UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

TEMA

"Sistema Integrado de Información Financiera para la Cooperativa de Ahorro y Crédito Unión Cochapamba"

APLICATIVO

"Módulo de Contabilidad para la Cooperativa de Ahorro y Crédito Unión Cochapamba"

AUTOR: EDGAR DANIEL PICUASI DUQUE.

DIRECTOR: DRA. CPA. MARÍA DE LA PORTILLA VERA MBA.

IBARRA – ECUADOR
2012

CERTIFICACIÓN

Certifico que la Tesis: "**SISTEMA INTEGRADO DE INFORMACIÓN FINANCIERA PARA LA COOPERATIVA DE AHORRO Y CRÉDITO UNIÓN COCHAPAMBA"** con el Aplicativo **"MÓDULO DE CONTABILIDAD"** ha sido realizada en su totalidad por el Sr. Edgar Daniel Picuasi Duque; portador de la cédula de identidad número: 100295498-8, bajo mi supervisión para lo cual firmo en constancia.

……..…….……

DRA. CPA. María De la Portilla Vera MBA.

DIRECTOR DE PROYECTO

CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE GRADO A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

Yo, **EDGAR DANIEL PICUASI DUQUE**, con cédula de identidad Nro100295498-8, manifiesto mi voluntad de ceder a la Universidad Técnica del Norte los derechos patrimoniales consagrados en la ley de propiedad intelectual del Ecuador, articulo 4, 5 y 6, en calidad de autor del trabajo de grado denominado: **"SISTEMA INTEGRADO DE INFORMACIÓN FINANCIERA PARA LA COOPERATIVA DE AHORRO Y CRÉDITO UNIÓN COCHAPAMBA"** con el aplicativo **"MÓDULO DE CONTABILIDAD"**, que ha sido desarrollada para optar por el título de Ingeniería en Sistemas Computacionales, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En mi condición de autor me reservo los derechos morales de la obra antes mencionada, aclarando que el trabajo aquí descrito es de mi autoría y que no ha sido previamente presentado para ningún grado o calificación profesional.

En concordancia suscribo este documento en el momento que hago entrega del trabajo final en formato impreso y digital a la biblioteca de la Universidad Técnica del Norte

……..

Nombre: Edgar Daniel Picuasi Duque.
Cédula: 100295498-8
Ibarra, a los 17 días del mes de Julio del 2012

BIBLIOTECA UNIVERSITARIA AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

La UNIVERSIDAD TÉCNICA DEL NORTE dentro del proyecto Repositorio Digital institucional determina la necesidad de disponer los textos completos de forma digital con la finalidad de apoyar los procesos de investigación, docencia y extensión de la universidad.

Por medio del presente documento dejo sentada mi voluntad de participar en este proyecto, para lo cual ponemos a disposición la siguiente investigación:

DATOS DE CONTACTO	
CEDULA DE IDENTIDAD	100295498-8
APELLIDOS Y NOMBRES	EDGAR DANIEL PICUASI DUQUE
DIRECCIÓN	AV. GUILLERMO G. UBIDIA – OTAVALO
EMAIL	danyk_lego@hotmail.com
TELÉFONO MOVIL	085112310

DATOS DE LA OBRA	
TITULO	"SISTEMA INTEGRADO DE INFORMACIÓN FINANCIERA PARA LA COOPERATIVA DE AHORRO Y CRÉDITO UNIÓN COCHAPAMBA" – MÓDULO DE CONTABILIDAD
AUTOR	EDGAR DANIEL PICUASI DUQUE
FECHA	17 DE JULIO DEL 2012
PROGRAMA	PREGRADO
TITULO POR EL QUE	INGENIERÍA EN SISTEMAS COMPUTACIONALES
DIRECTOR	DRA. CPA. MARÍA DE LA PORTILLA VERA MBA.

2. AUTORIZACIÓN DE USO A FAVOR DE LA UNIVERSIDAD

Yo, Edgar Daniel Picuasi Duque, con cédula de identidad Nro.100295498-8, en calidad de autor y titular de los derechos patrimoniales del trabajo de grado descrito anteriormente, hago entrega del ejemplar respectivo en forma digital y autorizo a la Universidad Técnica del Norte, la publicación de la obra en el Repositorio Digital Institucional y el uso del archivo digital en la biblioteca de la universidad con fines académicos, para ampliar la disponibilidad del material y como apoyo a la educación, investigación y extensión, en concordancia con la Ley de Educación Superior Artículo 143.

……..

Nombre: Edgar Daniel Picuasi Duque.
Cédula: 100295498-8
Ibarra, a los 17 días del mes de Julio del 2012

DEDICATORIA

A mis padres José Luis Picuasi y Matilde Duque por su comprensión y ayuda incondicional.

A mi madre en especial porque a pesar de las dificultades y lo dura que ha sido la vida para nosotros, ha sabido mantener a nuestra familia unidad y ser un ejemplo a seguir.

A mis hermanos Juan, Tocayo y Tamia a quienes quiero y admiro mucho, por ser personas de bien, por todos sus concejos y la confianza que depositaron en mí.

A toda mi familia y amigos quienes siempre me alentaron y motivaron a culminar mi carrera universitaria.

A todos mis compañeros de aulas quienes demostraron amistad sincera y formaron parte de mi desarrollo como ser humano y como profesional.

Edgar Picuasi Duque…

AGRADECIMIENTOS

A Dios por sobre todo, a él por regalarme salud, cuidarme, guiar mi camino y mantener unida a mi familia.

A todos los docentes de la facultad quienes han sabido poner lo mejor de sus conocimientos a nuestro alcance.

A todos aquellos amigos y familiares que de alguna manera u otra apoyaron mi carrera universitaria y confiaron en mí.

Edgar Picuasi Duque...

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El presente documento detalla todos los procesos que se siguieron para el desarrollo del Sistema Integrado de Información Financiera y en particular la implementación del **Módulo Contable** para la cooperativa de ahorro y crédito "Unión Cochapamba".

En la elaboración y desarrollo del Sistema Integrado de Información Financiera, el documento presenta cinco capítulos, en cada uno de ellos describe los procesos y metodologías utilizados en el transcurso de la elaboración del sistema.

El Sistema Integrado de Información Financiera consta de siete módulos: Módulo Administración y Seguridad, Módulo de Ahorros, Módulo Cajero, **Módulo Contabilidad**, Módulo Cartera, Módulo Clientes, Módulo Auditoría.

En el capítulo tres se describe el funcionamiento y algunos de aspectos generales del Módulo Contable motivo del presente documento.

SUMMARY

This document details all the processes that were followed for development of the Integrated Financial Information System and in particular the implementation of the **Accounting Module** for the credit union's "Cochapamba Union"
In the design and development of the Integrated Financial Information System, the document has five chapters.

In each of these describes the processes and methods used during system development. The Integrated Financial Information System consists of seven modules: Administration and Security Module, Savings, Cash Module, **Accounting Module**, Module Portfolio Client Module, Audit Module.

Chapter three describes the operation and some general aspects of the Accounting Module occasion of this document.

TEMA

"Sistema Integrado de Información Financiera para la Cooperativa de Ahorro y Crédito Unión Cochapamba"

APLICATIVO

"Módulo de Contabilidad para la Cooperativa de Ahorro y Crédito Unión Cochapamba"

PROBLEMA

La Cooperativa de Ahorro y Crédito Unión Cochapamba maneja todos los procesos contables con la ayuda de un sistema informático desactualizado e inseguro, lo cual provoca un flujo de información contable poco fiable y ocasiona retrasos en el plazo de entrega de información solicitada.

JUSTIFICACIÓN

El presente estudio de factibilidad se realiza para determinar la viabilidad del proyecto, el cual tiene como finalidad la automatización de los procesos contables de la Cooperativa de Ahorro y Crédito Unión Cochapamba, esto con el objetivo de agilizar los procesos para brindar un mejor servicio a sus clientes en general.

Mediante visitas realizadas a la Cooperativa se observó la falta de un sistema que les permita a ellos como institución agilizar y automatizar los procesos que hasta ahora se los ha realizado de manera incorrecta, además no cuenta con las respectivas políticas de seguridad, dado que en la Cooperativa no existe separación de funciones, es decir no hay un profesional asignado a cada área, esto hace que la información sea vulnerable a manipulaciones por parte de cualquier persona y con ello propensa a fraudes de todo tipo, al implementar este proyecto se pretende reducir el tiempo de respuesta de las diferentes instancias y oficinas de la Cooperativa.

OBJETIVO GENERAL

Analizar, Diseñar e Implementar el Módulo de Contabilidad para el Sistema Integrado de Información Financiera de la Cooperativa de Ahorro y Crédito "Unión Cochapamba" mediante el uso de herramientas libres y estándares abiertos.

OBJETIVOS ESPECÍFICOS

1. Analizar y establecer los requisitos para la arquitectura del módulo contable.
2. Estudiar las necesidades de la cooperativa en relación a la información contable generada.
3. Revisar reformas contables, en especial aquellas que son dirigidas a cooperativas.
4. Documentar la información y los procesos generados al desarrollar el módulo contable.
5. Investigar y establecer los procesos financieros que realiza la Cooperativa de Ahorro y Crédito Unión Cochapamba.
6. Investigar y estudiar el uso de herramientas libres apropiadas para utilizar en el desarrollo del proyecto.
7. Analizar las ventajas y desventajas que se obtiene al implementar sistemas web utilizando herramientas libres y estándares abiertos.

Capítulo I: Introducción

CONTENIDO:

1. Base legal de la Institución.
2. Situación actual diagnóstico.
3. Estructura de la entidad.
4. Información General.

1. Introducción

1.1. Base legal de la Institución

1.1.1. Constitución y Domicilio

La Cooperativa de Ahorro y Crédito "Unión Cochapamba" en adelante CUC con domicilio en la parroquia de Ambuquí, Comunidad de Chaupi Guaranguí, Cantón Ibarra de la Provincia de Imbabura, se constituye como una cooperativa sin fines de lucro y se regirá por la ley de Cooperativas, su Reglamento General, por otras leyes que fueren aplicables y por un estatuto único aprobado por los directivos pertinentes.

La responsabilidad de la Cooperativa ante terceros, está limitada a su capital social y al capital que hubieren suscrito personal en la entidad.

La Cooperativa tendrá duración indefinida pero sin embargo podrá disolverse y liquidarse por las causales y en forma establecida en la ley de Cooperativas, su Reglamento General y las normas señaladas en el estatuto.

1.1.2. Objetivos de la Cooperativa

Los objetivos de la Cooperativa son los siguientes:

1. Promover la cooperación económica entre sus socios, para cuyo cumplimiento recibirá los ahorros, certificados de aportación y depósitos que estos realicen, efectuará cobros y pagos así como todas aquellas operaciones necesarias para el fortalecimiento de la cooperativa crediticia dentro del marco legal permitido por las cooperativas.
2. Otorgar préstamos a sus miembros de conformidad al reglamento que para el efecto se establezca.
3. Proporcionar a sus asociados mayor capacidad en lo económico y social, mediante una adecuada educación cooperativista.
4. Establecer nexos dentro y fuera del país, en beneficio de la cooperativa.
5. Obtener fuentes de financiamiento interno y externo para el desarrollo de la institución.
6. Establecer otros servicios y otras actividades que estén encuadradas en la ley y Reglamento de Cooperativas y otras leyes que sean aplicables, que conlleven al mejoramiento social y económico de sus miembros.
7. Efectuar actividades inherentes a buscar financiamiento con la finalidad de capitalizar y otorgar más beneficios a los asociados.

1.1.3. Principios de la Cooperativa

La cooperativa regulara sus actividades, de conformidad y de acuerdo con los siguientes principios:

1. Igualdad de obligaciones y derechos de los socios que llenen los requisitos establecidos en la Ley de Cooperativas, su Reglamento General y el Estatuto.
2. Adhesión y retiro voluntario.
3. Control democrático, un socio es un voto.
4. Distribución de los excedentes económicos entre los socios en proporción a las operaciones o al trabajo efectuado por los socios de la cooperativa.
5. Interés limitado sobre los certificados de aportación que en ningún caso será mayor al señalado por la Ley o por el organismo estatal competente.
6. Neutralidad política y religiosa.
7. Fomento de la educación cooperativa.
8. Integración cooperativista.

1.1.4. Socios Fundadores

La Cooperativa, inicio sus actividades con la colaboración de 42 socios fundadores, todos ellos domiciliados en la comunidad de Chaupi Guaranguí, Parroquia de Ambuquí, del cantón Ibarra.

1.2. Situación actual diagnóstico.

1.2.1. Misión

"Ser una Cooperativa rural altamente eficiente y de amplia cobertura, con una permanente innovación de productos y servicios de ahorro y crédito con una cultura de calidad en todo el recurso humano".

1.2.2. Visión

"La Cooperativa de Ahorro y Crédito Unión Cochapamba hasta el año 2012, desarrollará e implementará estrategias administrativas y financieras que nos permitan tener sucursales grandes y competitivas, que generen microempresas sin descuidar el medio ambiente, de tal manera que sus socios interactúen como una familia en armonía con Dios y la naturaleza".

1.2.3. Valores Corporativos

1. **Servicio:** en la atención a nuestros clientes siempre demostramos entusiasmo, cordialidad, respeto y amabilidad porque sentimos la satisfacción por la asistencia a los demás.
2. **Cumplimiento:** mediante nuestra labor diaria demostramos el compromiso en la entrega de soluciones a problemas financieros del sector rural, este valor se refleja en la responsabilidad y organización con que tratamos a nuestros clientes.
3. **Compromiso:** demostrada en la convicción de nuestro grupo de trabajo en torno a los beneficios que trae el desempeño responsable y organizado.
4. **Honestidad y Transparencia:** base de la relación con nuestros usuarios, garantizando la integridad, el respaldo y seguridad.
5. **Responsabilidad Social:** Es la contribución activa y voluntaria al mejoramiento social, económico y ambiental con los sectores vulnerables**.** (Coop Unión Cochapamba, 2011)

1.2.4. FODA

La Cooperativa es una institución que tiene un buen prestigio financiero y es bien recibida por las comunidades rurales de la provincia de Imbabura, en especial del cantón Ibarra.
Existen algunos aspectos que alteran el desempeño normal de la institución, entre ellos el tecnológico, que hacen que, instituciones de estas características tengan que buscar mejores oportunidad e innovaciones constantes para cumplir con las metas trazadas.

A continuación se presenta un pequeño diagnostico a través de un análisis FODA.

FORTALEZAS	OPORTUNIDADES
1. Prestigio institucional. **2.** Directivos y personal comprometidos. **3.** Alianzas Estratégicas permiten brindar mayores servicios. **4.** Morosidad decreciente. **5.** Solvencia patrimonial. **6.** Alto nivel de rentabilidad. **7.** Servicio Social de la Comisión Educación. **8.** Ayuda social mediante actividades del mismo orden.	**1.** Desarrollo de micro y pequeña empresa. **2.** Posibilidad de utilizar nuevas fuentes de ingresos para aumentar los fondos de la CUC. **3.** Utilización de línea de Corporación Financiera Nacional para financiamiento de largo plazo. **4.** Apoyo de entidades del exterior como el proyecto Fondo Ítalo-Ecuatoriano. **5.** Apertura de Sucursales de la CUC. **6.** Creciente desarrollo cooperativo en el país.

DEBILIDADES	AMENAZAS
1. Servicio segmentado a comunidades de las zonas rurales **2.** Falta de sucursales **3.** Bajo nivel de control interno. **4.** No contar con un sistema de información integrado **5.** Eficiencia financiera inferior al promedio **6.** Demora en el trámite de créditos. **7.** Datos erróneos previos a la presentación de informes financieros. **8.** Corrección de errores manual. **9.** No existe plan estratégico.	**1.** Fortalecimiento del Sistema Financiero de la Cooperativa. **2.** Contexto socio-político del país. **3.** Situación económica financiera del país. **4.** Mercado competitivo. **5.** Política estatal sobre el sistema cooperativo. **6.** Creación de nuevas instancias reguladoras.

Tabla 1: Análisis de la Cooperativa (FODA)

Fuente: Propia

1.2.5. Situación

La CUC realiza una serie de procesos distribuidos en diferentes áreas específicas, dichos procesos se los realiza mediante el uso de un sistema programado en C++.

El cual muestra muchas limitaciones, porque no controla y administra todos los procesos de forma transparente y unificada, por ello emplea mucho tiempo en procesar la información, los datos no se guardan en ninguna base de datos, se manejan y almacenan de acuerdo a cada módulo en archivos visibles para el usuario, lo que hace que este sistema sea por demás vulnerable y susceptible a cualquier tipo de fraude o estafa.

Todos los módulos funcionan de forma separada e independiente uno del otro, repitiendo información importante en cada módulo y proceso, ocasionando pérdida de tiempo e inconsistencias en la información institucional.

El sistema actualmente usado no le permite a la Cooperativa acoplarse a las nuevas ordenanzas y manejo de cuentas emitidas por los respectivos órganos reguladores de entidades financieras, ya que no es flexible, modificable, adaptable o integrado.

1.2.6. Diagnostico 6M

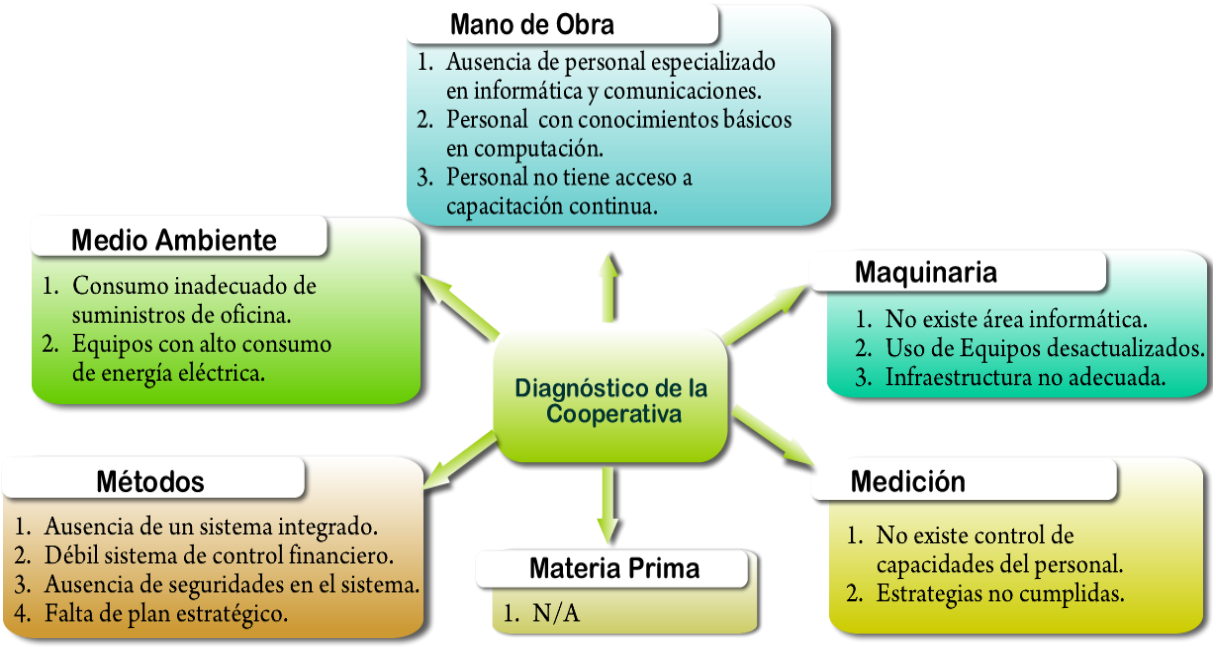


Figura 1: Diagnóstico de la Cooperativa (6 M)

Fuente: Propia

1.2.7. Objetivos y actividades del Plan Anual Operativo.

1. Ser una institución financiera rural posicionada en su área de intervención, con socios que están empoderados, con capacidad de dirigir y administrar los componentes socio-organizativos y financieros de la Cooperativa, en alianza con entidades afines. (FODA).
2. Capacitar a los nuevos directivos en finanzas populares.
3. Mantener alianzas con CODESARROLLO y FODEMI, para acceder a créditos.
    a. Realizar una propuesta conjunta con la Unión de Organizaciones Campesinas Cochapamba, para la utilización de los fondos que se mantiene en administración en FODEMI.
    b. Rotar al menos 200 créditos.
    c. Realizar tres asambleas generales, dos en el mes de enero, la una extraordinaria, para las elecciones de los nuevos directivos.
    d. Realizar cuatro reuniones de consejo administrativo.
    e. Realizar cuatro reuniones de consejo de vigilancia.
4. Preservar el medio ambiente como factor de desarrollo para las comunidades a través de la reforestación y forestación anual de por lo menos 20.000 plantas.
    a. Fomentar la capacitación en el Manejo de los Recursos Naturales in situ a cada beneficiario.

b. Producción de 8 .000 mil plantas forestales para las parcelas campesinas de los socios de la CUC.

5. Fomentar la cultura del ahorro y fortalecer el patrimonio de la Cooperativa vía incremento certificados de aportación por parte de los socios con un aporte de 20 USD. anuales.

a. Promover el ahorro y los depósitos a plazo fijo.

b. Reformar el Reglamento Interno en lo que corresponde a los Certificados de Aportación.

6. Promover la creación de micro emprendimientos productivos como un nuevo modelo de desarrollo comunitario, que permita mejorar el nivel de vida de los socios.

a. Establecer 2 alianzas estratégicas para el fomento de micro emprendimientos.

b. Apoyar en lo administrativo, a dos micro emprendimientos ya existentes (quesera y asociación 20 de diciembre).

c. Apoyar 4 micro-emprendimientos en la zona.

7. Fortalecer el talento humano y administrativo de la Cooperativa.

a. Fortalecer el equipo técnico - administrativo de la CUC con enfoque de género a través de capacitación, acuerdos y convenios.

b. Cumplir con las normas de seguridad y control interno de los bienes y recursos de la CUC (Seguros, Garantías) (Coop. Unión Cochapamba, 2011)

1.3. Estructura de la entidad

1.3.1. Organigrama Estructural

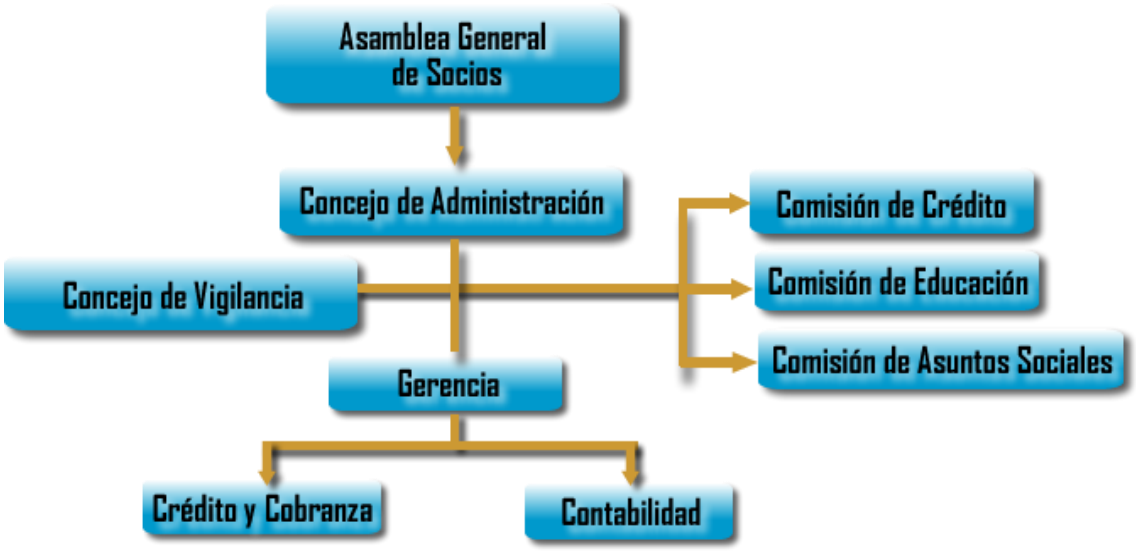


Figura 2: Organigrama Estructural

Fuente: CUC

La dirección, administración y control interno de la Cooperativa se ejercerá por medio de los siguientes organismos:

1. **Asambleas General de socios.-** Es la máxima autoridad de la Cooperativa y está constituida por todos los socios que consten en el registro respectivo y que estuvieren en uso y goce de sus derechos.
2. **Concejo de Administración.-** Es el organismo directo de la Cooperativa quien se atribuye la elección de Presidente, Gerente, Secretario y vocales pertinentes de acuerdo con la ley.
3. **Concejo de Vigilancia.-** Está compuesto por un número de miembros elegidos por la Asamblea General y cuyo objeto es vigilar la elección del presidente y secretario.
4. **Gerencia.-** Administración General de todos los asuntos que impliquen dinero la cooperativa.
5. **Comisiones.**
   a) **Comisión de Crédito:**
   Estará compuesta por miembros seleccionados por la Asamblea General o el Concejo de Administración, decidirá todo lo relacionado con las solicitudes de préstamo de los socios.
   b) **Comisión de Educación**:
   Nombrada por el Consejo de Administración elaborará, organizará, promoverá y presentará planes de trabajo educacional.
   c) **Comisión de Asuntos Sociales**:
   Nombrada por el Concejo de Administración tiene por finalidad estudiar, analizar y solucionar los problemas sociales, así como programar actividades de orden social.

1.4. Información General

1.4.1. Clientes, segmentado por valores

Rango en USD	Nro. Clientes
1-100	500
100-500	70
500-1000	13
1000-2000	14
2000-3000	8
3000-5000	8
>5000	7

Tabla 2: Rango de Clientes cooperativa

Fuente: CUC

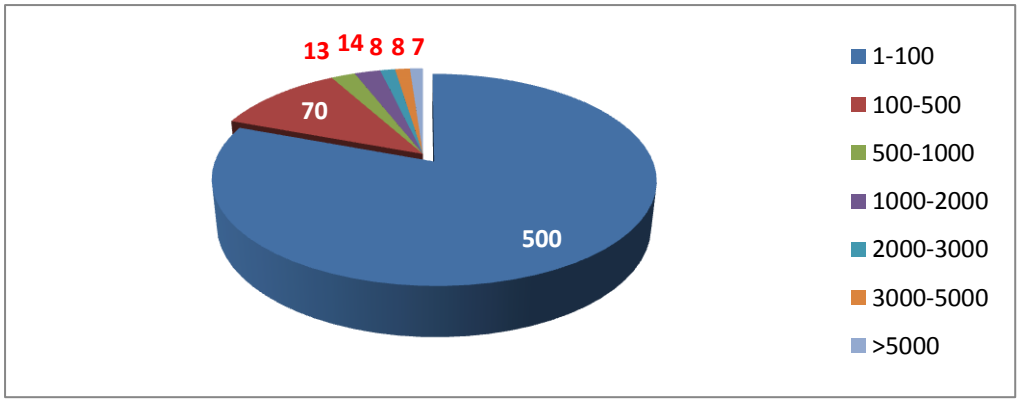


Figura 3: Clientes de Cooperativa

Fuente: CUC

1.4.2. Préstamos

Plazos y Montos de Préstamos	
Crédito	**Descripción**
Pecuarios	Se prestará un máximo de $1.500 USD, plazo máximo 2 años, pagos mensuales, trimestrales y máximo semestrales.
Agrícolas	Se prestará un máximo de $1.500 USD, plazo dependiendo del cultivo máximo 18 meses y pagos serán mensuales, trimestrales y máximo semestrales.
Salud	Serán solo en casos de enfermedades del socio o de los familiares, plazo máximo de 1 año, pagos trimestrales y se prestará máximo hasta $1.000 USD.
Viajes de Trabajo	Se entregará a personas que requieran fondos para viajes de trabajo en el exterior, se presta un monto máximo de $1000 USD.
Viviendas	Plazo máximo 4 años, pagos mensuales, trimestrales, semestrales, monto máximo $5.000 USD.
Compra de Terrenos	Plazo máximo será de 4 años pagaderos mensuales y trimestrales y un monto máximo de $5.000 USD.
Educación	Plazo hasta 18 meses, monto máximo hasta de $1.500 USD.
Crédito forestal	Plazo hasta 3 años, monto hasta $2.000 USD, pagos mensuales y trimestrales.
Crediempresas	Plazos hasta 4 años, monto máximo $5.000 USD, pagos mensuales y trimestrales.

Tabla 3: Tabla de Préstamos
Fuente: CUC

Capítulo II: Marco Teórico

CONTENIDO:

1. Normas legales.
2. Estándares abiertos.
3. Seguridad en sistemas informáticos.
4. Sistemas Operativos.
5. Servidor de Aplicaciones.
6. Plataformas y Frameworks
7. Herramienta de Reportes.

2. Marco Teórico

A continuación se describen aspectos generales de plataformas, estándares, herramientas y demás información de estudio necesarios para desarrollo del aplicativo.

2.1. Normas Legales

Las normas legales o jurídicas son reglas u ordenamientos dictados por alguna autoridad competente de acuerdo a un criterio de valor y cuyo incumplimiento trae una sanción.

2.1.1. Ley General de Instituciones del Sistema Financiero

Un sistema financiero es un conjunto de instituciones financieras es decir bancos, sociedades, grupos, etc., que realizan algún tipo de intermediación financiera con el público.
En el Ecuador el Sistema Financiero comprende el Banco Central del Ecuador, las instituciones financieras públicas, las instituciones financieras privadas y las demás instituciones controladas por la Superintendencia de Bancos y Seguros.

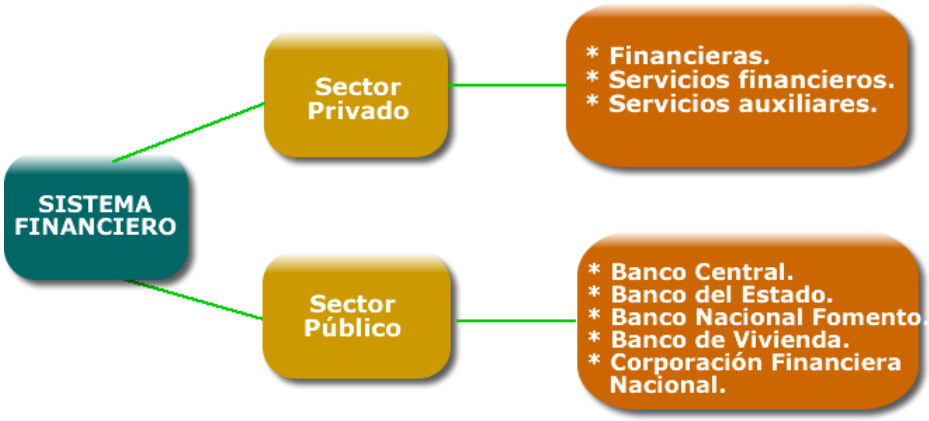


Figura 4: Estructura Sistema Financiero

Fuente: Propia

Los ordenamientos jurídicos en la historia relacionados con el sistema financiero han crecido y son reformadas en número y magnitud por la incorporación de nuevos bancos nacionales y extranjeros.
La normativa vigente regula la creación, organización, actividades, funcionamiento y extinción de las instituciones del sistema financiero privado. Además, la Ley contiene disposiciones para que la Superintendencia de Bancos ejerza control y vigilancia sobre estas instituciones.

Cada artículo de la Ley General de Instituciones del Sistema Financiero menciona aspectos trascendentales que permiten regular a las instituciones.

Art. 1:

"La Ley General de Instituciones del Sistema Financiero regula la creación, organización, actividades, funcionamiento y extinción de las instituciones del sistema financiero privado, así como la organización y funciones de la Superintendencia de Bancos y Seguros, en el campo de su competencia, entidad encargada de la supervisión y control del sistema financiero, en todo lo cual se tiene presente la protección de los intereses del público". (OAS, 2011)

2.1.2. Ley de Cooperativas

La Ley de Cooperativas es un precepto legal de aplicación obligatoria para el sistema cooperativo ecuatoriano, por mandato legal, corresponde a la Dirección Nacional de Cooperativas, ejecutar las políticas estatales de promoción del sector y realizar todos los trámites para la aprobación y registro de las organizaciones cooperativas.

La ley admite la autonomía, es decir además del Reglamento General de Cooperativas, se respeta sus propios estatutos y reglamentos internos.

La ley de cooperativas consta de varios artículos separados por capítulos y en cada uno de ellos de describen los procedimientos y normativas que deben seguir las instituciones cooperativas.

Dichos capítulos presentan los siguientes grupos:

1. Naturaleza y Fines
2. Constitución y Responsabilidad
3. De los Socios
4. Estructura Interna y Administración
5. Régimen Económico
6. Clasificación de las Cooperativas
7. Organizaciones de Integración Cooperativa
8. Fomento y Supervisión
9. Disolución y Liquidación
10. Beneficios y Sanciones
11. Disposiciones Especiales
12. Disposiciones Generales

Las cooperativas se rigen por los valores y principios universales del cooperativismo, los mismos que deberán ser aplicados en las resoluciones de carácter general emitidas para el sector cooperativo, los principios del cooperativismo son:

1. Adhesión abierta y voluntaria.
2. Control democrático de los socios.
3. Participación económica de los socios.
4. Autonomía e independencia.
5. Educación, capacitación e información.
6. Cooperación entre cooperativas.
7. Compromiso con la comunidad". (sbs.gob.ec, 2009)

2.1.3. Normas de Basilea

Las normas de Basilea son el resultado de las decisiones de los países más industrializados del mundo, quienes, representados por los Bancos Centrales y Organismos de Supervisión de cada país, aprobaron 25 Principios que sientan las bases para una regulación prudencial del sistema bancario de cada nación. El documento que contiene los Principios Básicos para la Supervisión Bancaria efectiva fue aprobado en 1997, pero periódicamente se revisa para ajustarlos a los requerimientos que las circunstancias exigen. (Normativa De Basilea, 2012)

Norma de Basilea I.- Dicho comité llegó a un acuerdo, conocido con el nombre de "Basilea I", dicho acuerdo consistía en imponer una norma regulatoria a todas las entidades financieras de dichos países. La norma (Basilea I) dice que todas las entidades financieras deben poseer en propiedad un mínimo del 8% de los llamados "activos de riesgo", en otras palabras, de los productos financieros que son considerados peligrosos y por ende, pudieran ser causantes de que la banca tuviera importantes pérdidas.

Norma de Basilea II.- Consiste en recomendaciones sobre la legislación y regulación bancaria y son emitidos por el Comité de supervisión bancaria de Basilea. El propósito de Basilea II, es la creación de un estándar internacional que sirva de referencia a los reguladores bancarios, con objeto de establecer los requerimientos de capital necesarios, para asegurar la protección de las entidades frente a los riesgos financieros y operativos.

Norma de Basilea III.- Representada por 44 bancos centrales y autoridades supervisoras.

Los aspectos más destacados son:

1. Mejorar y elevar la calidad del capital bancario exigido.
2. Reducir el riesgo sistémico y conceder suficiente tiempo para una transición manejable hacia el nuevo régimen.

2.2. Estándares abiertos

2.2.1. XML

Es un lenguaje de marcas generalizado (Extensible Markup Language), es un lenguaje usado para estructurar información en un documento o en general en cualquier fichero que contenga texto, como por ejemplo ficheros de configuración de un programa o una tabla de datos.
Ha ganado mucha popularidad en los últimos años debido a ser un estándar abierto y libre, creado por el Consorcio World Wide Web, W3C (los creadores de la www), en colaboración con un panel que incluye representantes de las principales compañías productoras de software.
El XML fue propuesto en 1996, y la primera especificación apareció en 1998. Desde entonces su uso ha tenido un crecimiento acelerado, que se espera que continúe durante los próximos años. (The World Wide Web Consortium, 2003)

2.2.1.1. Ventajas XML

Antes de ser lanzado el XML, ya existían otros lenguajes de marcas, como por ejemplo el HTML, basados en el lenguaje generalizado de marcas (SGML). El problema con el SGML es que por ser muy flexible y muy general, se torna difícil el análisis sintáctico de un documento y la especificación de la estructura. XML es más exigente que SGML en la sintaxis, lo que hace más fácil la construcción de librerías para procesarlo.
Comparado con otros sistemas usados para crear documentos, el XML tiene la ventaja de poder ser más exigente en cuanto a la organización del documento, lo cual resulta en documentos mejor estructurados. (The World Wide Web Consortium, 2003)

2.2.2. HTML

Una definición simple consiste en que es un lenguaje con el que se “escriben” páginas web.
Los diseñadores de páginas web utilizan este lenguaje para crear sus páginas, los programas que utilizan los diseñadores generan páginas HTML y los navegadores que utilizamos los usuarios muestran las páginas web después de leer el contenido, aunque es un lenguaje que utilizan los ordenadores y los programas de diseño, es fácil de aprender y escribir por parte de las personas, el nombre HTML está formado por las siglas de Hypertext Markup Language, en español lenguaje de marcado para hipertexto, que se definió teniendo en cuenta varias de las características más habituales que existían en ese momento para la publicación digital de contenidos.

Entre los conceptos utilizados en su creación, se encuentra el mecanismo de Hipertexto, de hecho, las letras “HT” del acrónimo HTML significan “hipertexto”.

El uso del sistema de hipertexto para crear documentos interactivos y que proporcionan información adicional cuando se solicita, es una de las claves del éxito del lenguaje, el elemento principal del hipertexto es el “hiperenlace”, también llamado “enlace web” o simplemente “enlace”.

Los enlaces se utilizan para establecer relaciones entre dos recursos web, aunque la mayoría de enlaces relacionan páginas web, también es posible enlazar otros recursos como imágenes, documentos, archivos y programas.

Una característica que no se suele tener en cuenta en los enlaces es que están formados por dos extremos y un sentido, en otras palabras, el enlace comienza en un extremo y apunta hacia el otro extremo. Cada uno de los dos extremos se llama “anchors” en inglés, que se puede traducir literalmente como “anclas”.

El lenguaje es un estándar reconocido en todo el mundo y cuyas normas define un organismo sin ánimo de lucro llamado W3C (World Wide Web Consortium), como se trata de un estándar reconocido por todas las empresas relacionadas con el mundo de Internet, una misma página HTML se visualiza de la misma manera en cualquier navegador de cualquier sistema operativo.

El propio W3C define el lenguaje HTML como “un lenguaje reconocido universalmente y que permite publicar información de forma global”. Desde su creación, el lenguaje HTML ha pasado de ser un lenguaje utilizado exclusivamente para crear documentos electrónicos a ser un lenguaje que se utiliza en muchas aplicaciones electrónicas: buscadores, tiendas online, banca electrónica.

(Corporation Creative Commons, 2009)

2.2.2.1. Elementos HTML

Los elementos son la estructura básica de HTML, los elementos tienen dos propiedades básicas: atributos y contenido.

Cada atributo y contenido tiene ciertas restricciones para que se considere válido al documento HTML.

Un elemento generalmente tiene una etiqueta de inicio (p.ej. <nombre-de-elemento>) y una etiqueta de cierre (p.ej. </nombre-de-elemento>). Los atributos del elemento están contenidos en la etiqueta de inicio y el contenido está ubicado entre las dos etiquetas (p.ej. <elemento="valor">Contenido</ elemento>). Algunos elementos, tales como
, no tienen contenido ni llevan una etiqueta de cierre.

El marcado estructural describe el propósito del texto, por ejemplo, <h2> Finansys</h2> establece «Finansys» como un encabezamiento de segundo nivel.

El marcado estructural no define cómo se verá el elemento, pero la mayoría de los navegadores web han estandarizado el formato de los elementos. (Wikipedia, es.wikipedia.org, 2007)

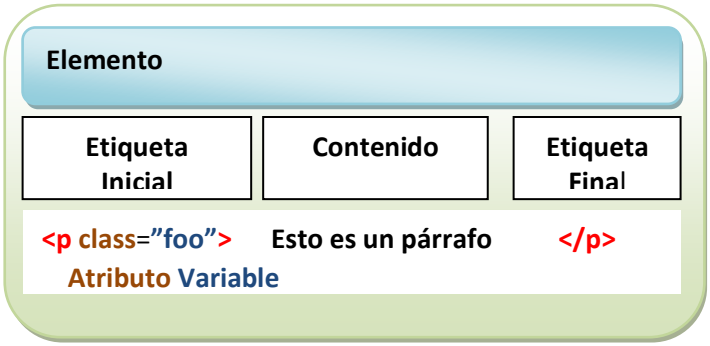


Figura 5: Estructura de un tag HTML

Fuente: (Wikipedia, es.wikipedia.org, 2007)

2.2.3. CSS (Cascading Style Sheets)

CSS es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML.

CSS es la mejor forma de separar los contenidos de la presentación y hoy en día se ha vuelto imprescindible para crear páginas web complejas.

En la separación de los contenidos y la vista presenta numerosas ventajas, ya que obliga a crear documentos HTML bien definidos y con significado completo (también llamados "documentos semánticos"), además mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes, mientras que el lenguaje HTML/XHTML se utiliza para marcar los contenidos, es decir, para designar lo que es un párrafo, lo que es un titular o lo que es una lista de elementos, el lenguaje CSS se utiliza para definir el aspecto de todos los contenidos, es decir, el color, tamaño y tipo de letra de los párrafos de texto, la separación entre titulares y párrafos, la tabulación con la que se muestran los elementos de una lista, etc. (Corporation Creative Commons, 2009)

La información de estilo puede ser adjuntada como un documento separado o en el mismo documento HTML, en este último caso podrían definirse estilos generales en la cabecera del documento o en cada etiqueta particular mediante el atributo "<style>".

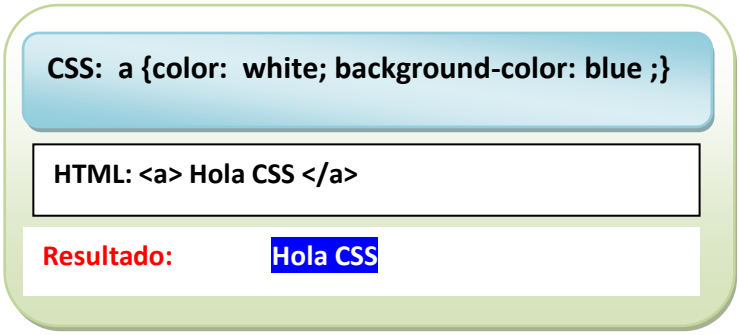


Figura 6: CSS en Práctica

Fuente: Propia

2.2.3.1. Formas de usar CSS

Para dar formato a un documento HTML, puede emplearse CSS de tres formas distintas:

1. Mediante CSS introducido por el autor del HTML.

   1.1. **Un estilo en línea (inline).-** es un método para insertar el lenguaje de estilo de página directamente dentro de una etiqueta HTML.
   Esta manera de proceder no es totalmente adecuada. El incrustar la descripción del formateo dentro del documento de la página Web, a nivel de código, se convierte en una manera larga, tediosa y poco selecta de resolver el problema de la programación de la página. Este modo de trabajo se podría usar de manera ocasional si se pretende aplicar un formateo con rapidez, No es todo lo claro o estructurado que debería ser, pero funciona.

   1.2. **Una hoja de estilo interna**.- es una hoja de estilo que está incrustada dentro de un documento HTML, dentro del elemento <head>, marcada por la etiqueta <style>. De esta manera se obtiene el beneficio de separar la información del estilo del código HTML propiamente dicho. Se puede optar por copiar la hoja de estilo incrustada de una página a otra (esta posibilidad es difícil de ejecutar si se desea para guardar las copias sincronizadas). En general, la única vez que se usa una hoja de estilo interna, es cuando se quiere proporcionar alguna característica a una página Web en un simple fichero, por ejemplo, si se está enviando algo a la página Web.

   1.3. **Una hoja de estilo externa.-** es una hoja de estilo que está almacenada en un archivo diferente al archivo donde se almacena el código HTML de la página Web. Esta es la manera de programar más potente, porque separa completamente las reglas de formateo para la página HTML de la estructura básica de la página.

2. Estilos CSS introducidos por el usuario que ve el documento, mediante un archivo CSS especificado mediante las configuraciones del navegador, y que sobrescribe los estilos definidos por el autor en una, o varias páginas web.
3. Los estilos marcados "por defecto" por los user agent, para diferentes elementos de un documento HTML, como por ejemplo, los enlaces. (Wikipedia, es.wikipedia.org, 1996)

2.2.4. JavaScript

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas.

Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario, técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios.

A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java. (Tom Negrino, 2011)

2.2.4.1. Formas de usar JavaScript

1) Dentro del mismo documento HTML/XHTML.

El código JavaScript se encierra entre etiquetas <script> y se incluye en cualquier parte del documento. Aunque es correcto incluir cualquier bloque de código en cualquier zona de la página, se recomienda definir el código JavaScript dentro de la cabecera del documento (dentro de la etiqueta <head>)

2) En un archivo externo.

Las instrucciones JavaScript se pueden incluir en un archivo externo de tipo JavaScript que los documentos HTML/XHTML enlazan mediante la etiqueta <script>. Se pueden crear todos los archivos JavaScript que sean necesarios y cada documento XHTML puede enlazar tantos archivos JavaScript como necesite.

3) Incluir JavaScript en los elementos HTML/XHTML.

Este último método es el menos utilizado, ya que consiste en incluir trozos de JavaScript dentro del código XHTML, El mayor inconveniente de este método es que ensucia innecesariamente el código XHTML de la página y complica el mantenimiento del código JavaScript. En general, este método sólo se utiliza para definir algunos eventos y en algunos otros casos especiales. (Corporation Creative Commons, 2009)

2.3. Seguridad en Sistemas Web

Un sistema web hace uso del protocolo HTTP para interactuar con los usuarios u otros sistemas, para ello el cliente utiliza un navegador, los problemas de seguridad pueden provenir de programas web, aunque en su mayor parte son consecuencia de fallos en la lógica y el diseño de la propia aplicación.

La necesidad de garantizar la seguridad en los datos que fluyen en el uso de un sistema web hacen que día a día se tomen medidas para mantener a salvo los datos.

Para que un sistema web se pueda definir como seguro debe tener en consideración estas cuatro características:

1. **Integridad:** La información sólo puede ser modificada por quien está autorizado y de manera controlada.
2. **Confidencialidad:** La información sólo debe ser legible para los autorizados.
3. **Disponibilidad:** Debe estar disponible cuando se necesita.
4. **No repudio:** El uso y/o modificación de la información por parte de un usuario debe ser irrefutable, es decir, que el usuario no puede negar dicha acción.

Los siguientes aspectos muestran posibles puntos vulnerables en aplicaciones web:

2.3.1. Seguridad en el Cliente

Un cliente es el usuario final del sistema web y es necesario que en él se tomen precauciones para evitar la vulnerabilidad contra ataques hacia el sistema, las acciones realizadas en el equipo deben ser de alguna manera controladas, para ello se puede optar por realizar verificaciones constantes del software que se instala, el uso de algún antivirus actualizado y demás programas o protecciones que eviten ataques.

2.3.2. Seguridad en el Servidor

El desarrollo y puesta en producción de una aplicación web requiere de varias herramientas, las mismas que deben cumplir un criterio de seguridad y tener las respectivas configuraciones que eviten ataques contra la aplicación.

1. **El servidor web.-** Se considera la cara pública de la organización y es sin duda un blanco de ataques hacia la información, por lo tanto es necesario tener precaución con los servicios y permisos que presta el servidor.
2. **Servidor de Aplicaciones.-** Generalmente gestiona la mayor parte de las funciones de lógica de negocio y de acceso a los datos de la aplicación. Es muy conveniente revisar periódicamente los ficheros de log (access_log y error_log en el servidor de aplicaciones Apache) para detectar posibles ataques al servidor de aplicaciones.
3. **Servidor de Bases de Datos.-** Las organizaciones suelen cometer varios errores que vuelven vulnerable a las bases de datos, como dejar las bases de datos de las pruebas en los servidores de producción, o enlazar datos sensibles a aplicaciones que dan cara al web y son de fácil hackeo, uno de los principales ataques son la inyección a SQL [lenguaje de consulta estructurado], pues los atacantes introducen tiras de código SQL a los débiles campos de las aplicaciones web.

2.3.3. Seguridad en la Aplicación

Para la implementación de la seguridad en las aplicaciones web se debe considerar las siguientes alternativas:

1) Control de acceso.- Un aspecto muy importante de una aplicación web es el control de acceso de los usuarios a zonas restringidas de la aplicación aquí intervienen dos conceptos:

1.1. Autentificación.- Es el proceso de determinar si un usuario es quien dice ser se puede hacer de varias maneras. Algunas de ellas son:

- ✓ Autentificación HTTP básica.
- ✓ Autentificación basada en la aplicación

1.2. Autorización.- Es el acto de comprobar si un usuario tiene el permiso adecuado para acceder a un cierto fichero o realizar una determinada acción, una vez que ha sido autentificado.

Diseñar el mecanismo de control de acceso exige:

- ✓ Determinar la información que será accesible por cada usuario.

- ✓ Determinar el nivel de acceso de cada usuario a la información.
- ✓ Especificar un mecanismo para otorgar y revocar permisos a los usuarios.
- ✓ Proporcionar funciones a los usuarios autorizados: identificación, desconexión, petición de ayuda, consulta y modificación de información personal, cambio de password, etc.
- ✓ Ajustar los niveles de acceso a la información a la política de seguridad de la organización.

2) **Validación de datos de entrada.-** El problema más frecuente que presentan las aplicaciones web es no validar correctamente los datos de entrada, esto da lugar a algunas de las vulnerabilidades más importantes de las aplicaciones, como la inyección SQL, el Cross-Site Scripting y el Buffer Overflow.

   Algunos de los aspectos a tomar en cuenta son:

   1. Fuentes de entrada.
   2. Inyección.
   3. Estrategias de protección.
   4. Vulnerabilidades específicas.

3) **Programación segura.-** Para evitar o al menos disminuir las vulnerabilidades de una aplicación web es muy importante seguir unas correctas prácticas de programación. algunas de las más importantes son:

   1. Inicialización de variables.
   2. Gestión de errores.
   3. Protección de información. (Universidad de Sevilla, 2011)

2.3.4. Seguridad en la Comunicación

SSL (Secure Socket Layer) es un protocolo para asegurar el transporte de datos entre el cliente y el servidor web, podemos reconocer una conexión HTTP sobre SSL porque aparece el prefijo ‘https’ en lugar de ‘http’ en la URL.

La manera en la que funciona el SSL es la siguiente:

Cuando el cliente pide una página SSL, el servidor envía un certificado que es obtenido de una autoridad certificadora confiable. El certificado contiene la llave pública del servidor. Después de asegurarse que el certificado es correcto y que el servidor es genuino, el cliente genera un número aleatorio, la llave de sesión. La llave es encriptada con la llave pública del servidor y enviada. El servidor desencripta el mensaje con su llave privada. Ahora ambos lados tienen una llave de sesión

conocida solamente por ellos dos. Toda comunicación desde y hacia ellos es encriptada y desencriptada con la llave de sesión.

Figura 7: HTTP sobre SSL

Fuente: (Owasp org, 2006)

SSL proporciona una comunicación segura entre cliente y servidor permitiendo la autentificación mutua, el uso de firmas digitales y garantizando la privacidad mediante encriptación (Owasp org, 2006)

2.4. Sistemas Operativos (S.O)

Un S.O es el software básico de una computadora que provee una interfaz entre el resto de programas del ordenador, los dispositivos hardware y el usuario.

Las funciones básicas del S.O son administrar los recursos de la máquina, coordinar el hardware y organizar archivos y directorios en dispositivos de almacenamiento, los sistemas operativos más utilizados son, Windows, Linux y Mac, con sus respectivas versiones.

El S.O es el principal elemento necesario para que una aplicación o programa se ejecute y funcione de manera correcta, al crear aplicaciones web y hacer uso de ellas, el S.O pasa a ser un medio, ya que no es necesario un tipo específico para su funcionamiento, siendo Finansys un sistema web es totalmente portable lo que quiere decir que no importa el S.O que se use, con la ayuda de un navegador o browser especificado, puede ser usado sin ningún problema. (Francisco Rueda Profesor, 2010)

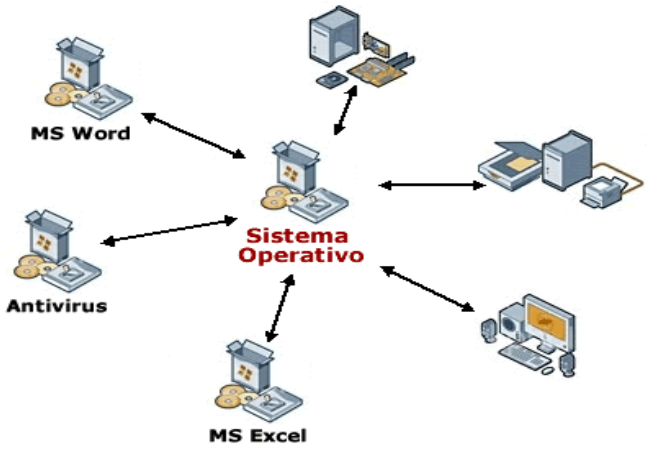


Figura 8: Sistema Operativo

Fuente: (Masadelante, 2011)

2.4.1. Clasificación de Sistemas Operativos

1. **Multiusuario:** Permite que dos o más usuarios utilicen sus programas al mismo tiempo. Algunos sistemas operativos permiten a centenares o millares de usuarios al mismo tiempo.
2. **Multiprocesador:** soporta el abrir un mismo programa en más de una CPU.
3. **Multitarea:** Permite que varios programas se ejecuten al mismo tiempo.
4. **Multitramo:** Permite que diversas partes de un solo programa funcionen al mismo tiempo.
5. **Tiempo Real:** Responde a las entradas inmediatamente, los sistemas operativos como DOS y UNIX, no funcionan en tiempo real. (Masadelante, 2011)

2.4.2. Características de los Sistemas Operativos

Las principales características de los S.O son las siguientes:

1. La instalación de un S.O hace más conveniente el uso de la computadora y por ende el uso de programas.
2. Hacen posible que se use de manera eficiente los recursos de la computadora y la correcta utilización de cada uno de ellos.
3. Pueden evolucionar aumentándoles nuevas funciones y servicios.
4. Administran correctamente cada uno de los dispositivos de hardware del computador, como el mouse, teclado, parlantes, impresoras, etc.
5. Organizan la información y le ordenan creando accesos rápidos para poder mostrarlos cuando el usuario lo requiera.
6. Facilita el acceso y la comunicación entre usuario, software, y dispositivos de hardware.

2.4.3. Fedora

Fedora es un S.O basado en Linux, consiste en una colección de software que hacen funcionar un computador de manera rápida y eficiente, es libre y gratuito, se mantiene gracias a una comunidad internacional (Proyecto Fedora) de ingenieros, diseñadores gráficos y usuarios que informan de fallos y prueban nuevas tecnologías. (Wikipedia, es.wikipedia.org, 2011)

Proyecto Fedora.- Es el nombre de una comunidad de personas en todo el planeta que comparten la filosofía, utilizan y construyen software libre, la meta de esta comunidad, es liderar la creación y la distribución tanto de código como de contenidos libres. Fedora es patrocinado por Red Hat, el proveedor de tecnología de código abierto más confiable en todo el mundo. Red Hat invierte en Fedora para estimular la colaboración y la innovación en tecnologías de software libre. (Wikipedia, es.wikipedia.org, 2011)

Seguridad Fedora.- SELinux ("Security-Enhanced Linux") se destaca entre las características de seguridad de Fedora, pues implementa una gran variedad de políticas de seguridad, incluyendo control de acceso obligatorio (MAC "Mandatory Access Control"), a través de los Módulos de Seguridad de Linux que están en el núcleo Linux del sistema. La distribución está liderando las distribuciones que incorporan SELinux, habiéndolo introducido en Fedora Core 2, sin embargo se lo desactivó como elemento predeterminado, pues alteraba radicalmente la forma en que el S.O funcionaba. Posteriormente fue activado por defecto en Fedora Core 3 introduciendo una política menos estricta. Fedora también tiene métodos propios para prevenir la sobrecarga del buffer y la utilización de rootkits. La verificación del buffer en tiempo de compilación, «Exec Shield» y restricciones en como la memoria del núcleo en /dev/mem puede ser accedida ayudan a prevenir esto.

En la actualidad han lanzado 16 versiones de este sistema operativo, durante sus primeras 6 versiones se llamó Fedora Core, debido a que solo incluía los paquetes más importantes del sistema operativo. La última versión es Fedora 16 denominada Verne.

Fedora cuenta con una buena documentación, traducida en alrededor de 40 idiomas y disponible en múltiples formatos, la documentación de Fedora describe como instalar y usar el sistema operativo Fedora y el software empaquetado por el Proyecto Fedora.

La Guía de Seguridad en Fedora está diseñada para asistir a usuarios de Fedora en el proceso de aprendizaje y prácticas de seguridad en estaciones de trabajo y servidores, para poder así evitar intrusiones locales y remotas, explotaciones, y actividades maliciosas. Con un conocimiento administrativo apropiado, vigilancia, y herramientas, los sistemas en Linux pueden ser funcionales y seguros, frente a los métodos de intrusión y explotación más comunes. (Fedora Project, 2008)

2.4.3.1. Características principales

1. Repositorios que Fedora recomienda usar solo los de código abierto o software libre.
2. Se destaca en seguridad y utiliza SELinux entre otras medidas de seguridad.
3. Soporta las arquitecturas x86, x86-64 y PowerPC.
4. Soporta redes instaladas sobre HTTP, FTP y NFS.
5. El entorno de escritorio por defecto es GNOME, e incluye KDE.
6. El gestor de buteo es GNUGRUB por defecto.
7. Fácil de instalar y configurar, incluyendo para esto instaladores y herramientas gráficas.
8. El sistema de archivos por defecto es ext3 sobre LVM.
9. Su navegador por defecto es Firefox desde su versión Fedora Core 3 y superior.
10. Incluye el paquete ofimático OpenOffice.org desde su versión 4.

2.4.4. CentOS

Es una distribución Linux de clase empresarial derivada de fuentes libres ofrecidas al público por Red Hat Enterprise Linux, normalmente utilizado como servidor, brinda un alto rendimiento en la administración de servicios.

Red Hat libera todo el código fuente del producto de forma pública bajo los términos de la licencia pública general de GNU y otras licencias. Los desarrolladores de CentOS usan ese código fuente para crear un producto final que es muy similar al Red Hat Enterprise Linux y está libremente disponible para ser descargado y usado por el público.

Existen otras distribuciones también derivadas de las fuentes de Red Hat. CentOS se ajusta plenamente a la política de redistribución del proveedor original y aspira a ser 100% compatible a nivel binario.

CentOS es desarrollado por un equipo de programadores del núcleo. A su vez los desarrolladores principales son apoyados por una activa comunidad de usuarios como los administradores de sistemas, administradores de red, empresarios, gerentes y colaboradores de todo el mundo. (Centos, 2011)

CentOS es una distribución empresarial plenamente funcional, estable y lista para usar sin costo, se puede usar como servidor web, servidor de correo, servidor de bases de datos o servidor de aplicaciones. CentOS incorpora todas las sofisticadas funcionalidades de compatibilidad con Windows incluidas en Red Hat Enterprise Linux. Se puede usar CentOS como controlador de dominios primarios o secundarios, o como servidor de clientes Windows basado en Samba y SMB.

CentOS es también un sistema de escritorio Linux plenamente funcional. Cuenta con la mayoría de las herramientas de escritorio más populares, como OpenOffice.org, Mozilla Firefox, Evolution, Gaim, X-chat, Konqueror, Gimp, Gnome y KDE.

Ya sea trabajando en un servidor empresarial, en un servidor para la oficina o incluso en un sistema común de escritorio, CentOS es una distribución estable, de alto rendimiento, fácil de administrar y de utilizar. (Centos, 2011)

2.4.4.1. Características

1. CentOS es sumamente estable y brinda todas las características para ser un servidor.
2. Actualizaciones continuas.
3. Soporta varios programas comerciales o no comerciales.
4. Existen repositorios que en suma alcanzan más de los diez mil paquetes (entre base, extras, rpmforge, atrpms y el de karan).
5. Yum permite realizar el trabajo de instalación o desinstalación. (Centos, 2011)

2.5. Servidor de Aplicaciones Apache Tomcat

2.5.1. Introducción a Apache Tomcat

Apache Tomcat (también llamado Jakarta Tomcat o simplemente Tomcat) funciona como un contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat implementa las especificaciones de los servlets y de Java Server Pages (JSP).

Tomcat es mantenido y desarrollado por miembros de la Apache Software Foundation y voluntarios independientes, los usuarios disponen de libre acceso a su código fuente y a su forma binaria en los términos establecidos en la Apache Software Licence.

Las primeras distribuciones de Tomcat fueron las versiones 3.0.x, las versiones más recientes son las 7.x, que implementan las especificaciones de Servlet 3.0 y de JSP 2.2.

A partir de la versión 4.0, Tomcat utiliza el contenedor de servlets Catalina. (Wikipedia T. , 2011)

2.5.2. Estructura del servidor Apache Tomcat

Las distribuciones binarias de Tomcat mantienen la estructura de directorios, teniendo en cuenta que la jerarquía de directorios de instalación de Tomcat incluye lo siguiente:

1. **bin.-** Este directorio contiene los scripts de inicio, parar, reiniciar y otros scripts y ejecutables para el funcionamiento del servidor.
2. **common.-** Este directorio contiene un sinnúmero de clases comunes que puede utilizar el Contenedor de ServletsCatalina y las aplicaciones web que se encuentren alojadas en el servidor web.
3. **conf.-** Este directorio contiene algunos archivos de configuración incluyendo server.xml que es el archivo principal de configuración de Tomcat y el archivo web.xml el cual configura los valores por defecto para las distintas aplicaciones desplegadas en Tomcat como cuando se usa Hibernate, RichFaces, etc. También en este directorio se encuentran los correspondientes DTD para la configuración de Tomcat. La función básica de los DTD es la descripción del formato de datos.
4. **logs.-** Aquí es donde Tomcat aloja los mensajes cuando se ejecuta, cierra o existen errores al ejecutar las aplicaciones.
5. **server**.- En este directorio existen clases que son utilizadas únicamente por el contenedor de Servlets Catalina.
6. **shared.-** Ubicación donde se encuentran las clases compartidas por todas las aplicaciones web alojadas en el servidor de aplicaciones Tomcat.

7. **webapps.-** Este directorio es donde se deben copiar o donde deben estar todos los archivos de una aplicación web.
8. **work.-** Se ubican los archivos que se crean durante la ejecución del servidor es decir el almacenamiento temporal de ficheros y directorios. . (Wikipedia T. , 2011)

2.5.3. Funcionamiento del servidor de aplicaciones Apache Tomcat

Dado que fue escrito en Java, funciona en cualquier sistema operativo que disponga de una máquina virtual Java.

Es cada vez más utilizado por las empresas en los entornos de producción debido a su contrastada estabilidad, resulta de gran utilidad para los programadores que deseen usar Tomcat como servidor web autónomo, en entornos con alto nivel de tráfico y alta disponibilidad.

Constituye además una excelente herramienta para los principiantes, existe detalles y documentación para usar Tomcat en todas las plataformas principales: Windows, Linux, Mac OS X, Solaris, y FreeBSD, con sus ficheros de configuración específicos, y consejos paso a paso para implementar y correr aplicaciones web eficazmente.

Figura 9: Integración con Apache Tomcat

Fuente: (Wikipedia T. , 2011)

2.5.4. Características de versiones Tomcat

A continuación se presentan las principales características de las últimas versiones del servidor de aplicaciones Tomcat.

Tomcat 6.x

1. Implementado de Servlet 2.5 y JSP 2.1
2. Soporte para Unified Expression Language 2.1
3. Diseñado para funcionar en Java SE 5.0 y posteriores
4. Soporte para Comet a través de la interfaz Comet Processor

Tomcat 7.x

1. Implementado de Servlet 3.0 JSP 2.2 y EL 2.2.
2. Mejoras para detectar y prevenir "fugas de memoria" en las aplicaciones web.
3. Limpieza interna de código.
4. Soporte para la inclusión de contenidos externos directamente en una aplicación web. (Wikipedia T. , 2011)

2.6. Base de Datos

Una base de datos o banco de datos es un conjunto de datos no redundantes, estructurados, organizados, pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. Las bases de Datos tienen muchos usos: nos facilitan el almacenamiento de grandes cantidades de información; permiten la recuperación rápida y flexible de información, con ellas se puede organizar y reorganizar la información, así como imprimirla o distribuirla en formas diversas. Una base de datos debe de contar con independencia lógica y física de los datos, esto se refiere a la capacidad de modificar alguna información específica sin que afecte a los demás registros**.** (Rincon, 2011)

2.6.1. PostgreSQL

Es un sistema avanzado de administración de Base de Datos objeto-relacionales (ORDBMS) de código abierto PostgreSQL es un gestor de bases de datos orientadas a objetos (SGBDOO o ORDBMS en sus siglas en inglés) muy conocido y usado en entornos de software libre, soporta un conjunto de funcionalidades avanzadas, lo que lo sitúa al mismo o a un mejor nivel que muchos SGBD comerciales.

PostgreSQL se distribuye bajo licencia BSD, lo que permite su uso, redistribución, modificación con la única restricción de mantener el copyright del software a sus autores, en concreto el PostgreSQL Global Development Group y la Universidad de California. PostgreSQL tiene una arquitectura que involucra muchos estilos, en su nivel más alto es un esquema clásico cliente-servidor, mientras que el acceso a la data es un esquema en capas.

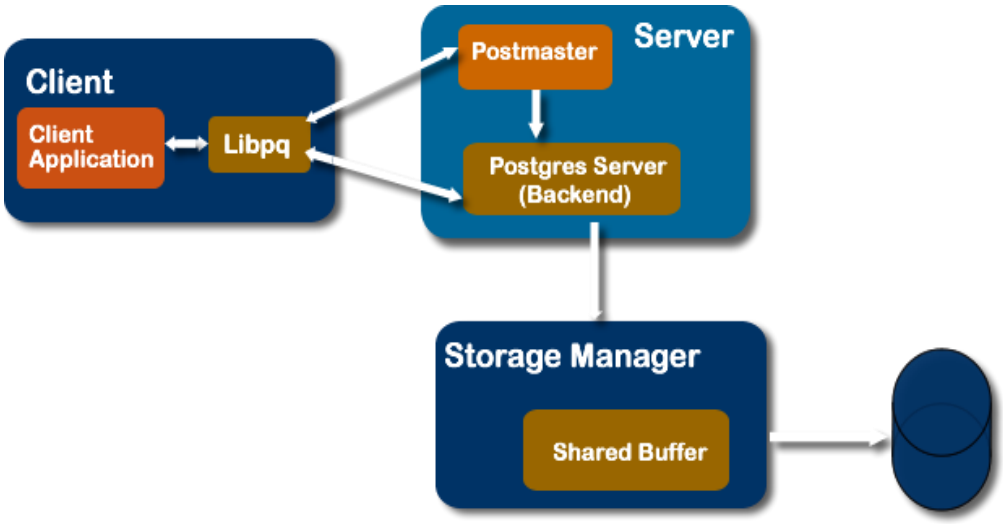


Figura 10: PostgreSQL System Concept Architecture

Fuente: (Postgresql, 2010)

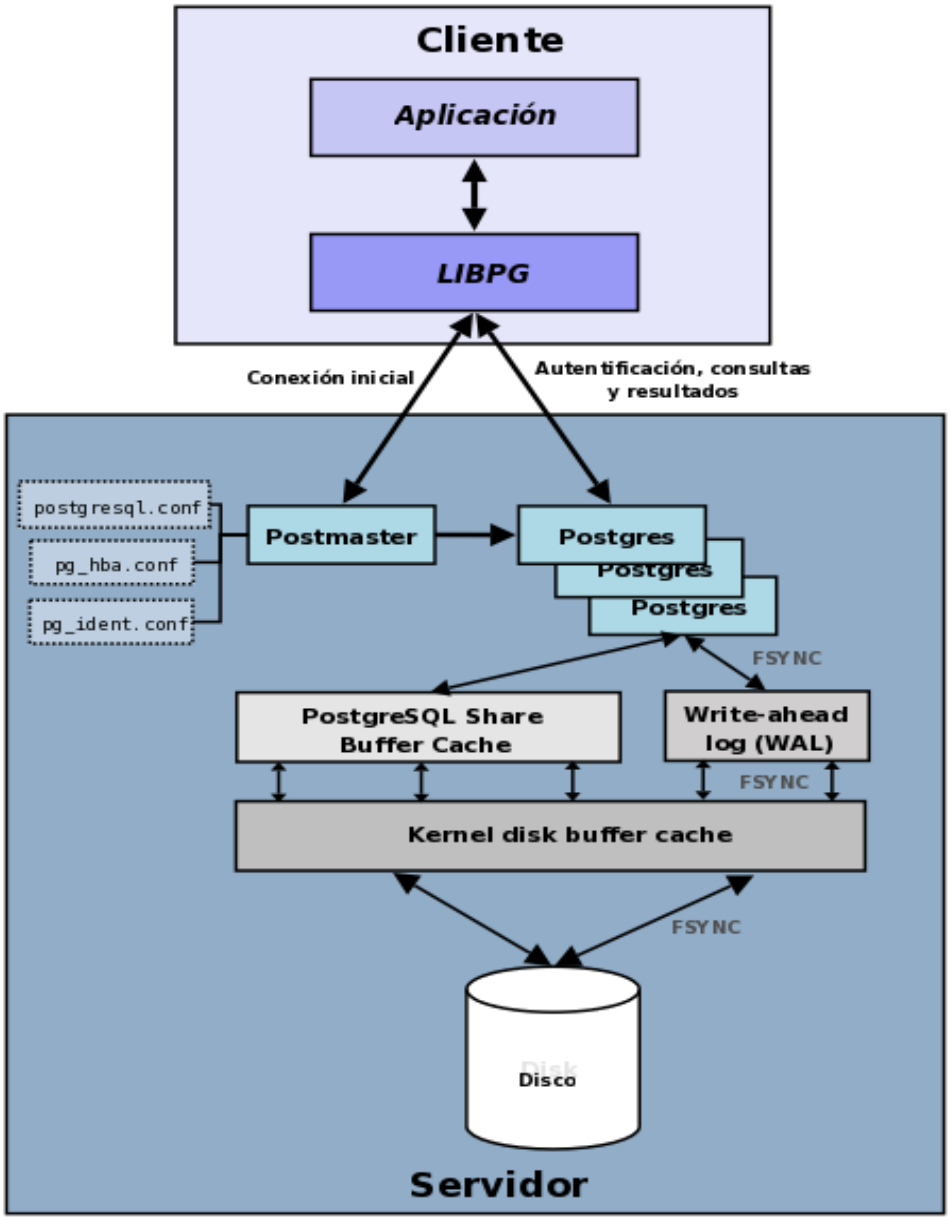


Figura 11: Componentes de PostgreSQL.

Fuente: (Postgresql, 2010)

1. **Aplicación cliente**: Esta es la aplicación cliente que utiliza PostgreSQL como administrador de bases de datos. La conexión puede ocurrir vía TCP/IP o sockets locales.
2. **Demonio postmaster**: Este es el proceso principal de PostgreSQL. Es el encargado de escuchar por un puerto/socket por conexiones entrantes de clientes. Tambien es el encargado de crear los procesos hijos que se encargaran de autentificar estas peticiones, gestionar las consultas y mandar los resultados a las aplicaciones clientes

3. **Ficheros de configuración**: Los 3 ficheros principales de configuración utilizados por PostgreSQL, postgresql.conf, pg_hba.conf y pg_ident.conf
4. **Procesos hijos postgres**: Procesos hijos que se encargan de autentificar a los clientes, de gestionar las consultas y mandar los resultados a las aplicaciones clientes
5. **PostgreSQL share buffer cache**: Memoria compartida usada por PostgreSQL para almacenar datos en caché.
6. **Write-Ahead Log (WAL)**: Componente del sistema encargado de asegurar la integridad de los datos (recuperación de tipo REDO)
7. **Kernel disk buffer cache**: Caché de disco del sistema operativo
8. **Disco**: Disco físico donde se almacenan los datos y toda la información necesaria para que PostgreSQL funcione. (Ing. Maribel Sabana Mendoza, 2010)

2.6.1.1. Características

1. **Atomicidad** (Indivisible) es la propiedad que asegura que la operación se ha realizado o no, y por lo tanto ante un fallo del sistema no puede quedar a medias.
2. **Consistencia** es la propiedad que asegura que sólo se empieza aquello que se puede acabar. Por lo tanto se ejecutan aquellas operaciones que no van a romper la reglas y directrices de integridad de la base de datos.
3. **Aislamiento** es la propiedad que asegura que una operación no puede afectar a otras. Esto asegura que dos transacciones sobre la misma información nunca generará ningún tipo de error.
4. **Durabilidad** es la propiedad que asegura que una vez realizada la operación, ésta persistirá y no se podrá deshacer aunque falle el sistema
5. Corre en casi todos los principales sistemas operativos: Linux, Unix, BSDs, Mac OS, Beos, Windows, etc.
6. Documentación muy bien organizada, pública y libre, con comentarios de los propios usuarios.
7. Soporte nativo para los lenguajes más populares del medio: PHP, C, C++, Perl, Python, etc.
8. Drivers: Odbc, Jdbc, .Net, etc.
9. Soporte de todas las características de una base de datos profesional (triggers, storeprocedures – funciones, secuencias, relaciones, reglas, tipos de datos definidos por usuarios, vistas, vistas materializadas, etc.)
10. Implementación del estándar SQL92 y SQL99.
11. Soporte de protocolo de comunicación encriptado por SSL

12. Extensiones para alta disponibilidad, nuevos tipos de índices, datos espaciales, minería de datos, etc.
13. Incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes, etc.
14. Permite la declaración de funciones propias, así como la definición de disparadores y la creación de tipos de datos propios.
15. Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.
16. Diseñada para entornos con altos volúmenes de tráfico. (Postgresql, 2010)

Algunos de los límites de PostgreSQL son:

Límite	Valor
Máximo tamaño base de dato	Ilimitado (Depende del sistema de almacenamiento)
Máximo tamaño de tabla	32 TB
Máximo tamaño de fila	1.6 TB
Máximo tamaño de campo	1 GB
Máximo número de filas por tabla	Ilimitado
Máximo número de columnas por tabla	250 - 1600 (dependiendo del tipo)
Máximo número de índices por tabla	Ilimitado

Tabla 4: Límites de PostgreSQL

Fuente: (Postgresql, 2010)

2.7. Plataformas y Frameworks

Para el desarrollo del aplicativo se optó por la utilización de plataformas y frameworks que se acoplen a las necesidades requeridas por de usuarios de la cooperativa, además se consideró la documentación existente y la facilidad de uso.
Dichas plataformas y frameworks se describen a continuación:

2.7.1. Java

La plataforma Java es el nombre de un entorno o plataforma de computación originaria de Sun Microsystems - Oracle, capaz de ejecutar aplicaciones desarrolladas usando el lenguaje de programación Java u otros lenguajes que compilen a bytecode y un conjunto de herramientas de desarrollo.
En este caso, la plataforma no es un hardware específico o un sistema operativo, sino más bien una máquina virtual encargada de la ejecución de las aplicaciones, y un conjunto de bibliotecas estándar que ofrecen una funcionalidad común. (wikipedia Java, 2010)

2.7.1.1. Características:

1. Bibliotecas de clases libre y reutilizables.
2. No es necesario reescribir los programas cuando se cambia la plataforma.
3. Se pierde menos tiempo depurando los programas.
4. Java es compilado. (Birnam Stewart, 2011)

2.7.2. Java EE (Java Platform, Enterprise Edition)

Anteriormente era conocida como J2EE hasta la versión 1.4, Java Enterprise Edition (JEE) es una plataforma que fue creada exclusivamente para creación de aplicaciones empresariales distribuidas, las cuales estarán escritas en el lenguaje de programación Java y que se ejecutan en un servidor de aplicaciones como Apache Tomcat, Jboss, etc.
Una aplicación distribuida es una aplicación con diferentes componentes que se ejecutan en entornos separados es decir en diferentes plataformas pero enlazadas a través de una red. JEE es un conjunto de especificaciones y permiten soluciones para el desarrollo, despliegue y gestión de aplicaciones de n capas.

Conjunto de especificaciones quiere decir que cualquier usuario o fabricante puede coger estas especificaciones y desarrollar un producto final que las cumpla con estas especificaciones un ejemplo de ellos es Glassfish ya que permite la ejecución de aplicaciones desarrolladas en la plataforma JEE, el cual incluye varias especificaciones de API, tales como JDBC, RMI, e-mail, JMS, Servicios Web, XML, etc., y define cómo coordinarlos. Java EE también configura algunas especificaciones únicas para Java EE para componentes. Estas incluyen Enterprise JavaBeans, servlets, jsp y varias tecnologías de servicios web. (wikipedia Java, 2010)

Java EE se puede dividir en varios contenedores en los cuales se pueden describir varios APIs para cada contenedor tal y como se muestra en el siguiente grafico que indica las relaciones entre los contenedores Java EE.

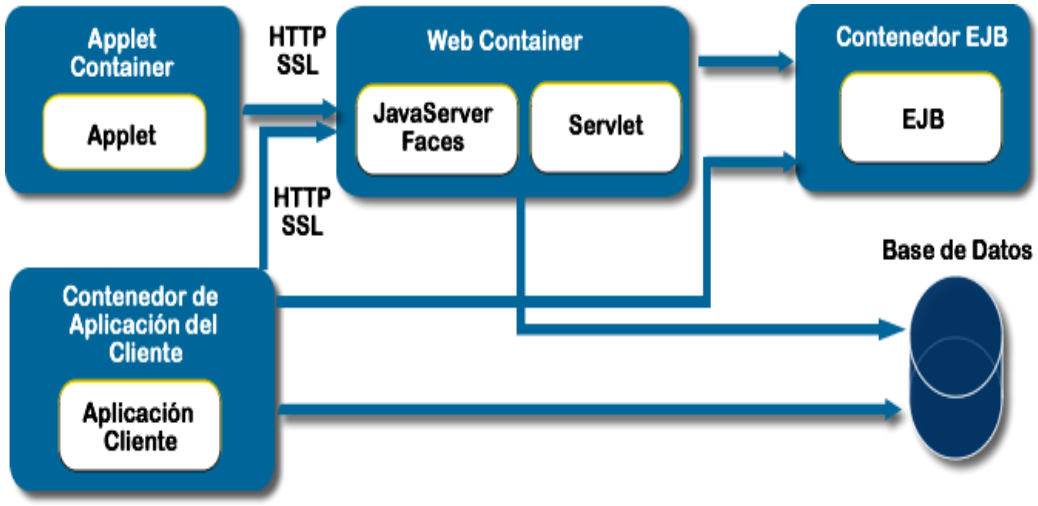


Figura 12: Contenedores de JEE

Fuente: (wikipedia Java, 2010)

2.7.2.1. Características de JEE

Las principales características de JEE entre las más relevantes e importantes son las siguientes:

1. **Portabilidad.-** Esta característica es una de las más importantes de la plataforma JEE ya que gracias a esta podemos ejecutar el software en diferentes plataformas, y además de esto podemos reutilizar el código fuente del software para crearse un nuevo código cuando el software pasa de una plataforma a otra. En cuanto mayor sea la portabilidad menor será la dependencia del software con respecto a la plataforma que se utiliza.
2. **Accesibilidad.-** Gracias a que JEE se mantiene como proyecto de software libre podemos disfrutar de esta plataforma sin pagar ningún tipo de licencias.

3. **Escalabilidad.-** Como característica de un sistema es generalmente un poco difícil de definir en cualquier caso, pero podemos decir que la plataforma JEE nos permite realizar aplicaciones totalmente escalables ya que a futuro podemos seguir añadiendo funcionalidades al sistema creado y sin ningún problema debido a que las aplicaciones son fáciles de entender por su separación en capas.
4. **Integrable.-** Permite que aplicaciones creadas con otras tecnologías formen parte de una aplicación Java, que estas puedan interactuar entre sí.
5. **Seguridad.-** Si una plataforma que es destinada para la creación de aplicaciones empresariales obviamente va a tener una buena seguridad, lo que nos va a permitir tener un control sobre usuarios y no permitirles acceder a la misma funcionalidad.
6. **Concurrencia.-** Permite que dos o más usuarios puedan acceder a una aplicación sin alterar su buen funcionamiento.

2.7.3. Arquitectura de JEE

La arquitectura de JEE está basada en nociones de capas, contenedores, componentes y servicios lo que implica un modelo de aplicaciones distribuidas en diversas capas o niveles, es así que las aplicaciones JEE suelen seguir una arquitectura de 3 capas como son las siguientes:

1. Capa de presentación
2. Capa de lógica de negocio
3. Capa de acceso a datos

La visión de la arquitectura es un esquema lógico, no físico. Cuando hablamos de capas nos referimos sobre todo a servicios diferentes (que pueden estar físicamente dentro de la misma máquina e incluso compartir servidor de aplicaciones y JVM. (Richard Monson-Haefel, 2011)

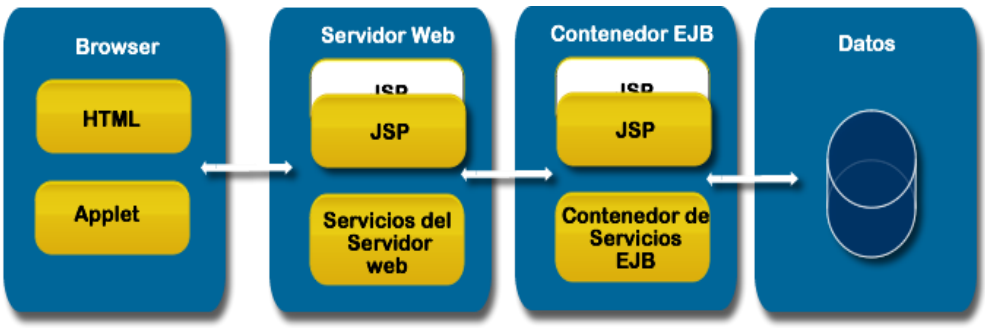


Figura 13: Arquitectura Java EE Web

Fuente: (Richard Monson-Haefel, 2011)

a) **Capa de Presentación.-** Esta capa es la que se encuentra visible a los ojos de los usuarios, es decir es la capa que donde interactúa el usuarios con el sistema.
b) **Capa de Lógica del negocio.-** En esta capa es donde se realizan todas las transacciones y operaciones con los datos obtenidos de la capa de acceso a datos para mostrarlos en la capa de presentación.
c) **Capa de acceso a datos.-** En esta capa es donde se realiza todas las operaciones CRUD de una aplicación, es decir que se realizan los métodos que acceden directamente a nuestro servidor de base de datos.

2.7.4. Netbeans IDE

NetBeans es un entorno de desarrollo integrado libre y gratuito sin restricciones de uso, hecho principalmente para el lenguaje de programación Java, existe además un número importante de módulos para extenderlo.
NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento. Sun Micro Systems fundó el proyecto de código abierto Netbeans en junio de 2000 y hoy en dia Sun Mricro Systems – Oracle continúa siendo el patrocinador principal de los proyectos.
La plataforma Netbeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos.
Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las API's de Netbeans y un archivo especial (manifest file) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos, debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma Netbeans pueden ser extendidas fácilmente por otros desarrolladores de software. (wikipedia NetBeans, 2010)

2.7.4.1. Características

1. Facilidad de uso.
2. Facilidad de instalación.
3. Característica/funciones extras.
4. Manual de instrucciones.
5. Soporte de fábrica.

2.7.5. JSF (JavaServer Page)

El desarrollo de aplicaciones o sistemas web sin el uso de una metodología o un patrón de diseño que permita mantener un orden y una separación de la codificación y demás elementos hace que en muchas ocasiones se mezcle en un mismo archivo JSP la interfaz de usuario, las reglas de validación, el acceso a la base de datos, etc., lo cual impide el mantenimiento de la aplicación y a la vez acorta su tiempo de vida útil.

Es por ello que se hace necesaria la utilización de herramientas que faciliten el desarrollo y nos permitan trabajar de manera ordenada. JSF nos ofrece un marco de trabajo que nos permite desarrollar aplicaciones, separando las diferentes capas de una arquitectura: presentación, reglas y entidades de negocio. (Ndeveloper, 2011)

JSF es un marco de trabajo que simplifica el desarrollo de interfaces de usuario en aplicaciones web basadas en tecnología JEE y en el patrón MVC para ello usa como alternativa JSP como la tecnología que permite hacer el despliegue de las páginas.

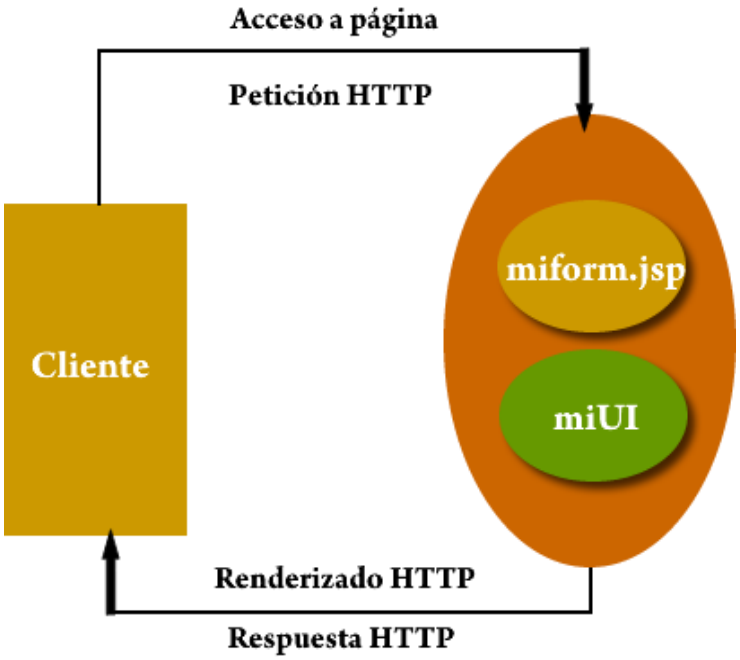


Figura 14: Diagrama de una aplicación JSF

Fuente: Propia

JavaServer Faces pretende facilitar la construcción de estas aplicaciones proporcionando un entorno de trabajo (framework) vía web que gestiona las acciones producidas por el usuario en su página HTML y las traduce a eventos que son enviados al servidor con el objetivo de regenerar la

página original y reflejar los cambios pertinentes provocados por dichas acciones. (Ndeveloper, 2011)

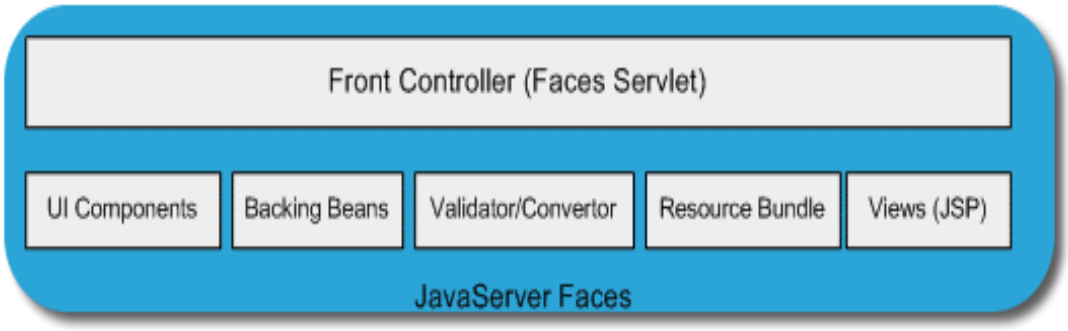


Figura 15: Componentes JSF

Fuente: (Ndeveloper, 2011)

2.7.5.1. Características

1. Útil en aplicaciones con tecnología basada en MVC.
2. Utiliza páginas JSP para generar las vistas, añadiendo una biblioteca de etiquetas propia para crear los elementos de los formularios HTML.
3. Asocia a cada vista con formularios un conjunto de objetos java manejados por el controlador (managed beans) que facilitan la recolección, manipulación y visualización de los valores mostrados en los diferentes elementos de los formularios.
4. Es extensible, pudiendo crearse nuevos elementos de la interfaz o modificar los ya existentes.
5. JSF tiene un alto nivel de abstracción que permite una mayor separación entre el diseño y el código permitiendo dividir realmente el desarrollo de una aplicación en diseño y desarrollo.
6. JSF incluye un conjunto central de componentes y herramientas que facilitan la creación de "widgets" estándar como "text boxes" o "radio buttons".
7. La comunicación entre los componentes, la entrada de datos y la validación forma parte de la tecnología, es decir, no hay que implementar nada de esto como ocurre en otras tecnologías.
8. La visualización de los mensaje de validación y errores de conversión también están estandarizados incluso soportan internacionalización.
9. JSF permite programar y tratar la vista (la interfaz de usuario) a través de componentes y está basada en eventos (pulsación de un botón, cambio en el valor de un campo, etc.).
10. JSF es muy flexible. Por ejemplo nos permite crear nuestros propios componentes.
11. La tecnología JavaServer Faces permite construir aplicaciones web que introducen realmente una separación entre el comportamiento y la presentación.
12. Proporciona una rica arquitectura para manejar el estado de los componentes, procesar los datos, validar la entrada del usuario, y manejar eventos. (Ndeveloper, 2011)

2.7.5.2. JSF con Netbeans

JSF se integra en Netbeans desde la versión 5.0, mediante bibliotecas las cuales contienen los siguientes archivos jar:

a) jsf-api.jar
b) jsf-impl.jar

Estas librerías contienen diferentes clases necesarias para usar los tags en las páginas jsp o xhtml.

En distribuciones de Netbeans actuales, el uso de jsf solo requiere crear un proyecto nuevo y especificar el uso de jsf, el cual ya viene integrado como herramienta del IDE.

2.7.6. Hibernate

La conexión e interacción de un sistema o aplicación con una base de datos es crucial dado que toda la información debe ser respaldada, para ello la disponibilidad debe ser alta, Hibernate permite usar los mismo conocimientos de programación orientada a objetos para el desarrollo de consultas y transacciones en general.

2.7.6.1. Definición

Hibernate es una herramienta de Mapeo objeto-relacional (ORM) para la plataforma Java, facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) o anotaciones en los beans de las entidades que permiten establecer estas relaciones.

Figura 16: Capa de Persistencia

Fuente: Propia

Permite diseñar objetos persistentes que podrán incluir polimorfismo, relaciones, colecciones, y un gran número de tipos de datos. De una manera muy rápida y optimizada podremos generar BDD en cualquiera de los entornos soportados: Oracle, DB2, MySql, etc., y lo más importante de todo, es open source, lo que supone, entre otras cosas, que no tenemos que pagar nada por adquirirlo.

Uno de los posibles procesos de desarrollo consiste en, una vez tengamos el diseño de datos realizado, mapear este a ficheros XML siguiendo la DTD de mapeo de Hibernate, desde estos podremos generar el código de nuestros objetos persistentes en clases Java y también crear BDD independientemente del entorno escogido.

Hibernate se integra en cualquier tipo de aplicación justo por encima del contenedor de datos, una posible configuración básica de hibernate es la siguiente:

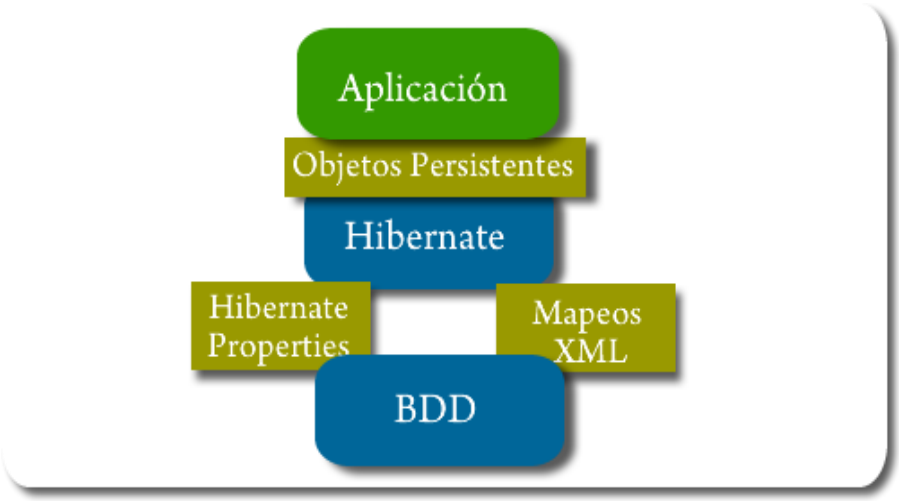


Figura 17: Configuración de Hibernate

Fuente: Propia

2.7.6.2. Características

- **a)** Usado en las bases de datos (modelo relacional).
- **b)** Uso en la memoria de la computadora (orientación a objetos).
- **c)** Uso de características de la POO.
- **d)** Hibernate convertirá los datos entre los tipos utilizados por Java y los definidos por SQL.
- **e)** Hibernate genera las sentencias SQL.
- **f)** Lenguaje propio de consulta de datos llamado HQL.
- **g)** ORM open source.
- **h)** Uso de Anotaciones o XML.

2.7.7. RichFaces como framework RIA.

RichFaces es un framework de código abierto que añade la capacidad de ajax en aplicaciones JSF sin recurrir a JavaScript.

RichFaces aprovecha al máximo el ciclo de vida de JavaServer Faces, es decir la validación, las instalaciones de conversión y gestión de los recursos estáticos y dinámicos. Los componentes de RichFaces vienen listos para su uso out-of-the-box, por lo que los desarrolladores pueden ahorrar tiempo de inmediato para aprovechar las características de los componentes para crear aplicaciones Web que proporcionan mejoras en la experiencia del usuario. RichFaces también incluye un fuerte apoyo para la skinnability de aplicaciones JSF. (Jboss org, 2011)

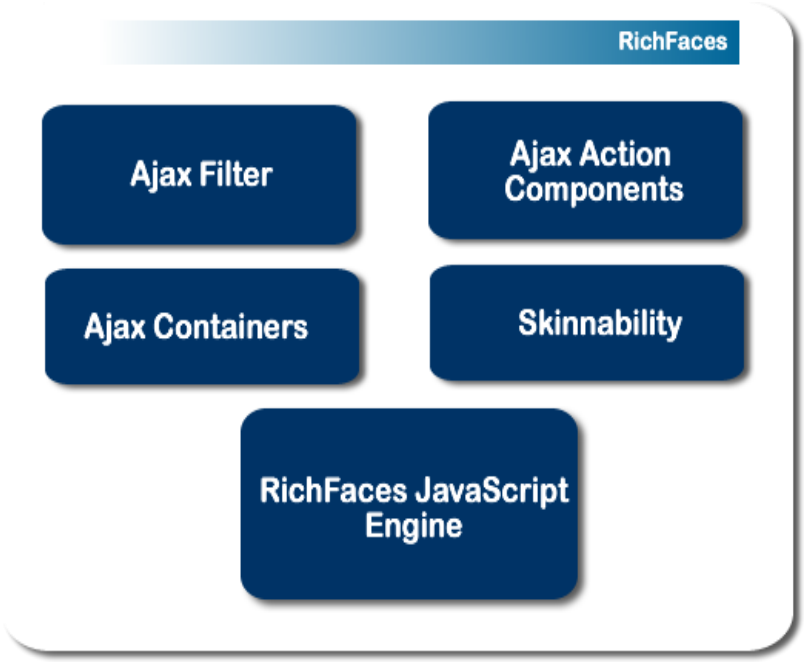


Figura 18: RichFace

Fuente: (Jboss org, 2011)

2.7.7.1. Características.

1. Permite intensificar el conjunto de beneficios JSF al trabajar con ajax. RichFaces se integra plenamente en el ciclo de vida de JSF. Mientras que con otros marcos sólo se dará acceso a las instalaciones de managed bean, las ventajas de RichFaces son la acción del cambio de valor de los listeners, dado que los validadores se invocan del lado del servidor y se convierten durante el ciclo request-response de ajax.
2. Se puede añadir más capacidad de ajax a los componentes ya existentes en aplicaciones JSF. El framework proporciona dos bibliotecas de componentes (Core ajax y UI).

La biblioteca central establece la funcionalidad de ajax en páginas ya existentes, y con ello no existe la necesidad de escribir ningún código JavaScript para reemplazar los componentes existentes con otras nuevas funciones ajax. RichFaces permite página de apoyo de todo el ajax en lugar del tradicional componente de apoyo de todo y le da la oportunidad de definir el evento en la página. Un evento invoca una petición ajax y las áreas de la página se sincronizan con el árbol de componentes JSF después de cambiar los datos en el servidor por la petición ajax, en relación con los eventos activados en el cliente.

3. Con RichFaces es posible crear vistas rápidas de carácter complejo basándose en los componentes. Además es posible utilizarlas sin problemas con otras bibliotecas de otros fabricantes, así que se tiene más opciones para el desarrollo de aplicaciones.
4. Permite escribir componentes propios y personalizados con una función de soporte para ajax. Los recursos del paquete de aplicaciones con las clases de Java. Además de su núcleo, la funcionalidad de ajax RichFaces ofrece un soporte avanzado para la gestión de los recursos diferentes: imágenes, código JavaScript y hojas de estilo CSS. El marco de recursos hace posible empaquetar fácilmente estos recursos en archivos JAR, junto con el código de sus componentes personalizados.
5. Generar fácilmente recursos binarios on-the-fly. Marco de recursos puede generar imágenes, sonidos, hojas de cálculo Excel, etc. on-the-fly para que sea posible, por ejemplo para crear imágenes con el enfoque familiar de la "Java Graphics2D" de la biblioteca.
6. Crear una interfaz de usuario moderna rica look-and-feel con skin basado en la tecnología. RichFaces proporciona una función que permite skinnability fácil definir y gestionar los diferentes esquemas de color y otros parámetros de la interfaz de usuario con la ayuda de los parámetros del skin con nombre.
7. RichFace permite probar y crear los componentes, actions, listeners y las páginas a la vez. El marco de pruebas no sólo pondrá a prueba los componentes, sino también cualquier otra funcionalidad del lado del servidor o del lado del cliente, incluyendo el código JavaScript. Lo que es más, que va a hacer todo esto sin el despliegue de la aplicación de prueba en el contenedor de Servlets. (Jboss org, 2011)

2.7.7.2. Requisitos Técnicos.

- **a)** Java.
- **b)** JavaServer Faces.
- **c)** Servidor de aplicaciones Java o el contenedor de servlets.
- **d)** Navegador (en el lado del cliente).
- **e)** Framework RichFaces.

2.7.7.3. Compatibilidad.

a) JDK 1.5 o superior.

2.7.7.4. Servidores importantes que soportan.

a) Apache Tomcat 5,5 a 6,0.
b) BEA WebLogic 9,1 a 10,0.
c) Sun Application Server 9 (J5EE).
d) GlassFish V2, V3.
e) JBoss 4.2.x – 5.
f) Websphere 7.0. y superior.
g) Geronimo 2.0 o superior.

2.7.7.5. Navegadores Compatibles

Linux	Mac OS
Firefox 3.0 y superior.	Safari 3.0 o superior
Opera 9.5 y superiores	Firefox 3.5 y superior
Windows	
Firefox 3.0 y superior.	
Google Chrome.	
Opera 9.5 y superiores.	
Safari 3.0 o superior.	

Tabla 5: Navegadores Compatibles a RichFace

Fuente: (Jboss org, 2011)

2.7.7.6. Bibliotecas necesarias.

a) commons-beanutils-1.7.0.jar
b) commons-collections-3.2.jar
c) commons-digester-1.8.jar
d) commons-logging-1.0.4.jar
e) jhighlight-1.0.jar

2.8. Herramienta de Reportes

Para la creación de reportes del proyecto se elige a JasperReports que es una de las herramientas más populares para la emisión de reportes que tiene integración con Java, además de contar con integración HQL (Hibernate Query Lenguage) para Hibernate, lo cual facilita la creación de consultas a la base de datos.
Su principal propósito es crear documentos de tipo páginas, preparados para imprimir en una forma simple y flexible. Se usa comúnmente con IReport, un front-end gráfico de código abierto para la edición de informes.
Está bajo GNU, por lo que es software libre. (Jsanroman, 2012)

2.8.1. JasperReports

Es una herramienta gratuita y open source que se compone de un conjunto de librerías Java para facilitar la generación de informes en nuestras aplicaciones tanto web como de escritorio. Los informes se definen en un fichero XML el cual será compilado por las librerías jasperreport y generarán un fichero .jasper que utilizaremos para rellenar y mostrarlo en los informes finales

JasperReports está completamente escrita en Java y se puede utilizar en una gran variedad de aplicaciones de Java, incluyendo JEE o aplicaciones web, para generar contenido dinámico. El Sistema Operativo para JasperReports no es un problema ya que funciona perfectamente tanto en plataformas de Microsoft Windows como en GNU Linux y en ambas plataformas se ha comprobado su buen funcionamiento.

2.8.2. Requerimientos de JasperReports

Para el funcionamiento de JasperReports se necesita básicamente 2 cosas principales la primera es JDK 1.4 o una versión superior es decir la máquina virtual de Java la cual hace posible el funcionamiento de cualquier aplicación Java y como se había explicado anteriormente que JasperReports está completamente escrita en Java entonces es requisito principal tener instalado el JDK. Y también se necesita de las siguientes librerías a más de las de JasperReports para empezar a generar reportes.

1. jasperreports-javaflow-4.0.2.jar
2. jasperreports-4.0.2.jar
3. jasperreports-applet-4.0.2.jar
4. jasperreports-fonts-4.0.2.jar

Para la generación de Reportes a través de Hibernate es necesaria también la librería que hsqldb.jar

2.8.3. Funcionamiento de JasperReports

El funcionamiento de JasperReports es similar a un compilador y a un intérprete, es decir que traduce cada instrucción o sentencia del archivo XML a un lenguaje máquina e inmediatamente se ejecuta y se crea el archivo propio de JasperReports (.jasper).

El usuario se encarga de diseñar el reporte codificándolo en XML con las etiquetas y atributos definidos en un archivo llamado jasperreports.dtd el cual es parte de JasperReports.

En el archivo XML el usuario define básicamente todos los parámetros que se utilizaran en el reporte, describiendo así donde colocar texto, imágenes, líneas, detalles de tablas, rectángulos, cómo adquirir los datos, como realizar ciertos cálculos para mostrar campos calculados, etc.

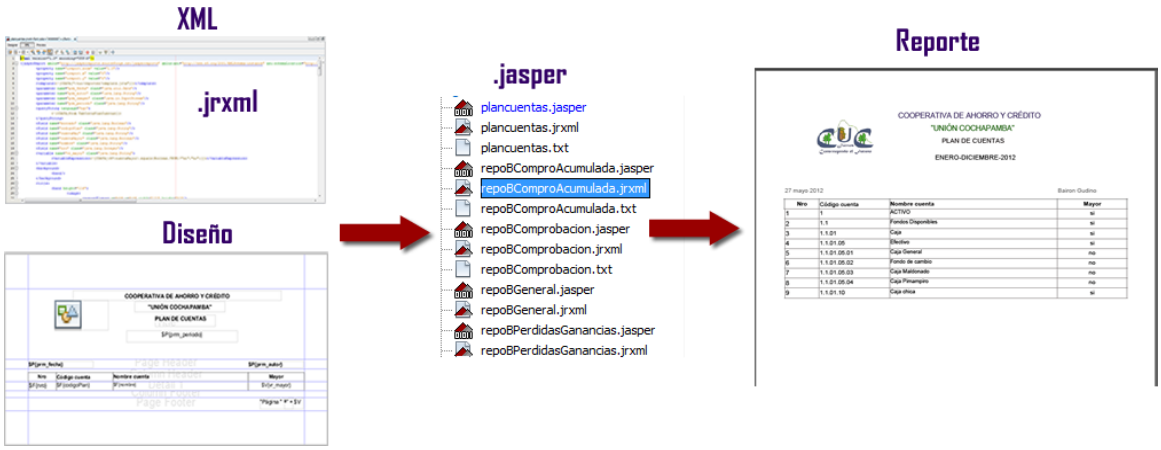
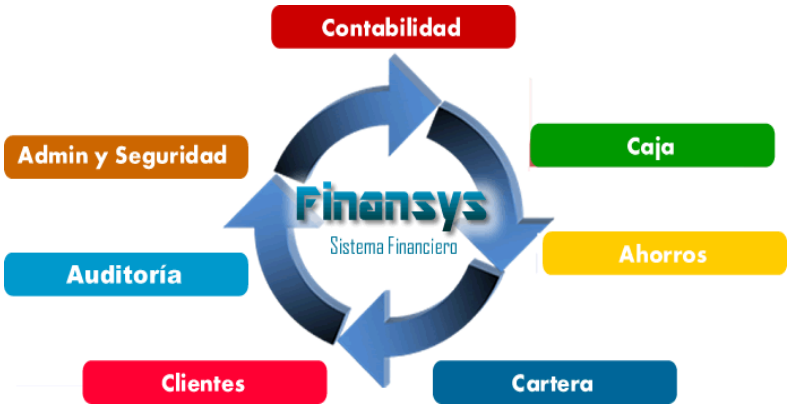


Figura 19: Reportes Jasperreport

Fuente: Propia

Capítulo III: Descripción y Funcionamiento



CONTENIDO:

1. Vista General.
2. Características del Sistema.
3. Dependencias en la Implementación.
4. Funcionamiento del Sistema.
5. Arquitectura del Sistema.
6. Diagramas Entidad Relación.
7. Diagrama General de Casos de Uso.
8. Descripción del Módulo Contabilidad.

3. Funcionamiento del Sistema

3.1. Vista General

En este documento la Administración Financiera se define como un sistema que integra los subsistemas de Administración y Seguridad, Ahorros, Cartera, Clientes, Cajas, Contabilidad, y Auditoría los cuales se encuentran interrelacionados en cuanto a su normatividad, operatividad e información que generan, manteniendo así las características y necesidades propias de cada uno de estos subsistemas.

A partir de la descripción de las funciones y procesos que generan información, se ha diseñado el Sistema FINANSYS teniendo como módulos principales los siguientes:

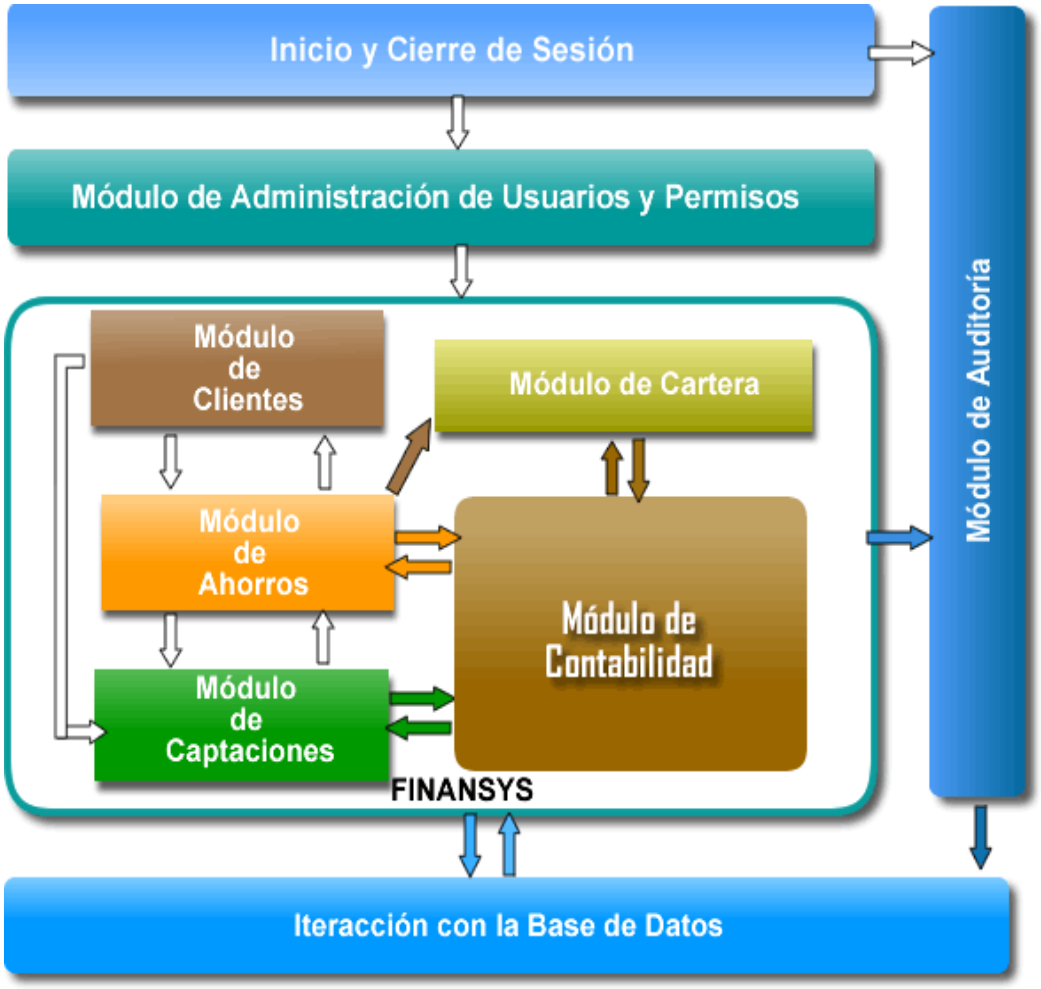


Figura 20: Módulos del Sistema

Fuente: Propia

3.2. Características del Sistema

Característica	Descripción
Administración Agrupada	Se instala y actualiza únicamente del lado del servidor ya que es una aplicación Web y no requiere que se apliquen dichos cambios a los clientes ya que los clientes consumen del servidor.
Visualización en navegadores web.	Podrá funcionar en cualquier navegador que tenga soporte para java y java Script ya que la aplicación fue desarrollada con tecnologías de código abierto como JEE (Java Enterprise Edition) pero se recomienda el uso del navegador Firefox.
Facilidades en el registro de Información	La interfaz es amigable y de fácil manejo lo que agilita el proceso de capacitaciones.
La Disponibilidad es alta	Los usuarios podrán acceder inmediatamente al sistema desde cualquier punto de la Intranet a través de cualquier navegador.
Eficiencia en la presentación de Información.	Al ser un sistema integrado se puede acceder rápidamente a la información que se encuentran en los diferentes módulos del sistema y tener toda la información necesaria para una tarea en una misma pantalla ya que se puede definir menús y roles personalizados
Fácil de Usar	El diseño y la funcionalidad son muy simples ya que consta de un menú intuitivo, fácil navegación entre opciones del sistema y su información es consistente.
Facilidad para analizar la información.	A través de los diferentes reportes y opciones de búsquedas y consultas con los que consta el sistema.
Sistema Multiplataforma	El acceso al sistema se podrá realizar desde cualquier navegador web.

Tabla 6: Características del Sistema

Fuente: Propia

3.3. Dependencias para la implementación de Finansys

Para la correcta funcionalidad se deberá implementar primeramente un ambiente de pruebas similar al de producción para evitar posibles errores y conflictos de información.

3.3.1. Costos

Para la implementación del sistema se prevé la adquisición de un servidor con características adecuadas para el funcionamiento del sistema. El sistema no aplica costos de licenciamiento por ser desarrollado con herramientas de código abierto.

3.3.2. Licenciamiento

Las tecnologías utilizadas para el desarrollo e implementación del sistema no necesitan de algún pago para la adquisición de licencias ya que el licenciamiento de estas herramientas está regido por las cláusulas de GPL establecido en el software de libre. Para el desarrollo del Sistema Integrado de Información Financiera se utilizará las siguientes herramientas:

1. Como metodología de Desarrollo RUP (Rational Unified Process).
2. PostgreSQL como servidor de Base de Datos
3. Framework Hibernate para mapeo de datos
4. JEE (Java Enterprise Edition) como plataforma de Desarrollo.
5. JSF (Java Server Faces) para la interfaz
6. RichFaces para la integración de funcionalidades Ajax en JSF.
7. HTML, XML, CSS, Ajax, JavaScript, estándares de código abierto.
8. JasperReports para presentación de Reportes en formato PDF.

3.3.3. Instalación

No existe mayor problema en la instalación ya que esta se la realizará en forma centralizada, es decir que la información estará ubicada en un solo punto llamado servidor, al cual los clientes accederán a través de un navegador web desde cualquier punto de red de la Intranet de la Institución.

3.4. Funcionamiento.

El Sistema Integrado de Información Financiera fue creado utilizando la plataforma de Desarrollo JEE (Java Enterprise Edition) para la cual es recomendable utilizar MVC (Modelo Vista Controlador) como patrón base para diseñar aplicaciones interactivas. La interface gráfica del sistema utiliza JSF y RichFaces para diseñar la parte que visualizara el usuario, teniendo como template principal el proceso de autenticación de usuario con la página index.jsp.

El flujo de la aplicación inicia por la página de login o Autenticación, si un usuario intenta acceder directamente a otra página, la aplicación verifica que no se ha autenticado y le re direcciona a la página de autenticación. Luego de autenticarse el usuario podrá navegar por las opciones que presenta cada módulo de la aplicación dependiendo del rol y el tipo de usuario.

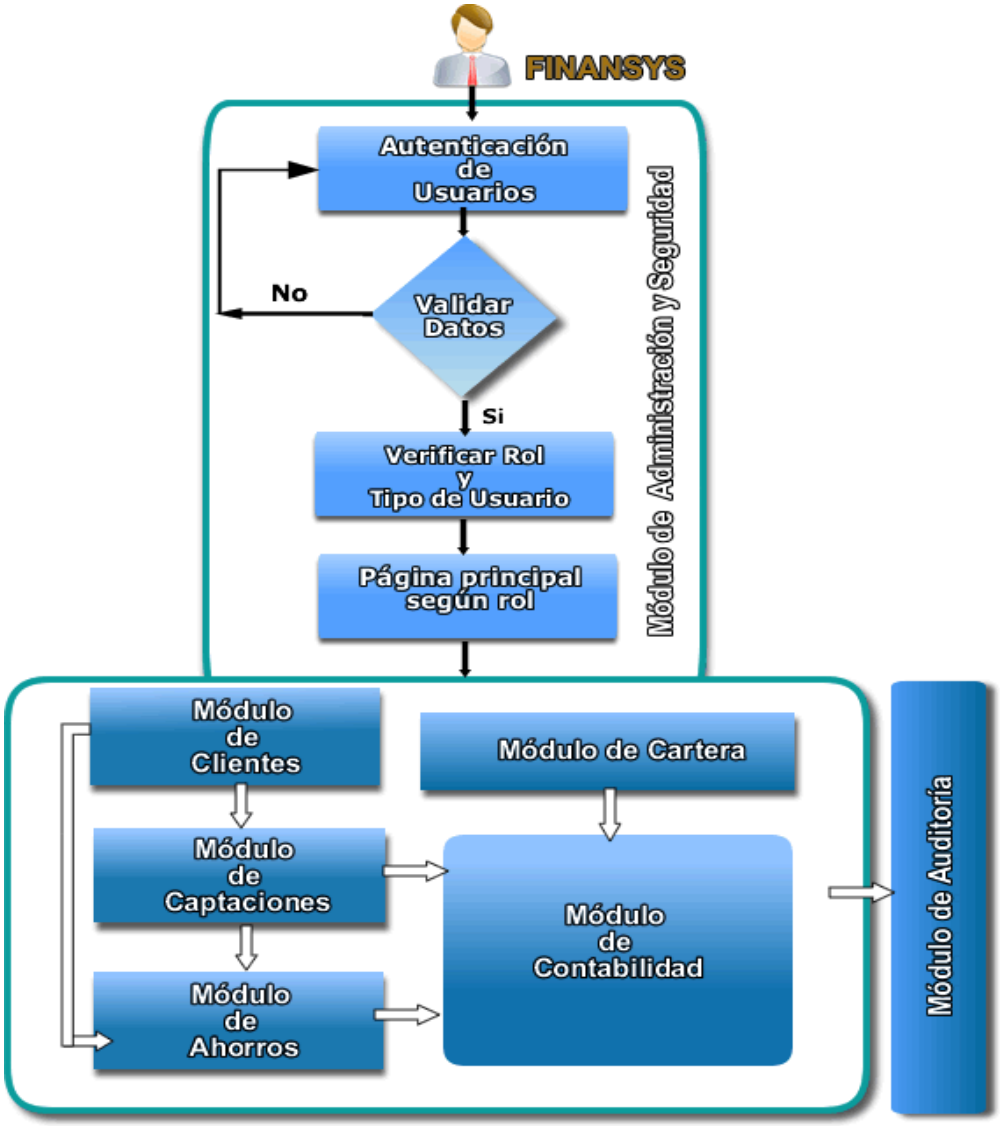


Figura 21: Funcionamiento General del Sistema

Fuente: Propia

3.5. Arquitectura del Sistema

La arquitectura MVC organiza el diseño de una aplicación interactiva mediante la separación de la presentación de los datos y el comportamiento de la aplicación.

a) El modelo representa la estructura de los datos en la aplicación, es decir que en esta capa se realizan las consultas y manipulación de datos (CRUD), además de los métodos, funciones u operaciones con estos datos.

b) La vista es la capa que presenta los datos al usuario, es decir la interfaz gráfica de la aplicación.

c) El controlador traduce las acciones de los usuarios realizadas en la vista y llama a los métodos de negocio creados en el modelo y selecciona la vista apropiada según haya sido el requerimiento del usuario.

Resumiendo se puede decir que un modelo se encarga del estado de la aplicación y su funcionalidad, una vista de la presentación de la aplicación y el controlador del comportamiento en respuesta a una acción del usuario.

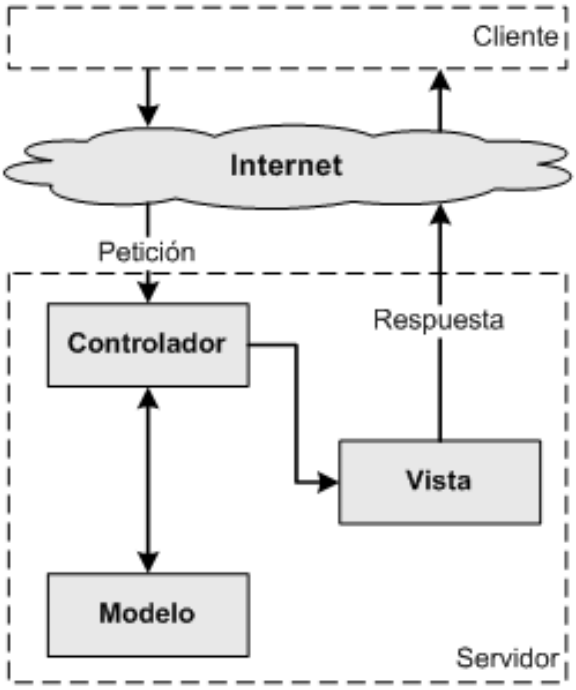


Figura 22: Arquitectura de la Aplicación

Fuente: (Libros web, 2012)

3.6. Diagrama entidad relación Finansys

El diagrama Entidad-Relación de la BDD de Finansys, se encuentra en anexos, 8.1

3.7. Descripción del Módulo Contable

A continuación se describe el Módulo Contable [MC] del sistema Finansys.

3.7.1. Módulo de Contabilidad

La Contabilidad en una cooperativa se va a encargar de estudiar, medir y analizar el patrimonio, con el fin de tomar decisiones y realizar un control de las actividades monetarias, para ello se basa en la presentación de la información, previamente registrada, de manera sistemática.

Estas técnicas producen estructuradamente información cuantitativa y valiosa, expresada en unidades monetarias acerca de las transacciones de ciertos eventos económicos identificables y cuantificables que la afectan.

La finalidad de la contabilidad es suministrar información en un momento dado y de los resultados obtenidos durante un período de tiempo, que resulta de utilidad a los usuarios de la contabilidad en la toma de sus decisiones, tanto para el control de la gestión pasada, como para las estimaciones de los resultados futuros, dotando tales decisiones de racionalidad y eficiencia

El MC de Finansys se encarga del manejo de la información financiera contable de la CUC, la información a tratar se realiza en base a las peticiones y requerimientos de los usuarios de este módulo.

En general gestiona todos los comprobantes creados en un determinado periodo contable y periodo mes de la CUC.

3.7.2. Acciones del Módulo Contable.

Para el uso del **MC** el usuario que tenga los privilegios de acceso a este módulo tendrá a disposición las acciones detalladas a continuación.

Paso	Acción
1	**Autenticarse como usuario contador**
2	**Ingresar al menú Contabilidad**
3	El menú presenta las siguientes opciones: **3.1** Plan de Cuentas. **3.2** Comprobantes. **3.3** Tareas Contables. **3.4** Control de Presupuesto. **3.5** Consultas. **3.6** Reportes.
3.1	**Administrar Plan de Cuentas** 1. Crear cuenta contable. 2. Editar cuenta contable. 3. Borrar cuenta contable.
3.2	**Administrar Comprobantes** 1. Crear comprobantes contables. 2. Editar comprobantes de todo tipo. 3. Borrar comprobantes de todo tipo. 4. Buscar comprobantes. 5. Filtrar comprobantes por número, tipo, concepto, fecha. 6. Buscar Comprobantes de periodos cerrados. 7. Crear tipos de comprobantes. 8. Editar tipos de comprobantes. 9. Borrar tipos de comprobantes. 10. Reenumerar comprobantes.
3.3	**Realizar Tareas Contables** 1. Verificar contables. 2. Crear periodos contables. 3. Modificar periodos contables. 4. Borrar periodos. 5. Usar período contable. 6. Usar periodos mes. 7. Editar periodos mes. 8. Conciliación Bancaria. 9. Mayorizar y cerrar saldos del mes. 10. Abrir comprobantes del mes. 11. Asientos de cierre de fin de año.

6	**Administrar Control de Presupuesto** 1. Generar cuentas de control de presupuesto. 2. Revisar cuentas de control de presupuesto.
7	**Realizar Consultas** 1. Saldos por cuenta. 2. Movimientos por cuenta. 3. Movimientos por fecha.
8	**Generar Reportes** 1. Plan de cuentas. 2. Comprobantes continuos. 3. Balance de comprobación. 4. Balance de comprobación acumulada. 5. Balance General. 6. Balance de Pérdidas y Excedentes. 7. Libro diario. 8. Movimientos por cuenta.

Tabla 7: Acciones del MC

Fuente: Propia

3.7.3. Funcionamiento del Módulo Contabilidad (MC)

La forma general en la que funciona el módulo contable se representa en el siguiente gráfico:

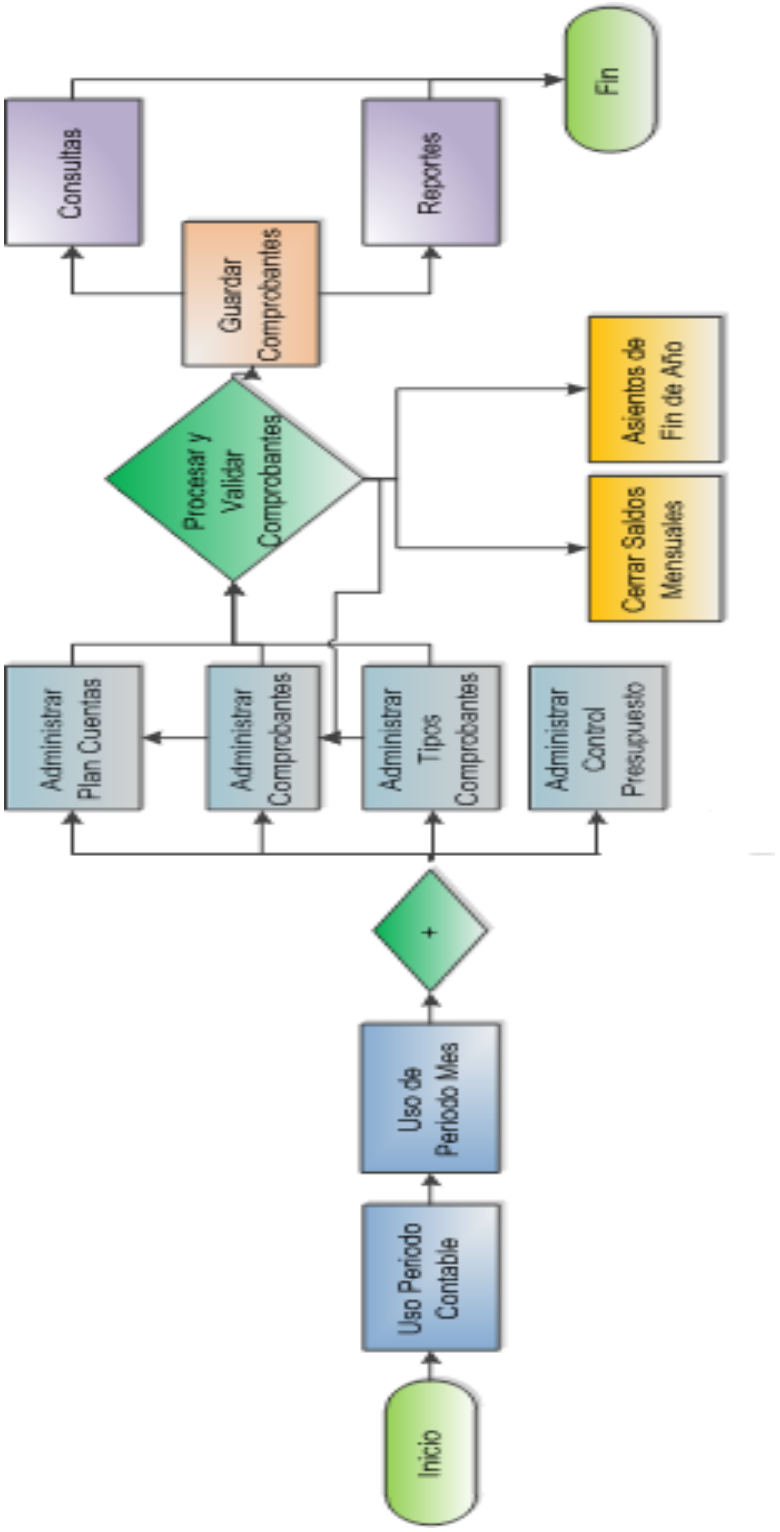


Figura 23: Funcionamiento MC

Fuente: Propia

3.7.4. Casos de Uso: MC – Finansys

A continuación se presenta el caso de uso del módulo contable, el detalle y la descripción del mismo se podrá observar en los Anexos.

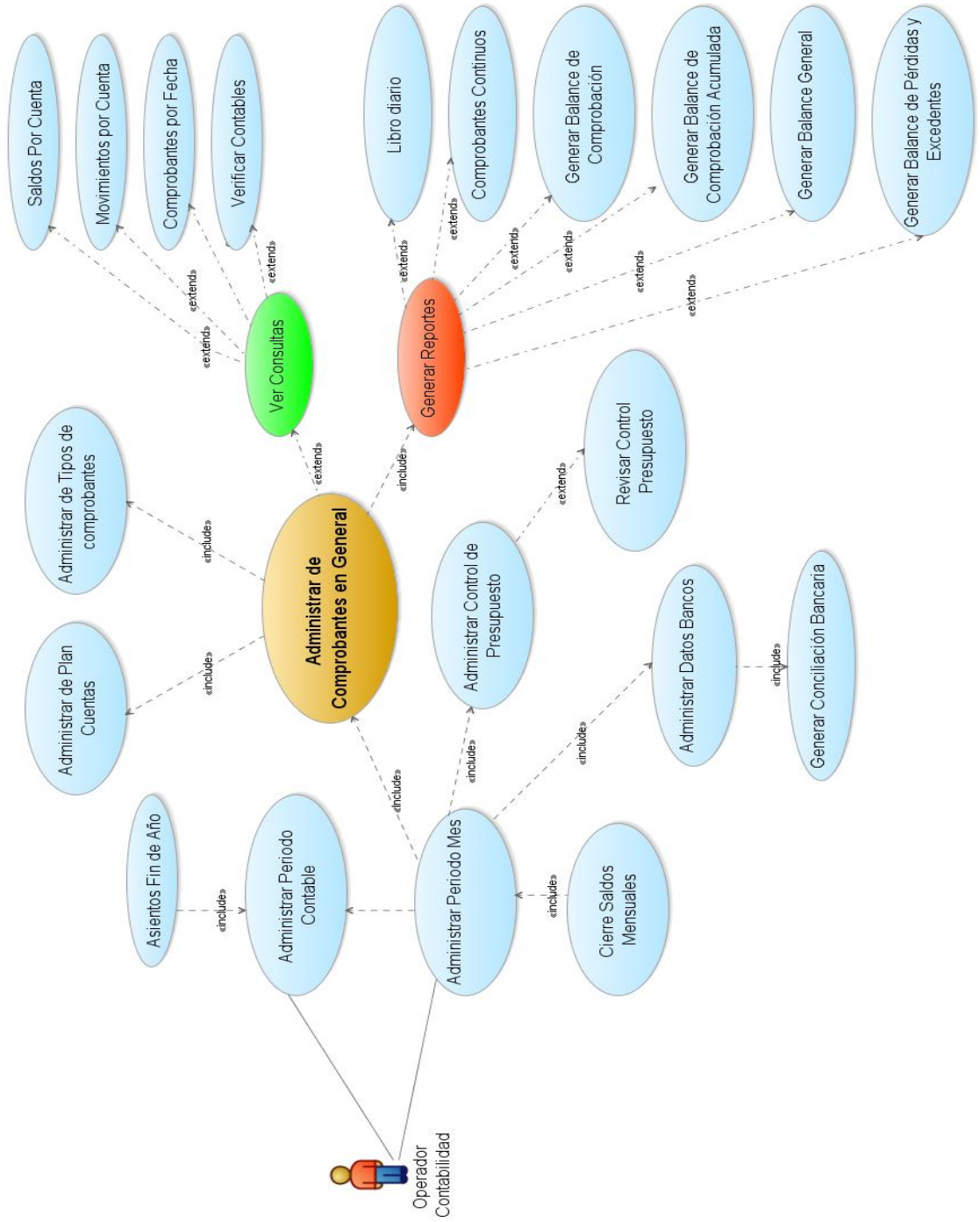


Figura 24: U.C. Módulo contable

Fuente: Propia

3.7.5. Especificación de Caso de Uso: Administración de Periodos Contables

3.7.5.1. Descripción Breve

El caso de uso describe la Administración de los Periodos Contables, el proceso de uso y la vital importancia de un periodo contable para Finansys.

3.7.5.2. Flujo Básico de Eventos

1.- Los periodos contables deberán contener las fechas de inicio y fin del periodo contable las cuales son usadas para la creación de los periodos mes.

La tabla de periodos contables se estructurará de la siguiente manera:

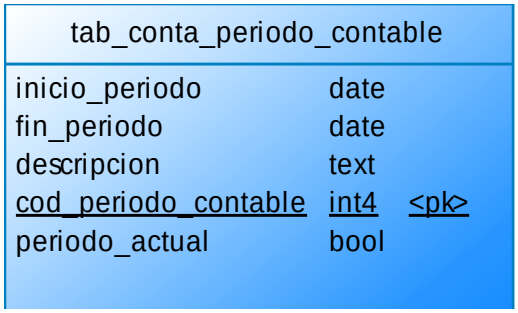


Figura 25: Tabla periodo contable

Fuente: Propia

TAB_CONTA_PERIODO_CONTABLE		
Campo	**Tipo**	**Observación**
COD_PERIODO_CONTABLE	INTEGER	PK_PERIODO_CONTABLE AUTOINCREMENT
INICIO_PERIODO	DATE	UK_INICIO_PERIODO Fecha inicio de periodo
FIN_PERIODO	DATE	Fecha fin de periodo contable
DESCRIPCION	TEXT	Descripción del periodo contable
PERIODO_ACTUAL	BOOLEAN	TRUE--- Periodo actual

Tabla 8: Tabla descripción periodo contable

Fuente: Propia

Nota: Toda sucursal o agencia utilizarán un único periodo contable para cada año.

2.- El Sistema se iniciará con un periodo contable registrado en la base de datos por defecto, el cual a su vez hace referencia a un periodo mes también creado por defecto y perteneciente al periodo contable (ver caso: 02UCAdministrarPeriodoMes).

Figura 26: Inicio periodo contable

Fuente: Propia

3.- Una vez en el sistema el Operador Contable puede administrar los periodos contables, los periodos contables son de vital importancia dado que todos los comprobantes generados en MC-Finansys y en los demás módulos se registran perteneciendo a un periodo contable y por ende a un periodo mes.

Figura 27: Administrar Periodo

Fuente: Propia

3.7.5.3. Precondiciones

3.7.5.3.1. Registro previo de Periodo Contable

El sistema deberá contener como primer registro un periodo contable por defecto el será usado como periodo inicial.

3.7.5.3.2. Registro previo de Periodo Mes

El sistema deberá contener un periodo mes (ver caso: 02UCAdministrarPeriodoMes), el cual pertenece a un periodo contable y a una sucursal.

3.7.5.3.3. Registro previo de Sucursales

El sistema deberá tener registradas sucursales para la creación de periodos mensuales

3.7.6. Especificación de Caso de Uso: Administración de Periodos Mes

3.7.6.1. Descripción Breve

El caso de uso describe el proceso de revisión y uso de un Periodo mes, además la vital importancia que tienen dentro de Finansys.

3.7.6.2. Flujo Básico de Eventos

1. Al iniciar el sistema, inicia con un periodo mes por defecto el cual pertenece a un periodo contable (ver caso: 02UCAdministrarPeriodoContable).
2. La tabla de un periodo mes deberá contener las fechas de inicio del periodo mes las cuales son de vital importancia para poder crear comprobantes de todos los módulos.

La tabla para los periodos mensuales se estructurará de la siguiente manera:

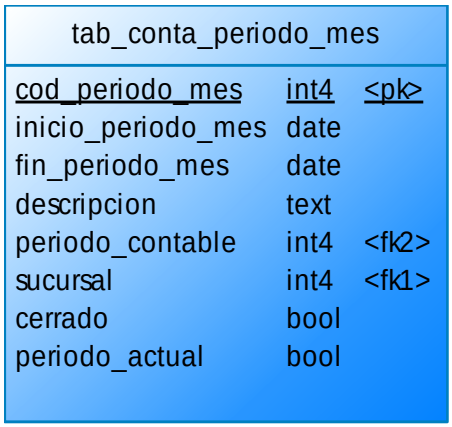


Figura 28: Tabla periodo mes

Fuente: Propia

TAB_CONTA_PERIODO_MES		
Campo	**Tipo**	**Observación**
COD_PERIODO_MES	INTEGER	PK_PERIODO_MES
INICIO_PERIODO_MES	DATE	
FIN_PERIODO_MES	DATE	FECHA FIN DE PERIODO MES
DESCRIPCION	TEXT	DESCRIPCIÓN DEL PERIODO MES
PERIODO_CONTABLE	INTEGER	FK_PERIODO_MES_PERIODO_CONTABLE
SUCURSAL	INTEGER	FK_PERIODO_MES_ADMINISTRACION_SUCURSAL
CERRADO	BOOLEAN	
PERIODO_ACTUAL	BOOLEAN	

Tabla 9: Tabla periodo mes descripción

Fuente: Propia

Nota: Los datos de esta tabla no podrán ser borrados desde las vistas, solo pueden ser modificadas.

3. Una vez en el sistema el Operador Contable puede revisar los periodos contables y modificar en fechas o uso actual si así lo desea, No se permitirá borrar, ni crear un periodos mes dado que estos se crean al cerrar saldos mensuales y de permitirse crearlos podrían generarse inconsistencias, la interfaz de usuario se verá de la siguiente manera:

Figura 29: Administración periodos mes

Fuente: Propia

3.7.6.3. Precondiciones

3.7.6.3.1. Registro previo de Periodo Contable

El sistema deberá contener como primer registro un periodo contable por defecto el cual será usado como periodo inicial, además deberá existir un periodo mes con fecha inicial mas no fecha final, esta se crea al cerrar los saldos mensuales(ver caso: 01CUAdministrarPeriodoContable).

3.7.7. Especificación de Caso de Uso: Administración Plan de Cuentas

3.7.7.1. Descripción Breve

El caso de uso describe el proceso de administración del plan de cuentas, necesario para la mayoría de módulos de Finansys.

3.7.7.2. Flujo Básico de Eventos

El usuario podrá administrar un plan de cuentas el cual se estructura en una tabla con los siguientes campos.

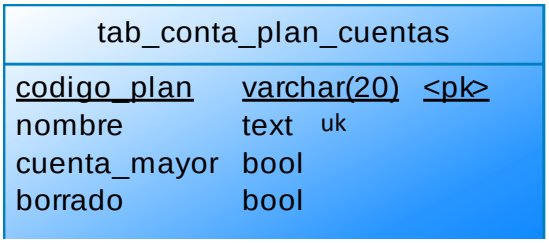


Figura 30: Tabla plan de cuentas

Fuente: Propia

TAB_CONTA_PLAN_CUENTAS		
Campo	**Tipo**	**Observación**
CODIGO_PLAN	CHARACTER VARYING(20)	PK_TAB_CONTA_PLAN_CUENTAS
NOMBRE	TEXT	Uk - Unique
CUENTA_MAYOR	BOOLEAN	
BORRADO	BOOLEAN	

Tabla 10: Plan Cuentas descripción

Fuente: Propia

Nota: La tabla presenta una estructura genérica para ingresar cualquier plan de cuentas.

La interfaz de usuario presentará un listado de todas las cuentas contables ordenadas por su código, con casillas de filtro para buscar alguna cuenta específica.

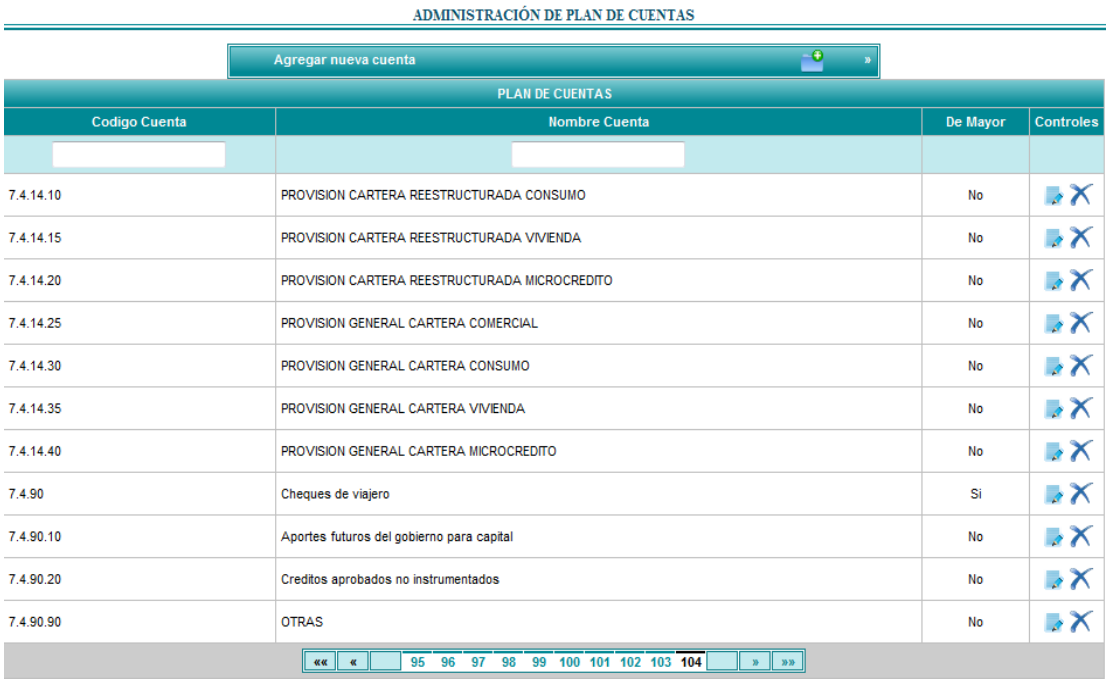


Figura 31: Administración Plan Cuentas

Fuente: Propia

1. El usuario editará una cuenta contable dando click en el icono editar, el cual le permitirá modificar el nombre de la cuenta contable y el atributo de Mayor el cual indica si la cuenta es de grupo o una cuenta individual, si es una cuenta de grupo que contiene subcuentas no este atributo no se cambiará.

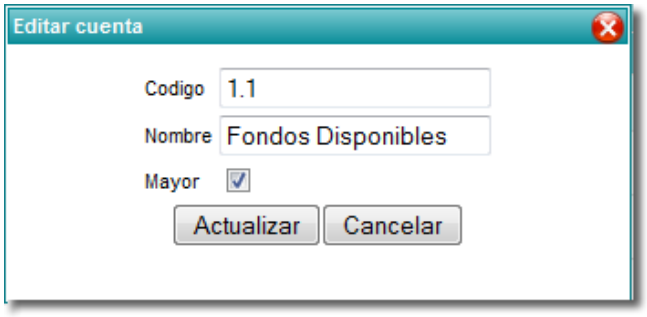


Figura 32: Editar cuenta

Fuente: Propia

2. El usuario eliminará una cuenta contable dando click en el icono eliminar, el cual borra la cuenta contable a nivel de usuario, más no a nivel de la base de datos, si la cuenta fue borrada permanecerá en un estado que le permite volver a ser restaurada.

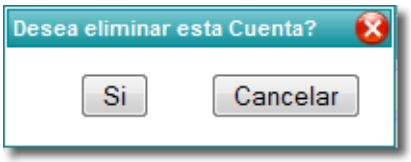


Figura 33: Eliminar cuenta

Fuente: Propia

3. Si el usuario desea agregar una nueva cuenta contable, el usuario elegirá una cuenta de mayor, el sistema le sugiere una cuenta subcuenta perteneciente a este grupo.
4. El usuario dará click en el botón Guardar y se almacenará la cuenta.
5. Una vez creada la cuenta el usuario podrá hacer uso de la misma para crear comprobante.

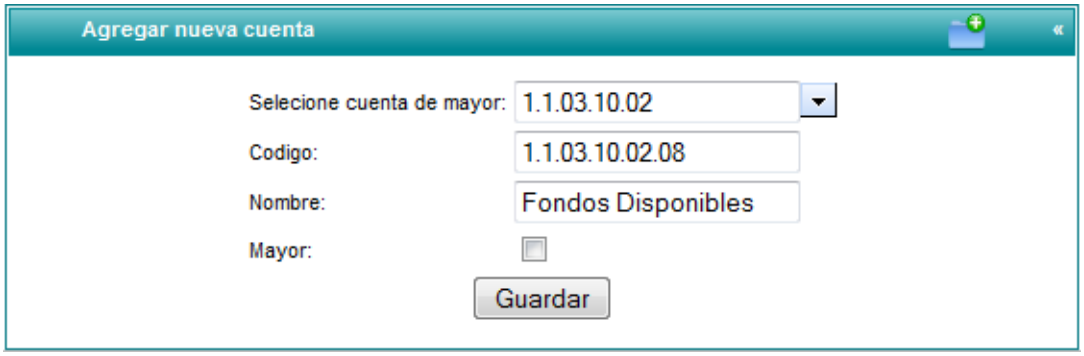


Figura 34: Nueva cuenta

Fuente: Propia

3.7.8. Especificación de Caso de Uso: Administración de Comprobantes

3.7.8.1. Descripción Breve

El caso de uso describe el proceso de administración de todos los comprobantes es decir aquellos generados en MC-Finansys (Módulo Contable) y aquellos creados en los demás módulos.

3.7.8.2. Flujo Básico de Eventos

3.7.8.2.1. Estructura de tablas en maestro-detalle.

Los comprobantes se configuran como maestro – detalle dado que un comprobante contiene dos o más movimientos, la estructura de las dos tablas será configurará de la siguiente manera:

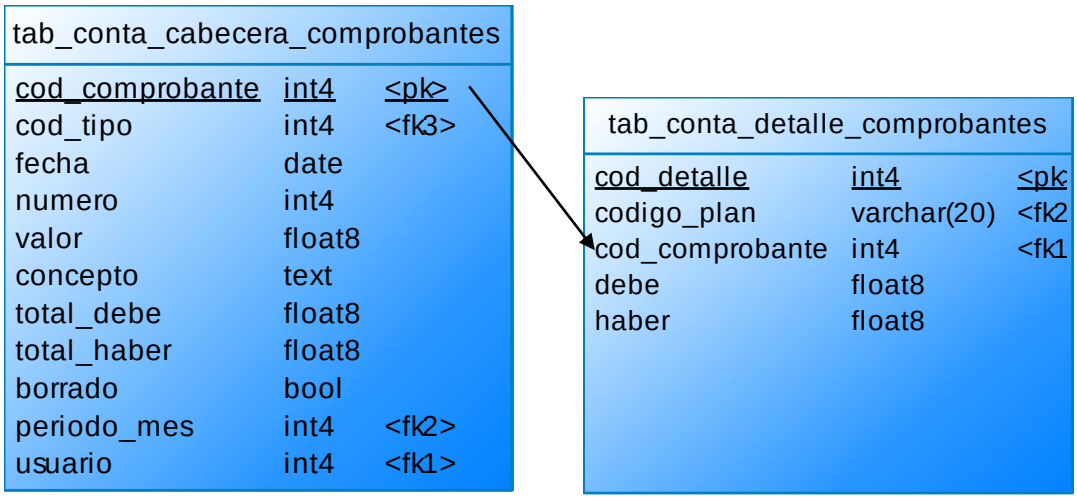


Figura 35: Tabla Comprobantes

Fuente: Propia

TAB_CONTA_CABECERA_COMPROBANTES		
Campo	**Tipo**	**Observación**
COD_COMPROBANTE	INTEGER	PK_TAB_CONTA_CABECERA_COMPROBA
COD_TIPO	INTEGER	FK_CAB_COMP_TIPO_COMP
FECHA	DATE	Fecha del comprobante
NUMERO	INTEGER	Número del comprobante
CONCEPTO	TEXT	Descripción del comprobante
VALOR	DOUBLE	Total del comprobante
TOTAL_DEBE	DOUBLE	
TOTAL_HABER	DOUBLE	
BORRADO	BOOLEAN	
PERIODO_MES	INTEGER	FK_CAB_COMP_PERIODO_MES
USUARIO	INTEGER	FK_CAB_COMP_ADMIN_USUARIO

TAB_CONTA_DETALLE_COMPROBANTES		
Campo	**Tipo**	**Observación**
COD_DETALLE	INTEGER	PK_TAB_CONTA_DETALLE_COMPROBAN
CODIGO_PLAN	CHARACTER VARYING(20)	FK_DETA_COMP_PLAN
COD_COMPROBANTE	INTEGER	FK_DETA_COMP_CAB
DEBE	DOUBLE	
HABER	DOUBLE	

Tabla 11: Tabla Comprobantes

Fuente: Propia

Nota: De eliminarse un dato del maestro se eliminarán todos sus detalles.

3.7.8.3. Interfaces de usuario

Una vez en el sistema el Operador Contable puede administrar los comprobantes en general y de manera particular los comprobantes de tipo contable.

3.7.8.4. Creación de comprobantes contables.

La interfaz le permitirá ingresar datos de un comprobante contable con sus respectivos detalles que son los movimientos contables realizados para el comprobante.

Figura 36: Administración comprobantes

Fuente: Propia

3.7.8.5. Búsqueda y administración de comprobantes en general.

La interfaz presentará una lista de todos los comprobantes de todo tipo, siempre y cuando pertenezcan a un periodo mes que aún no estén cerrados.

Figura 37: Buscar comprobantes

Fuente: Propia

Los demás Casos de Uso del Módulo Contable se encuentran en el CD de anexos.

Capítulo IV: Diseño y Desarrollo del Aplicativo

CONTENIDO:

1. Fase de Inicio
2. Fase de Elaboración
3. Fase de Construcción
4. Fase de Transición

4. Diseño y Desarrollo del Aplicativo

4.1. Fase de Inicio

4.1.1. Documento de Visión

4.1.1.1. Introducción

4.1.1.1.1. Propósito

El propósito de este documento es definir en alto nivel los requerimientos del Sistema Integrado de Información Financiera en adelante Finansys, el cual se encargará de realizar todo el proceso financiero de la CUC de manera detallada y automática, el detalle de cómo Finansys cubrirá las necesidades de los usuarios se especifica en los casos de uso algunos de ellos especificados en el capítulo 3.

4.1.1.1.2. Alcance

Este documento de visión se aplica a Finansys que será implementado por estudiantes de la UTN – FICA.

4.1.1.1.3. Definiciones, Siglas y Abreviaturas

Todas las siglas y abreviaturas con sus respectivas definiciones constan en el Glosario.

4.1.1.1.4. Referencias

1. Glosario.
2. Acta de trabajo No. 01.
3. Resumen del Modelo de Casos de Uso.

4.1.1.2. Posicionamiento

4.1.1.2.1. Oportunidad de negocio

A partir de los procedimientos ya establecidos en la CUC, y como parte de la renovación y automatización establecido por los directivos, se determina la creación del sistema Finansys que permita mejorar toda la gestión de las actividades relacionadas al manejo de la información financiera generada en la entidad.

El manejo de la información financiera generada por la CUC en la actualidad se realiza en base a un sistema implementado en lenguaje C, el cual presenta limitaciones, porque no controla y administra todos los procesos de forma unificada empleando mucho tiempo en procesar la información, esto provoca realizar procesos manuales que dificultan la agilidad con la que debería trabajar una institución como esta, dichos procesos manuales se sustentan en la elaboración de documentos físicos y digitales que impiden un flujo de información adecuado para la CUC.

Luego de un análisis realizado por los desarrolladores, se pudo establecer que el sistema que actualmente cubre las necesidades de la CUC presenta algunas falencias o dificultades, entre ellas se destacan las que se menciona a continuación.

1. **No existe un gestor de base de datos**

El sistema actual usa archivos visibles al usuario para el manejo de la información lo que hace que este sistema sea por demás vulnerable y susceptible a cualquier tipo error.

2. **Parametrización no adecuada**

Se realizan modificaciones continuas al programa fuente lo cual provocando duplicidad de esfuerzos y un gasto.

3. **Tecnología desactualizada**

La tecnología de la aplicación que se mantiene en funcionamiento es desactualizada; pues ha sido implementada en una herramienta que descontinuo su uso.

4. **Falta de control y auditoría**

El sistema informático actual, carece de controles y pistas de auditoría que permitan validar la información y establecer la eficiencia y eficacia con que se está operando.

5. **Seguridad deficiente**

El sistema actual tiene falencias de seguridad importantes que comprometen la confidencialidad de la información.

Con referencia a lo expuesto anteriormente, la directiva general de la CUC ha planteado rediseñar la aplicación informática existente para permitir trabajar de una manera más eficiente e integrada en un único sistema financiero.

El nuevo sistema Finansys entonces brindará las facilidades necesarias para el manejo de la información financiera generada por la cooperativa.

Por otro lado, rediseñar e implementar un nuevo sistema, obliga a que se cree en la nueva aplicación la mayoría de las tareas de los módulos del sistema antiguo, pero de una manera más eficiente y amigable con el usuario, también se desarrollaran nuevos módulos necesarios para mantener una aplicación controlada y efectiva, de esta manera los módulos serán:

1. Administración y Seguridad.
2. Clientes.
3. **Contabilidad.**
4. Préstamos y Cartera.
5. Ahorros.
6. Caja.
7. Auditoría Informática.

4.1.1.2.2. Definición del problema

El problema de:	**1.** Redundancia de procesos en el desarrollo de actividades diarias. **2.** No poseer un sistema integrado que administre todos los procesos de gestión de la información financiera. **3.** La aplicación actual carece de seguridad. **4.** Aplicación construida con tecnología que ya no tiene soporte. **5.** Dificultad para el mantenimiento de la aplicación actual.
Que afecta a:	**1.** Todos los usuarios de la Cooperativa involucrados con la gestión de la información financiera. **2.** Personal que tiene dificultad para mantener el sistema en producción.
El impacto de ello es:	**1.** Falta de información consistente de la gestión financiera. **2.** Existen muchas actividades y procesos manuales que no permiten una gestión eficiente. **3.** Inconsistencia en la información generada.
Una solución exitosa debería:	**1.** Solucionar los requerimientos internos de los involucrados en el proceso de la gestión financiera. **2.** Cubrir las necesidades de integración.

	3. Automatizar algunos de los procesos que hasta ahora son realizados de forma manual. **4.** Brindar información confiable sobre el manejo financiero de la cooperativa de manera fácil y oportuna. **5.** Implementar una solución informática de calidad soportada por una metodología eficiente de desarrollo de software.

Tabla 12: Problema

Fuente: Propia

4.1.1.3. Descripción de los interesados y usuarios

4.1.1.3.1. Resumen de los interesados

Interesados son todas aquellas personas directamente involucradas en la definición y alcance del proyecto.

A continuación se presenta la lista de los mismos:

Nombre interesado	Descripción	Responsabilidades
Coordinador del proyecto. (Gerente General)	Responsable del proyecto a nivel directivo de la Cooperativa de Ahorros y Crédito Unión Cochapamba.	**1.** Establecer los lineamientos generales para el desarrollo del proyecto. **2.** Coordinar a nivel directivo los diferentes requerimientos que surjan en el desarrollo del sistema. **3.** Proveer de los recursos necesarios para el correcto desarrollo del proyecto.
Responsable del proyecto (Contador)	Responsable del proyecto por parte de la Cooperativa (Desarrollador de la tesis).	**1.** Responsable del análisis y diseño del proyecto. **2.** Gestiona el correcto desarrollo del proyecto en lo referente a la construcción e implantación.
Responsable funcional (Contador)	Responsables del proyecto por parte de cada uno de los departamentos de la Cooperativa.	**1.** Responsables de coordinar con los diferentes usuarios la correcta determinación de los requerimientos y la correcta concepción del sistema.

Jefe Departamental (Contador)	Responsable de proporcionar la información para el registro de productos, clientes, préstamos y demás datos financieros.	**1.** Definir la estructura de la información que se utilizará para el registro de clientes, préstamos y demás datos financieros que se registraran en la Cooperativa.	**Tabla 13: Lista de Interesados**

Fuente: Propia

4.1.1.3.2. Resumen de los usuarios

Los usuarios son todas aquellas personas involucradas directamente en el uso del sistema Finansys.

A continuación se presenta una lista de los usuarios:

Nombre	Descripción	Responsabilidad
Administrador del sistema	Persona del Centro de Cómputo que administra el sistema Finansys.	Administrar eficazmente el sistema (creación de nuevos usuarios y gestionar acceso a usuarios, facilitar mantenimiento al sistema frente a nuevos requerimientos).
Administrador funcional del sistema	Persona de la Cooperativa de Ahorros y Crédito Unión Cochapamba que administra el sistema Finansys.	Administrar funcionalmente el sistema: creación de nuevos productos, definir los tipos de transacciones, nuevos grupos de clientes, tipos de préstamos, etc.
Usuario del sistema	Personal que labora en los diferentes departamentos de la Cooperativa de Ahorros y Crédito Unión Cochapamba que administra el sistema Finansys	Ingresan la información referente a cada uno de los departamentos de la Cooperativa como son registro de información personal para el ingreso de nuevos clientes, créditos, retiros, etc.

Nombre	Descripción	Responsabilidad
Usuario de gestión del sistema	Personal de la Cooperativa de Ahorro Crédito Unión Cochapamba	**1.** Validar la información proveniente de los diferentes departamentos de la Cooperativa generada a través del Finansys. **2.** Consolidar la información. **3.** Aprobación de Socios y Créditos

Tabla 14: Usuarios del sistema

Fuente: Propia

4.1.1.3.3. Entorno actual de los usuarios

1. El personal de los diferentes departamentos de la Cooperativa serán usuarios del sistema Finansys, y beneficiará así a toda la cooperativa ya que permitirá registrar y llevar un control adecuado de todos los procesos y movimientos financieros que se realizan diariamente en cada uno de los departamentos de la Cooperativa.
2. Los usuarios manejan la información de cada módulo de forma independiente debido a que no existe un sistema que integre todos los módulos en un solo sistema.
3. Debido a que es muy costoso adquirir licencias de software privativo el sistema será implementado en su totalidad con software libre y el lineamiento general es tener las aplicaciones en plataforma web.
4. Actualmente existen sistemas para instituciones financieras como Bancos y Cooperativas de Ahorro y Crédito pero hay que tomar en cuenta que cada institución tiene lineamientos diferentes por lo que hace difícil la adquisición de estos sistemas ya elaborados, lo que conlleva a las instituciones a diseñar su propio sistema que se ajuste a sus necesidades.
5. Los Módulos de Ahorros, Captaciones, Clientes, Administración y Seguridad, Cartera, Auditoría y Contabilidad formarán parte del Sistema Integrado de Información Financiero, los mismos que deberán interactuar entre sí.

4.1.1.3.3.1. Coordinador de Proyecto

Representante	Flandes Ibarra – Gerente.
Descripción	Responsable a nivel directivo del proyecto por parte de la Cooperativa de Ahorro y Crédito Unión Cochapamba.
Tipo	Gerente de la Cooperativa.
Responsabilidades	**1.** Establecer los lineamientos generales para el desarrollo del proyecto. **2.** Coordinar a nivel directivo los diferentes requerimientos que surjan en el desarrollo del sistema.
Criterios de éxito	**1.** Mantener activo el sistema luego de su implementación. **2.** Mantener la funcionalidad del sistema.
Implicación	Revisor de la Administración.
Entregables	N/A.
Comentarios	**1.** Mantener una relación constante con el desarrollo del proyecto. **2.** Brindar apoyo a nivel gerencial cuando sea necesario.

Tabla 15: Coordinador del proyecto

Fuente: Propia

4.1.1.3.3.2. Responsable del proyecto

Representante	Esteban Gudiño, Bairon Gudiño y Edgar Picuasi.
Descripción	Responsables del proyecto por parte de la Cooperativa de Ahorro y Crédito Unión Cochapamba.
Tipo	Analistas de sistemas.
Responsabilidades	**1.** Responsable del análisis, diseño e implementación del proyecto. **2.** Gestiona el correcto desarrollo del proyecto en lo referente a

	la construcción e implantación.
Criterios de éxito	**1.** Cumplir con el cronograma determinado. **2.** Cumplir con los requerimientos establecidos para el sistema. **3.** Obtener un sistema de calidad que cumpla con los requerimientos funcionales establecidos.
Implicación	Jefe de proyecto (Project Manager)
Entregables	**1.** Documento de visión. **2.** Resumen del modelo de casos de uso. **3.** Especificaciones del modelo de casos de uso. **4.** Diseño ER de la base de datos y el diccionario. **5.** Especificaciones complementarias.

Tabla 16: Responsable del proyecto

Fuente: Propia

4.1.1.3.3.3. Responsable funcional

Representante	Dra. Irene de la Cruz – Contadora.
Descripción	Responsables del proyecto por parte de la Cooperativa de Ahorro y Crédito Unión Cochapamba.
Tipo	Experto en el tema.
Responsabilidades	**1.** Responsable de coordinar con los diferentes usuarios la correcta determinación de los requerimientos y la correcta concepción del sistema. **2.** Coordinar las pruebas de validación del nuevo sistema. **3.** Coordinar y asegurar la capacitación de los usuarios.
Criterios de éxito	Obtener un sistema de calidad que cumpla con los requerimientos funcionales establecidos.
Implicación	Aprueba las especificaciones funcionales y las pruebas realizadas.

Entregables	**1.** Documento de revisión de las especificaciones funcionales. **2.** Documento de revisión de las pruebas funcionales.

Tabla 17: Resumen funcional

Fuente: Propia

4.1.1.3.4. Perfiles de usuario

4.1.1.3.4.1. Administrador del sistema

Representante	Dra. Irene de la Cruz – Contadora.
Descripción	Persona que administra el sistema Finansys.
Tipo	Operador, Analista de Sistemas.
Responsabilidades	Administrar funcionalmente el sistema: gestionar acceso a usuarios, dar mantenimiento al sistema frente a nuevos requerimientos.
Criterios de éxito	N/A.
Implicación	N/A.
Entregables	**1.** Bitácora de control de nuevos requerimientos. **2.** Bitácora de control de incidencias del nuevo sistema.

Tabla 18: Administrador del sistema

Fuente: Propia

4.1.1.3.4.2. Administrador funcional del Sistema

Representante	Dra. Irene de la Cruz – Contadora.
Descripción	Persona de la Cooperativa que administra el sistema Finansys.
Tipo	Experto en el Tema.

Responsabilidades	Creación de nuevas cuentas de usuarios, definición de periodos contables, etc.
Criterios de éxito	N/A.
Implicación	N/A.
Entregables	N/A.
Comentarios	N/A.

Tabla 19: Administrador funcional

Fuente: Propia

4.1.1.3.4.3. Usuario del sistema

Representante	Dra. Irene de la Cruz – Contadora. Srta. Anita Fueres.
Descripción	Personal de los diferentes departamentos CUC que harán uso de cada uno de los módulos del Finansys.
Tipo	Personal de los diferentes departamentos de la CUC.
Responsabilidades	**1.** Ingresar la información concerniente en cada uno de los módulos del Finansys. **2.** Realizar consultas de información de clientes. **3.** Realizar transacciones, etc.
Criterios de éxito	N/A
Implicación	N/A
Entregables	N/A
Comentarios	N/A

Tabla 20: Usuario del sistema

Fuente: Propia

4.1.1.3.5. Necesidades de los interesados y usuarios

Necesidades	Priori-dad	Inquietudes	Solución Actual	Solución propuesta
Diseñar un sistema que facilite la consolidación e integración de información mediante la unificación de módulos y automatización de procesos de la CUC.	Alta	El sistema debe consolidar la información para facilitar registro y administración de información.	Sistema desarrollado en C++ pero no es un sistema que integra todos los módulos en un solo sistema.	Implementar el módulo de Ahorros con herramientas de software actuales que nos permita integrar todos los módulos en un solo sistema.
Elaborar el sistema utilizando herramientas y software libre que facilite y agilice su desarrollo.	Alta	Se debe utilizar software libre para el desarrollo del sistema.	N/A	Desarrollar el sistema utilizando las herramientas de desarrollo como la plataforma JEE, frameworks JSF, Hibernate, RichFaces, servidor de aplicaciones apache, estándares abiertos como CSS.
La interfaz del sistema debe ser fácil de manejar, cumpliendo con todos los requerimientos establecidos.	Alta	Cumplir con todos los requerimientos de los usuarios.	N/A	Desarrollo con la ayuda de los expertos en cada uno de los departamentos de la CUC.
Lograr la implementación del sistema en el menor tiempo posible para su utilización desde el próximo año.	Media	Cumplir con las especificaciones de los usuarios.	Actualmente se trabaja con el sistema por módulos independientes.	Trabajar con el Finansys desde el año 2012 ya con el sistema implementado y en funcionamiento.

Tabla 21: Necesidades de interesados

Fuente: Propia

4.1.1.3.6. Alternativas y competencia

4.1.1.3.6.1. Adquirir un sistema desarrollado.

Se ha mostrado interés en buscar alternativas externas para solucionar los diversos requerimientos, pero en la actualidad no existen herramientas en el mercado que se adapten a las necesidades específicas de la Cooperativa.

4.1.1.4. Vista General del Producto

Esta sección provee información a alto nivel de las funciones del sistema a implantar y de las interfaces con otras aplicaciones existentes.

4.1.1.4.1. Perspectiva del producto

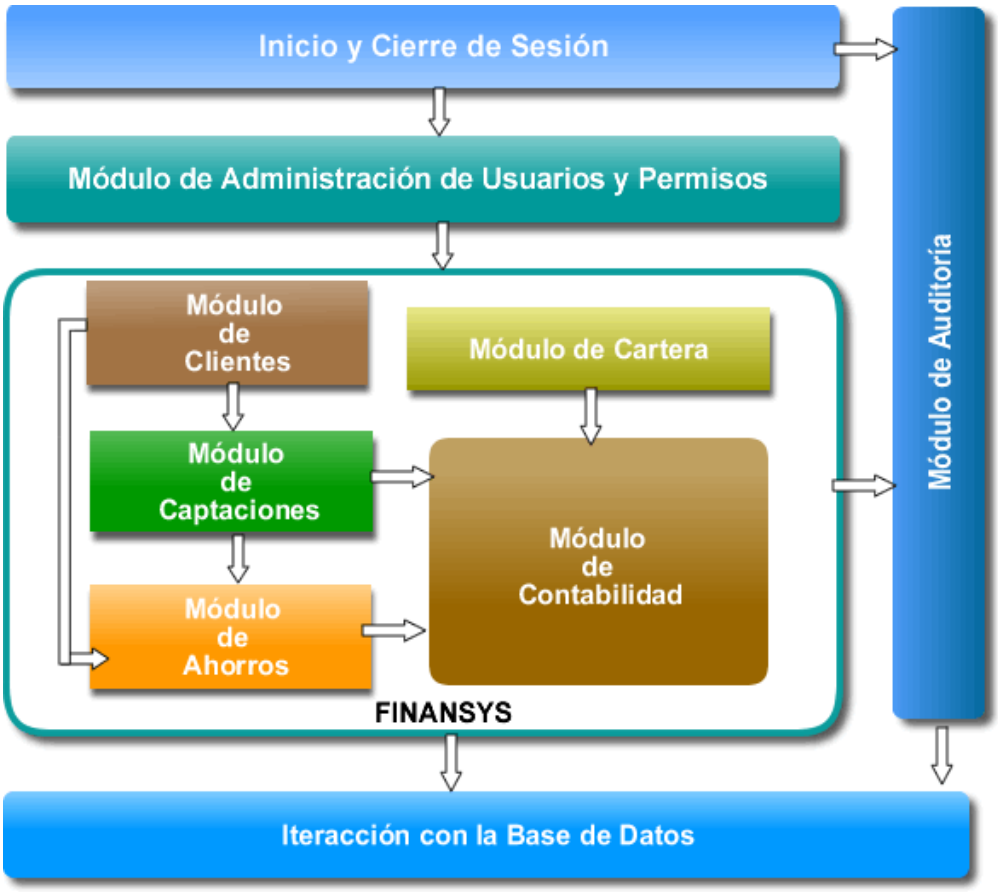


Figura 38: Perspectiva del producto

Fuente: Propia

4.1.1.4.2. Resumen de capacidades

Finansys, mejorará todo el proceso de gestión financiera y permitirá además:

1. Eliminar la elaboración manual de varios procesos redundantes.
2. Controlar la seguridad en el manejo de información.

4.1.1.4.2.1. Módulo Contabilidad.

Beneficios para el usuario	Características que lo soportan
El registro de la información contable será confiable, automático y completo.	El ingreso de información realizada por el encargado del manejo del módulo contable será sistematizado y ordenado. El responsable del manejo del módulo contable tendrá una herramienta de registro y manejo de información congruente y sincronizada con el sistema integrado.
Los usuarios o usuario del módulo contarán con una herramienta unificada.	Al contar con un sistema computacional, los usuarios podrán mejorar sus procesos. Además evitará el uso manual de formatos y documentos no unificados.
Se tendrá alta disponibilidad de uso del sistema.	El acceso a la información a través del sistema web permitirá a los usuarios un acceso inmediato desde cualquier punto de la intranet de la cooperativa.
Facilidades para el análisis de la información.	Brindará diferentes reportes y funciones de consulta.
Contará con información detallada.	Oportunidad de mantener un control sobre la información que se desea mostrar o realizar un reporte.

Tabla 22: Módulo contabilidad

Fuente: Propia

4.1.1.4.3. Suposiciones y dependencias

En la implantación del sistema Finansys, la cooperativa deberá contar con los servicios necesarios.

4.1.1.4.4. Costos y precios

Costos			
N°	**Descripción**	**Costo Cooperativa**	**Costo Tesista**
1	Servidor HP	1500	0.00
2	Computador portátil	0.00	800
	Subtotal	1500	800
N°	**Descripción**	**Costo Cooperativa**	**Costo Tesista**
1	IDE Netbeans	0.00	0.00
2	Servidor de aplicaciones Apache Tomcat	0.00	0.00
3	Base de datos PostgreSQL	0.00	0.00
4	Framework Hibernate	0.00	0.00
5	JDK 6.x y JRE 6.x	0.00	0.00
	Subtotal	0.00	0.00
N°	**Descripción**	**Costo Cooperativa**	**Costo Tesista**
1	Cursos de Capacitación Continua FICA	40.00	40.00
2	Cursos externos	0.00	150.00
	Subtotal	0.00	230.00
N°	**Descripción**	**Costo Cooperativa**	**Costo Tesista**
1	Resmas de hojas de papel bond	0.00	20.00
2	Copias de documentos y libros	0.00	50.00
3	DVDs	0.00	10.00
4	Memoria Flash	0.00	30.00
5	Libros	0.00	80.00
6	Internet	0.00	40.00
7	Cartuchos y recargas de tinta	0.00	50.00
8	Anillados y empastados	0.00	120.00
9	Lápices, esferos y cuadernos	0.00	5.00
	Subtotal	0.00	405.00
N°	**Descripción**	**Costo Cooperativa**	**Costo Tesista**
1	Movilizaciones	0.00	300.00
2	Imprevistos	0.00	50.00
	Subtotal	0.00	250.00
TOTAL COSTOS		1500.40	1685.00

Tabla 23: Tabla de Costos

Fuente: Propia

4.1.1.4.5. Licenciamiento e instalación

1. Dado que se usará software libre no se requiere ningún tipo de licencia.
2. La instalación del Sistema será realizada por los estudiantes que han desarrollado el sistema.

4.1.1.5. Características del producto

4.1.1.5.1. Facilidad de acceso y uso

Finansys será desarrollado utilizando tecnología Web, desarrollando la parte visual con RichFaces optimizando de esta forma el rendimiento y facilidad de uso por parte de los usuarios, creando pantallas con una presentación clara y atractiva gracias a títulos barras de estado, íconos intuitivos, tipografía legible.

4.1.1.5.2. Unificación de la información

Unos de los principales objetivos del Finansys es registrar y unificar la información de la Cooperativa y proporcionar herramientas con criterios de búsqueda parametrizables, ahorrando mucho tiempo a la hora de obtener la información.

4.1.1.5.3. Mejor control y validación de la información

Al tratarse de un sistema financiero, la validación de los datos es un proceso muy importante a tener en cuenta en Finansys, validando correctamente los flujos de datos de entrada, para verificar que el tipo de datos sea el esperado y avisando al usuario cuando exista algún error. Logrando que la información guardada sea congruente y adecuada.

4.1.1.6. Restricciones

Por cuanto la Cooperativa en la actualidad no cuenta con sucursales, se sugiere un entorno de red local o intranet, por tal motivo el sistema no será subido al internet porque para que el sistema esté en línea se necesitan las debidas seguridades para funcionar en ese entorno.

4.1.1.7. Rangos de calidad

El desarrollo del Sistema Finansys se elaborará siguiendo la Metodología de Desarrollo de Software RUP, contemplando los parámetros que la metodología define para asegurar la producción de software de alta calidad y que se ajuste a las necesidades de los usuarios finales a tiempo.

4.1.1.8. Precedencia y Prioridad

Las prioridades del sistema son:

1. Mantener la información unificada y congruente.
2. Realizar de forma correcta los procesos financieros de la Cooperativa.
3. Interactuar entre los diferentes módulos del sistema.
4. Proporcionar una herramienta que tenga una interfaz amigable con el usuario.
5. Desarrollar un sistema multiusuario con el respectivo control de acceso y privilegios a cada usuario de acuerdo al rol que éste tenga.
6. Desarrollar métodos fáciles de búsqueda de información.

4.1.1.9. Otros requerimientos del producto

4.1.1.9.1. Requisitos del Sistema

Para realizar Finansys se necesita cumplir con los requisitos detallados a continuación:

1. Un computador servidor con las siguientes aplicaciones y características:
    a. Sistema operativo Centos.
    b. Base de datos PostgresQL versión 8.4 sobre un servidor Centos.
    c. Servidor web Apache Tomcat versión 6.0.26 en un servidor Centos.
    d. Puerto de red.
2. Red de área local.
3. Navegador Web Firefox desde la versión 4.
4. No requiere una conexión a internet.
5. Facilidad de uso.
6. Que sea administrable.
7. Que realice cálculos exactos.

4.1.2. Plan de Desarrollo

4.1.2.1. Introducción

Este Plan de Desarrollo del Software es una versión preliminar preparada para ser incluida en la propuesta elaborada como respuesta al proyecto Finansys para la Cooperativa de Ahorro y Crédito Unión Cochapamba.
Este documento presenta una visión general del enfoque de desarrollo propuesto.

El desarrollo de Finansys utilizará la metodología de unificación de procesos, la misma que permitirá utilizar las normas para definir el proyecto y de esta manera organizar el desarrollo la documentación generada.
El enfoque del desarrollo constituye una configuración del proceso RUP de acuerdo a las características del proyecto, seleccionando los roles de los participantes, las actividades a realizar y los artefactos (entregables). Este documento es a su vez uno de los artefactos de RUP.

4.1.2.1.1. Propósito

El propósito del Plan de Desarrollo de Software es proporcionar la información necesaria para definir, planificar y controlar el desarrollo de un sistema admisible. En él se describe el enfoque de desarrollo del software.

Participantes:

Los usuarios del Plan de Desarrollo del Software son:

1. Analistas del proyecto, los mismos que definen los tiempos, esquemas, recursos y herramientas a utilizarse en el desarrollo del cronograma de actividades.
2. Los miembros del equipo de desarrollo que ejecutan y desarrollan el contenido de la planificación del cronograma.

4.1.2.1.2. Alcance

El Plan de Desarrollo del Software describe el plan global usado para el desarrollo de Finansys.
El detalle de las iteraciones individuales se describe en los planes de cada iteración, documentos que se aportan en forma separada.

Durante el proceso de desarrollo en el artefacto "Visión" se definen las características del producto a desarrollar, lo cual constituye la base principal de cada etapa.

Para la versión 1.0 del Plan de Desarrollo del Software, nos hemos basado en la captura de requisitos por medio del stakeholder representante de la CUC para hacer una estimación aproximada, una vez comenzado el proyecto y durante la fase de Inicio se generará la primera versión del artefacto "Visión", el cual se utilizará para refinar este documento. Posteriormente, el avance del proyecto y el seguimiento en cada una de las iteraciones ocasionará el ajuste de este documento produciendo nuevas versiones actualizadas.

4.1.2.2. Vista General del Proyecto

4.1.2.2.1. Propósito, Alcance y Objetivos

La información que a continuación se incluye ha sido extraída de las diferentes reuniones que se han realizado con el stakeholder de la CUC desde el inicio del proyecto, Dra. Irene de la Cruz y Sr. Flandes Ibarra.

La CUC como entidad financiera busca satisfacer las necesidades económicas de sus socios, en las diferentes actividades tales como: Ahorro, Crédito, Vivienda, Transporte, Salud, Educación, Agricultura y Ganadería, en una institución de estas características es previsible la adaptación a los nuevos sistemas de información y a la evolución tecnológica.

Por ello, la CUC considera necesario el desarrollo de un nuevo sistema integrado de información financiera dado que el actual sistema usado presenta varias deficiencias entre ellas, está desarrollado con tecnología antigua, cada módulo funciona por separado esto provoca que muchos de los procesos existentes se repitan, entre otros, de ahí la necesidad de implementar un sistema informático integral basado en una arquitectura moderna, utilizando software libre y de esta manera contribuir con el desarrollo y crecimiento de la cooperativa.

El proyecto detallará la propuesta para el desarrollo de los módulos, como interactúan entre sí, como se conforman para un buen funcionamiento y con ello cumplir los objetivos planteados en el documento "Visión".

a) **Módulo de Administración y Seguridad**

1. Administración de todos los usuarios del sistema.
2. Administración de sucursales de la cooperativa.
3. Administración de los roles que se asignan a los usuarios.
4. Creación de tipos de usuario.
5. Administrar parámetros generales.

6. Autenticación de usuarios.
7. Administración de sesiones.

b) Módulo de Ahorros

1. Cuentas de Ahorros.
2. Otros Ingresos.
3. Realizar Bloqueos y Desbloqueos.
4. Depósitos a Plazo Fijo.
5. Realizar Transferencias, Débitos y Créditos.
6. Consultas.
7. Reportes.

c) Módulo de Cajero

1. Apertura de Cuenta.
2. Depósito de Ahorros.
3. Retiros en Efectivo.
4. Retiros con Cheque.
5. Pago de Préstamos.
6. Registro Otros Ingresos.
7. Apertura de Caja.
8. Arqueos de Caja.
9. Consultas.

d) Módulo de Contabilidad

1. Administrar plan de cuentas.
2. Crear y administrar comprobantes.
3. Crear y administrar los tipos de comprobante.
4. Crear y administrar periodos contables.
5. Crear y administrar periodos mensuales.
6. Administrar información de bancos.
7. Cerrar y mayorizar saldos del mes.
8. Realizar conciliación bancaria.
9. Generar control de presupuesto.
10. Administrar asientos de fin de año.

11. Realizar estados financieros.
12. Realizar consultas.
13. Realizar reportes.

e) Módulo de Cartera

1. Crear grupos y tipos de préstamos.
2. Simular Préstamo.
3. Generar solicitudes de préstamos.
4. Generar Tabla Amortización.
5. Estudio de Factibilidad del Préstamo.
6. Aprobación o negación de préstamos.
7. Pago de Cuotas.
8. Consultas.
9. Reportes.

f) Módulo de Clientes.

1. Administración de provincias, cantones, parroquias y comunidades.
2. Creación de clientes naturales y jurídicos.
3. Administración de datos personales cliente.
4. Administración de datos de personas relacionados a cliente.
5. Cierre de cuentas.
6. Consultas.
7. Reportes.

g) Módulo de Auditoría.

1. Auditar sesiones.
2. Auditar acciones en la base de datos.
3. Consultas.
4. Reportes.

4.1.2.2.2. Suposiciones y Restricciones

Las suposiciones y restricciones respecto del sistema, y que se derivan directamente de las entrevistas con el stakeholder de la CUC son:

1. Sistemas seguros; protección de la información.
2. Seguridad de transacciones e intercambio de información.
3. Adaptación a la normativa de protección de datos.
4. Adaptabilidad y la facilidad de uso.

Como es natural, la lista de suposiciones y restricciones se incrementará durante el desarrollo del proyecto, particularmente una vez establecido el artefacto "Visión".

4.1.2.2.3. Entregables del proyecto

En la metodología RUP, a lo largo de todo el desarrollo del software se van generando documentos entregables, que son documentos explicativos de cada actividad, la mayoría de documentos manejan versiones porque pueden ir variando a lo largo de todo el proceso de elaboración del proyecto.

A continuación se describen los artefactos que serán utilizados por el proyecto.

1) Plan de Desarrollo del Software.

Describe el plan global usado para el desarrollo de Finansys.

2) Glosario.

Es un entregable que contiene una recopilación de términos poco conocidos o de difícil interpretación con su respectiva explicación o significado.

3) Modelo de Casos de Uso.

El modelo de Casos de Uso presenta una descripción de los pasos para llevar a cabo algún proceso o funcionalidad del sistema. Se representa mediante Diagramas de Casos de Uso.

4) Visión.

Este documento presenta el enfoque del producto desde la perspectiva del cliente, detallando las necesidades y características del producto.

5) Especificaciones de Casos de Uso.

Para aquellos casos de uso que su funcionalidad no sea evidente, se realiza una explicación detallada utilizando un documento plantilla; dicha plantilla incluye según la especificación del caso de uso: flujo básico de eventos, flujo alternativo de eventos, precondiciones y pos condiciones.

6) Prototipos de Interfaces de Usuario.

Se trata de prototipos que permiten al usuario hacerse una idea de las interfaces que proveerá el sistema.

7) Modelo de Dato.

Este documento describe la representación lógica de la estructura de los datos almacenada en una base de datos, describiendo los tipos de datos y la forma de relacionarse con la demás tablas, así como también las restricciones de integridad.

8) Casos de Prueba.

Cada prueba es especificada mediante un documento que establece las condiciones de ejecución, las entradas de la prueba, y los resultados esperados. Estos casos de prueba son aplicados como pruebas de regresión en cada iteración.
Cada caso de prueba llevará asociado un procedimiento de prueba con las instrucciones para realizar la prueba.

9) Lista de Riesgos.

Este documento contiene la lista de los riesgos conocidos en el proyecto, organizados descendentemente de acuerdo a la importancia que tenga el riesgo; También contiene acciones específicas de contingencia o mitigación del riesgo.

10) Manual de Instalación

Este documento proporciona una guía detallada de los pasos necesarios para la correcta instalación del sistema.

11) Producto

El sistema Finansys se entregará almacenado en un CD con los mecanismos apropiados para su instalación.

12) Actas de Trabajo

Las actas de trabajo contienen las actividades, responsabilidades, observaciones y revisiones que se realicen en el desarrollo del proyecto.

4.1.2.2.4. Evolución del Plan de Desarrollo del Software

El Plan de Desarrollo del Software se revisará semanalmente y se refinará antes del comienzo de cada iteración.

4.1.2.3. Organización del Proyecto

4.1.2.3.1. Participantes en el Proyecto

El personal involucrado en el desarrollo del sistema son los siguientes:

Analistas.- Con experiencia en el entorno de desarrollo del proyecto, con el fin de que los prototipos puedan ser lo más cercanos posibles al producto final.

Programadores.- Los conocimientos que deben tener son: amplio conocimientos en el lenguaje de programación JAVA, administración de la base de datos PostgreSQL, diseño UML, metodología RUP.

4.1.2.3.2. Interfaces Externas

La CUC proporcionará los requisitos del sistema y los encargados de evaluar los artefactos de acuerdo a cada subsistema.
El equipo de desarrollo interactuará activamente con ellos para la especificación y validación de los artefactos generados.

4.1.2.3.3. Roles y Responsabilidades

A continuación se describen las principales responsabilidades de cada uno de los puestos en el equipo de desarrollo durante las fases de Inicio y Elaboración, de acuerdo con los roles que desempeñan en RUP.

Puesto	Responsabilidad
Analista	Captura, especificación y validación de requisitos, interactuando con el cliente y los usuarios mediante entrevistas. Colaboración en la elaboración de las pruebas funcionales y el modelo de datos.

Programador	Son los responsables del desarrollo del proyecto, desarrollando los módulos del proyecto, creando versiones de los módulos hasta llegar a la versión final.

Tabla 24: Roles y responsabilidades

Fuente: Propia

4.1.2.4. Gestión del Proceso

4.1.2.4.1. Plan del Proyecto

En esta sección se presenta la organización en fases e iteraciones y el calendario del proyecto.

4.1.2.4.1.1. Plan de las Fases

El desarrollo se llevará a cabo en base a fases con una o más iteraciones en cada una de ellas. La siguiente tabla muestra una la distribución de tiempos y el número de iteraciones de cada fase (para las fases de Construcción y Transición es sólo una aproximación muy preliminar)

Fase	Nro. Iteraciones	Duración
Fase de Inicio	1	5 semanas
Fase de Elaboración	1	8 semanas
Fase de Construcción	2	16 semanas
Fase de Transición	1	4 semanas

Tabla 25: Plan de fases

Fuente: Propia

Los hitos que marcan el final de cada fase se describen en la siguiente tabla.

Descripción	Hito
Fase de Inicio	En esta fase desarrollará los requisitos del producto desde la perspectiva del usuario, los cuales serán establecidos en el artefacto Visión, previamente establecidos los requerimientos en las actas de trabajo. Los principales casos de uso serán identificados y se hará un refinamiento del Plan de Desarrollo del Proyecto. La aceptación del cliente / usuario del artefacto Visión y el Plan de Desarrollo marcan el final de esta fase.
Fase de Elaboración	En esta fase se analizan los requisitos y se desarrolla un prototipo de arquitectura (incluyendo las partes más relevantes y / o críticas del sistema). Al final de esta fase, todos los casos de uso correspondientes a requisitos que serán implementados en la primera release de la fase de Construcción deben estar analizados y diseñados (en el Modelo de Análisis / Diseño). La revisión y aceptación del prototipo de la arquitectura del sistema marca el final de esta fase. En nuestro caso particular, por no incluirse las fases siguientes, la revisión y entrega de todos los artefactos hasta este punto de desarrollo también se incluye como hito. La primera iteración tendrá como objetivo la identificación y especificación de los principales casos de uso, así como su realización preliminar en el Modelo de Análisis / Diseño, también permitirá hacer una revisión general del estado de los artefactos hasta este punto y ajustar si es necesario la planificación para asegurar el cumplimiento de los objetivos. Ambas iteraciones tendrán una duración de una semana.
Fase de Construcción	Durante la fase de construcción se terminan de analizar y diseñar todos los casos de uso, refinando el Modelo de Análisis / Diseño. El producto se construye en base a 2 iteraciones, cada una produciendo una release a la cual se le aplican las pruebas y se valida con el cliente / usuario. Se comienza la elaboración de material de apoyo al usuario. El hito que marca el fin de esta fase es la versión de la release 3.0, con la capacidad operacional parcial del producto que se haya considerado como crítica, lista para ser entregada a los usuarios para pruebas beta.
Fase de Transición	En esta fase se prepararán dos releases para distribución, asegurando una implantación y cambio del sistema previo de manera adecuada, incluyendo el entrenamiento de los usuarios. El hito que marca el fin de esta fase incluye, la entrega de toda la documentación del proyecto con los manuales de instalación y todo el material de apoyo al usuario, la finalización del entrenamiento de los usuarios y el empaquetamiento del producto.

Tabla 26: Fases del proyecto

Fuente: Propia

4.1.2.4.1.2. Objetivos de las Iteraciones

Fase	Iteración	Descripción	Hitos Asociados	Objetivos
Incepción	Iteración Inicial	Definición del modelo de negocio y plan de proyecto.	Actas de Trabajo. Documento de Visión.	**1.** Clasificar de forma general los requerimientos del Usuario. **2.** Determinar el alcance e impacto del proyecto. **3.** Determinar la factibilidad del proyecto desde el punto de vista del negocio.
Elaboración	Desarrollar Prototipo	Desarrollar el prototipo de la Arquitectura. Definir y diseñar todos los casos de uso.	Documento de Arquitectura.	**1.** Definir arquitectura y herramientas. **2.** Realizar la lista de riegos técnicos. Prototipo temprano para revisión de usuarios.
Construcción	**Primera Iteración:** Desarrollar versión beta del producto.	Implementar y probar los casos de uso para la versión beta.	Versión beta del producto	Implementación de todas las características claves obtenidas de la arquitectura realizada previamente y desde la prospectiva del usuario.
	Segunda Iteración: Desarrollar el reléase inicial.	Implementación y desarrollo de los casos de uso restantes, corregir errores y observaciones de la versión beta.	Producto (Software)	Software revisado por los usuarios involucrados en cada uno de los módulos del sistema. Alta calidad del software
	Tercera Iteración: Desarrollo	Corregir y mejorar los defectos del reléase inicial.	Producto (Software)	Toda la funcionalidad del sistema completa para su liberación.

	del sistema completo.	Desarrollo del sistema completo.		
Transición	Liberar el Software.	Empaqueta e instalar el reléase.	Software liberado	Sistema en producción.

Tabla 27: Objetivos de iteraciones

Fuente: Propia

4.1.2.4.1.3. Calendario del Proyecto

A continuación se presenta un calendario de las principales tareas del proyecto incluyendo sólo las fases de Inicio y Elaboración. Como se ha comentado, el proceso iterativo e incremental de RUP está caracterizado por la realización en paralelo de todas las disciplinas de desarrollo a lo largo del proyecto, con lo cual la mayoría de los artefactos son generados muy tempranamente en el proyecto pero van desarrollándose en mayor o menor grado de acuerdo a la fase e iteración del proyecto.
La siguiente figura ilustra este enfoque, en ella lo ensombrecido marca el énfasis de cada disciplina (workflow) en un momento determinado del desarrollo.

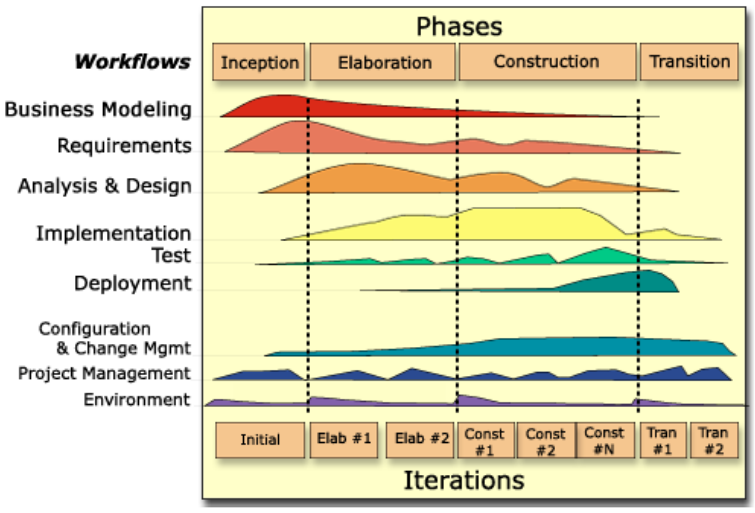


Figura 39: Calendario del proyecto

Fuente: Propia

Para este proyecto se ha establecido el siguiente calendario. La fecha de aprobación indica cuándo el artefacto en cuestión tiene un estado de completitud suficiente para someterse a revisión y aprobación, pero esto no quita la posibilidad de su posterior refinamiento y cambios.

Disciplinas / Artefactos generados o modificados durante la Fase de Inicio	Comienzo	Aprobación
Modelado del Negocio		
Modelo de Casos de Uso del Negocio y Modelo de Objetos del Negocio	Semana 1 14/10 – 20/10	Semana 3 28/10 – 3/11
Requisitos		
Glosario	Semana 1 14/10 – 20/10	Semana 3 28/10 – 3/11
Visión	Semana 2 21/10 – 27/10	Semana 3 28/10 – 3/11
Modelo de Casos de Uso	Semana 3 28/10 – 3/11	siguiente fase
Especificación de Casos de Uso	Semana 3 28/10 – 3/11	siguiente fase
Análisis / Diseño		
Modelo de Análisis / Diseño	Semana 2 21/10 – 27/10	siguiente fase
Modelo de Datos	Semana 2 21/10 – 27/10	siguiente fase
Implementación		
Modelo de Implementación	Semana 3 28/10 – 3/11	siguiente fase
Pruebas		
Casos de Pruebas Funcionales	Semana 3	siguiente fase

	28/10 – 3/11	
Gestión de Cambios y Configuración	Durante todo el proyecto	
Gestión del proyecto		
Plan de Desarrollo del Software en su versión 1.0 y planes de las Iteraciones	Semana 1 14/10 – 20/10	Semana 3 28/10 – 3/11
Ambiente	Durante todo el proyecto	

Tabla 28: Calendario de fechas

Fuente: Propia

Disciplinas / Artefactos generados o modificados durante la Fase de Elaboración	Comienzo	Aprobación
Modelado del Negocio		
Modelo de Casos de Uso del Negocio y Modelo de Objetos del Negocio	Semana 1 14/10 – 20/10	aprobado
Requisitos		
Glosario	Semana 1 14/10 – 20/10	aprobado
Visión	Semana 2 21/10 – 27/10	aprobado
Modelo de Casos de Uso	Semana 3 28/10 – 3/11	Semana 5 11/12 – 17/12
Especificación de Casos de Uso	Semana 3 28/10 – 3/11	Semana 5 11/12 – 17/12
Análisis / Diseño		

Modelo de Datos	Semana 2 21/10 – 27/10	Revisar en cada iteración
Implementación		
Modelo de Implementación	Semana 3 28/10 – 3/11	Revisar en cada iteración
Pruebas		
Casos de Pruebas Funcionales	Semana 3 28/10 – 3/11	Revisar en cada iteración
Gestión de Cambios y Configuración	Durante todo el proyecto	
Gestión del proyecto		
Plan de Desarrollo del Software en su versión 2.0 y planes de las Iteraciones	Semana 4 4/11 – 10/11	Revisar en cada iteración
Ambiente	Durante todo el proyecto	

Tabla 29: Calendario de fechas

Fuente: Propia

4.1.2.4.2. Seguimiento y Control del Proyecto

4.1.2.4.2.1. Gestión de Requisitos

Los requisitos del sistema son especificados en el artefacto Visión. Cada requisito tendrá una serie de atributos tales como importancia, estado, iteración donde se implementa, etc., estos atributos permitirán realizar un efectivo seguimiento de cada requisito.

Los cambios en los requisitos serán gestionados mediante una solicitud de cambio, las cuales serán evaluadas y distribuidas para asegurar la integridad del sistema y el correcto proceso de gestión de configuración y cambios.

4.1.2.4.2.2. Control de Plazos

El calendario del proyecto tendrá un seguimiento y evaluación semanal por el jefe de proyecto y por la persona encargada del seguimiento y control asignada en la institución.

4.1.2.4.2.3. Control de Calidad

Los defectos detectados en las revisiones y formalizados también en una solicitud de cambio tendrán un seguimiento para asegurar la conformidad respecto de la solución de dichas deficiencias.
Para la revisión de cada artefacto y su correspondiente garantía de calidad se utilizarán las guías de revisión y checklist (listas de verificación) incluidas en RUP.

4.1.2.4.2.4. Gestión de Riesgos

A partir de la fase de Inicio se mantendrá una lista de riesgos asociados al proyecto y de las acciones establecidas como estrategia para mitigarlos o acciones de contingencia. Esta lista será evaluada al menos una vez en cada iteración.

4.1.2.4.2.5. Gestión de Configuración

Se realizará una gestión de configuración para llevar un registro de los artefactos generados y sus versiones. También se incluirá la gestión de las Solicitudes de Cambio y de las modificaciones que éstas produzcan, informando y publicando dichos cambios para que sean accesibles a todo los participantes en el proyecto. Al final de cada iteración se establecerá una baseline (un registro del estado de cada artefacto, estableciendo una versión), la cual podrá ser modificada sólo por una Solicitud de Cambio aprobada.

4.1.2.5. Referencias

1. Proyecto SIGPRE Automatización del proceso para la gestión de la reforma presupuestaria de EMELNORTE de Edwin Roberto López Hinojosa.
2. Sistema para Gestión de Artículos Deportivos LSI de Patricio Letelier Torres y César López Rodríguez.
3. Visita Virtual de la infraestructura física y tecnológica de la Universidad Técnica del Norte de Christian Guerrón y Roger Catucuamba.

4. Documentación de Rational Unified Process, manuals de ayuda, tutoriales, etc.

4.1.3. Actas de trabajo

4.1.3.1. Introducción

Las actas de trabajo, son documentos necesarios en esta fase de inicio, estas permiten levantar una gran cantidad de requerimientos en cada una de las entrevistas acordadas con los interesados del sistema, además sirven de respaldo en los alcances del proyecto.
En el desarrollo del Módulo Contable de Finansys se han levantado un total diez actas de trabajo, y en cada una de ellas se obtienen requisitos indispensables para el desarrollo del sistema.

4.1.3.1.1. Propósito

El propósito de las Actas de Trabajo es levantar los procesos de información necesaria para definir y planificar el desarrollo del sistema.

4.1.3.1.2. Acta de Trabajo nro. 1

ACTA DE TRABAJO No. 1		
1. Proyecto: Sistema integrado de información financiera para la cooperativa de ahorro y crédito "Unión Cochapamba"		
Tema a tratar: Levantamiento de requerimientos funcionales del sistema a implantar.		
Fecha: 14/07/2011		
Participantes:		
Nombre	**Cargo**	**Firma**
Dra. Irene Mena	Contadora	
Edgar Picuasi	Tesista	
Observaciones:		
Para el desarrollo del sistema integrado de información financiera para la cooperativa de ahorro y crédito "Unión Cochapamba" se recopilan datos necesarios para estructurar el diseño del módulo.		

Se obtiene los siguientes requerimientos y se tratan los siguientes temas: • Revisión del sistema anterior y preguntas de los módulos que contiene el sistema. • Identificación de las dificultades del sistema anterior. • Propuestas para definir el nombre que del sistema Finansys. • Propuesta para desarrollo web con herramientas libres.
Compromisos adquiridos:
- Describir las herramientas a ser utilizadas a los directivos.

El listado siguiente de actas de trabajo se encuentra en el cd de anexos del proyecto.

1. **02ATMCPlanCuentas.**
2. **03ATMCComprobantes.**
3. **04ATMCPeriodos**
4. **05ATMCTiposControl**
5. **06ATMCMayorizarCerrar**
6. **07ATMCBancos**
7. **08ATMCConsultas**
8. **09ATMCReportes**

4.2. Fase de Elaboración

4.2.1. Documento de arquitectura

4.2.1.1. Introducción

4.2.1.1.1. Propósito

El presente documento presenta una vista de alto nivel de la arquitectura del Módulo Contable (MC) de Finansys, objetivos, restricciones, los casos de uso más relevantes, los patrones de arquitectura aplicados y las principales decisiones sobre el diseño del módulo.

4.2.1.1.2. Alcance

El alcance de la arquitectura es definir los componentes arquitectónicos del sistema para tener una visión clara del sistema que describa los aspectos funcionales del mismo.

4.2.1.2. Representación de la Arquitectura

El presente documento presenta la arquitectura como una serie de vistas los modelos han sido desarrollados usando Visio, Jdeveloper y el lenguaje UML.
El modelo propuesto por RUP para representar la arquitectura utiliza el siguiente conjunto de vistas:

1. **Vista de Casos de Uso:** lista los casos de uso o escenarios del modelo de casos de uso que representen funcionalidades centrales del sistema final, que requieran una gran cobertura arquitectónica o aquellos que impliquen algún punto especialmente delicado de la arquitectura.
2. **Vista Lógica:** describe las partes arquitectónicamente significativas del modelo de diseño, como ser la descomposición en capas, subsistemas o paquetes. Una vez presentadas estas unidades lógicas principales, se profundiza en ellas hasta el nivel que se considere adecuado.
3. **Vista de Procesos:** describe la descomposición del sistema en threads y procesos pesados. Indica que procesos o grupos de procesos se comunican o interactúan entre sí y los modos en que estos se comunican.

4. **Vista de Deployment:** describe uno o más escenarios de distribución física del sistema sobre los cuales se ejecutará y hará el deploy del mismo. Muestra la comunicación entre los diferentes nodos que componen los escenarios antes mencionados, así como el mapeo de los elementos de la Vista de Procesos en dichos nodos.
5. **Vista de Implementación:** describe la estructura general del Modelo de Implementación y el mapeo de los subsistemas, paquetes y clases de la Vista Lógica a subsistemas y componentes de implementación.
6. **Vista de Datos:** describe los elementos principales del Modelo de Datos, brindando un panorama general de dicho modelo en términos de tablas, vistas, índices, etc.

4.2.1.3. Objetivos y Restricciones de la Arquitectura

Existen requerimientos y restricciones de relevancia para la definición de la arquitectura:

1. Diseño basado en capas de propósito claro y concreto y con alto grado de cohesión.
2. Desacoplamiento entre capas que permita el fácil reemplazo de los mismos.
3. Capas altamente reutilizables.
4. Todos los requerimientos descritos en el documento de Visión deben ser tomados en consideración para el desarrollo de la arquitectura definida.

4.2.1.4. Vista de Casos de Uso

4.2.1.4.1. Proceso

4.2.1.4.1.1. Proceso Módulo Contabilidad

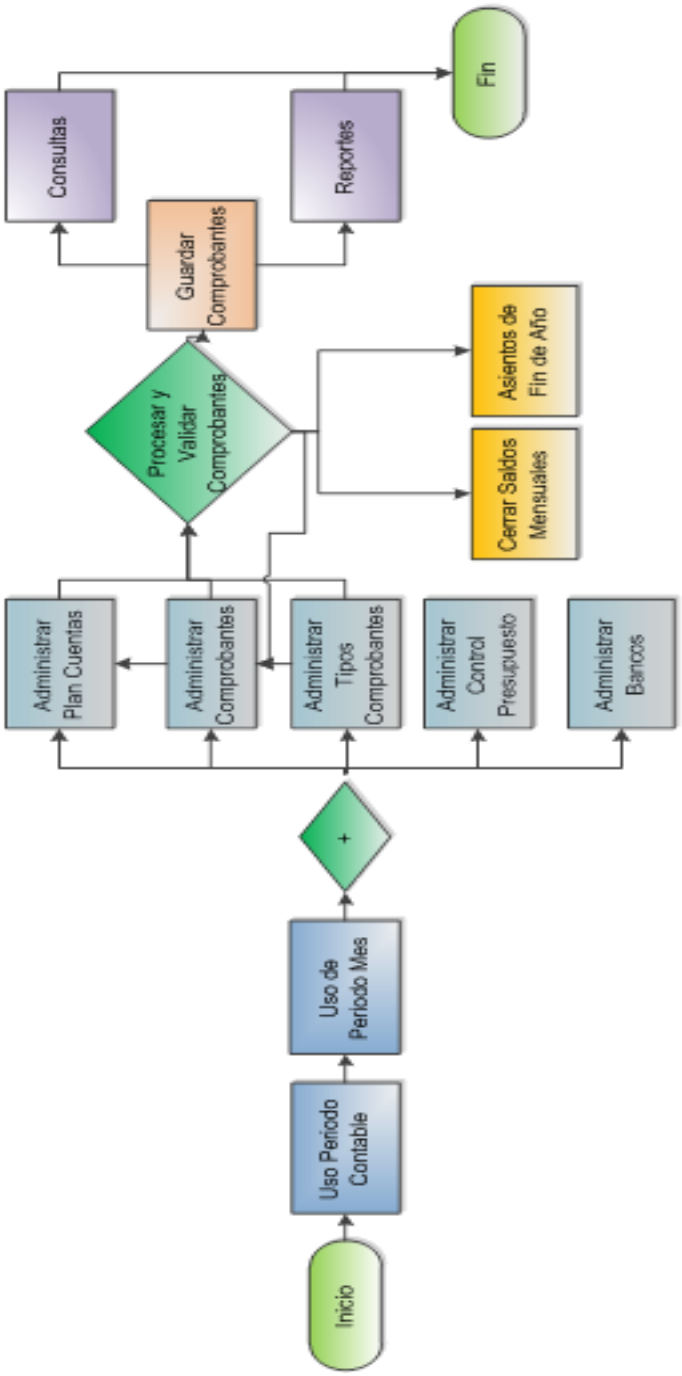


Figura 40: Procesos del MC

Fuente: Propia

4.2.1.4.2. Actores

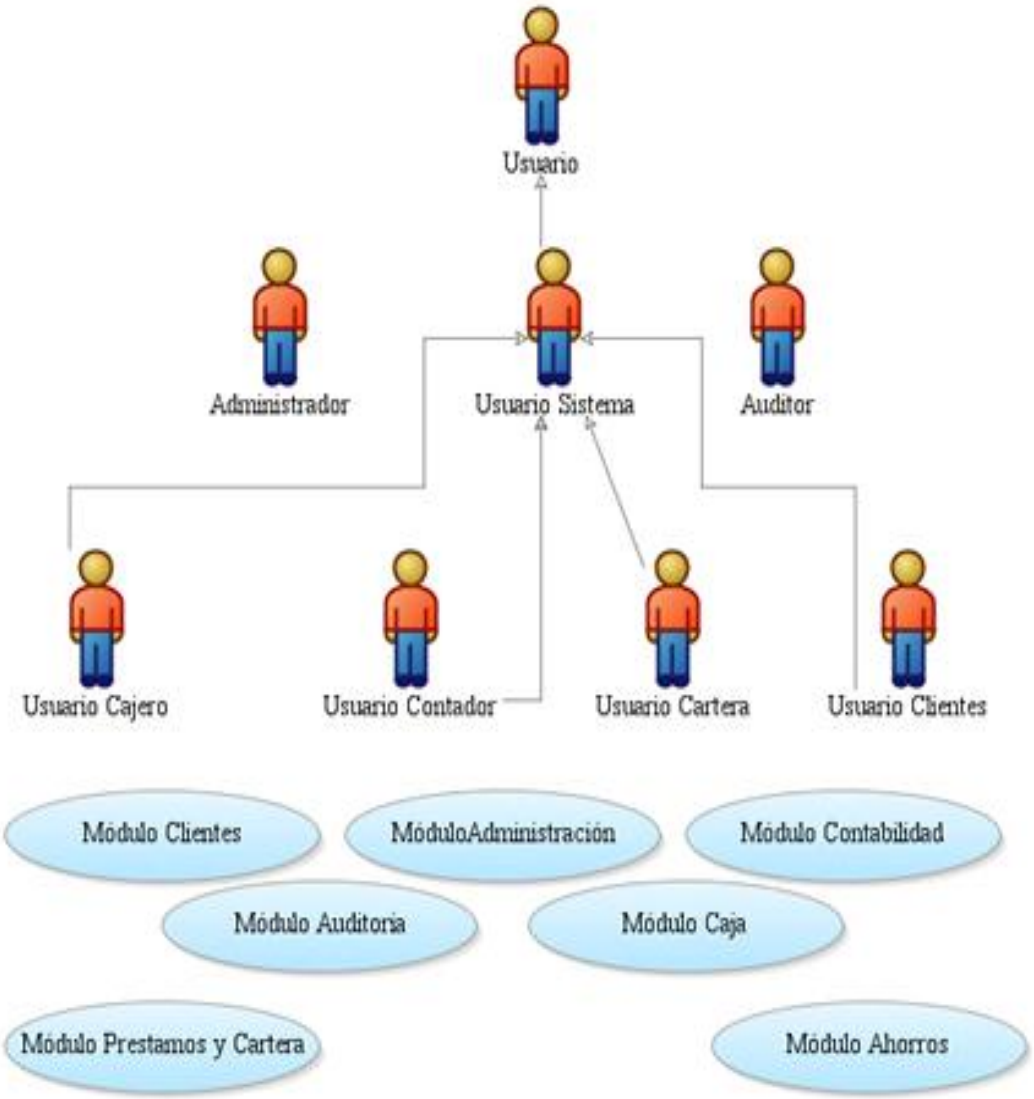


Figura 41: Actores

Fuente: Propia

Nro.	Stakeholder	Descripción
1	Usuario	Usuario general del sistema que puede ingresar requerimientos y consultar la base de datos
2	Administrador	Administrador del sistema quién gestiona los objetos y recursos del sistema
4	Usuario Auditor	Realiza consultas de ingresos y salidas de usuarios, movimientos (ingreso, modificación, eliminación) de la información del sistema.
5	Usuario Contador	Procesa todas las operaciones contables que implica el sistema financiero.
6	Usuario Clientes	Se encarga de actividades relacionadas con clientes nuevos y existentes.
7	Usuario Cartera	Analiza la simulación, viabilidad, registro y asignación de préstamos a los clientes.
8	Usuario Cajero	Se encarga de realizar todas las transacciones de los clientes en las actividades de depósitos retiros, pagos de préstamos, etc.

Tabla 30: Descripción de clientes

Fuente: Propia

4.2.1.4.3. Modelos de Casos de Negocio

Presenta una visión global del sistema desde el punto de vista del negocio.

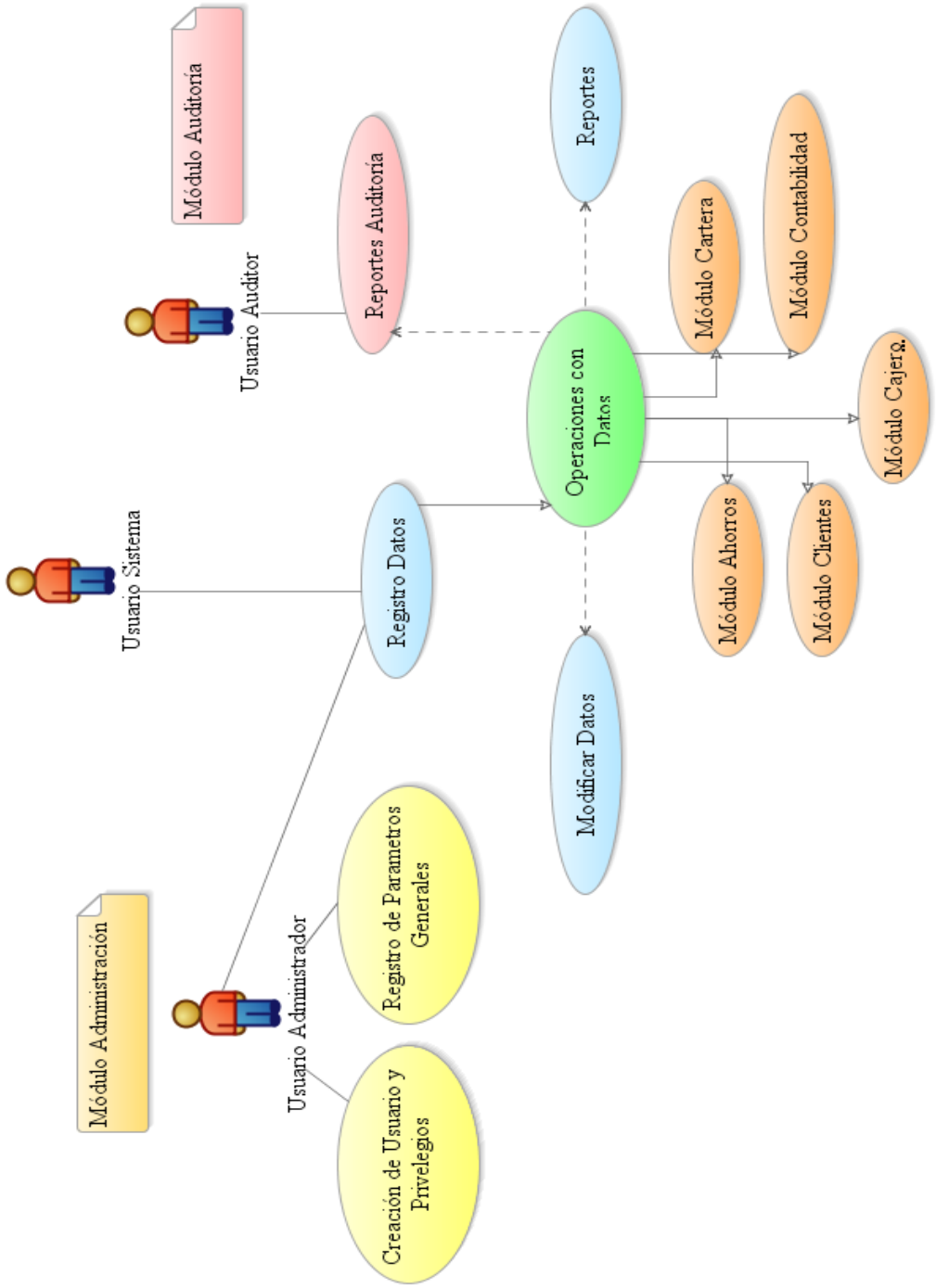


Figura 42: Casos de Negocio

Fuente: Propia

4.2.1.4.4. Modelos de Casos de Uso Finansys

El gráfico con todos los casos de uso con cada uno de los módulos se encuentra en anexos. 8.2.

4.2.1.4.5. Prioridad de Casos de Uso

Caso de Uso	Prioridad para el Negocio	Prioridad Técnica
Módulo Contable		
1. Administración Plan de Cuentas	Alta	Alta
2. Administración de Comprobante	Alta	Alta
3. Administrar Tipos de Comprobante	Media	Media
4. Administración de Periodos Contables	Alta	Alta
5. Administración de Periodos Mensuales	Alta	Media
6. Cerrar Saldos Mes	Alta	Alta
7. Conciliación Bancaria	Alta	Alta
8. Control de Presupuesto	Media	Media
9. Revisar Control	Baja	Media
10. Asientos de Fin de Año	Alta	Alta
11. Consultas	Alta	Alta
12. Reportes	Alta	Alta

Tabla 31: Prioridad de Casos de Uso

Fuente: Propia

4.2.1.4.6. Modelos de Caso de Uso

4.2.1.4.6.1. Modelo de Caso de uso Módulo de Contabilidad

La figura de casos de uso mc fue detallada en el capítulo 3.

4.2.1.4.7. Descripción de los Casos de Uso más relevantes

4.2.1.4.7.1. Descripción Módulo Contabilidad

1. Administración Plan de Cuentas

Permite administrar todas las cuentas contables, es decir la creación, modificación y eliminación de todo el plan de cuentas.

2. Administración de Comprobantes

Controla todos los comprobantes generados en todos los módulos y los utiliza para crear toda la información contable.

3. Administrar Tipos de Comprobante

Crea tipos de comprobantes necesarios para crear un comprobante, cada módulo genera un comprobante con un tipo de comprobante específico.

4. Administración de Periodos Contables

Permite utilizar un período contable necesario para segmentar la información generada por el sistema.

5. Administración de Periodos Mensuales

Permite utilizar periodos mes necesarios para segmentar la información contenida dentro de un período contable.

6. Cerrar Saldos Mes

Cierra los saldos del mes, guarda los saldos finales de las cuentas de movimiento y da paso a un nuevo período mes.

7. Conciliación Bancaria

Permite realizar la conciliación de bancos.

8. Control de Presupuesto

Permite crear un control de presupuesto por año, necesario para trazar fondos en cuanto a ingresos y gastos, no altera a los saldos de las cuentas es solo informativa.

9. Revisar Control

Permite revisar el control de presupuesto dependiendo de un período contable.

10. Asientos de Fin de Año

Realiza los asientos de fin de año para dar paso a un nuevo período contable.

11. Consultas

Permite acceder a consultas necesarias para el usuario contador.

12. Reportes

Permite generar reportes contables necesarios para comprobar la efectividad de los movimientos y el desempeño de la cooperativa.

4.2.1.5. Vista de Restricciones

En esta vista se presentan las restricciones normativas, de estándares y de tecnología, a las cuales está sujeto tanto el proceso de desarrollo como el producto desarrollado, incluidas en las categorías soporte, implementación e interfaces.

4.2.1.5.1. Normativas

Existen restricciones y normativas, dictadas por el gobierno para entidades financieras las cuales el sistema actual no le permite a la Cooperativa acoplarse en el manejo de la información financiera generada por la institución.

4.2.1.5.2. Licenciamiento

Dado que el Sistema financiero a desarrollar utiliza en su totalidad varias herramientas, estándares y tecnologías libres implementadas en java con licencia GNU GPL, el Sistema financiero no necesita inversión alguna en el licenciamiento de las herramientas a ser utilizadas.

4.2.1.5.3. Estándares

UML

Todo artefacto utilizado para comunicación y documentación, tanto entre miembros del equipo de desarrollo como con los usuarios, está basado en UML.

Interfaz Web

La interfaz de usuario debe estar orientada a la web. Debe ser posible visualizar el contenido utilizando cualquiera de los browsers más difundidos

4.2.1.5.4. Tecnología

El desarrollo completo del Sistema debe realizarse bajo los siguientes parámetros tecnológicos

Elemento	Tecnología o Frameworks	Observaciones
Base de Datos	PostgreSQL	Gestor de bases de datos, de gran escalabilidad, multiplataforma, estable y confiable, diseñado para ambientes de alto volumen, libre bajo licencia BSD.
Framework	JSF	Tecnología y Framework para aplicaciones java de entornos web, con un modelo de programación de capas, potente por su fácil desarrollo, desempeño, adopción.
	RichFaces	Librería de código abierto basada en Java para componentes visuales en JSF, que permite crear aplicaciones web con Ajax
	MVC	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
IDE	NetBeans	Entorno de Desarrollo integrado libre, principalmente para el lenguaje Java.
Servidor Web	Tomcat	Servidor Web libre, compatible con Java, muy fiable, multiplataforma.
Tipo MIME	CSS	Lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XHTML, separando la estructura y presentación de un documento.
	HTML	Lenguaje de marcado predominante para la elaboración de páginas web. Es usado para describir la estructura y el contenido en forma de texto
	JavaScript	Lenguaje de programación interpretado, usado principalmente para mejorar la interfaz de usuario y páginas web dinámicas.
Control de Versiones	CVS	Mantiene el registro de todo el trabajo y los cambios en los ficheros que forman parte de un proyecto, permitiendo que distintos desarrolladores colaboren.
Sistema Operativo	De Desarrollo: Fedora	Distribución Linux muy estable, de propósitos generales basada en RPM respaldada por Red Hat.
	De Producción: Centos	Distribución Linux popular en el segmento de servidores web, derivada y con total compatibilidad de Red Hat.

Herramienta de Reportes	JasperReports	Herramienta escrita en Java para la creación de informes con contenido enriquecido.
Metodología	RUP	Metodología estándar utilizada para el análisis, implementación y documentación de sistemas orientados a objetos

Tabla 32: Tecnologías Empleadas.

Fuente: Propia

4.2.1.5.5. Sistemas Existentes

4.2.1.5.5.1. Sistema Financiero CUC

La CUC (Cooperativa Unión Cochapamba) en la actualidad trabaja con un sistema programado en C++, El cual se encarga en la totalidad de procesar toda la información financiera generada, sin embargo dicho sistema presenta limitaciones, porque no controla y administra todos los procesos de forma unificada, además los datos no se guardan en ningún gestor de base de datos sino en archivos planos.

4.2.1.6. Vistas Lógicas

Finansys comprende 3 vistas principales como son las siguientes: Arquitectura del sistema, arquitectura lógica y arquitectura de los módulos.

4.2.1.6.1. Arquitectura del Sistema

Finansys será desarrollado utilizando tecnología Web, desarrollando la parte visual con RichFaces optimizando de esta forma el rendimiento y facilidad de uso por parte de los usuarios, creando pantallas con una presentación clara y atractiva gracias a títulos barras de estado, íconos intuitivos, tipografía legible, etc., la figura que se muestra a continuación indica la arquitectura básica de desarrollo y separa sus funcionalidades en tres capas diferenciando claramente el funcionamiento y el proceso del sistema. Esta lógica se basa en el patrón arquitectónico MVC (Modelo Vista Controlador).

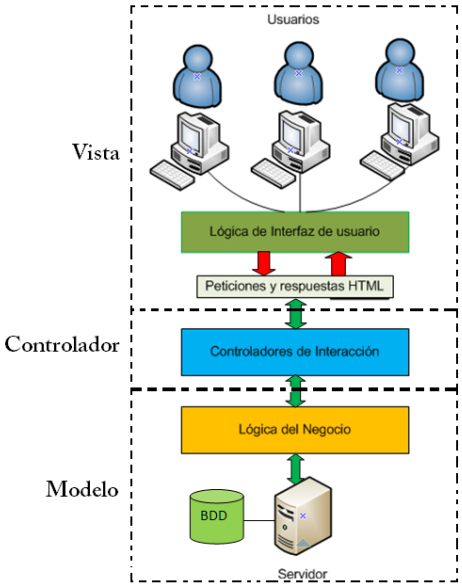


Figura 43: Patrón Arquitectónico MVC.

Fuente: Propia

La Arquitectura MVC fue planteada para disminuir el esfuerzo y complejidad de programación necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos logrando con esto una buena funcionalidad y sobre todo entendibles lo que permite fácilmente la creación de nuevas funciones.

Sus capas principales son que el Modelo, las Vistas y los Controladores las cuales se manipulan como entidades separadas, lo que significa que cualquier cambio realizado en el Modelo se manifieste automáticamente en las Vistas.

4.2.1.6.2. Arquitectura Lógica

Como refinamiento de la arquitectura del sistema especificada anteriormente, a continuación se muestra la arquitectura lógica del sistema.

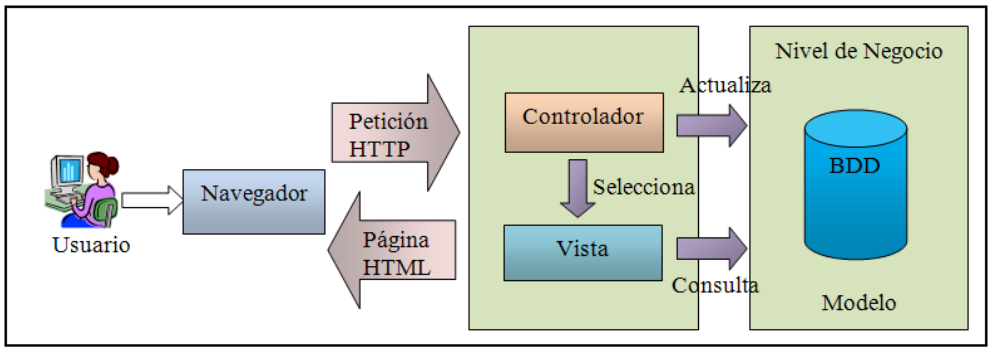


Figura 44: Arquitectura Lógica

Fuente: Propia

4.2.1.6.2.1. Interfaz de Usuario

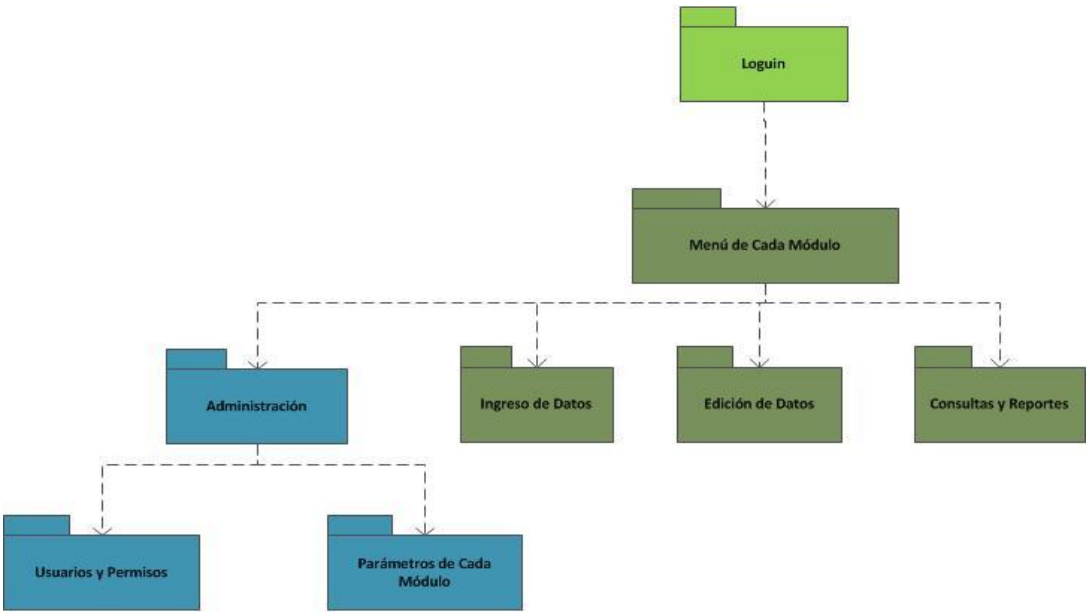


Figura 45: Interfaz de Usuario

Fuente: Propia

4.2.1.6.3. Servicios del Sistema

Al tratarse de un sistema financiero, la validación de los datos es un proceso muy importante a tener en cuenta en Finansys, validando correctamente los flujos de datos de entrada, para verificar que el tipo de datos sea el esperado y avisando al usuario cuando exista algún error. Logrando que la información guardada sea congruente y adecuada.

4.2.1.6.4. Infraestructura

Corresponde a los requerimientos de infraestructura que necesita el sistema Finansys para su funcionamiento.

4.2.1.6.4.1. Aplicación

La aplicación será publicada en el servidor Web Tomcat, y estará accesible a los usuarios del sistema a través de la red local.

4.2.1.6.4.2. Datos

Para mantener la información estable y confiable se guarda una la base de datos PostgreSQL.

4.2.1.6.4.3. Sistema Operativo

La aplicación será instalada y configurada en una máquina con Centos como su sistema operativo.

4.2.1.6.5. Vista de Datos

La Siguiente vista muestra gráficamente la estructura de la base de datos, presentando las tablas con sus respectivos campos y las relaciones existentes entre tablas.

4.2.1.7. Vista de Despliegue

El siguiente diagrama de despliegue muestra las relaciones físicas de los distintos nodos que componen el sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación.

Como se puede observar en la figura, un usuario podrá acceder a la aplicación mediante un browser siendo especificado Firefox como browser necesario el cual ejecutará la página Jsf-RichFace con extensión jsp, el mismo que muestra la portada de login que permite el acceso al sistema. La conexión a la base de datos se realiza a través de Hibernate (Herramienta ORM).

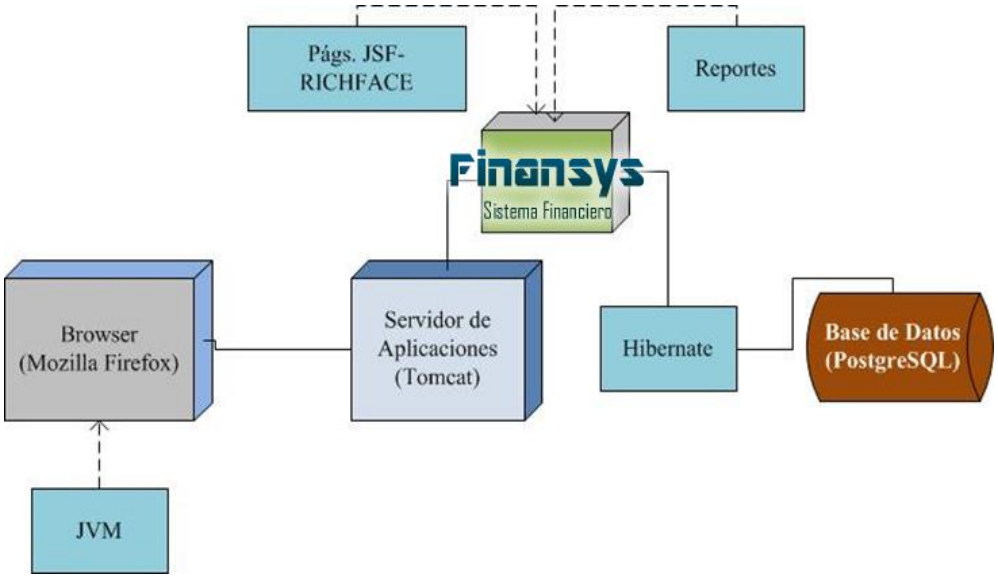


Figura 46: Vista de Despliegue.

Fuente: Propia

4.2.2. Casos de uso

El siguiente listado de casos de uso se encuentra en el CD de anexos.

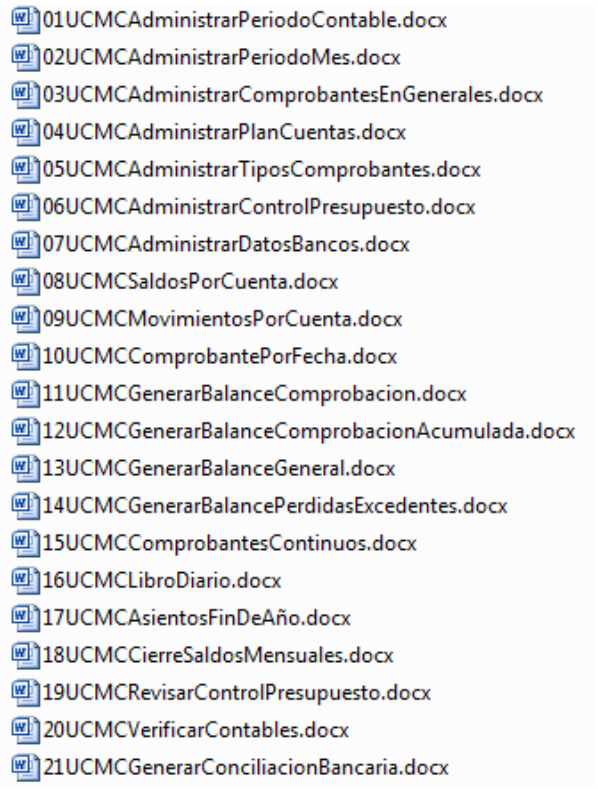


Figura 47: Listado de Casos de Uso

Fuente: Propia

4.3. Fase de Construcción

4.3.1. Modelo de datos

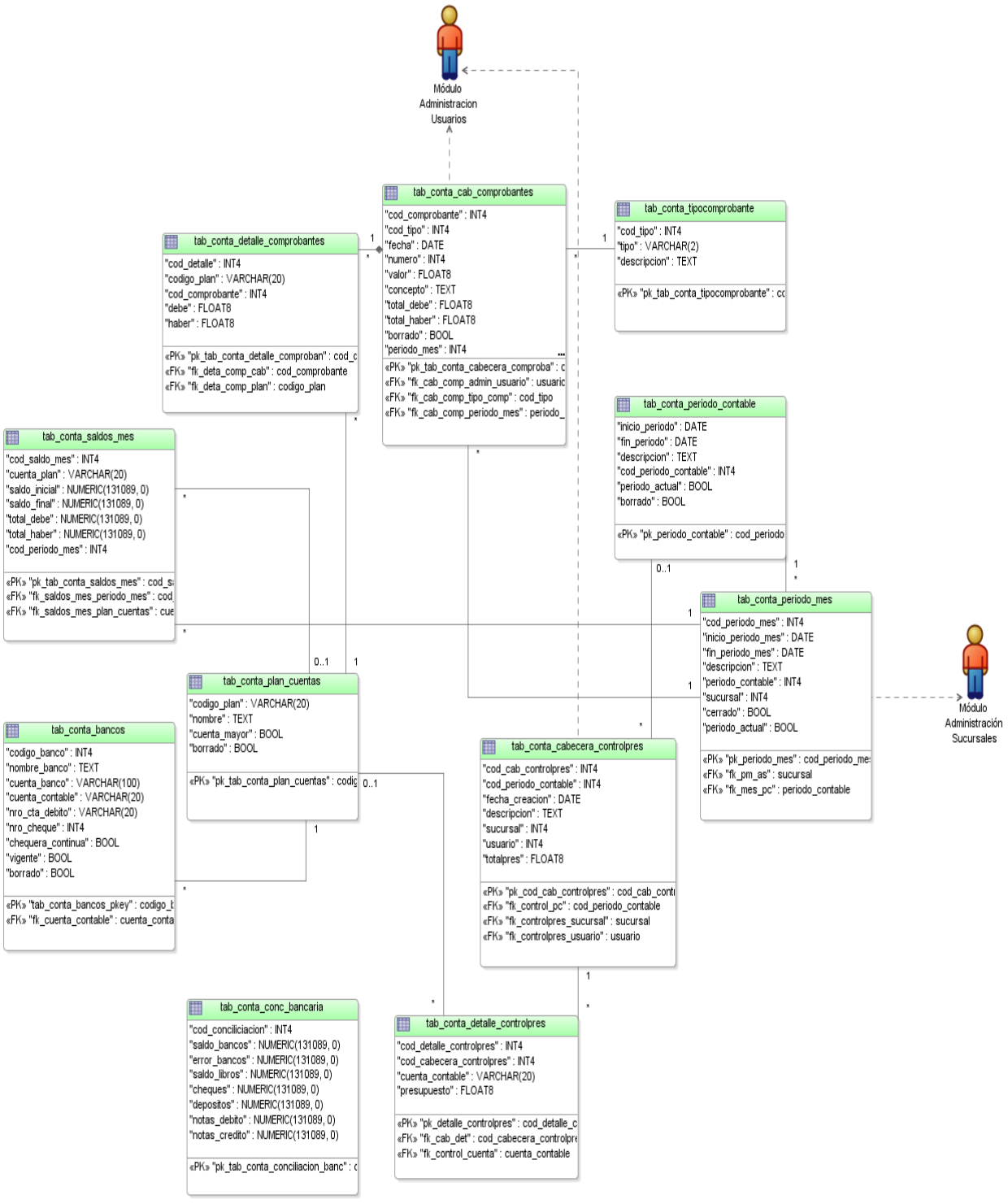


Figura 48: Modelo Base de Datos Módulo Contable.

Fuente: Propia

4.3.2. Plan de Pruebas

4.3.2.1. Introducción

El presente documento presenta el Plan de Pruebas como respuesta al proyecto Finansys para la cooperativa de ahorro y crédito "Unión Cochapamba" (en adelante CUC).

4.3.2.1.1. Propósito

El propósito del Plan de Pruebas consiste en la creación de pruebas rutinarias a Finansys con el objetivo de identificar posibles errores e inconsistencias.

4.3.2.1.2. Alcance

Este Plan de Pruebas aplica a todos los componentes necesarios para registrar, modificar o borrar información de Finansys durante el proceso financiero.

4.3.2.2. Estrategia de Pruebas

La estrategia define como se realizarán las pruebas, las consideraciones principales para la estrategia de prueba son las técnicas a ser usadas y el criterio para saber cuándo las pruebas están completas.

4.3.2.2.1. Tipos de Pruebas

4.3.2.2.1.1. Pruebas de Integridad de Datos

Objetivo:	Asegurar la integridad de datos
Técnica:	**1.** Registrar datos con tipos válidos. **2.** Registrar datos en entidad que tengan relación con otras. **3.** Revisar el esquema de base de datos para asegurarse que los datos se han guardado satisfactoriamente y de acuerdo a los estándares definidos.
Criterio de completitud	**1.** Todos los métodos de acceso y procesos de la Base de datos funcionan como fueron diseñados.

Consideraciones especiales	**1.** Se debe utilizar un conjunto pequeño de datos para incrementar la visibilidad de cualquier evento anormal o inesperado. **2.** Los datos de pruebas deberían ser reales y de uso común.

Tabla 33: Pruebas de Integridad de Datos.

Fuente: Propia

4.3.2.2.1.2. Pruebas del Sistema

Objetivo:	Asegurar la apropiada navegación dentro del sistema, ingreso de datos, procesamiento y recuperación.
Técnica:	Ejecute cada caso de uso, flujo básico o función utilizando datos válidos e inválidos, para verificar que: **1.** Los resultados esperados ocurren cuando se utiliza un dato válido. **2.** Los mensajes de error o de advertencia aparecen en el momento adecuado, cuando se utiliza un dato inválido. **3.** Cada regla de negocios es aplicada adecuadamente.
Criterio de completitud	**1.** Todas las pruebas planeadas han sido ejecutadas. **2.** Todos los defectos que se identificaron han sido tenidos en cuenta.
Consideraciones especiales	Considerar aspectos que impactan la implementación y ejecución de las pruebas del Sistema

Tabla 34: Pruebas del sistema

Fuente: Propia

4.3.2.2.1.3. Pruebas del Ciclo del negocio

Objetivo:	Asegurar que el sistema funciona de acuerdo con el modelo de negocios emulando todos los eventos en el tiempo y en función del tiempo.
Descripción de la prueba:	Las pruebas del ciclo de negocio deberían emular las actividades ejecutadas en el a través del tiempo. Debería identificarse un período, como por ejemplo un año, y las transacciones y actividades que podrían ocurrir durante un período de un año deberían ejecutarse. Incluyendo todos los ciclos y eventos diarios, semanales y mensuales.
Técnicas:	Ejecute cada caso de uso, flujo básico o función utilizando datos válidos e inválidos, para verificar que: **1.** Incremente el número de veces en que una función es ejecutada para simular diferentes usuarios sobre un período especificado **2.** Todas las fechas o funciones que involucren tiempos serán probadas con datos válidos e inválidos de fechas o periodos de tiempo. **3.** Todas las funciones ocurren en un período de tiempo serán ejecutadas en el tiempo apropiado. **4.** Los resultados esperados ocurren cuando los datos válidos son usados. **5.** Los mensajes de error o de advertencia aparecen en el momento adecuado, cuando se utiliza un dato inválido. **6.** Cada regla de negocios es aplicada adecuadamente.
Criterio de completitud	**1.** Todas las pruebas planeadas han sido ejecutadas. **2.** Todos los defectos que se identificaron han sido tenidos en cuenta.
Consideraciones especiales	**1.** Las fechas y eventos del sistema pueden requerir actividades especiales de soporte. **2.** Se requiere un modelo de negocios para identificar requisitos y procedimientos de prueba apropiados.

Tabla 35: Pruebas del negocio
Fuente: Propia

4.3.2.2.2. Pruebas de Interfaz de Usuario

Objetivo:	Verificar lo siguiente: La navegación a través de los objetos de la prueba refleja la funcionalidad del proyecto; Se realiza una navegación de todos los menús y los formularios de cada submenú. Los elementos de los formularios tales como botones, íconos, alertas, etc., deben mantener un solo formato.
Descripción de la prueba:	Las pruebas de interfaz de usuario verifican la adecuada interacción del usuario con el software. El objetivo es verificar que cada interfaz corresponda con la acción que realiza y que la interfaz tenga una adecuada navegación.
Técnicas:	Con la ayuda de los usuarios que usarán el sistema, se les pide que usen el sistema realizando las actividades y procesos cotidianos en el sistema. Los usuarios del sistema son reales y trabajan en su área de trabajo normal. Los desarrolladores no están presentes. Los usuarios son advertidos que el sistema puede fallar.
Criterio de completitud	Se establece un período de pruebas, en el que los errores detectados no sean clasificados como críticos para el sistema.
Consideraciones especiales	Se debe establecer el mecanismo de comunicación entre los usuarios y los desarrolladores para que los errores que se detecten puedan ser solucionados.

Tabla 36: Pruebas de interfaz de usuario

Fuente: Propia

4.3.2.2.3. Pruebas de Desempeño

Objetivo:	Validar y verificar los requisitos de desempeño especificados para el sistema. Validar el tiempo de respuesta para las transacciones o procesos bajo las siguientes condiciones: Volumen normal anticipado. Volumen máximo anticipado.
Descripción de la prueba:	Las pruebas de desempeño miden los tiempos de respuesta que tiene el sistema y otros aspectos sensibles al tiempo. Las pruebas se realizan varias veces, cambiando entre una y otra cargas diferentes. La prueba inicial debe ejecutarse con una carga similar a la esperada, mientras que la prueba final se ejecuta utilizando una carga máxima esperada. Las pruebas se pueden utilizar para calibrar el desempeño del sistema en función de condiciones como el hardware, usuarios, etc. Algunas características que pueden afectar el desempeño son: **1.** Cuellos de botella en el disco. **2.** Cuellos de botella en el CPU. **3.** Capacidad de almacenamiento. **4.** Capacidades físicas del hardware donde está alojado el sistema. **5.** Virus informáticos. **6.** Congestión en la red de datos.
Técnicas:	Con la ayuda de los usuarios que usarán el sistema, se les pide que usen el sistema realizando las actividades y procesos cotidianos en el sistema. Los usuarios del sistema son reales y trabajan en su área de trabajo normal. Se pide a los usuarios que todos accedan a una determinada

	acción o proceso al mismo tiempo. Los desarrolladores no están presentes. Los usuarios son advertidos que el sistema puede fallar.
Criterio de completitud	Se establece un período de pruebas, en el que los errores detectados no sean clasificados como críticos para el sistema.
Consideraciones especiales	Se debe establecer el mecanismo de comunicación entre los usuarios y los desarrolladores para que los errores que se detecten puedan ser solucionados.

Tabla 37: Pruebas de desempeño
Fuente: Propia

4.3.2.2.4. Pruebas de Seguridad y Control de Acceso

Objetivo:	Seguridad en el Funcionamiento y Datos, verificar que los usuarios puedan acceder solo aquellas funciones y datos para los cuales se le ha otorgado permisos al momento de crear su perfil de usuario. Seguridad en Administración del Sistema, comprobar que solo aquellos usuarios con permisos privilegiados puedan acceder a las funciones parametrizables del sistema y opciones del sistema.
Descripción de la prueba:	Las pruebas se realizará enfocándonos en dos aspectos principales: **1.** Seguridad en la aplicación controlando el acceso a determinada información y funciones del negocio, y **2.** Seguridad del sistema realizando registro de accesos de usuarios al sistema.
Técnicas:	**1.** Seguridad de Datos y Funciones, identificar los tipos de usuarios y asignar funciones a las que tiene acceso el tipo de usuarios para asignarle al usuario. **2.** Realizar pruebas para cada tipo de usuario y verificar los

	permisos creado transacciones para cada tipo de usuario. **3.** Re direccionamiento a la página de autenticación si el usuario aún no se ha logeado o registrado para el acceso al sistema. **4.** Modificar los tipos de usuario y verificar si los permisos han cambiado para el usuario.
Criterio de completitud	Para cada tipo de dato se puede asignar las funciones y datos apropiados para su desempeño.
Consideraciones especiales	El acceso al sistema debe ser revisado con el administrador de la red y de la base de datos. Con esto podremos visualizar cualquier anomalía.

Tabla 38: Pruebas de acceso

Fuente: Propia

4.3.2.3. Herramientas

Tarea	Herramientas
Registro de Defectos	Microsoft Word
Otras Herramientas de Prueba	SQLPLUS
Gestión de Proyecto	JDEVELOPER Microsoft Project Microsoft Word
Herramientas DBMS	PGADMIN III

Tabla 39: Herramientas

Fuente: Propia

4.3.2.4. Recursos

Rol	Recurso Requerido	Responsabilidad Específica
Administrador de Pruebas	Esteban Gudiño, Bairon Gudiño Edgar Picuasi	Proveer las directrices de las pruebas. Adquirir los recursos Necesarios.
Diseñador de Pruebas	Esteban Gudiño, Bairon Gudiño Edgar Picuasi	Identificar y priorizar las pruebas. Generar Plan de Pruebas
Gestión de Proyecto	Esteban Gudiño, Bairon Gudiño Edgar Picuasi	Responsables de ejecutar las pruebas, registro de resultados.
Administrador de BDD	Esteban Gudiño, Bairon Gudiño Edgar Picuasi	Administrar y asegurar los datos de pruebas.

Tabla 40: Recursos

Fuente: Propia

4.3.2.5. Entregables

Entregable	Propietario	Revisión/Distribución
Plan de Pruebas	Esteban Gudiño, Bairon Gudiño Edgar Picuasi	Coordinadores del proyecto

Tabla 41: Entregables

Fuente: Propia

4.3.3. Lista de Riesgos

Ranking	Descripción del Riesgo e Impacto	Estrategia reducción del riesgo
7	La liberación del sistema Finansys podría no estar lista para el mes de enero del 2012, mes en que se inicia un nuevo período contable.	Aumentar el esfuerzo.
7	Que la cooperativa no haga la adquisición del servidor a tiempo.	Adquisición del servidor.
5	Que las características del servidor donde será alojado el sistema, no cumpla con los requerimientos necesarios para un correcto funcionamiento del sistema.	Antes de su uso especificar los debidos requerimientos de hardware y software.
3	Abandono del proyecto por parte de los desarrolladores.	Aumentar el compromiso de terminar todo el sistema.
3	Incompatibilidad del navegador de internet de los usuarios.	Instalar el navegador compatible con el sistema.
3	Modificación de los requerimientos puestos en marcha.	Realizar actas de reunión de trabajo, definir bien los requerimientos y que el interesado apruebe dicha acta con la respectiva firma.
3	Saturación de la red informática por parte de los usuarios en actividades ajenas al sistema.	Sugerir políticas de uso de internet.
2	Presencia de virus en las computadores de los usuarios finales.	Instalar antivirus, actualizar constantemente.

Tabla 42: Lista de riesgos

Fuente: Propia

4.4. Fase de Transición

4.4.1. Manual de instalación

4.4.1.1. Introducción

4.4.1.1.1. Copiar el software al escritorio

En el CD del proyecto se encuentra una carpeta con todo el software necesario, copiar todos esos archivos y pegarlos en el escritorio. Se abre una consola, debe de autenticarse como súper usuario (root), acceder desde consola al directorio del escritorio (*cd /home/admin/Desktop)* y dar todos los permisos a todos los archivos (*chmod 777 *)*.
Luego de realizar este paso, estamos listos para proceder a instalar el software.

4.4.1.1.2. Instalación de la máquina virtual de Java

Desde la consola se debe digitar el siguiente comando: #rpm –ivh jdk-7u2-linux-i586.rpm

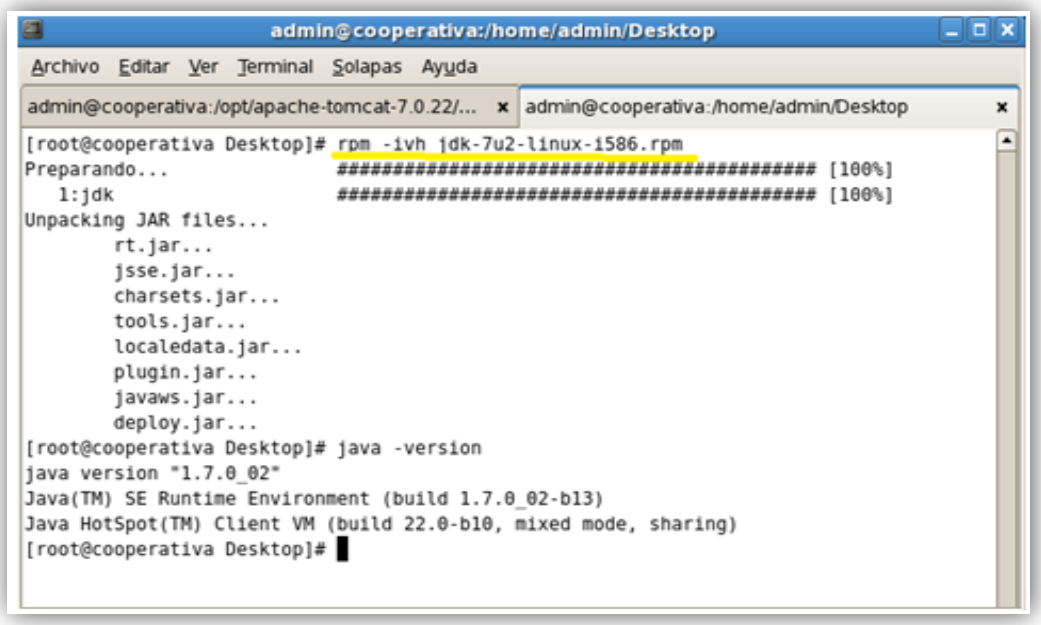


Figura 49: Instalación de la máquina virtual de Java.

Fuente: Propia

4.4.1.1.3. Configuración de JAVA_HOME

En el paso anterior, el jdk se instala en: /usr/java/jdk.1.7.0_02, el siguiente paso es configurar el JAVA_HOME asignando la ruta donde se instaló el jdk como se indica en la figura:

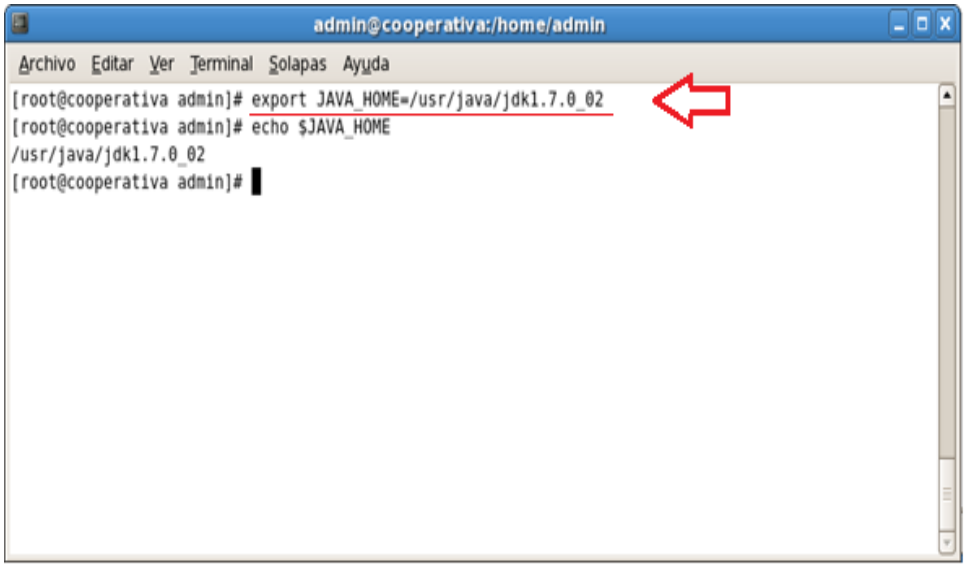


Figura 50: Java home

Fuente: Propia

Del mismo modo se procede a configurar el PATH de la siguiente forma:

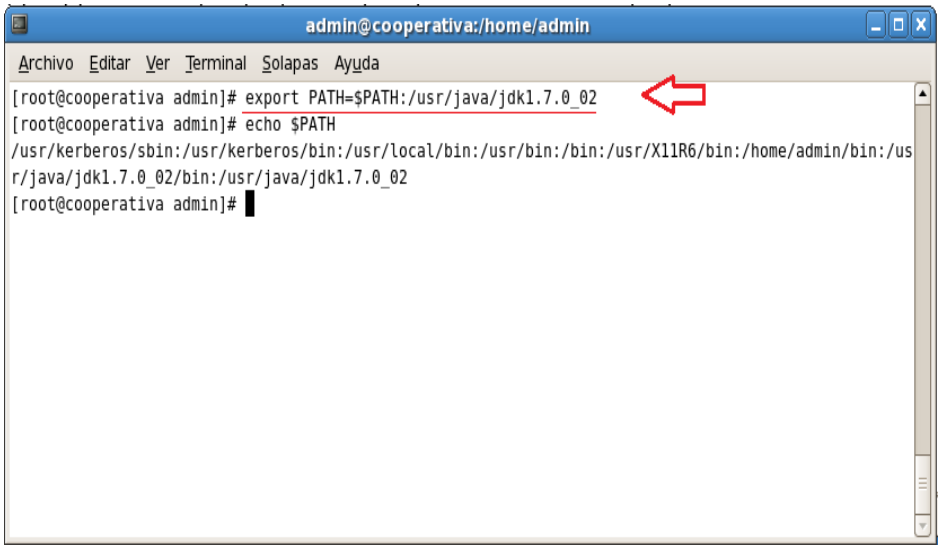


Figura 51: Configuración de java path

Fuente: Propia

4.4.1.1.4. Instalación de PostgreSQL

Desde la consola se debe digitar el siguiente comando: # ./ postgresql-9.1.2-1-linux.run

A continuación se carga un asistente gráfico de instalación, los pasos son los siguientes:

Figura 52: Asistente instalación de PostgreSQL - paso 1.

Fuente: Instalación propia

Figura 53: Asistente de instalación de PostgreSQL – paso 2.

Fuente: Instalación propia

Figura 54: Asistente de instalación de PostgreSQL – paso 3.

Fuente: Instalación propia

Figura 55: Asistente de instalación de PostgreSQL – paso 4.

Fuente: Instalación propia

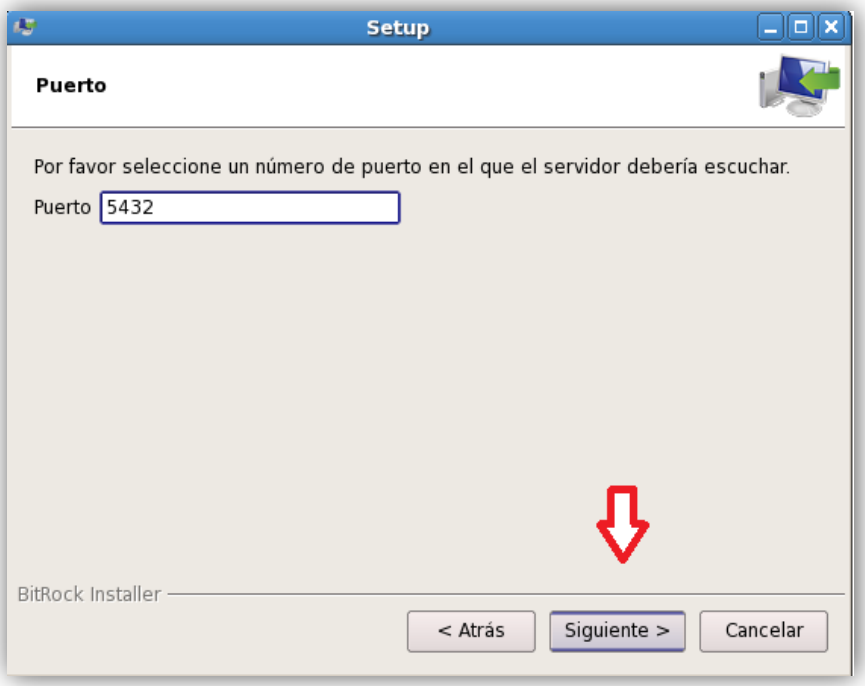


Figura 56: Asistente de instalación de PostgreSQL – paso 5.

Fuente: Instalación propia

Figura 57: Asistente de instalación de PostgreSQL – paso 6.

Fuente: Instalación propia

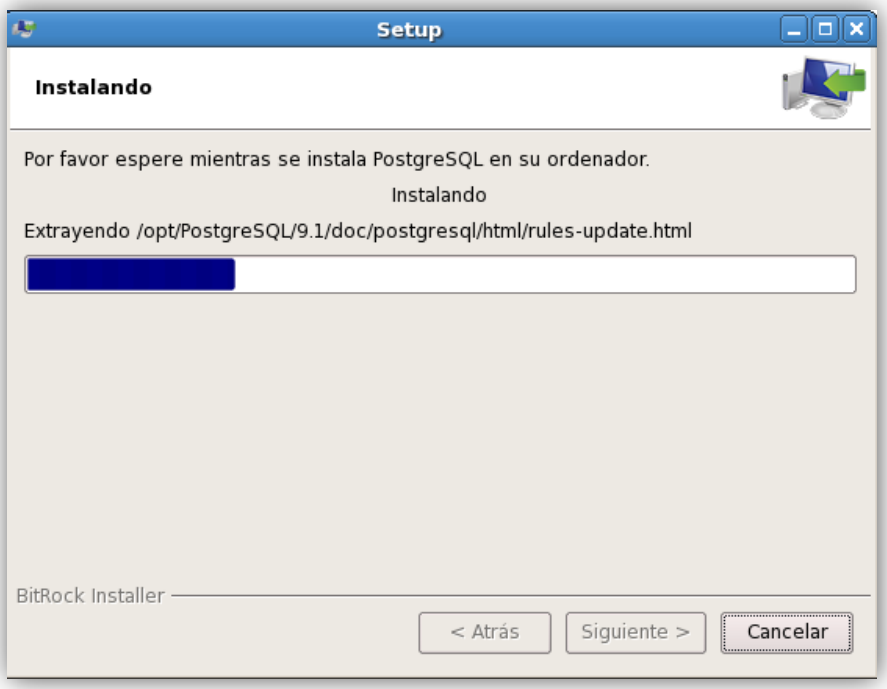


Figura 58: Asistente de instalación de PostgreSQL – paso 7.

Fuente: Instalación propia

Figura 59: Asistente de instalación de PostgreSQL – paso 8.

Fuente: Instalación propia

Una vez instalado la base de datos, procedemos a abrir PgAdmin III, para realizar acciones en el motor de base de datos.Se debe autenticar con el password proporcionado en el proceso de instalación. Dentro de PgAdmin III, dar click derecho en Login Roles y seleccionar New Login Role como muestra la imagen.

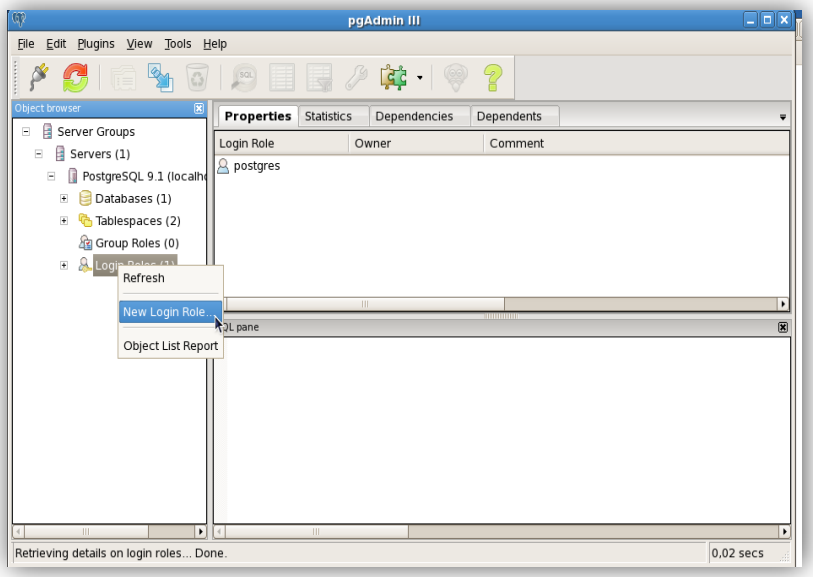


Figura 60: Creación de nuevo usuario.

Fuente: Instalación propia

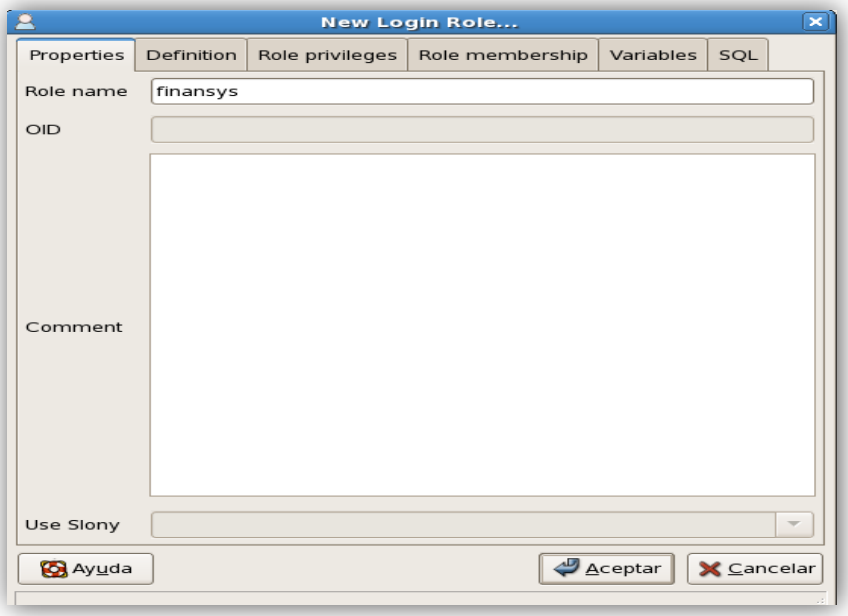


Figura 61: Creación del rol finansys.

Fuente: Instalación propia

Luego en la pestaña Definition asignar el password que tendrá dicho usuario.

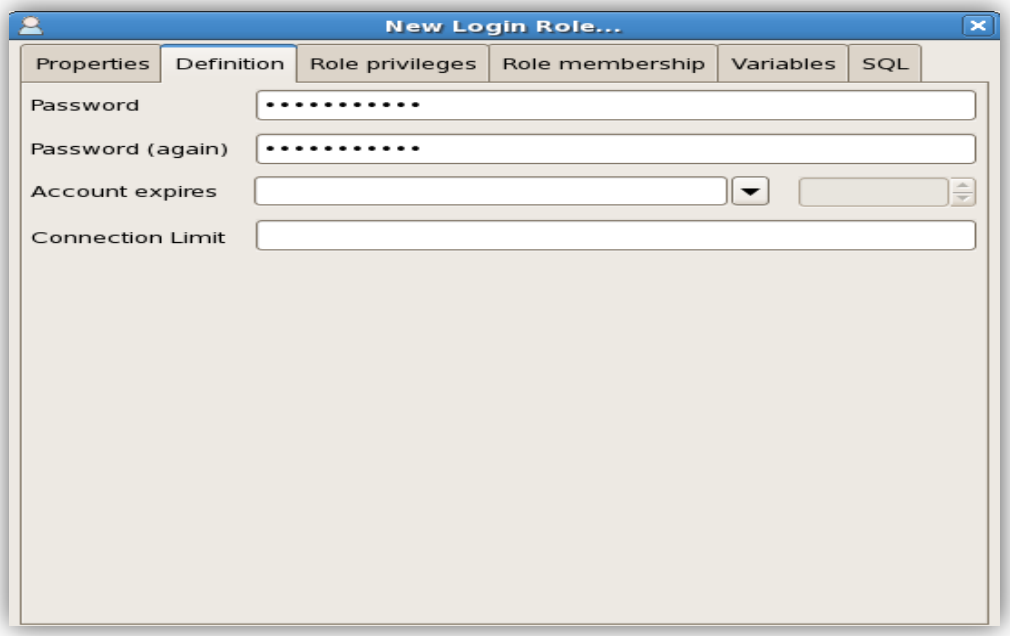


Figura 62: Asignar contraseña al rol.

Fuente: Instalación propia

De igual forma se procede a crear una nueva base de datos dando click derecho en la opción Databases y seleccionando la opción new database.

Crear la base de datos con el nombre finansys y elegir al usuario creado como se muestra en el gráfico.

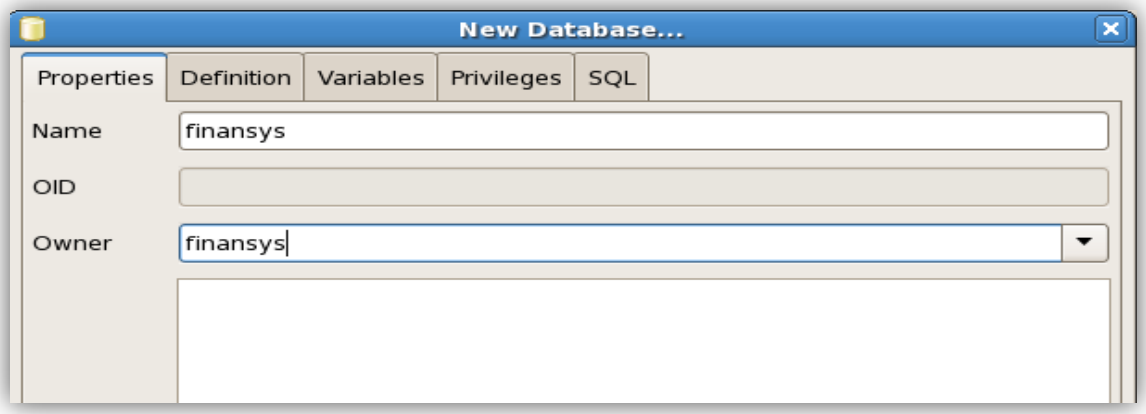


Figura 63: Creación de una nueva base de datos.

Fuente: Instalación propia

Por último se procede a cargar el backup, el mismo que contiene el script de las tablas y algunos datos básicos que necesita el sistema.

En la base de datos creada, dar click derecho y seleccionar la opción Restore.

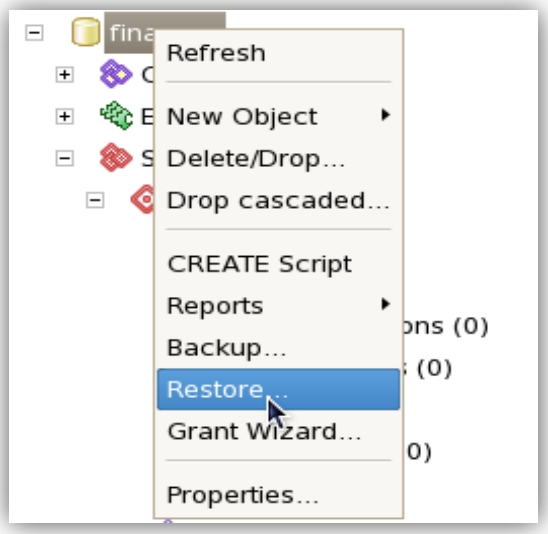


Figura 64: Subir backup a la base de datos.

Fuente: Instalación propia

Seleccionar el archivo con extensión .backup, asignarle el usuario creado y presionar el botón Restore.

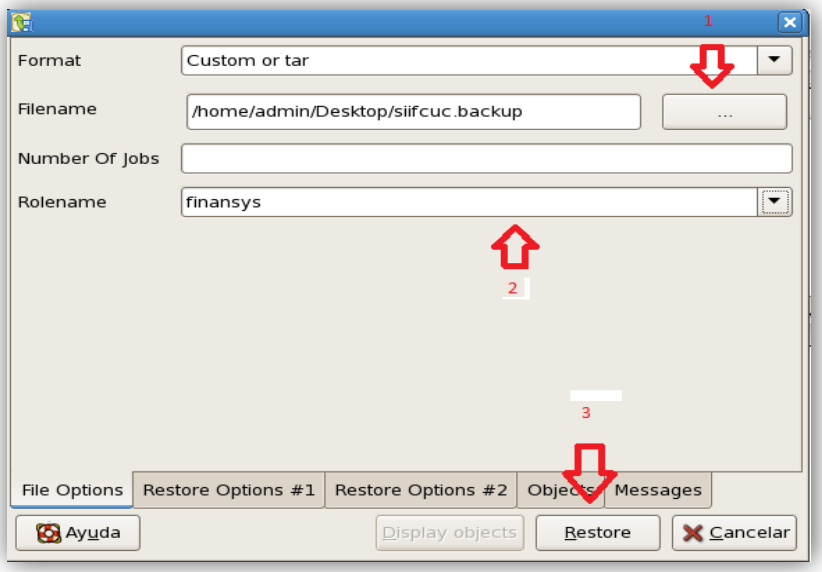


Figura 65: Subir backup a la base de datos.

Fuente: Instalación propia

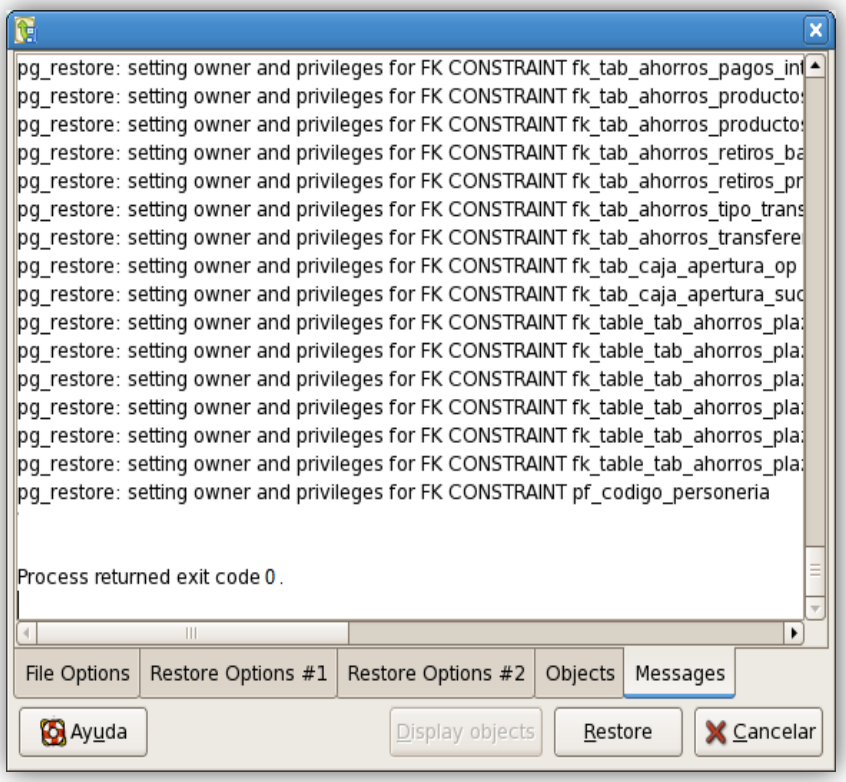


Figura 66: Subir backup a la base de datos.

Fuente: Instalación propia

4.4.1.1.5. Instalación y Configuración de Apache Tomcat.

En el escritorio, donde se copiaron todos los archivos, existe un archivo comprimido llamado apache-tomcat-7.0.22-x86.zip, se accede nuevamente a la consola que estaba abierta y ubicada en el directorio del escritorio, para descomprimir el archivo de digita el siguiente comando

unzip apache-tomcat-7.0.22-x86.zip

Al concluir la ejecución del comando anterior, se crea en el escritorio una carpeta llamada apache-tomcat-7.0.22,

mv apache-tomcat-7.0.22 /opt/

cd /opt/ apache-tomcat-7.0.22/bin

chmod 777 *

Hasta ahí ya está el apache tomcat listo; para iniciar el servicio hay que ejecutar el fichero startup.sh.

Para hacer que el servicio se inicie automáticamente al inicial el sistema operativo se debe seguir los siguientes pasos:

cd /etc/init.d/

vi apache

El script apache debe quedar de la siguiente forma:

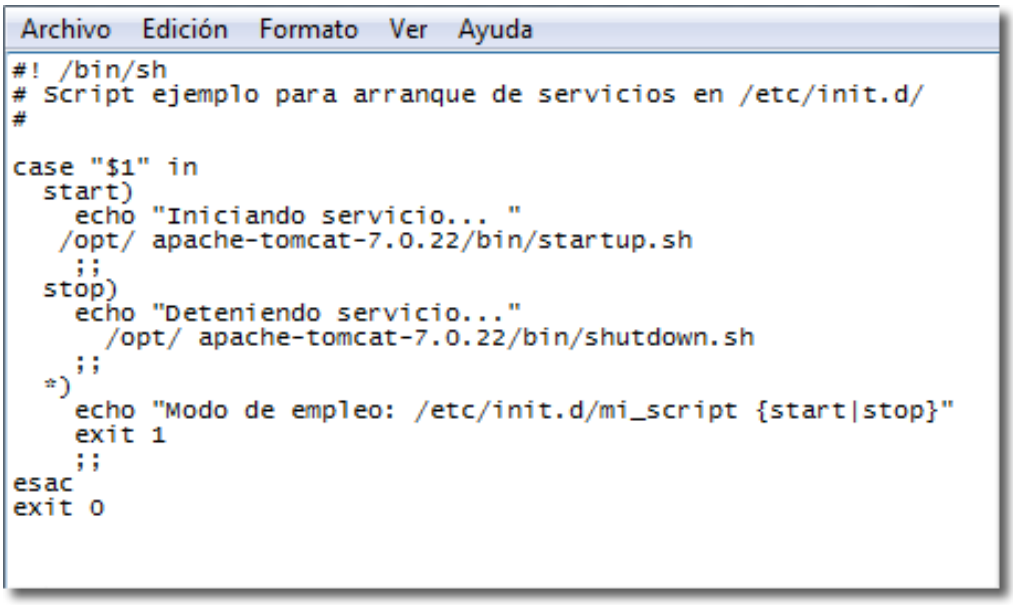


Figura 67: Código Instalación Apache

Fuente: Instalación propia

Posteriormente, hemos de crear un enlace simbólico en el runlevel correspondiente para que se ejecute cada vez que arranquemos la máquina

#ln -s /etc/init.d/apache /etc/rc3.d/S98apache

Y listo.
Reiniciar el servidor para verificar que el servicio si se inicia automáticamente al inicial el sistema.

Una vez iniciado el sistema, para verificar que el servidor Apache Tomcat este funcionado correctamente, en el path en un web browser digitar localhost y debe cargar la siguiente página:

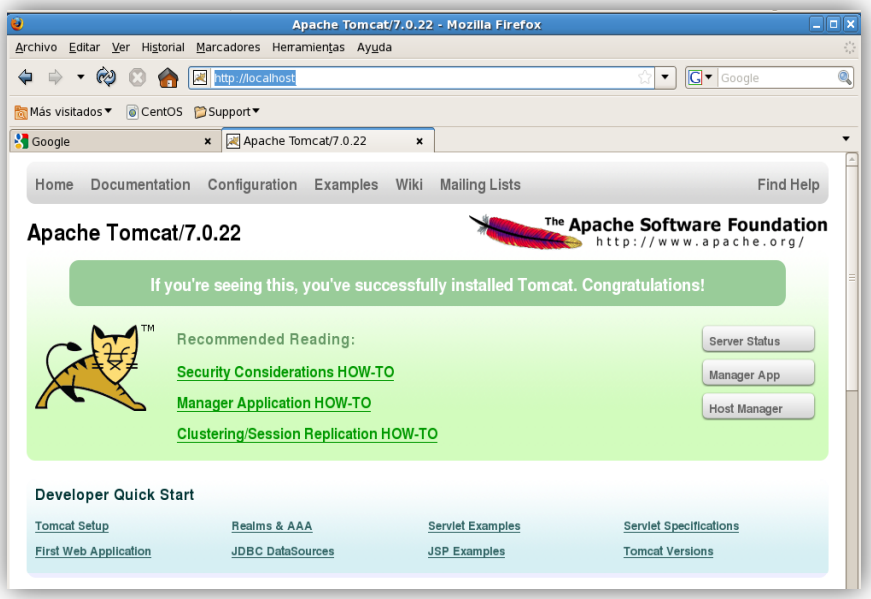


Figura 68: Página de administración de Apache Tomcat.

Fuente: Instalación propia

Por último el archivo con extensión .war ubicado en el escritorio, pegarlo en

/opt/apache-tomcat-7.0.22/webapp/ y listo.

Para verificar que la aplicación funcione correctamente en el navegador firefox digitar http://localhost/Finansys

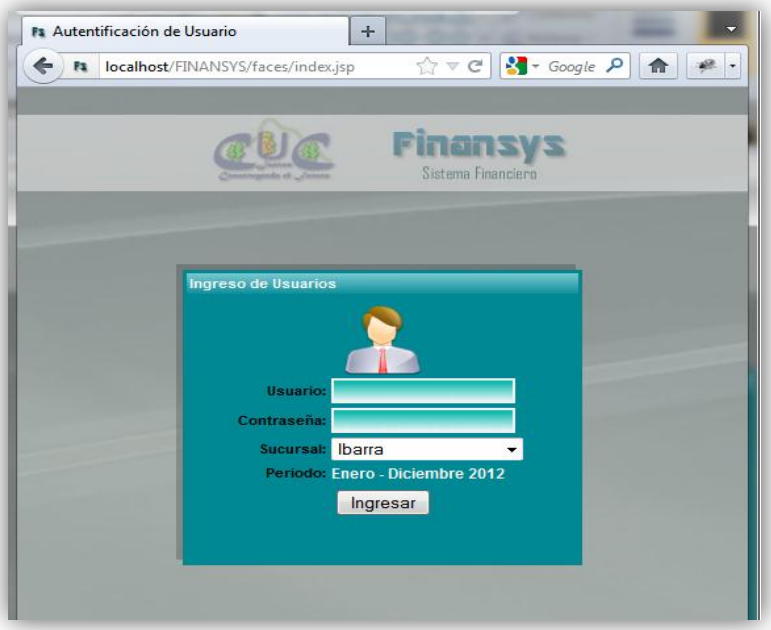


Figura 69: Página de inicio del sistema.

Fuente: Propia

4.4.1.1.6. Configuración del SSL para HTTPS

Después de configurar correctamente el JAVA_HOME, se procede a generar una clave con el algoritmo de encripación RSA, a dicha clave debemos asignarle un alias; Se debe configurar de la siguiente manera:

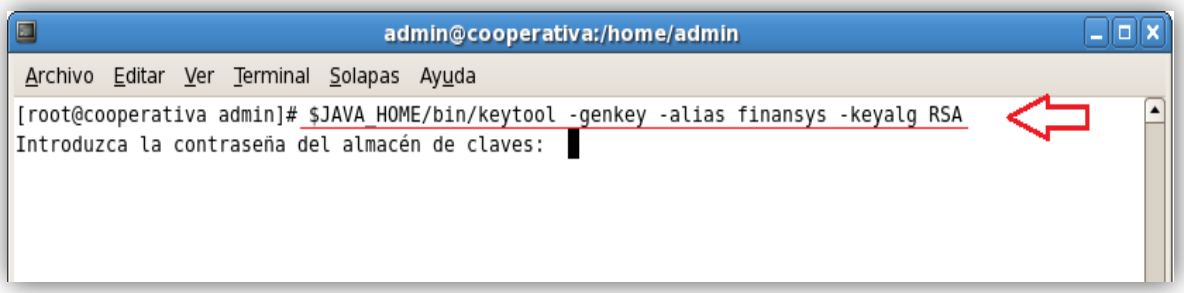


Figura 70: Generación de la clave.

Fuente: Propia

Luego se debe ingresar una contraseña, se debe completar unos datos que le va solicitando:

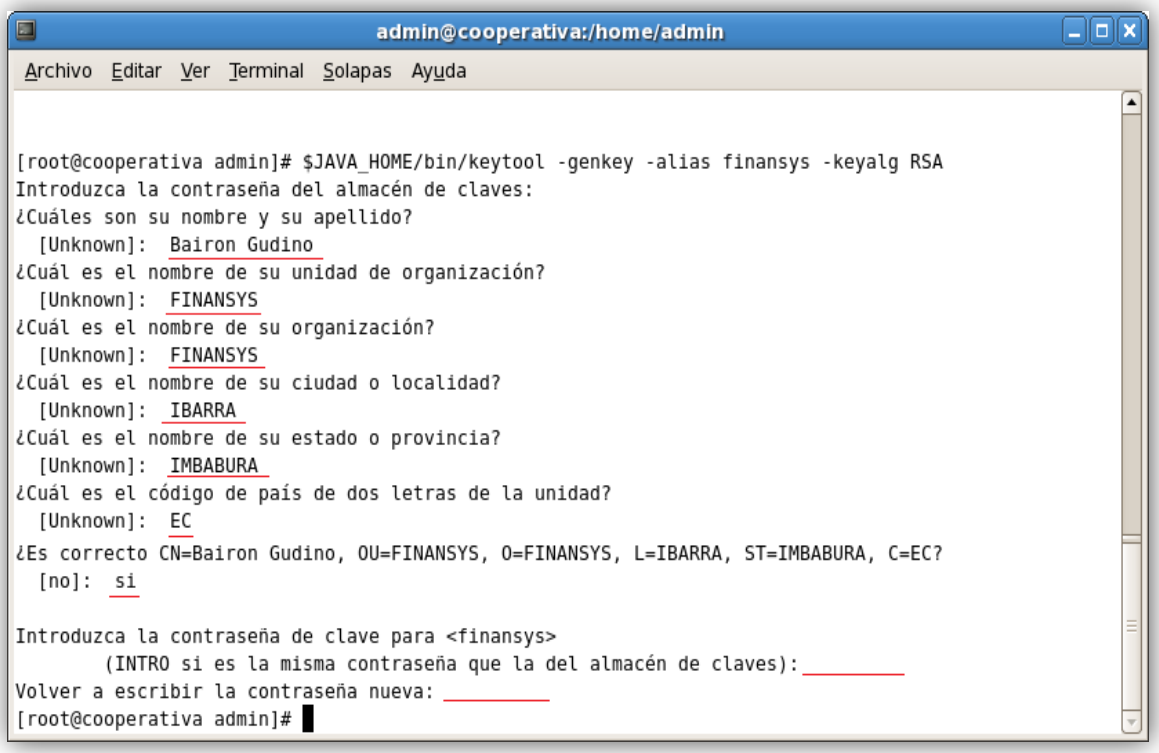


Figura 71: Parámetros de la clave.

Fuente: Propia

Para terminar la configuración del SSL, se procede a configurar el servidor Apache Tomcat. El Archivo server.xml ubicado en la carpeta /opt/apache-tomcat-7.0.22/conf, debe quedar de la siguiente forma:

Para verificar la configuración, reiniciar el servidor apache y en un navegador web acceder al siguiente link: https://localhost:8443/Finansys/

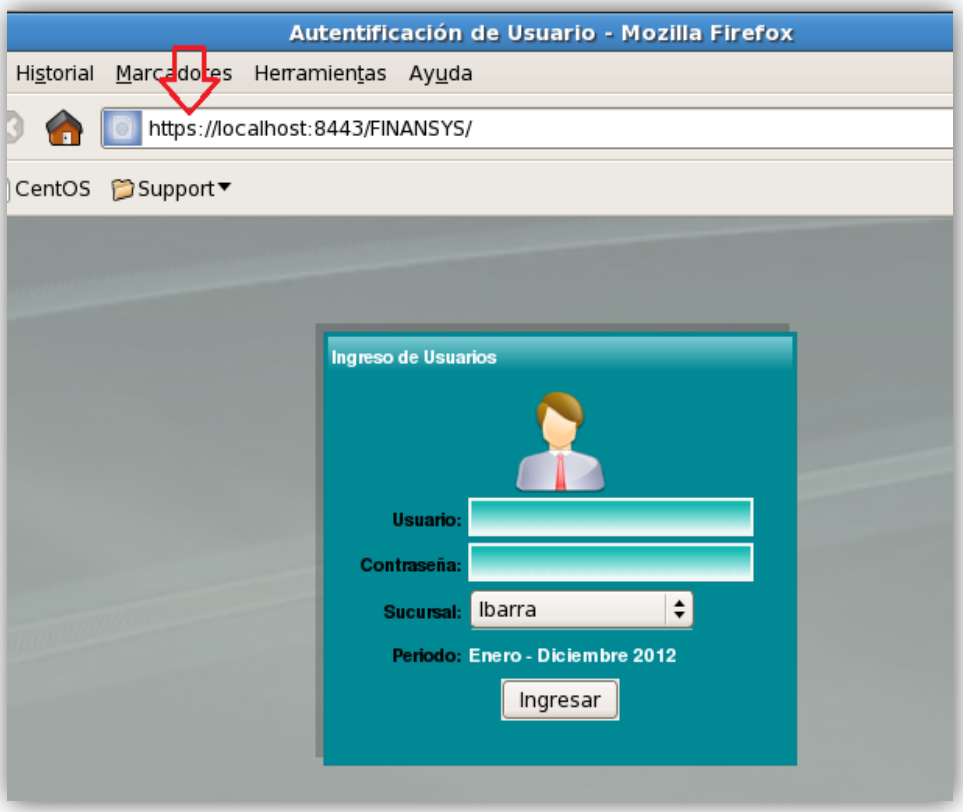


Figura 72: Página de inicio del sistema en modo HTTPS.

Fuente: Propia

Capítulo V: Conclusiones y Recomendaciones

CONTENIDO:

5. Conclusiones.
6. Recomendaciones.

5. Conclusiones y Recomendaciones

5.1. Conclusiones

1. La indagación y recopilación de requerimientos funcionales del sistema es trascendental ya que esto permitió la obtención de un módulo contable eficaz y con ello un sistema financiero de calidad.
2. Contar con una metodología para el desarrollo del software, se constituye como indispensable dado que aclara la perspectiva de lo deseado por el usuario.
3. La utilización de un lenguaje como java permite mantener un control completo del desarrollo de la aplicación.
4. El utilizar herramientas libres resulta benéfico para la institución, dado que no costea licencias y puede utilizar el software sin ninguna restricción.
5. En una entidad financiera como en este caso resulta conveniente y de gran beneficio utilizar una aplicación cliente-servidor, dado que todas las operaciones se mantienen centralizadas y todos acceden a un mismo recurso.
6. El fusionar diferentes estándares abiertos en el desarrollo de la aplicación permite crear aplicaciones más amigables con el usuario final.

5.2. Recomendaciones

1. Es aconsejable utilizar una metodología en este caso RUP en la recopilación de la información de los diversos procesos y el análisis de requerimientos ya que permiten un desarrollo óptimo y claro del software.
2. Para el desarrollo de un sistema integrado que conste de varios módulos, es necesario utilizar una herramienta de versionamiento ya que facilita el desarrollo centralizado.
3. Es necesario tomar muy en cuenta las disposiciones pedidas por los usuarios finales y demás involucrados en el uso del sistema, dado que ellos tienen la experiencia en el manejo de la información y demás procesos.
4. Utilizar buenas prácticas de programación permite construir aplicaciones de fácil mantenimiento y sobre todo escalables.
5. Capacitar al personal con el sistema desarrollado es muy importante ya que de esta manera el usuario final se familiariza con la funcionalidad del aplicativo y despeja cualquier duda.
6. Mantener la información clara a nivel de código y de base de datos resulta necesario, ya que en el desarrollo del sistema integrado, los demás módulos pueden verificar que contiene cada campo o método.
7. Documentar los errores obtenidos en el proceso de codificación resulta útil, ya que en posteriores casos se puede hacer uso de las sugerencias previas.
8. Es necesario tratar temas de costos de software, dado que en el entorno laboral surgen preguntas por parte de personas interesadas y no es fácil presentar un precio real.

6. Glosario

6.1. Introducción.

El presente glosario forma parte del sistema "Finansys" y es una recopilación de términos empleados en todo el sistema.
Este glosario no es un diccionario informático; representa una guía general de las definiciones y conceptos que se usan con frecuencia.

6.2. Palabras

6.2.1. Actions.- Métodos de capa de vista, para el uso de componentes en páginas jsp.

6.2.2. API.- **I**nterfaz de **P**rogramación de **A**plicaciones (**A**pplication **P**rogramming **I**nterface) es el conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

6.2.3. Artefacto.- Un artefacto es un producto tangible resultante del proceso de desarrollo de software. En ocasiones un artefacto puede referirse a un producto terminado, pero más habitualmente se refiere a la documentación generada a lo largo del desarrollo del producto.

6.2.4. Buteo Boot.- Es la secuencia de arranque, es el proceso que inicia el sistema operativo cuando el usuario enciende una computadora. Se encarga de la inicialización del sistema y de los dispositivos.

6.2.5. Casos de Uso.- Es una descripción de los pasos o las actividades que deberán realizarse para llevar a cabo algún proceso.

6.2.6. CSS.- El nombre hojas de estilo en cascada viene del inglés **C**ascading **S**tyle **S**heets.

6.2.7. CUC.- Cooperativa de Ahorro y Crédito "Unión Cochapamba".

6.2.8. Finansys.- Nombre del Sistema Integrado de Información Financiera.

6.2.9. Framework.- es un esquema (un esqueleto, un patrón) para el desarrollo y/o la implementación de una aplicación.

6.2.10. GNU/Linux.- GNU/Linux es uno de los términos empleados para referirse a la combinación del núcleo o kernel libre similar a Unix denominado Linux, que es usado con herramientas de sistema GNU.

6.2.11. HTML.- (Siglas de **H**yper **T**ext **M**arkup **L**anguage) Lenguaje de marcado de hipertexto.

6.2.12. Jdeveloper.- Es un entorno de desarrollo integrado desarrollado por Oracle.

6.2.13. JSF.- es una tecnología y framework para aplicaciones Java basadas en web que simplifica el desarrollo de interfaces de usuario en aplicaciones Java EE.

6.2.14. Live CD.- es un sistema operativo almacenado en un medio extraíble, que puede ejecutarse desde éste sin necesidad de instalarlo en el disco duro de una computadora.

6.2.15. On-the-fly.- Sobre marcha, en proceso

6.2.16. Open Source.- Código abierto, es el término con el que se conoce al software distribuido y desarrollado libremente.

6.2.17. ORM.- Es el **M**apeo **O**bjeto-**R**elacional (**O**bject-**R**elational **M**apping) es una técnica de programación para convertir datos entre el lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional, utilizando un motor de persistencia.

6.2.18. POO.- Programación **O**rientada a **O**bjetos.

6.2.19. RichFaces.- Framework de código abierto que añade la capacidad de ajax en aplicaciones JSF sin recurrir a JavaScript.

6.2.20. Rup.- El **P**roceso **U**nificado de **R**ational (**R***ational* **U***nified* **P***rocess*) es un proceso de desarrollo de software y junto con el UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

6.2.21. SElinux.- es una característica de seguridad de Linux que provee una variedad de políticas de seguridad.

6.2.22. Servlets.- Los servlets, son objetos que corren dentro y fuera del contexto de un contenedor de servlets (ej: Tomcat) y extienden su funcionalidad.

6.2.23. skin.- Mascara, interfaz visual de distintas características.

6.2.24. Skinnability.- Cambiar de skin.

6.2.25. SMB.- Server **M**essage **B**lock es un Protocolo de red que permite compartir archivos e impresoras entre nodos de una red.

6.2.26. SO.- Sistemas Operativos

6.2.27. Tag.- Etiqueta. Las etiquetas le dicen al browser las instrucciones y características necesarias para presentar la página en la pantalla.

6.2.28. UML.- Lenguaje **U**nificado de **M**odelado (**U***nified* **M***odeling* **L***anguage*). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema.

6.2.29. Visio.- Microsoft Visio es un software de dibujo vectorial para Microsoft Windows. Las herramientas que lo componen permiten realizar diagramas de oficinas, diagramas de bases de datos, diagramas de flujo de programas, UML, y más.

6.2.30. Vista de Deployment.- es un tipo de diagrama UML que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes.

6.2.31. YUM.- Es una colección de herramientas que sirven para automatizar el proceso de instalación, actualización, configuración y eliminación de paquetes de software.

6.2.32. W3C. - Consorcio World Wide Web.

7. Referencias bibliográficas

Libros:

1) Birnam Stewart. (2011). *Java Distribuido.* E.E.U.U: McGrawHill.

2) Francisco Rueda Profesor, U. d. (2010). *Sistemas Operativos.* México: McGraw Hill.

3) Ing. Maribel Sabana Mendoza. (2010). *Libro de postgreSQL.* Quito: Grupo Editorial Megabyte.

4) Coop Unión Cochapamba. (2011). *Estatuto de la Cooperativa "Unión Cochapamba.* Ibarra: N/A.

5) Coop. Unión Cochapamba. (2011). *Plan Operativo Anual.* Ibarra: N/A.

Web:

1) Centos. (01 de 05 de 2011). *Centos*. Recuperado el 02 de 01 de 2012, de http://www.centos.org/

2) Corporation Creative Commons. (31 de 08 de 2009). *Libros Web*. Recuperado el 2 de 03 de 2012, de http://www.librosweb.es/

3) Fedora Project. (2 de 09 de 2008). *Security Guide*. Recuperado el 28 de 05 de 2012, de http://docs.fedoraproject.org/esES/Fedora/16/html/Security_Guide/index.html

4) Jboss org. (07 de 05 de 2011). *Richfaces*. Recuperado el 10 de 07 de 2012, de http://docs.jboss.org/richfaces/latest_3_3_X/en/devguide/html/Introduction.html

5) Jsanroman. (05 de 05 de 2012). *Jasperreport*. Recuperado el 21 de 07 de 2012, de http://jsanroman.net/2007/11/C2BFque-es-jasper-reports-2/

6) Libros web. (01 de 01 de 2012). *librosweb.es*. Recuperado el 01 de 08 de 2012, de http://www.librosweb.es/

7) Masadelante. (02 de 05 de 2011). *Sistemas Operativos*. Recuperado el 03 de 04 de 2012, de http://www.masadelante.com/faqs/sistema-operativo

8) Ndeveloper. (01 de 04 de 2011). *Ndeveloper*. Recuperado el 03 de 06 de 2012, de http://www.ndeveloper.com/ndeveloperDocuments/documents/nDeveloper_JavaServerFaces.pdf

9) Normativa De Basilea. (2012). *Reformas Introducidas en el Sistema Financiero Ecuatoriano.* Quito: Norma.

10) OAS. (12 de 06 de 2011). *Ley General Institucion Financiera*. Recuperado el 14 de 03 de 2012, de http://www.oas.org/juridico/mla/sp/ecu/sp_ecu-mla-law-finance.html

11) Owasp org. (15 de 01 de 2006). *Owasp.org*. Recuperado el 02 de 04 de 2012, de www.owasp.org/documentation/topten

12) Postgresql. (12 de 04 de 2010). *postgresql.org*. Recuperado el 07 de 05 de 2012, de http://www.postgresql.org.pe/articles/introduccion_a_postgresql.pdf

13) Richard Monson-Haefel. (2011). *Enterprise JavaBeans 2da Edición.* E.E.U.U: O' REILLY.

14) Rincon. (01 de 03 de 2011). *Concepto de Base de Datos*. Recuperado el 06 de 05 de 2012, de http://html.rincondelvago.com/concepto-de-base-de-datos.html

15) sbs.gob.ec. (12 de 29 de 2009). *Decreto Cooperativas*. Recuperado el 01 de 02 de 2012, de http://www.sbs.gob.ec/medios/PORTALDOCS/downloads/normativa/decreto_194_cooperativas_29_dic_09.pdf

16) The World Wide Web Consortium, W. (23 de Abril de 2003). *The World Wide Web Consortium (W3C)*. Recuperado el 03 de 02 de 2012, de http://www.w3.org/

17) Tom Negrino, D. S. (2011). *Javascript.* Estados Unidos : Tontitown.

18) Universidad de Sevilla. (12 de 04 de 2011). *LSI*. Recuperado el 02 de 03 de 2012, de http://www.lsi.us.es/~quivir

19) Wikipedia. (17 de 12 de 1996). *es.wikipedia.org*. Recuperado el 03 de 03 de 2012, de http://es.wikipedia.org/wiki/Hojas_de_estilo_en_cascada

20) Wikipedia. (08 de 04 de 2007). *es.wikipedia.org*. Recuperado el 01 de 03 de 2012, de http://es.wikipedia.org/wiki/HTML

21) Wikipedia. (20 de 04 de 2011). *es.wikipedia.org*. Recuperado el 07 de 04 de 2012, de http://es.wikipedia.org/wiki/Fedora_(distribución_Linux)

22) wikipedia Java. (03 de 07 de 2010). *Java.* Recuperado el 01 de 05 de 2012, de http://es.wikipedia.org/wiki/Plataforma_Java

23) wikipedia NetBeans. (01 de 04 de 2010). *NetBeans*. Recuperado el 29 de 05 de 2012, de http://es.wikipedia.org/wiki/NetBeans

24) Wikipedia, T. (03 de 05 de 2011). Recuperado el 23 de 05 de 2012, de http://es.wikipedia.org/wiki/Tomcat

8. Anexos

8.1. Diagrama entidad relación Finansys

8.2. Casos Modelos de Casos de Uso Finansys

Los siguientes archivos de anexos se los encontrará en el cd del proyecto:

8.3. Manual de usuario módulo contable.

8.4. Manual técnico módulo contable.

8.5. Entregables de RUP.

8.5.1. Actas de trabajo módulo contable.

8.5.2. Casos de Uso módulo contable.