos implementan funciones específicas que no tienen necesariamente que funcionar en otras.

Aparte de esta universalidad, el SQL posee otras dos características muy apreciadas. Por

una parte, presenta una potencia y versatilidad notables que contrasta, por otra, con su accesibilidad de aprendizaje.

Entre los SGBD multiusuario con excelente rendimiento encontramos a Microsoft SQL Server, Oracle, Firebird, Informix, MySQL y PostgreSQL, entre otros. Demostrándose así que en la actualidad, el software libre ha alcanzado un éxito merecido, por su viabilidad y por sus ventajas desde el punto de vista económico, llevándolo a ser citado entre los softwares más usados y recomendados.

Común es la pregunta entre las personas que se adentran por primera vez en el mundo de las bases de datos libres: ¿MySQL o PostGreSQL? En realidad no es una pregunta asociada específicamente a los "novatos", ya que incluso los profesionales dedicados a este campo se realizan muchas veces esta misma pregunta.

1.8.1 MySQL como gestor de bases de datos relacionales.

Para el desarrollo de este trabajo se selecciona MySQL porque es uno de los gestores de bases de datos más populares desarrollados bajo la filosofía de software libre. Es un gestor multihilo y multisuque puede utilizarse gratuitamente y su código fuente se encuentra disponible.

Inicialmente, MySQL carecía de elementos considerados esenciales en las bases de datos relacionales, tales como integridad referencial y transacciones. A pesar de ello, atrajo a los desarrolladores de páginas Web con contenido dinámico, justamente por su simplicidad. Fue desarrollado originalmente para manejar grandes bases de datos mucho más rápido que las soluciones existentes y ha sido usado con éxito en entornos de producción de alto rendimiento durante varios años. Aunque se encuentra en constante desarrollo, ofrece actualmente un conjunto muy rico y útil de funciones.

Mysql es un verdadero multiusuario, multihilo servidor de bases de datos SQL(Structured Query Language). Mysql es una implementación Cliente/Servidor que consiste en un servidor demonio mysqld y muchos programas clientes y librerías.(Detron HB and Monty Program KB 2000)

1.9 Servidor Web Apache

El servidor Web de apache presenta un sin número de características que lo hacen muy potente, como por ejemplo, es gratis, es decir es software libre, es popular, es sencillo de configurar, es versátil ya se puede instalar sobre Unix o sobre Windows. El 60 % de los sitios actuales en el mundo están montados sobre servidores apache.

1.9.1 Django con Apache y mod_python

Apache con mod_python es actualmente la configuración más robusta para usar Django en un servidor en producción.

Mod_python es un plugin de Apache que embebe Python dentro de Apache y carga código Python en memoria cuando el servidor se inicia. El código permanece en memoria a lo largo de la vida del proceso Apache, lo que repercute en aumentos significativos de performance comparado con otros arreglos de servidor.

1.10 Conclusiones parciales

En este Capítulo se ha realizado un estudio de algunas de las tecnologías de desarrollo web existentes en la actualidad, de manera concluyente el presente trabajo se desarrollará en el lenguaje de programación python por las características antes mencionadas y por tener el autor amplios conocimientos de dicho lenguaje, además se empleará el uso de Django como framework para el desarrollo de la aplicación, por las grandes facilidades de programación que brinda y por otras características ya mencionadas, como gestor de bases de datos se utilizara MySQL y la aplicación se servirá en un servidor apache con el módulo mod-python instalado.

CAPÍTULO II. DISEÑO E IMPLEMENTACIÓN DEL SOFTWARE PARA LA PLANIFICACIÓN Y EL CONTROL DE INVERSIONES.

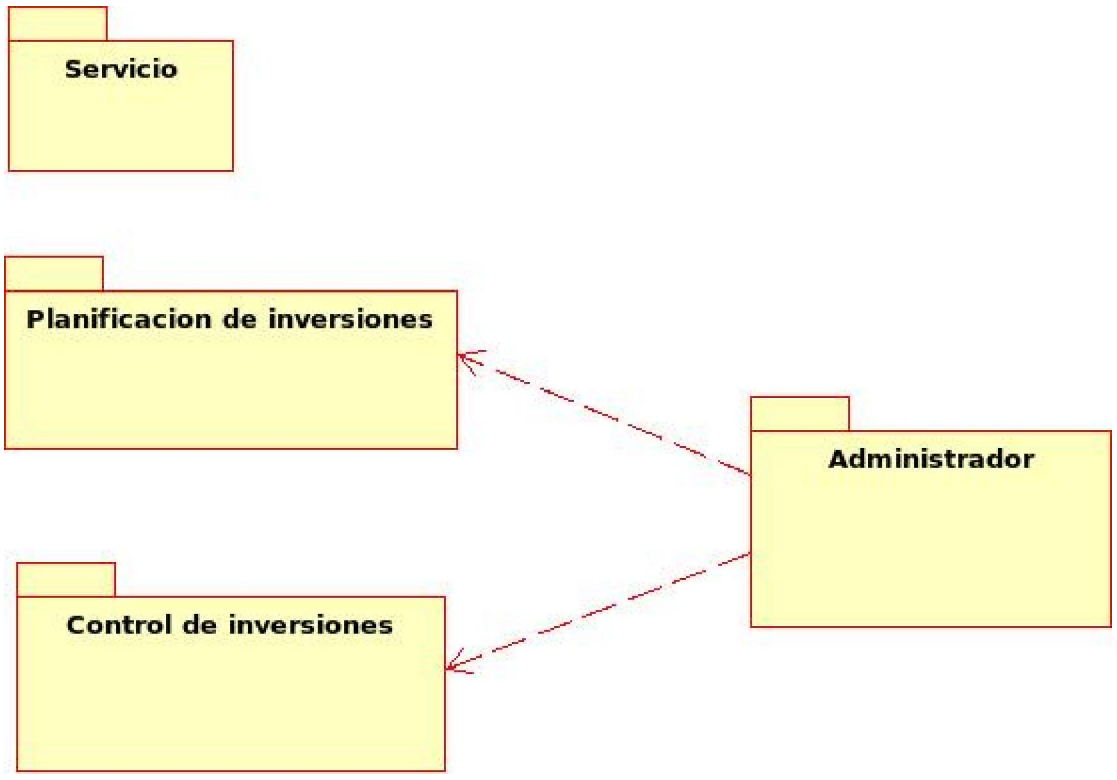
En este capitulo se especifican los detalles de diseño e implementación de cada uno de los módulos que conforman el software para la planificación y el control de inversiones en el proyecto PENDOCUVE, así como las relaciones existentes entre estos módulos.

2.1 Diseño de los módulos del Software.

En el diseño de cada unos de los módulos del software se hará uso de los diagramas que aporta el UML para dar una mejor visión de las funcionalidades implementadas.

2.1.1 Diagrama de paquetes.

Este software cuenta con cuatro paquetes, lo cuales quedan detallados en el siguiente diagrama de paquetes.

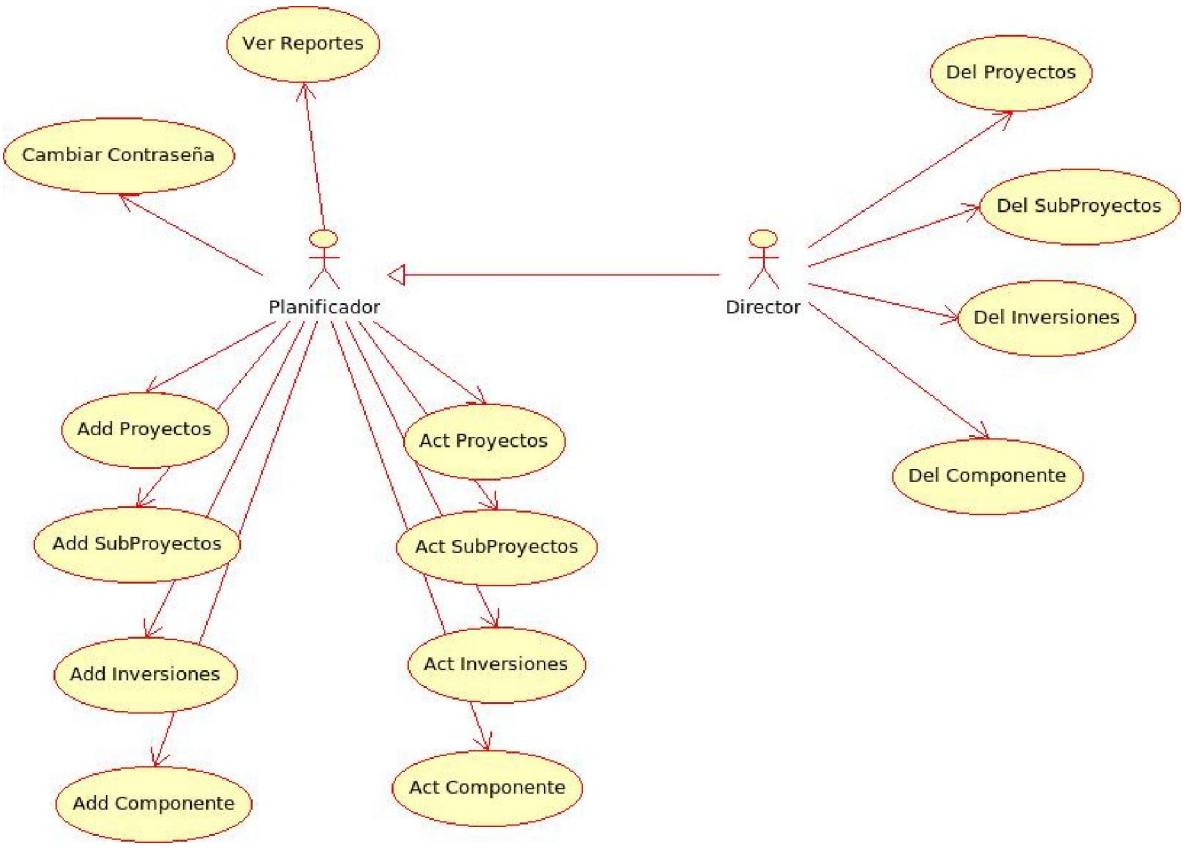


2.1.1.1 Diseño del paquete planificación de inversiones.

Basado en los estudios realizados, surge la creación de este paquete, el cual brinda la posibilidad de realizar la planificación de la inversión, cumpliendo uno de los objetivos parciales de esta investigación. El paquete esta formado por la implementación de los casos de usos, siendo estos todas las acciones para la planificación de la inversión.

2.1.1.1.1 Diagramas de casos de usos.

Para este paquete se presenta 14 casos de usos, donde cada uno describe las acciones posibles a realizar por el usuario determinado. Se presenta continuación el diagrama de casos de usos de este paquete.



Para un mejor entendimiento de estos casos de uso a continuación se realiza una breve descripción de los mismos.

- **Ver Reportes:** El usuario puede ver todos los reportes establecidos para la planificación de inversiones, además el usuario tiene acceso a todos los reportes parciales generados de forma automática por el software.
- **Cambiar Contraseña:** El usuario puede cambiar su contraseña.
- **Add Proyectos :** El usuario puede adicionar proyectos.
- **Add Subproyectos:** El usuario puede adicionar subproyectos a los proyectos previamente creados
- **Add Inversiones:** El usuario puede adicionar inversiones para cada subproyecto.
- **Add Componente:** El usuario puede adicionar componentes, ya sea componente de construcción y montaje, componente de equipos o componente otros.
- **Act Proyectos:** El usuario puede actualizar un proyecto previamente creado
- **Act Subproyectos:** El usuario puede actualizar un subproyecto previamente creado.
- **Act Inversiones**: El usuario puede actualizar una inversión previamente creada.
- **Act Componente:** El usuario puede actualizar cualquiera de los tres componentes previamente creados.
- **Del Proyectos:** El usuario puede eliminar de manera definitiva un proyecto, eliminando en caso de que el usuario tengas los privilegios necesarios todo lo que dependa de el, cumpliéndose la integridad referencial.
- **Del Subproyectos:** El usuario puede eliminar subproyectos de forma definitiva, además se cumple lo del caso de uso anterior.
- **Del Inversiones:** El usuario puede eliminar inversiones de forma definitiva, además se cumple lo del caso de uso anterior.
- **Del Componente:** El usuario puede eliminar componentes de forma definitiva, además se cumple lo del caso de uso anterior.

2.1.1.1.2 Representación de las clases utilizadas en el paquete.

Para lograr una mayor organización y modularización se crearon clases con diversas funcionalidades, detalladas a continuación. Algunas de las clases utilizadas no presentan funcionalidades pues se crearon con la intención de generar formularios a partir de ellas para la introducción de datos, estas clases no serán especificadas, otras clases se relacionan con una meta clase denominada *admin* donde en ella se implementan todas las funcionalidades.

Clase Proyecto

Esta clase representa el concepto de Proyectos, posee un método para obtener proyectos, y se relaciona con la meta clase *admin*

Clase Acreedor

Esta clase representa el concepto de Acreedores

Clase SubProyecto

Esta clase representa el concepto de SubProyectos y es hija de la clase Proyectos.

Clase Inversiones

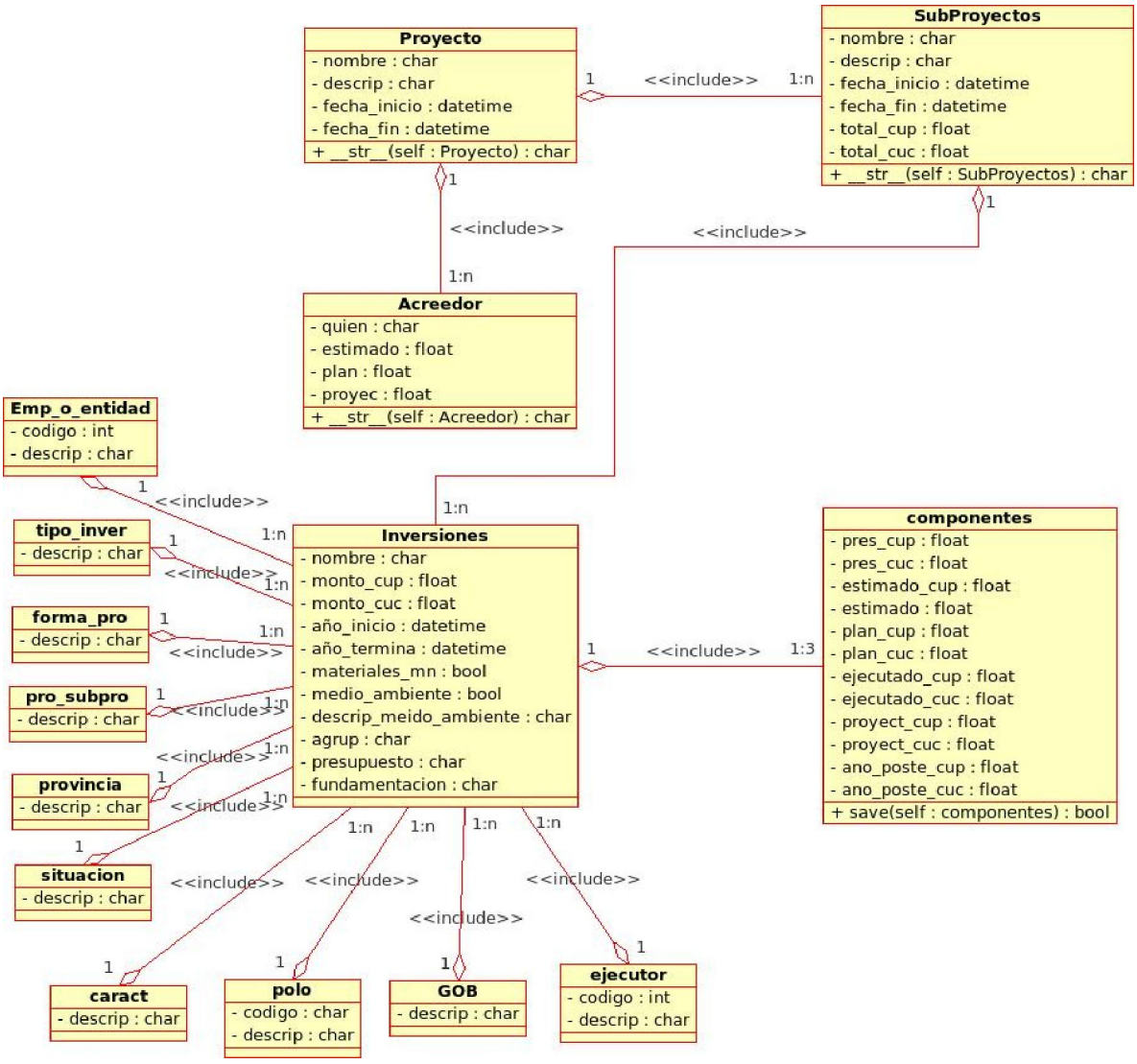
Esta clase representa el concepto de Inversiones, el control de esta clase se realiza a través de la meta clase *admin*

Clase Componentes

Esta clase representa el concepto de componentes, los cuales son, componente de construcción y montaje, componente de equipos y componente otros.

2.1.1.1.3 Diagramas de clases del paquete.

A continuación se muestra como queda definido el diagrama de clases para este paquete.



2.1.1.2 Diseño del paquete control de inversiones.

Este paquete se identifica como un controlador de datos, esta diseñado para obtener y utilizar los metadatos del software publicados por el paquete planificar inversiones. Estos elementos quedan expresados como recursos y son localizados mediante URLs, de forma tal que este paquete constituye una herramienta más para el control de las inversiones.

2.1.1.2.1 Diagramas de casos de usos.

A continuación se muestra un diagrama donde queda representado los casos de usos de este paquete para un total de 23 casos de usos.

Descripción de los casos de usos:

- **Add Contratos :** el usuario puede adicionar contratos
- **Add Facturas:** el usuario puede adicionar facturas asociadas a contratos existentes en el sistema
- **Add Cheques:** el usuario puede adicionar cheques para el pago de una o más facturas
- **Add Aprobación :** el usuario puede adicionar aprobación
- **Add Ent. Inversionista:** el usuario puede adicionar entidades inversionistas
- **Add Repres. Contrato:** el usuario puede adicionar representantes de contratos
- **Add Especialidades:** el usuario puede adicionar especialidades
- **Act Contratos:** el usuario puede actualizar contratos que ya existan en el sistema

- **Act Facturas:** el usuario puede seleccionar facturas para actualizar su información
- **Act Cheques:** el usuario puede actualizar cheques
- **Act Aprobación:** el usuario puede actualizar aprobaciones
- **Act Ent. Inversionista:** el usuario puede actualizar entidades inversionistas
- **Act Repres. Contrato:** el usuario puede actualizar representantes de contratos
- **Act Especialidades:** el usuario puede actualizar especialidades
- **Del Contratos:** el usuario puede eliminar de forma definitiva contratos, esta eliminación provocará la eliminación de todos los objetos en cascada que dependían de dicho contrato
- **Del Facturas:** el usuario puede eliminar facturas, cumpliéndose la misma condición que lo anterior.
- **Del Cheques:** el usuario puede eliminar facturas
- **Del Aprobación:** el usuario puede eliminar aprobaciones
- **Del Ent. Inversionista:** el usuario puede eliminar entidades inversionistas
- **Del Repres. Contrato:** el usuario puede eliminar representantes de contratos
- **Del Especialidades:** el usuario puede eliminar especialidades

2.1.1.2.2 Representación de las clases utilizadas en el paquete.

Se han desarrollado varias clases para un mejor manejo de la información. Se presentan como quedan las mencionadas clases.

Clase Inversionista

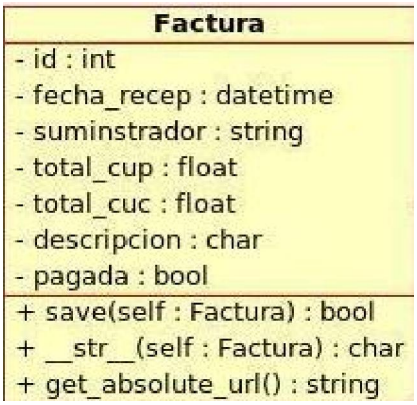
Esta clase representa el concepto de Inversionistas.

Clase Contrato

Esta clase representa el concepto de Contrato, presenta un método save() personalizado.

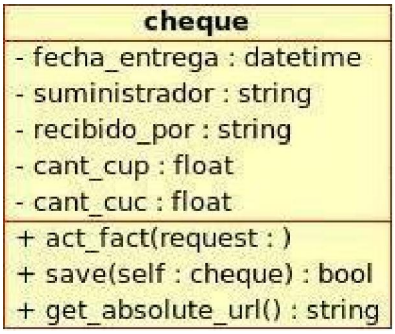
Clase Factura

Esta clase representa el concepto de Factura, y guarda relación con la clase contrato.



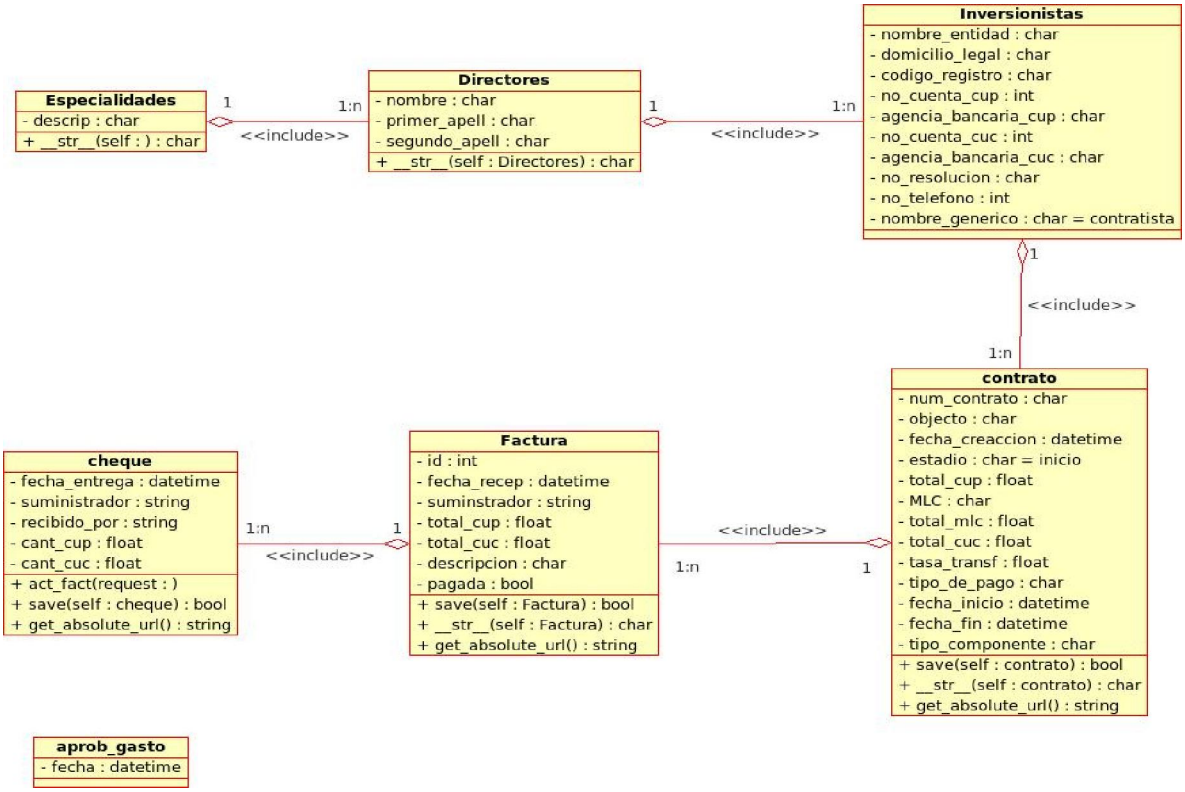
Clase Cheque

Esta clase representa el concepto de Cheque, esta relacionada con facturas y posee métodos que se ejecutan en cada inserción de un objeto cheque.



2.1.1.2.3 Diagrama de clases del paquete control de inversiones.

A continuación se muestra como queda definido el diagrama de clases del paquete control de inversiones.

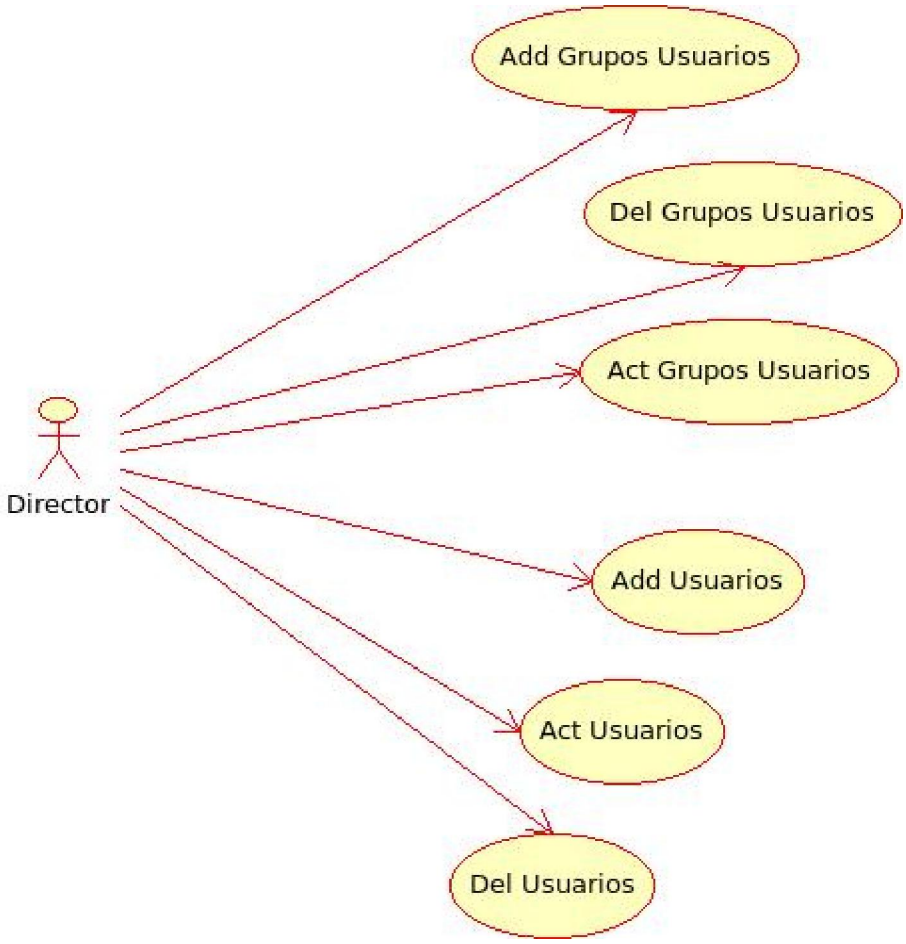


2.1.1.3 Diseño del paquete administrador.

Este paquete tiene como función principal el control total de todos los objetos del sistema, y tiene los privilegios de eliminar dichos objetos. Además se encarga de todo el control de acceso por usuarios y grupos de usuarios.

2.1.1.3.1 Diagrama de casos de usos.

A continuación se muestra los seis casos de usos correspondientes paquete administrador:



Descripción de los casos de usos:

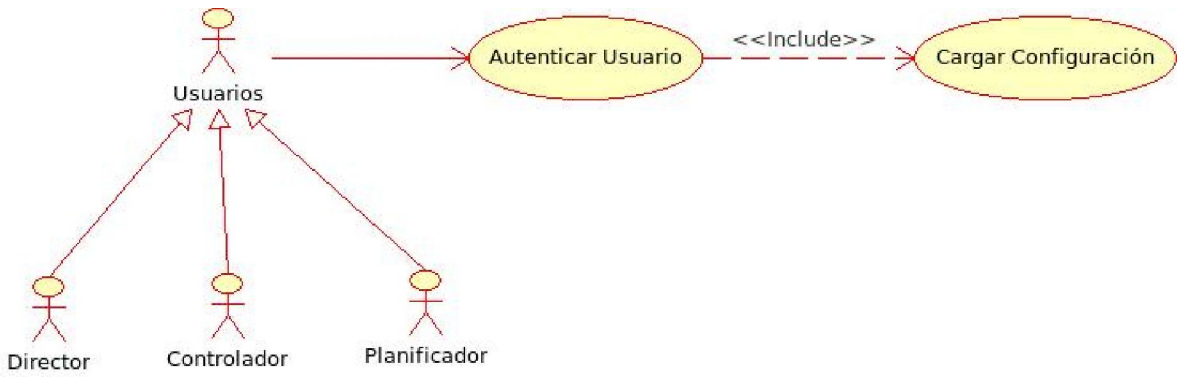
- **Add Grupos Usuarios:** el usuario puede adicionar grupos de usuarios, especificando todos los permisos para el grupo.
- **Del Grupos Usuarios:** el usuario puede eliminar grupos de usurarios previamente creados

- **Act Grupos Usuarios:** si existen grupos de usuarios creados, estos puede ser modificados por un usuario que viva con ese derecho
- **Add Usuarios:** el usuario administrador puede crear usuarios en el sistema, indicando a que grupo pertenece, así como una serie de variables a llenar, además puede adicionarle permisos extras además del grupo.
- **Act Usuarios:** el usuario puede actualizar la información de un usuario determinado.
- **Del Usuarios:** el usuario puede eliminar usuarios de forma definitiva del sistema.

2.1.1.4 Diseño del paquete servicio.

Este paquete tiene como función principal dar servicio al usuario en cuestión, es el que controla el acceso al sistema y es el encargado de mostrarle a cada usuario su nivel de acceso según sus privilegios.

2.1.1.4.1 Diagrama de casos de usos.



Descripción de los casos de usos.

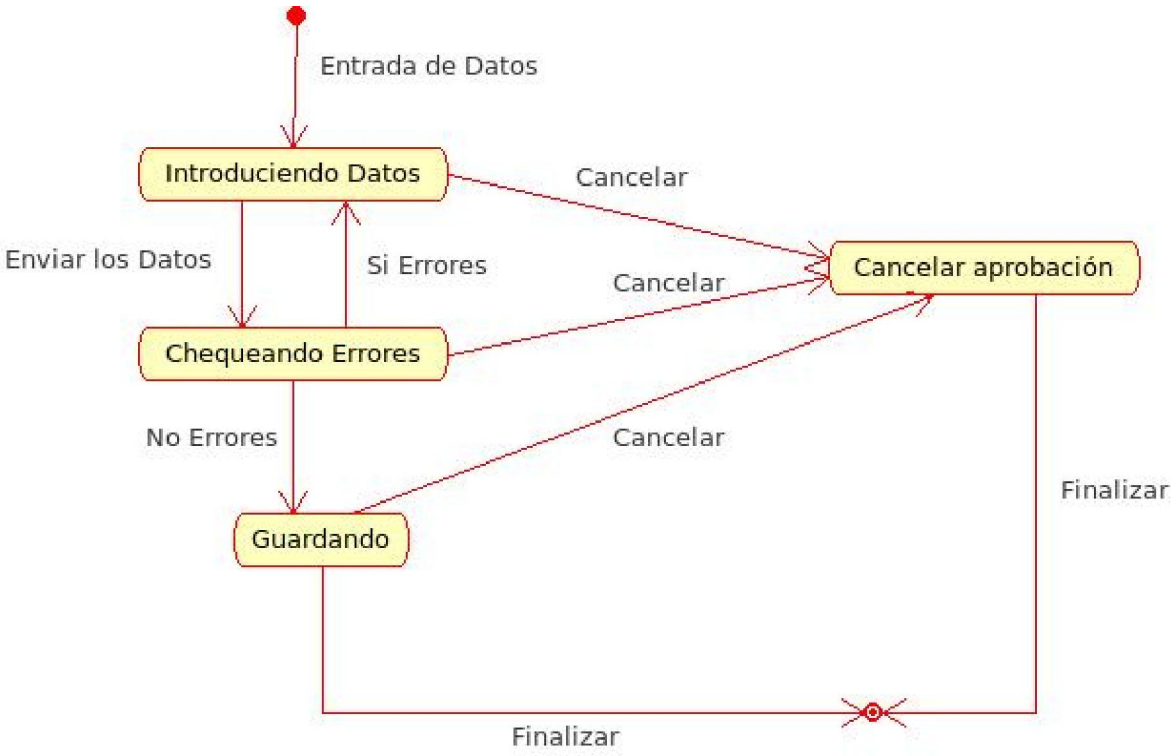
- **Autenticar Usuario:** el usuario introduces su nombre de usuario y contraseña correspondiente y el sistema verificara si es correcto
- **Cargar Configuración:** si el usuario es autenticado en el sistema, el sistema cargar su configuración personalizada, mostrando los objetos a los que el usuario tiene acceso y con todos sus permisos de acceso.

2.1.2 Diagrama de estados.

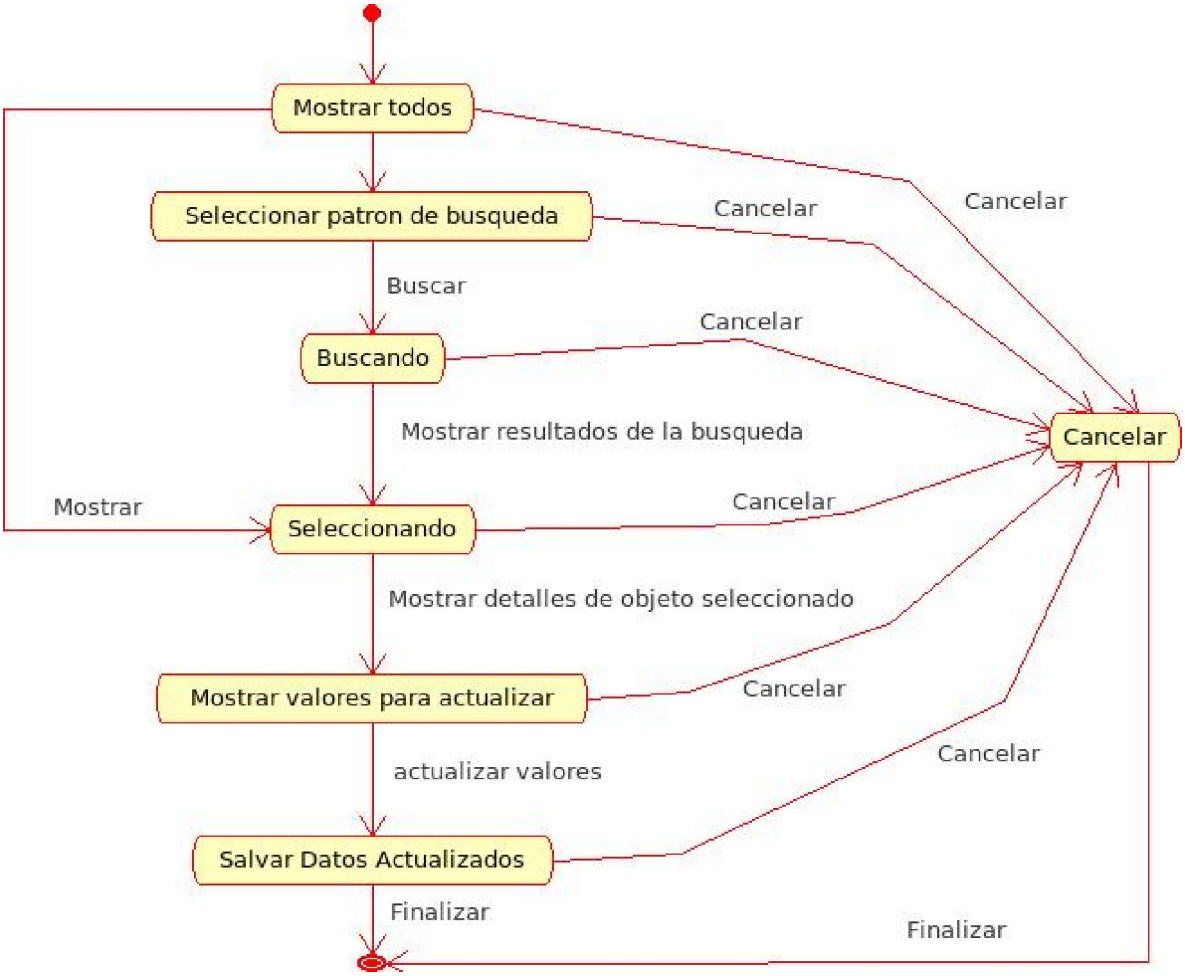
Los casos de usos de cada paquete quedan detallados en los siguientes diagramas que muestran cada unos de los estados donde se encontrara la aplicación cada ves que el usuario ejecute una acción determinada.

Los casos de uso de actualización de todos los paquetes presentan un diagramas de estado similares variando en los datos que se maneja en los diferentes estados, por dicha razón se mostraran los diagramas de estados representando a grupos de casos de usos.

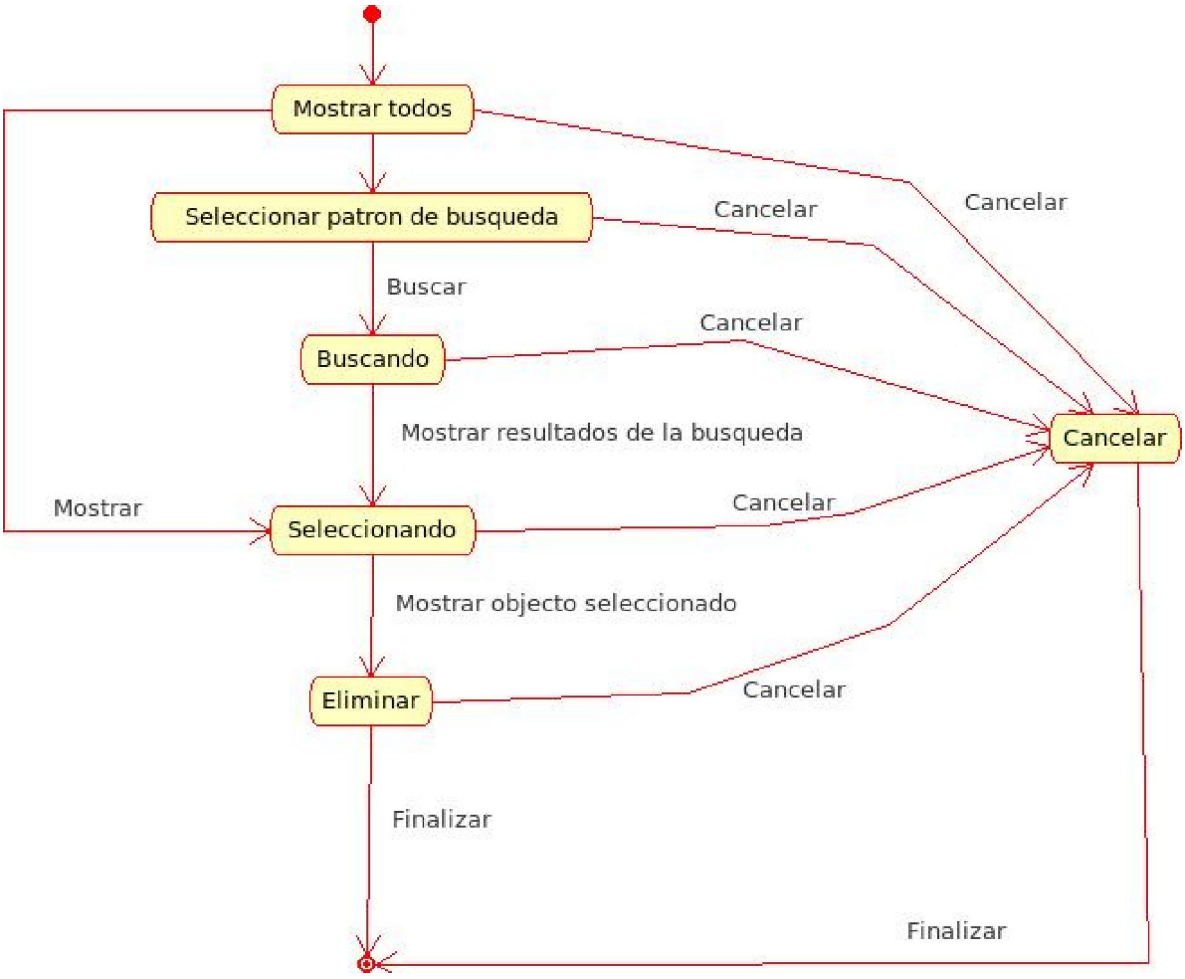
- **Diagrama de estado para los casos de usos de adicionar (Add)**



- **Diagrama de estado para los casos de usos de actualizar (Act)**



- **Diagrama de estado para los casos de usos de eliminar (Del)**



2.2 Implementación de los módulos del Software.

Este punto se centrará en algunas tecnologías, softwares utilizados y detalles esenciales en la implementación de cada uno de los módulos que forman el software.

2.2.1 Eclipse

Eclipse es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Esta plataforma, típicamente ha sido usada para desarrollar entornos de desarrollo integrado (del inglés IDE).

El entornos de desarrollo integrado (IDE) de Eclipse emplea módulos (en inglés *plug-in*) para proporcionar toda su funcionalidad al frente de la plataforma de cliente rico, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas,

las necesite el usuario o no. Este mecanismo de módulos es una plataforma ligera para componentes de software. Adicionalmente a permitirle a Eclipse extenderse usando otros lenguajes de programación como son C/C++ y Python, permite a Eclipse trabajar con lenguajes para procesado de texto com Latex, aplicaciones en red como Telnet y Sistemas de gestión de bases de datos. La arquitectura plugin permite escribir cualquier extensión deseada en el ambiente, como sería la Gestión de configuración . Se provee soporte para Java y CVS en el SDK de Eclipse. Y no tiene por qué ser usado únicamente para soportar otros lenguajes de programación.

La definición que da el proyecto Eclipse acerca de su software es: "*una especie de herramienta universal - un IDE abierto y extensible para todo y nada en particular*".

En cuanto a las aplicaciones clientes, eclipse provee al programador con frameworks muy ricos para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de software, aplicaciones web, etc. Por ejemplo, GEF (Graphic Editing Framework - Framework para la edición gráfica) es un plugin de Eclipse para el desarrollo de editores visuales que pueden ir desde procesadores de texto wysiwyg hasta editores de diagramas UML, interfaces gráficas para el usuario (GUI), etc. Dado que los editores realizados con GEF "viven" dentro de Eclipse, además de poder ser usados conjuntamente con otros plugins, hacen uso de su interfaz gráfica personalizable y profesional.(actores varios n.d.)

2.2.1.1 Plug In PyDev

PyDev es un plug-in para Eclipse, el es un entorno de desarrollo integrado usado para programar en python, soportando descomposición de código y debug de forma gráfica.(autores varios n.d.)

2.2.2 Ejemplos de rutinas en Python usando Django

Definición de una clase en django, esta es la clase de componentes, observar como se establecen las relaciones con otras clases:

```
class C_Equipo(models.Model):
    pres_cup = models.DecimalField(editable=False, max_digits=9, decimal_places=2, help_text="miles de pesos")
    pres_cuc = models.DecimalField(editable=False, max_digits=9, decimal_places=2)
```

```
    estimado_cup = models.DecimalField(max_digits=9, decimal_places=2, help_text="Estimado a ejecutar al terminar el ano")
    estimado_cuc = models.DecimalField(max_digits=9, decimal_places=2, help_text="Estimado a ejecutar al terminar el ano")
    plan_cup = models.DecimalField(max_digits=9, decimal_places=2)
    plan_cuc = models.DecimalField(max_digits=9, decimal_places=2)
    ejecutado_cup = models.DecimalField(editable=False, max_digits=9, decimal_places=2, default=0)
    ejecutado_cuc = models.DecimalField(editable=False, max_digits=9, decimal_places=2, default=0)
    proyect_cup = models.DecimalField("Proyeccion CUP", blank=True, max_digits=9, decimal_places=2, default=0)
    proyect_cuc = models.DecimalField("Proyeccion CUC", blank=True, max_digits=9, decimal_places=2, default=0)
    ano_poste_cup = models.DecimalField(blank=True, max_digits=9, decimal_places=2, default=0)
    ano_poste_cuc = models.DecimalField(blank=True, max_digits=9, decimal_places=2, default=0)
    inversion = models.OneToOneField(Inversione)

    def save(self):
        from inversion.planificar.views import act_monto_inver
        self.pres_cup = self.plan_cup + self.proyect_cup + self.ano_poste_cup
        self.pres_cuc = self.plan_cuc + self.proyect_cuc + self.ano_poste_cuc
        super(C_Equipo, self).save()
        act_monto_inver(self.inversion_id)
```

```
    class Meta():
        verbose_name = ('Componente Equipos')
        verbose_name_plural = ('Componentes Equipos')
```

Ejemplo de uso de la meta clase Admin y de la meta clase Forms, perteneciente a inversiones:

```
class InversioneForms(forms.ModelForm):
    model = Inversione
    def clean_ano_termina(self):
        inicio = self.cleaned_data['ano_inicio']
        fin = self.cleaned_data['ano_termina']

        if fin < inicio:
            raise forms.ValidationError("Esta fecha tiene que ser mayor que la de inicio")
        return fin

    def clean_descrip_medio_ambiente(self):
        ma = self.cleaned_data['inver_medio_ambiente']
        dma = self.cleaned_data['descrip_medio_ambiente']

        if ma <> "N" and dma == "":
            raise forms.ValidationError("Este campo es obligatorio")
        if ma == "N" and dma <> "":
```

```
            raise forms.ValidationError("Este campo tiene que estar vacio, pues no es inversion del medio ambiente")


class InversioneAdmin(admin.ModelAdmin):
    form = InversioneForms
    list_display = ('nombre', 'monto_cup', 'monto_cuc')
    list_filter = ['subproyectos']
    radio_fields = {"materiales_mn":admin.VERTICAL, "inver_medio_ambiente":admin.VERTICAL}
admin.site.register(Inversione, InversioneAdmin)
```

2.2.3 Seguridad del sistema

Todo desarrollador Web necesita considerar la seguridad como un aspecto fundamental de la programación Web. Desafortunadamente, se da el caso de que implementar la seguridad es difícil -- los atacantes sólo necesitan encontrar una única vulnerabilidad, pero los defensores deben proteger todas y cada una.

Django intenta mitigar esta dificultad. Está diseñado para protegerte automáticamente de muchos de los errores de seguridad comunes que los nuevos (y aun los experimentados) desarrolladores Web cometen.

2.4 Conclusiones Parciales

A partir del diseño e implementación del software para la planificación y el control de inversiones del proyecto PENDOCUVE desarrollado en el presente trabajo se pueden arribar a las siguientes conclusiones:

- Se diseño una base de datos para el almacenamiento de la información que genera él proyecto.
- Se implementó los módulos de servicio, administración, planificación y control.
- Se creó una interfaz de desarrollo para el cliente usando tecnologías de desarrollo Web.

CAPÍTULO III. MANUAL DE USUARIOS

En este capítulo se describen las facilidades de uso para los usuarios, tanto directores, planificadores o controladores de PlaCoIn, el cual necesita requerimientos mínimos para su funcionamiento:

- Procesador Intel P III/700 MHz o superior.
- 128 Mb de memoria RAM.
- Sistema operativo Windows 95 o superior, Linux.
- Del lado del cliente puede usarse cualquier navegador web, se recomienda Firefox.

3.1 Instalación del Sistema

Para configurar PlaCoIn (django), previamente debe instalarse el servidor Web Apache con el módulo mod_python activado. Esto usualmente significa tener una directiva LoadModule en el archivo de configuración de Apache. Puede lucir parecido así:

```
LoadModule python_module /usr/lib/apache2/modules/mod_python.so
```

Luego, editar el archivo de configuración de Apache y agregar lo siguiente:

```
<Location "/">
        SetHandler python-program
        PythonHandler django.core.handlers.modpython
        SetEnv DJANGO_SETTINGS_MODULE inversion.settings
        PythonPath "['/ruta/al/proyecto', '/ruta/a/django'] + sys.path"
        PythonDebug On
</Location>
```

Aquí se debe tener en cuenta copiar el software en la raíz del servidor web de apache para que <Location "/"> tenga sentido de lo contrario habría que reemplazar “/” de <Location "/"> por la ruta donde esté el proyecto.

Luego se debe proceder a reiniciar el servidor Apache:

```
$ /etc/init.d/apache reload
```

3.2 Manual de Usuario

Inicio de una sesión de trabajo

La sesión de trabajo del software comienza mostrando su ventana de presentación, la cual se usa para autenticarse en la aplicación (véase figura 3.1).

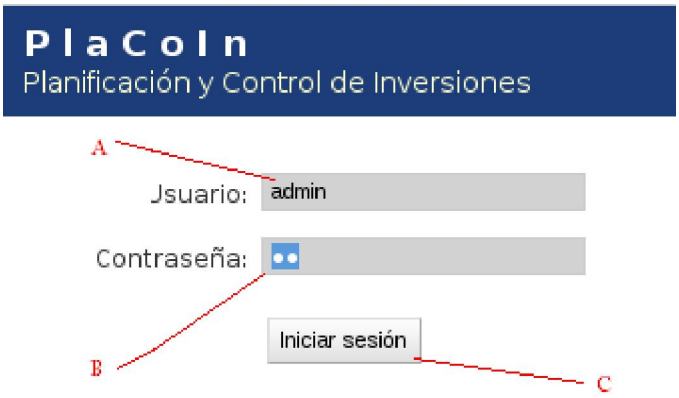


Figura 3. Ventana de Autenticación

Para acceder a las funcionalidades de la aplicación el usuario necesita identificarse:

A: el usuario introduce un nombre de usuario.

B: el usuario introduce su contraseña.

C: cuando el usuario presiona el botón “Iniciar sesión” el sistema carga la configuración adecuada para ese usuario, teniendo en cuenta los permisos asignados previamente a ese usuario.

3.2.1 Vistas del Administrador.

En la vista del administrador se pueden ejecutar las siguientes acciones mostradas en la figura 3.2:

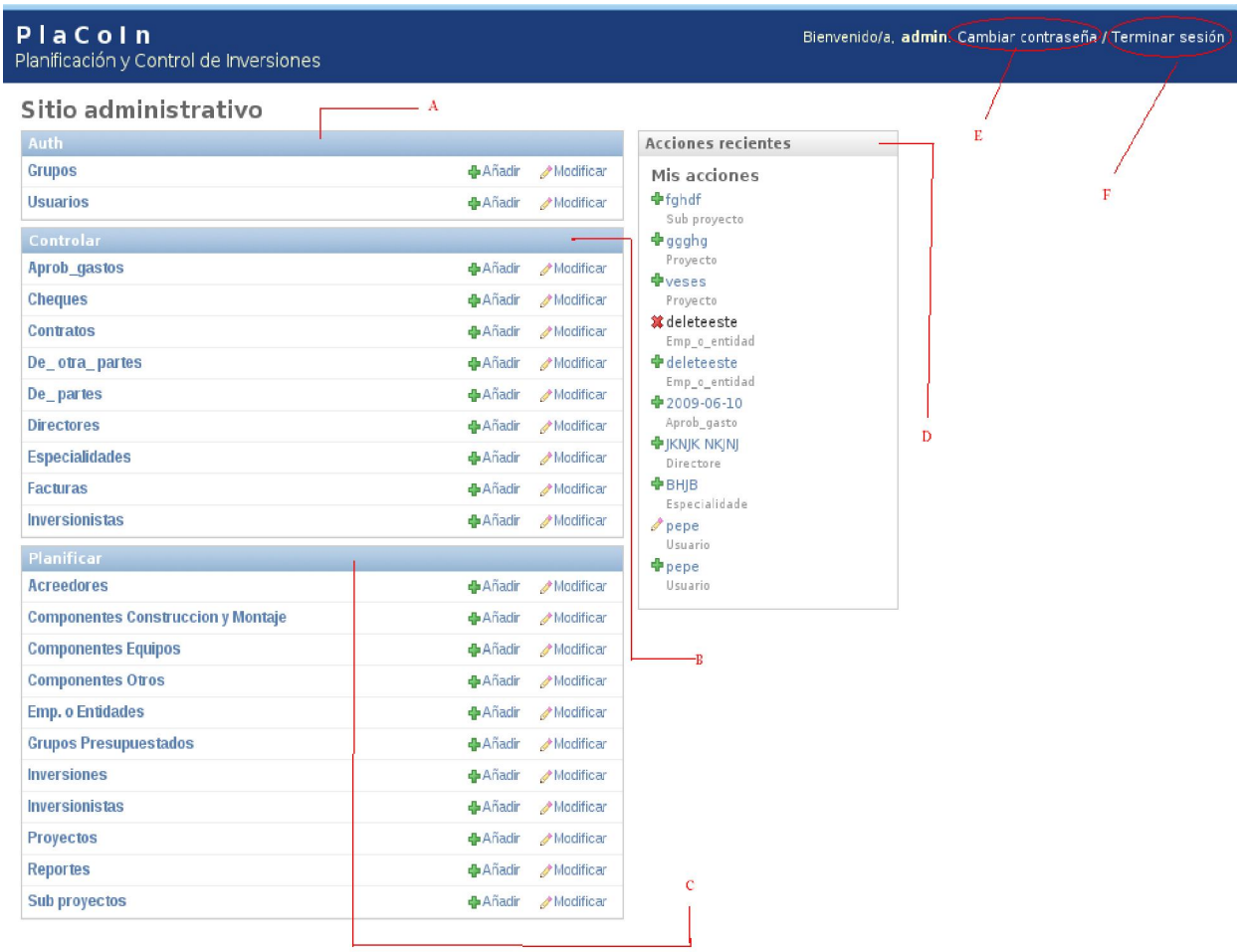


Figura 3.1 Vista del administrador

A: módulo donde se agrupan todas las acciones correspondientes a usuarios y grupos de usuarios, el administrador es el único capaz de manipular este módulo.

B: módulo o paquete "controlar" aquí se agrupan todas las acciones y objetos que están relacionados con el control de la inversión, el administrador es el usuario con privilegios para eliminar objetos en este módulo, al no ser que se especifique lo contrario manualmente.

C: módulo o paquete "planificar" aquí se agrupan todas las acciones y objetos que están relacionados con la planificación de la inversión, el administrador es el usuario con privilegios para eliminar objetos en este módulo, al no ser que se especifique lo contrario manualmente.

D: este es el panel de acciones frecuente, el cual puede ser usado por cualquier usuario del sistema para acceder de forma rápida hasta la decima acción realizada por el usuario operario, los objetos que son eliminados no pueden ser recuperados de ninguna forma, es por eso que el administrador es el único que goza de ese derecho.

E: todos los usuarios tienen pueden cambiar su contraseña cuando quieran.

F: terminar la sesión de trabajo, esto es importante realizar cada vez que se concluya el trabajo.

Como se explico previamente el administrador es el usuario con privilegios de poder tratar con usuarios y grupos de usuarios, como se muestra en la figura 3.3 para grupos de usuarios.

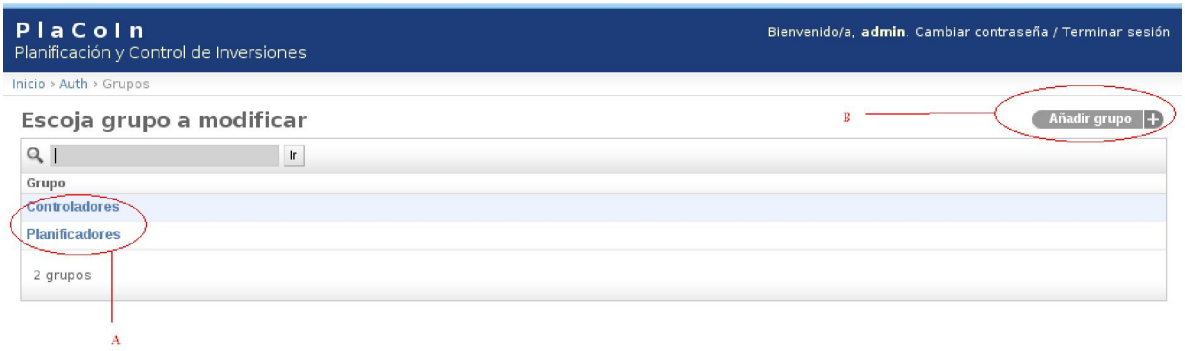


Figura 3.3 modificar grupo

A: lista de todos los grupos creados.

B: añadir un grupo nuevo al sistema, con sus respectivos permisos, como muestra la figura 3.4.

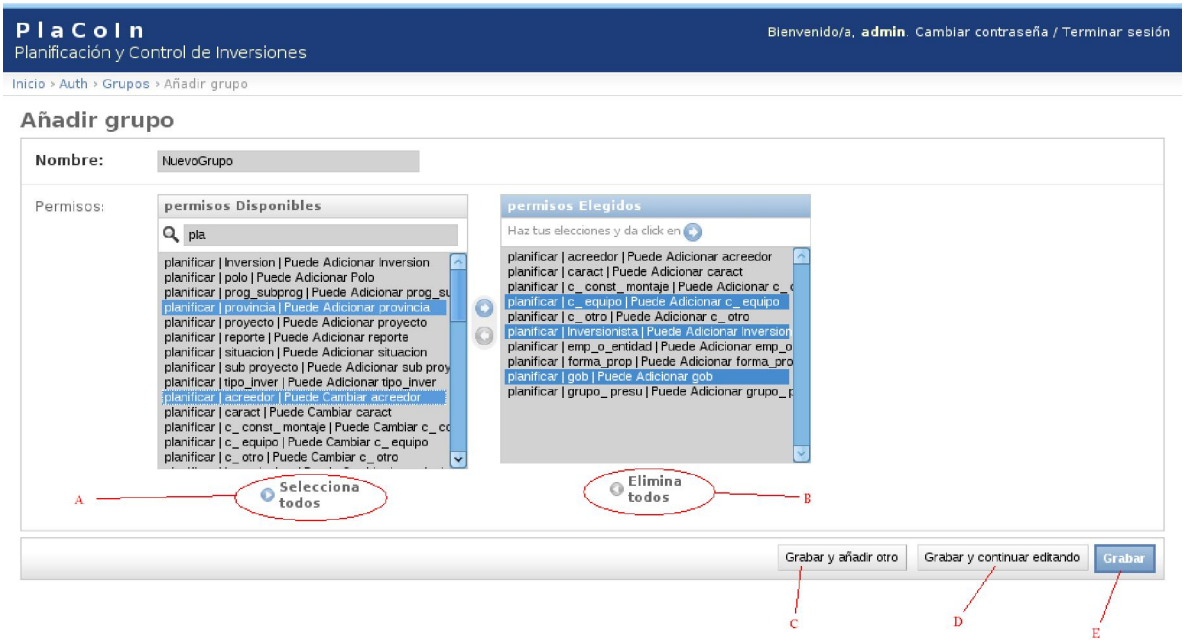


Figura 3.4 formulario para adicionar grupo

A: permite seleccionar todos los permisos dentro de la caja.

B: permite eliminar todos los permisos seleccionados

C: la opción “Grabar y añadir otro”, adiciona el nuevo grupo y muestra el formulario en blanco nuevamente para que el usuario adicione otro nuevo grupo.

D: la opción “Grabar y continuar editando”, permite al usuario salvar sus cambios y continuar con la vista del propio formulario para continuar editando.

E: la “Grabar”, permite al usuario salvar el nuevo objeto y regresar a la vista donde se muestra la lista de todos los objetos.

De igual manera para el tratado de los usuarios como se muestra en la figura 3.5.

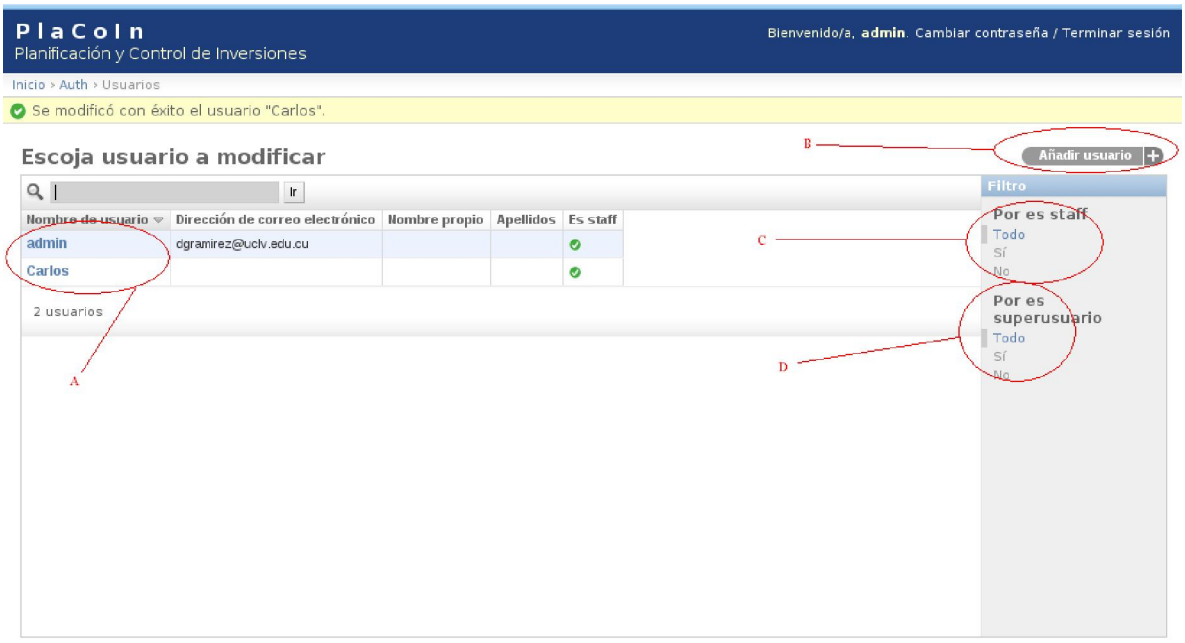


Figura 3.5 control de usuarios

A: lista de todos los usuarios creados en el sistema. Se puede acceder algún usuario específico para su modificación.

B: adicionar un nuevo usuario.

C: filtrar los objetos según las variables especificadas, en este caso filtrar por los usuarios que son staff en el sistema.

D: de igual manera, si el usuario es súper usuario, también puede filtrar.

Cuando se da clic en el la pestaña de adicionar un nuevo usuario se muestra un formulario para introducir la creación del mismo, como se muestra en la figura 3.6

Figura 3.6 crear un usuario

A: introducir datos esenciales para la creación del usuario.

Luego usted puede enriquecer más los datos del usuario creado adicionando más información del mismo en el siguiente formulario que se muestra una vez oprimido el botón grabar del formulario anterior, ver figura 3.7. En el formulario que se muestra en la figura 3.7 perteneciente como se acaba de decir al completamiento de la creación del usuario se ven datos como son nombre, apellidos, si estará activo, a que grupo de usuarios pertenece, y se le puede adicionar permisos adicionales que no estén en el grupo, entre otras opciones.

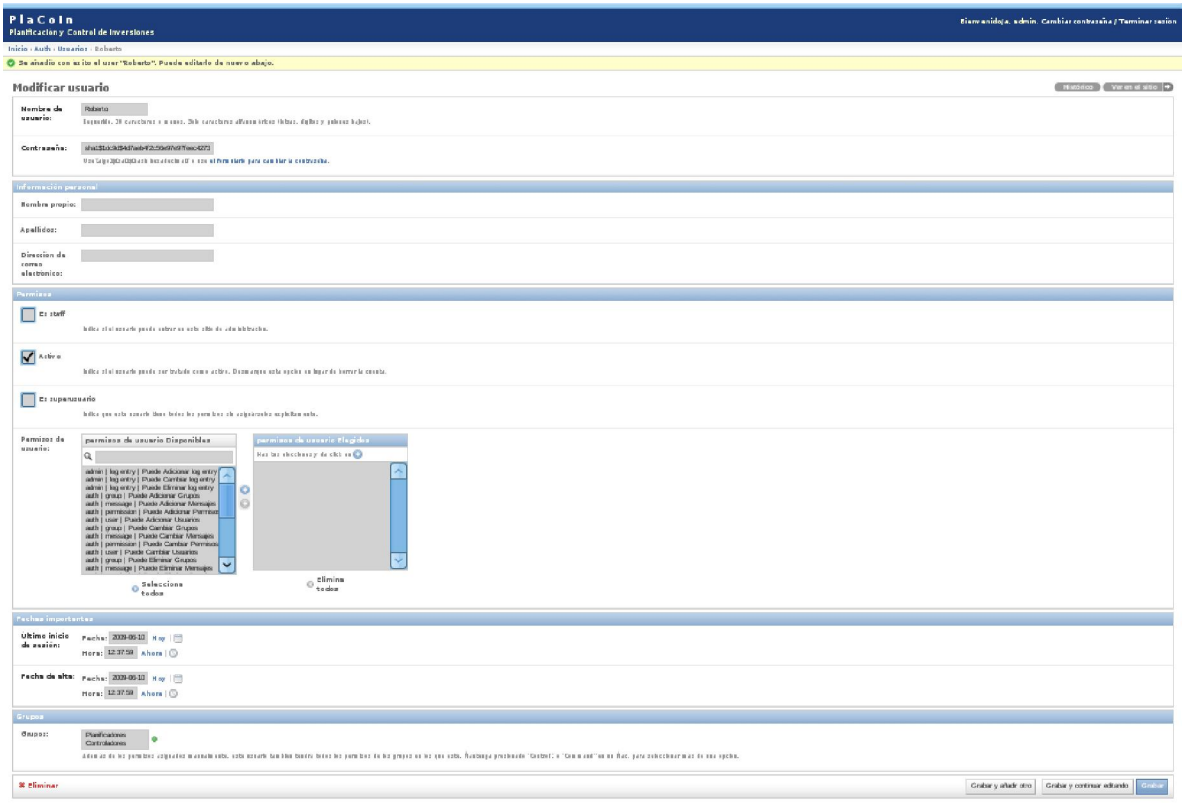


Figura 3.7 crear usuarios

Como se había mencionado previamente los usuarios con privilegios son los únicos capaces de poder eliminar objetos del sistema, al no ser que se especifique lo contrario, ver figura 3.8.

Figura 3.8 Eliminar un objeto

A: clic aquí para eliminar un objeto determinado.

B: esta opción le permite a los usuarios que tengan permisos para ejecutarla de ver el historial de todos lo objetos del sistema, mostrando el comportamiento de los usuarios con ese objeto, indicando cuando se ha modificado y quien lo ha modificado. Ver figura 3.9.

Figura 3.9 Histórico

Cuando se va a eliminar un objeto se eliminara todos los demás objetos que dependan de el, cumpliendo la integridad referencial, ver figura 3.10. El sistema le hará una pregunta de verificación y le mostrará todos los objetos que se eliminaran en ese momento.

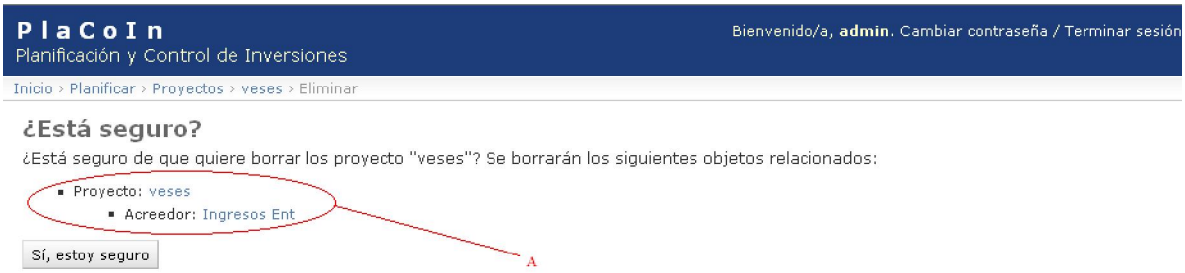


Figura 3.10 Eliminación en cascada

A: objetos a eliminar en cascada con la eliminación de un objeto X.

Todos los usuarios tienen el derecho de cambiar su contraseña cuando ellos deseen, dando clic en el vinculo que se muestra en la figura 3.2 el sistema le mostrará un formulario para el cambio de la contraseña, ver figura 3.11.

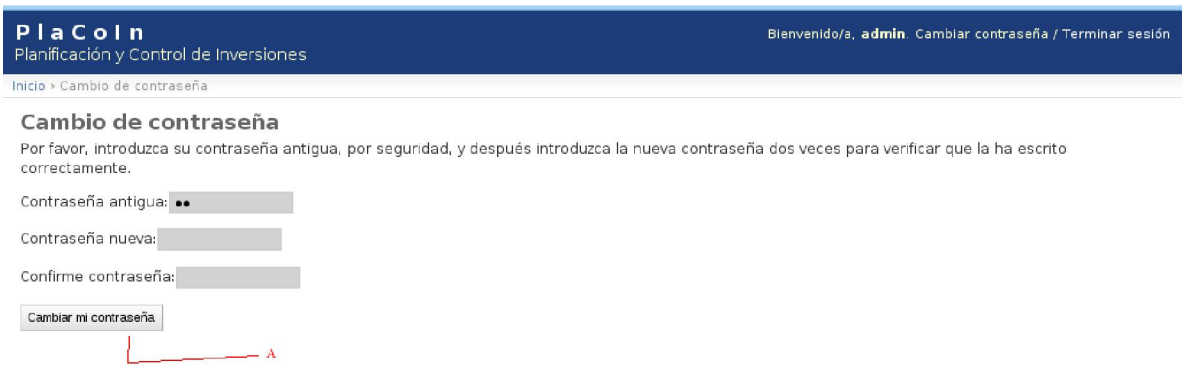


Figura 2.11 Cambiar contraseña

A: si el chequeo que realiza el sistema con las contraseñas es satisfactorio en sistema le cambiara la contraseña al usuario en cuestión.

3.2.2 Vistas del Planificador

Esta vista será para todos los usuarios del sistema que pertenezcan al grupo de usuario de los planificadores, luce como la vista del administrador pero solo será capaz de ver los objetos que estén relacionados con la planificación como se ve en la figura 3.12 mostrada a continuación.

Figura 3.12 Vista del planificador

La estructura de trabajo es la misma que en las vistas del administrador solo se diferencian en la funcionalidad, en la figura 3.13 se muestra un listado de los proyectos donde el usuario puede seleccionar uno para su edición.

Figura 3.13 Proyectos

Cuando se selecciona un proyecto u otro objeto se muestra un formulario con todos los valores previo de ese objeto listo para la edición, ver figura 3.14.

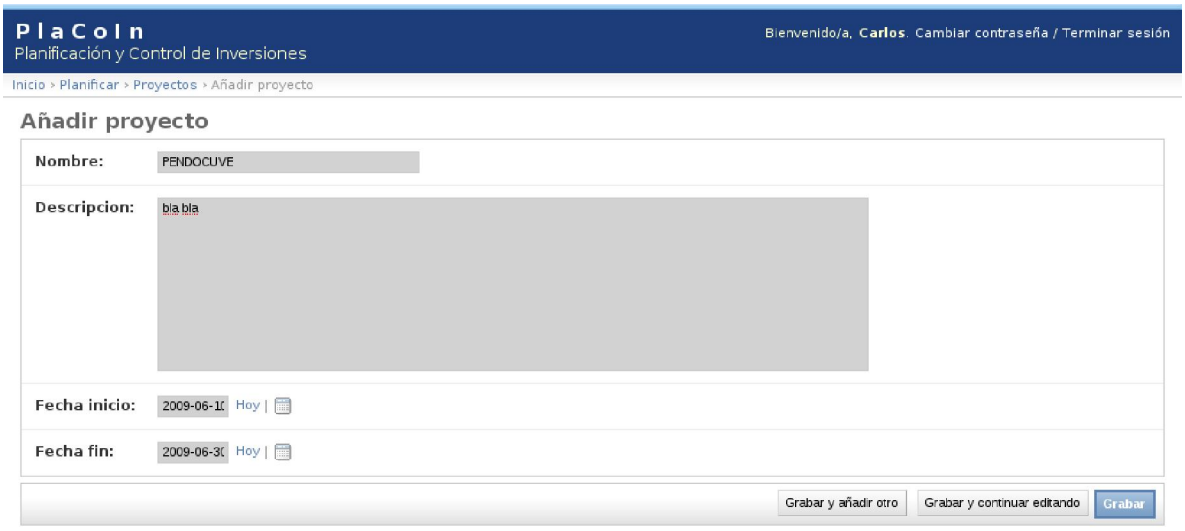


Figura 3.14 Edición de un proyecto

Los filtros de búsqueda son usados también por los usuarios planificadores como se muestra en la figura 3.15 donde se tiene un listado de todo los sub proyectos y al lado un filtro por proyectos para mostrar todos los sub proyectos de un proyecto dado.

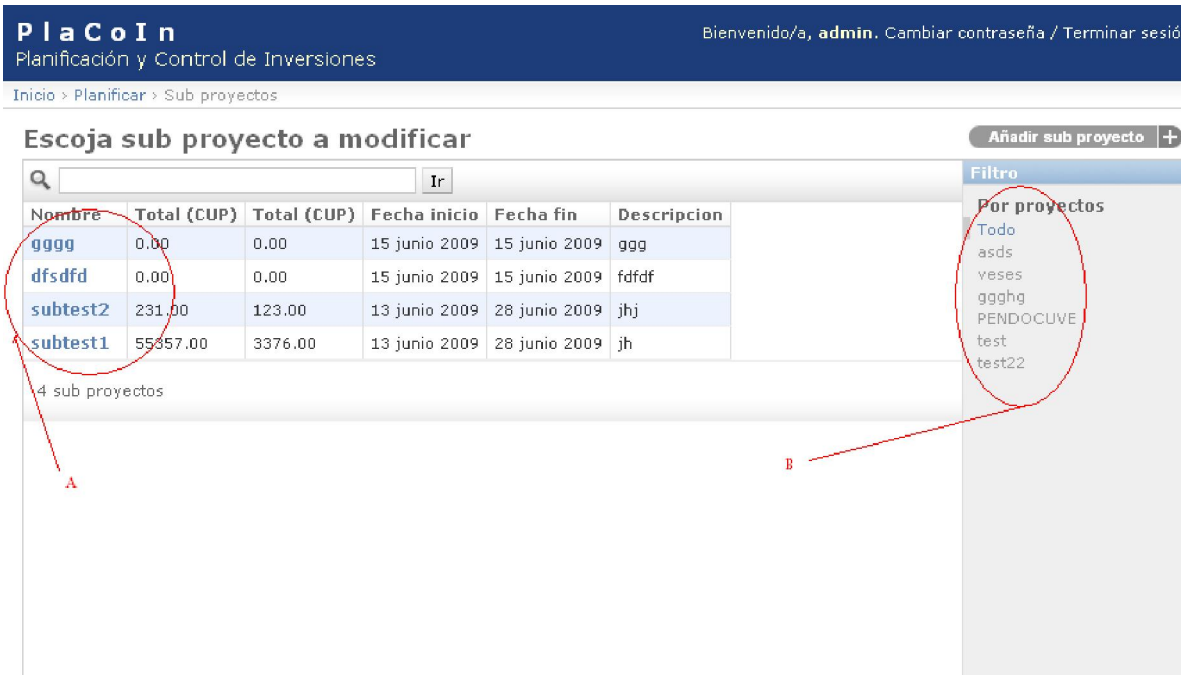


Figura 3.15 Sub proyectos

A: listado de sub proyectos

B: filtrado por proyectos.

Como aquí los objetos están relacionados, por ejemplos proyectos con sub proyectos en los formularios de creación de algunos objetos aparecen los listados de los objetos a relacional, ver figura 3.16.

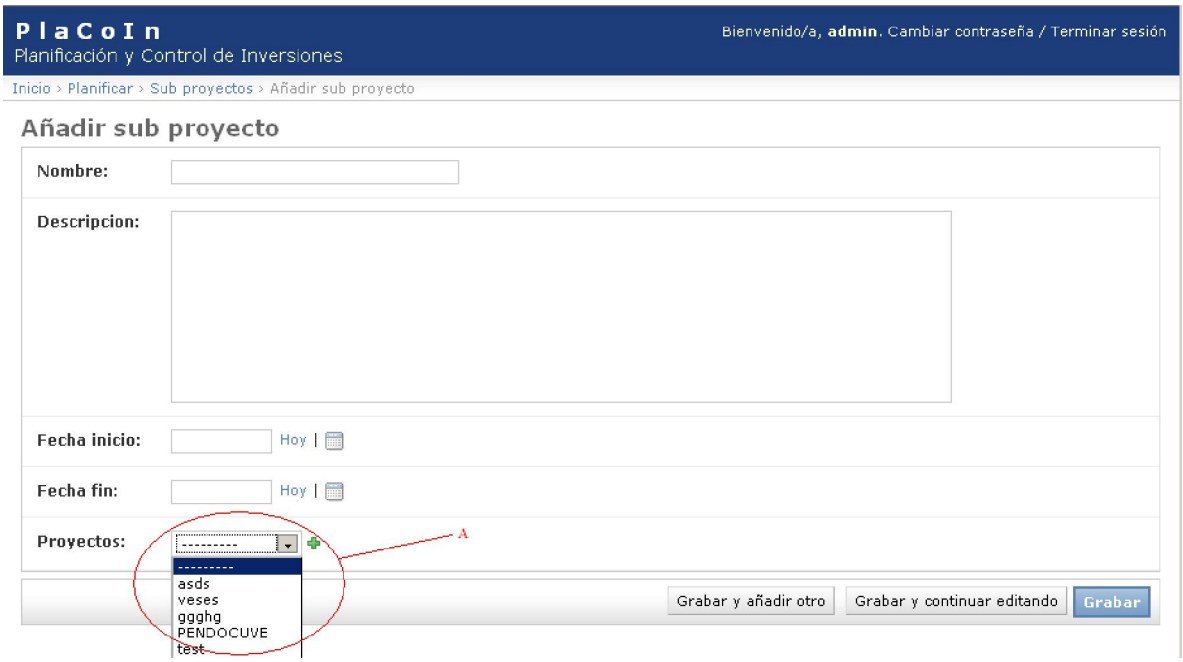


Figura 3.16 Crear sub proyectos

A: proyectos con quien se relacionará el sub proyecto a crear.

Otro ejemplo de filtros esta en las inversiones, ver figura 3.17

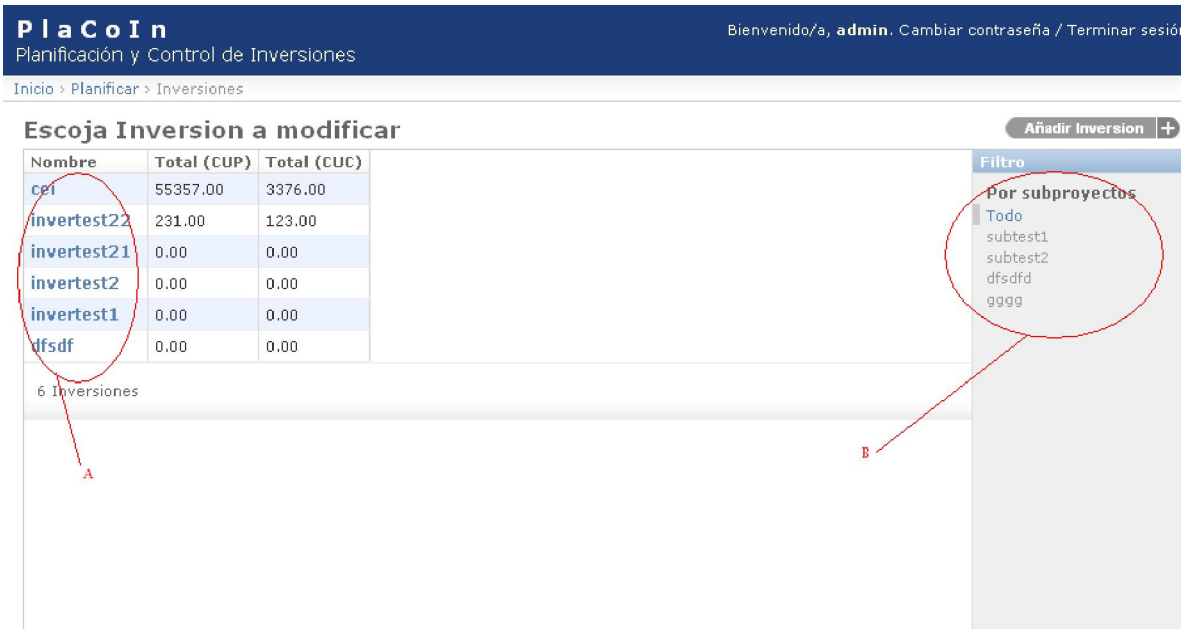


Figura 3.17 Inversiones

Vista del componente construcción y montaje (mantiene la misma estructura):

Figura 3.18 Construcción y montaje

Formulario para adicionar un componente, en este caso componente de construcción y montaje los demás componentes son análogos a este:

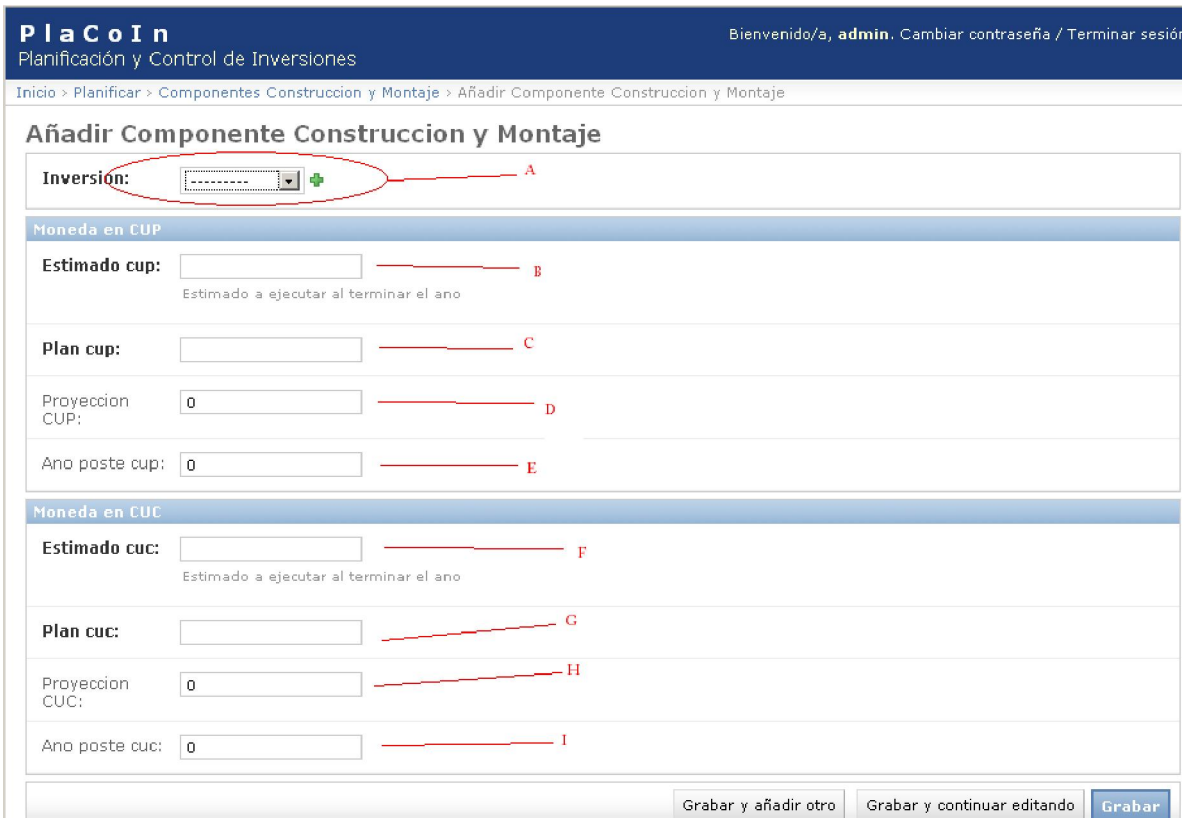


Figura 3.19 Adicionar componente

A: seleccionar a que inversión va a pertenecer ese componente

B: valor estimado en CUP para el año que se planifica

C: valor del plan en CUP para el año que se planifica

D: valor de la proyección en CUP, este valor no es obligatorio, puede convertirse en un campo calculado

E: este campo puede también ser opcional, pues de no ponerse nada se determina su valor.

F, G, H, I: son igual que del A-D pero para valores en CUC.

Formulario para añadir grupos presupuestados:

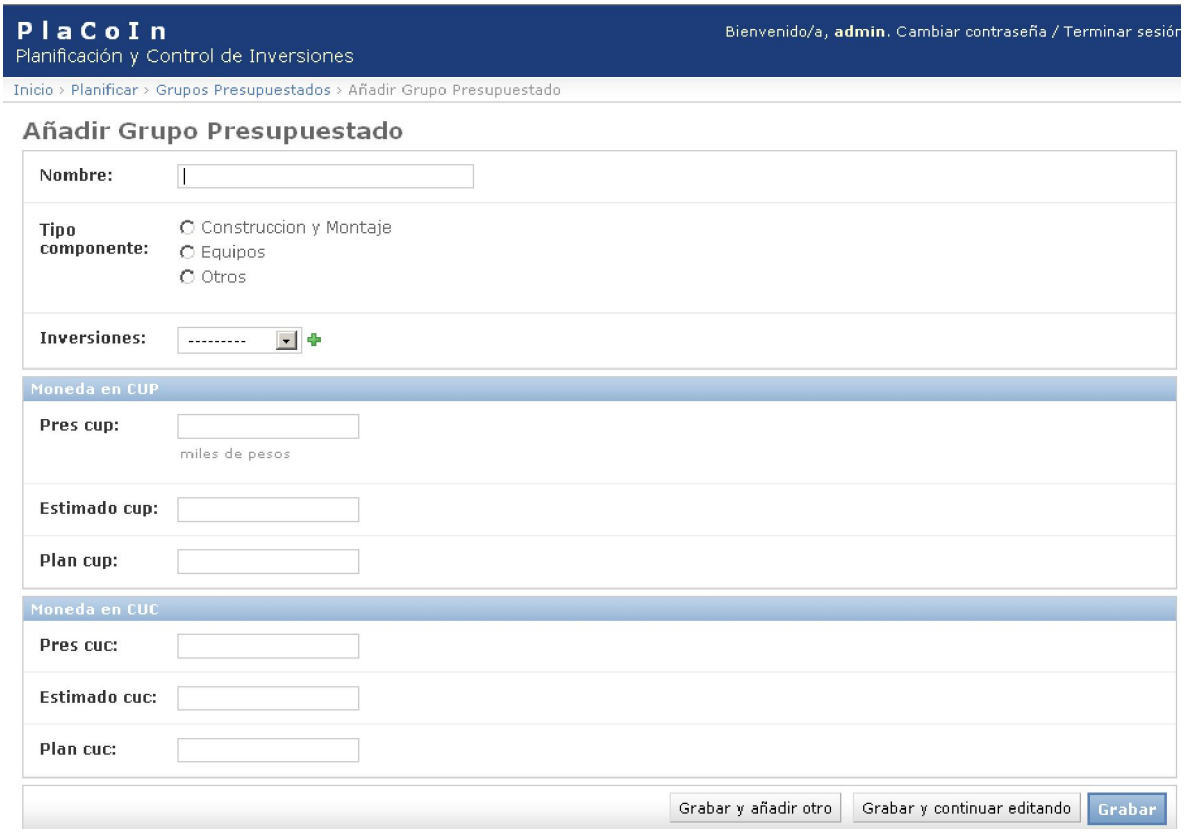


Figura 3.20 adicionar grupo presupuestado

3.2.3 Vistas del controlador

De igual manera que las anteriores vistas, para el controlador se mantiene la misma estructura, ver figura 3.21.

Figura 3.21 Vista controlar

Al igual que las listas de objetos y los filtros:

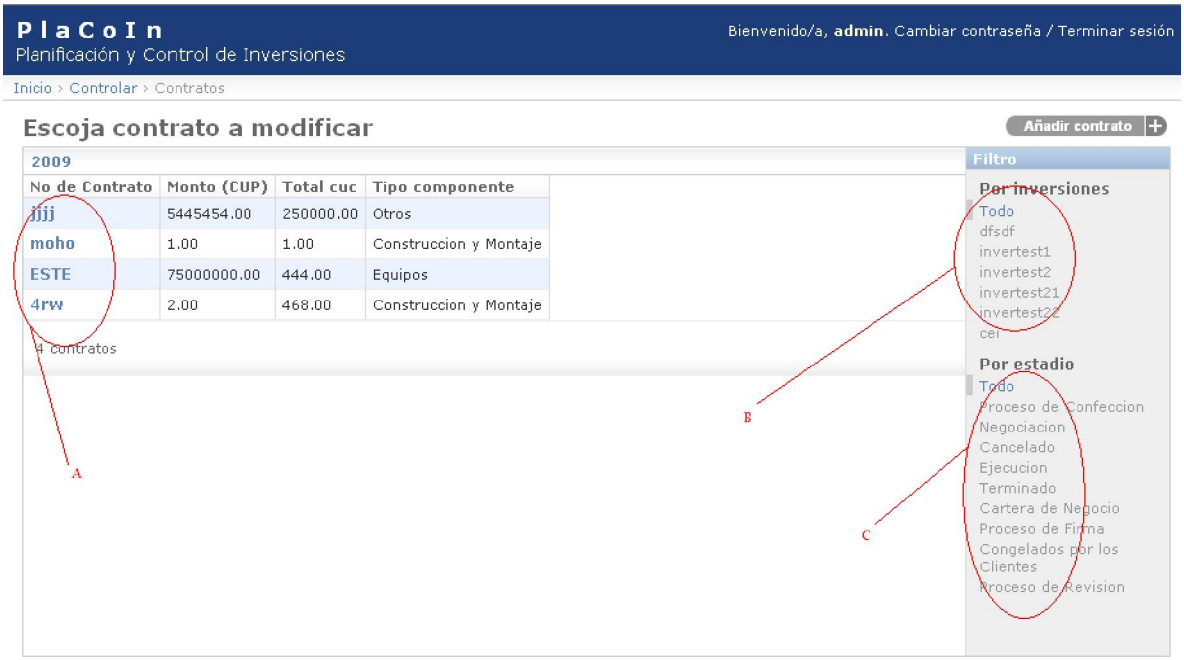


Figura 3.22 Contratos

Los formularios cuentan con campos que son de tipo fecha y estos vienen de la mano de un calendario en javascript para facilitar la puesta de la fecha, además existen campos que se puede adicionar en el momento que se va a utilizar, ver figura 3.23.

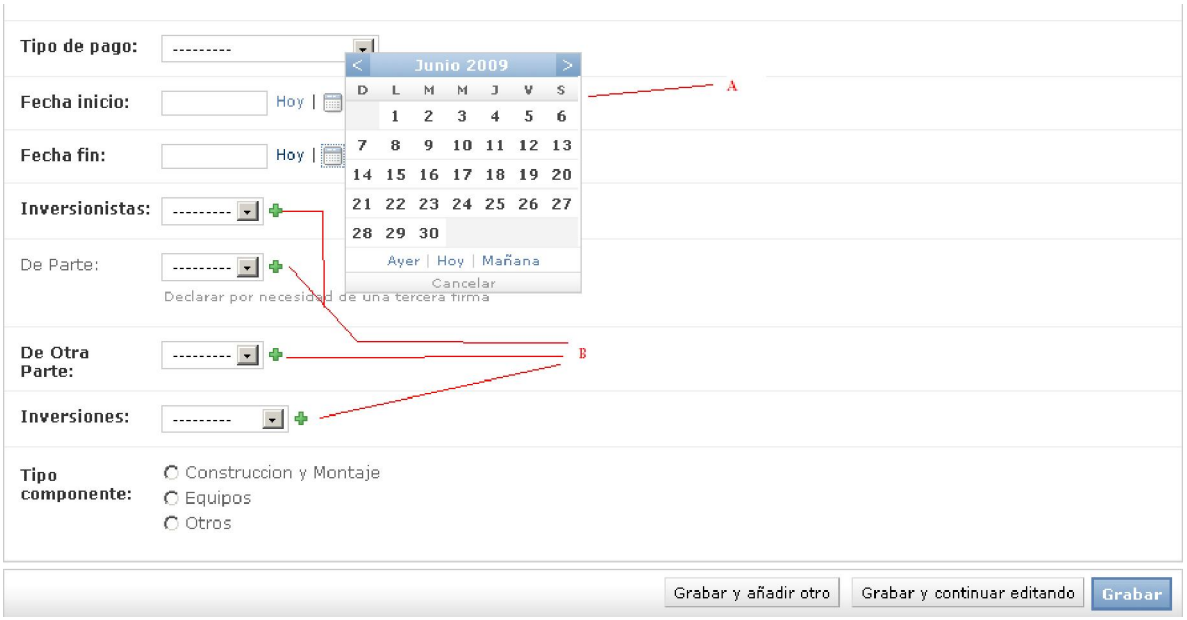


Figura 3.23 Fragmento del formulario de inversión

A: calendario para ubicar de forma eficaz la hora y fecha correspondiente.

B: iconos de acceso a otros formularios para la creación de esos campos.

CONCLUSIONES

Como resultado de esta investigación se desarrolló un software plataforma Web que le permite planificar y controlar las inversiones del proyecto PENDOCUVE, de forma tal que el usuario final tenga la facilidad de definir inversiones para su posterior control y un mejor manejo de la información, cumpliéndose el objetivo general planteado y concluyéndose que:

- Se realizó un diagnostico de la información que se requería para la planificación y el control de inversiones del proyecto PENDOCUVE
- Se elaboró un estudio de tecnologías "libres" disponibles para la elaboración de un software, específicamente orientado al desarrollo Web.
- Se diseñó los módulos de software correspondientes que automatizan la planificación y el control de inversiones.
- Se implementaron los módulos de software que automatizan la planificación y el control de inversiones.

RECOMENDACIONES

Teniendo en consideración que el modelo propuesto es extensible se recomienda:

- Incorporar a PlaCoIn otras informaciones a controlar en los módulos de planificación y control, para extenderlo a otros proyectos.
- Implementar otras funciones para el manejo de las inversiones.
- Introducir el concepto del control por suministro para las inversiones.

BIBLIOGRAFÍA

actores varios, Framework. In WikiPedia. Available at: http://es.wikipedia.org/wiki/Framework.

actores varios, UML. In *WikiPedia*. Available at: http://es.wikipedia.org/wiki/UML.

Adrian Holovaty, 2007. *The Django Book* 1st ed.,

Adrian Holovaty & Jacob Kaplan-Moss, 2008. *El Libro de Django*,

autores varios, python. In *Wikipedia*. Available at: http://es.wikipedia.org/wiki/Python.

Detron HB and Monty Program KB, 2000. *MySQL Reference Manual* 3rd ed.,

Grady BOOCH, James RUMBAUGH & Ivar JACOBSON, *El Lenguaje Unificado de Modelado*,

Hans Petter Langtangen, *Python Scripting for Computational Science* 3rd ed.,

Simon Willison, 2005. An introduction to Django. Available at: http://simon.incutio.com/.

Yenier Cruz Brito, 2007. *SEPADMEX: Framework para el descubrimiento y descripción de los elementos de SEPAD.* Universidad Central de las Villas.

Taily Galindo, Misión Vuelvan Caras. , 14.

actores varios, Eclipse. In *WikiPedia*. Available at: http://es.wikipedia.org/wiki/Eclipse_(software).

autores varios, PyDev. In *WikiPedia*. Available at: http://en.wikipedia.org/wiki/PyDev.

Luke Welling & Laura Thomson, Programación. Desarrollo Web con PHP y MySQL.

José Antonio Gallego Vásquez. Guía prática para usuarios. Desarrollo Web con PHP y MySQL.

ANEXOS

LA COMISIÓN ORGANIZADORA DE LA XXXVIII JORNADA CIENTÍFICA ESTUDIANTIL UNIVERSITARIA

OTORGA A:

Dagron R. González Ramírez.

El PRESENTE

DIPLOMA DE PREMIACIÓN

RELEVANTE

27 de Mayo del 2009.
Universidad Central "Marta Abreu" de Las Villas

Presidente Comisión

Diploma de premiación

FACULTAD MATEMÁTICA-FÍSICA-COMPUTACIÓN

UNIVERSIDAD CENTRAL "MARTA ABREU" DE LAS VILLAS

DIPLOMA

A: Dayron R. González Ramírez

Por haber obtenido el **PREMIO RELEVANTE** en la
JORNADA CIENTÍFICO ESTUDIANTIL
con el trabajo titulado

Software para la planificación y control de inversiones en el proyecto endógeno Cuba – Venezuela

Comisión: Programación e Ingeniería de Software

Otorgado el 6 de mayo de 2009 en
la Facultad Matemática-Física-Computación,
Universidad Central "Marta Abreu" de Las Villas

Dra. Leticia Arco García
Vicedecana de Investigación y Postgrado

Dra. Yanet Rodríguez Sarabia
Decana Facultad MFC

Diploma de premiación