

UNIVERSIDAD DE ORIENTE
NÚCLEO DE ANZOÁTEGUI
ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS
DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS



**DESARROLLO DE UN SISTEMA AUTOMATIZADO PARA
FACTURACIÓN Y RECOBRO DE SERVICIOS TELEFÓNICOS
ASOCIADO A LA GERENCIA AIT, PDVSA MATURÍN, BASADO
EN TECNOLOGÍA WEB Y ESTÁNDARES ABIERTOS.**

PRESENTADO POR:
Br. Douglas Rafael Gene Pereira.

Trabajo de Grado presentado ante la Universidad de Oriente
como requisito parcial para optar al título de
INGENIERO EN COMPUTACIÓN

Puerto La Cruz, Agosto de 2010

UNIVERSIDAD DE ORIENTE
NÚCLEO DE ANZOÁTEGUI
ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS
DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS



**DESARROLLO DE UN SISTEMA AUTOMATIZADO PARA
FACTURACIÓN Y RECOBRO DE SERVICIOS TELEFÓNICOS
ASOCIADO A LA GERENCIA AIT, PDVSA MATURÍN, BASADO
EN TECNOLOGÍA WEB Y ESTÁNDARES ABIERTOS.**

Tutora:

Ing. Gabriela Veracierta
Tutora Académica

Puerto La Cruz, Agosto de 2010

UNIVERSIDAD DE ORIENTE
NÚCLEO DE ANZOÁTEGUI
ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS
DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS



**DESARROLLO DE UN SISTEMA AUTOMATIZADO PARA
FACTURACIÓN Y RECOBRO DE SERVICIOS TELEFÓNICOS
ASOCIADO A LA GERENCIA AIT, PDVSA MATURÍN, BASADO
EN TECNOLOGÍA WEB Y ESTÁNDARES ABIERTOS.**

Jurado Calificador:

Ing. José Luis Bastardo

Jurado Principal

Ing. Claudio Cortínez

Jurado Principal

Ing. Gabriela Veracierta

Tutora Académica

Puerto La Cruz, Agosto de 2010

RESOLUCIÓN

De acuerdo al Artículo número 41 del Reglamento de Trabajos de Grado:

“Los Trabajos de Grado son de la exclusiva propiedad de la Universidad de Oriente, y sólo podrán ser utilizados para otros fines con el consentimiento del Consejo de Núcleo respectivo, quien deberá participarlo previamente al Consejo Universitario, para su autorización”.

DEDICATORIA

A Dios por la oportunidad que me da para desarrollarme como persona y profesional.

A mis padres Bernardina Pereira y William Gene, que siempre lucharon por darnos la posibilidad de ser mejores personas, y a pesar de todo lo sufrido, siempre conte con ese gran apoyo por parte de la familia.

A mis hermanos Wendy Gene y William Gene por ser parte de mi vida y por todo lo que han hecho por mí, en especial a ti hermanita de la cual siempre estaré agradecido por ayudarme, tu ayuda fue vital para lograr terminar esta meta.

A mis dos hermosos angelitos mis hijas Rachel Gene y Richelle Gene las amo con todo mi corazón.

Para mis abuelos que se que al lado de Dios siempre están cuidándome.

AGRADECIMIENTO

A Dios por haberme obsequiado una hermosa familia, el don de la salud y la energía necesaria para siempre tratar de alcanzar mis metas.

A mis padres porque sin ellos no hubiese logrado esta importante meta.

A mis dos hermosas niñas que por ellas y para ellas debía de terminar esta etapa de mi vida.

A mi familia por apoyarme siempre en las buenas y en las malas, en especial a mis abuelos que se, desde donde quiera que estén, siempre han estado protegiéndome y guiándome para que no pierda mi rumbo.

A la madre de mis dos pequeñas niñas, por todo el apoyo en su momento, fue parte de todo este proyecto, gracias por todo, de corazón espero que Dios te llene de felicidad.

A la Universidad de Oriente, “La casa más Alta”, por haberme dado la oportunidad de estudiar y de ser mi segundo hogar durante estos años, siempre recordare que *del pueblo venimos y hacia el pueblo vamos*.

A mis profesores quienes tuvieron la tarea de impartir tantos conocimientos en mí, guiándome en mi aprendizaje, les estaré eternamente agradecido.

A todos mis amigos de la UDO (Biel-Piero, Daniel, Francisco, David, Jaisfel, Samuel, Amable, por mencionar varios) con quienes compartí muy buenos momentos, muchachos adelante y a trabajar por nuestro país.



A mis amigos de AIT por brindarme su confianza y hacerme parte del equipo y ayudare siempre que lo necesite.

A las grandiosas personas con las que compartí en Maturín mientras realizaba este proyecto (Fabricio, Yenireth, Ronnie, Leidy, Anibal). Especialmente a Fabricio Bravo que se convirtió en unos de mis mejores amigos, gracias por todo.

A mis asesores Ings. Gabriela Veracierta y Yulimar Salazar quienes fueron mis guías para la elaboración de este humilde trabajo que hoy les presento, gracias totales.

RESUMEN

El proceso de recobro de facturación telefónica que lleva a cabo la gerencia AIT (Automatización, Informática y Telecomunicaciones) del Distrito Norte, consiste en realizar el recobro a las diferentes Gerencias que forman parte del Distrito, pero este proceso actualmente se lleva a cabo de forma manual, lo cual genera contratiempos y retrasos en el recobro a las distintas gerencias del Distrito. Esta acción es fundamental para controlar el presupuesto de la Gerencia por medio del reembolso de los servicios prestados, razón por la cual la gerencia de AIT específicamente el proceso de Cadena de Suministros, se vio en la necesidad de solicitar el desarrollo de un software para la automatización del proceso de recobro de facturación telefónica, para obtener beneficios de seguridad de la información, disminución de esfuerzo y tiempo. El desarrollo del proyecto estuvo basado en el Proceso Unificado de Desarrollo de Software, la Técnica UML y su extensión WEBML, para la implementación se utilizaron las herramientas de PHP y MYSQL cumpliendo con las normas de la empresa, en cuanto al desarrollo de aplicaciones con software libre de acuerdo al decreto 3.390.

ÍNDICE GENERAL

RESOLUCIÓN	iv
DEDICATORIA	v
AGRADECIMIENTO	vi
RESUMEN	viii
ÍNDICE GENERAL	ix
ÍNDICE DE FIGURAS	xiv
ÍNDICE DE TABLAS	xvii
CAPÍTULO I INTRODUCCIÓN	19
1.1 Planteamiento del Problema.....	19
1.2 Objetivos.....	22
1.2.1 Objetivo General.....	22
1.2.2 Objetivos Específicos.....	22
CAPÍTULO II MARCO TEÓRICO	24
2.1 Antecedentes.....	24
2.2 Marco Teórico.....	26
2.2.1 Software.....	26
2.2.2 Objetivos de la ingeniería de software.....	26
2.2.3 Base de Datos.....	27
2.2.3.1 Arquitectura de un Sistema de Base de Datos.....	27
2.2.3.2 Sistema de gestión de base de datos.....	28
2.2.3.3 Objetivos de los SGBD.....	29
2.2.3.4 Diseño de Bases de Datos.....	30
2.2.3.5 Modelos de Bases de Datos.....	31
2.2.3.6 Modelos utilizados con frecuencia en las Bases de Datos.....	32
2.2.3.7 Requerimientos de las bases de datos.....	33
2.2.4 Navegador web.....	33



2.2.5 Internet	34
2.2.6 Sistema web	35
2.2.7 Interfaz web.....	35
2.2.8 Estructura web.....	35
2.2.9 Lenguajes de programación	36
2.2.9.1 PHP	37
2.2.9.2 JavaScript	38
2.2.9.3 Frameworks.....	38
2.2.9.4 Ext-js (Extend JS)	39
2.2.9.5 Ajax.....	41
2.2.9.6 Protocolo http.....	42
2.2.10 MySQL.....	44
2.2.11 Proceso Unificado de Desarrollo de Software	44
2.2.11.1 Fase de Inicio	46
2.2.11.2 Fase de Elaboración	48
2.2.11.3 Fase de Construcción	49
2.2.11.4 Fase de Transición	49
2.2.12 Modelados de Páginas Web	50
2.2.12.1 Lenguaje Unificado De Modelado (UML)	51
2.2.12.2 WebML (Lenguaje de Modelado Web)	56
2.2.12.3 WebML Vs. UML.....	66
2.2.13 Servidor Web Apache	70
2.2.14 Software Libre (SL)	70
CAPÍTULO III FASE DE INICIO.....	72
3.1 Introducción	72
3.2 Flujo de trabajo requisito	73
3.2.1 Comprender el contexto del sistema	73
3.2.1.1 Modelo de dominio	74
3.2.2 Colección de requisitos	77



3.2.2.1 Alcance.....	78
3.2.2.2 Posicionamiento	78
3.2.3 Riesgos del sistema	80
3.2.4 Identificación de usuarios (actores del sistema)	85
3.2.5 Modelo de casos de uso	86
3.2.6 Descripción de casos de uso del sistema SAFREST.....	87
3.2.6.1 Caso de uso: Acceder al sistema	87
3.2.6.2 Caso de uso: Generar reporte	87
3.2.6.3 Caso de uso: Consultar datos de teléfono	87
3.2.6.4 Caso de uso: Consultar datos de teléfono	88
3.2.6.5 Caso de uso: Consultar teléfono por montos	89
3.2.6.6 Caso de uso: Modificar indicador	89
3.2.6.7 Caso de uso: Consultar teléfono por indicador	89
3.2.6.8 Caso de uso: Gestionar teléfonos	90
3.2.6.9 Caso de uso: Cargar datos de teléfono	90
3.2.6.10 Caso de uso: Actualizar datos de teléfono	90
3.2.6.11 Caso de uso: Gestionar roles	90
3.2.6.12 Caso de uso: Agregar datos gerenciales.....	91
3.2.7 Requisitos de dispositivos específicos	91
3.2.8 Requisitos no funcionales	91
3.3 Flujo de trabajo análisis	92
3.3.1 Análisis de requerimientos y/o requisitos	93
3.3.2 Especificación de los grupos de usuarios formalmente descritos	93
3.3.3 Especificación de los casos de usos por usuarios	93
3.3.3.1 Actor Consultor.....	94
3.3.4 Diagramas de análisis.....	95
3.3.4.1 Diagrama de análisis para el caso de uso Acceder.....	96
3.3.4.2 Diagrama de análisis para el caso de uso Cargar datos de teléfono....	96
3.3.5 Diagramas de colaboración.....	97



3.3.5.1 Diagrama de colaboración para el caso de uso Acceder	98
3.3.5.2 Diagrama de colaboración para el caso de uso Cargar datos de teléfono	99
3.4 flujo de trabajo diseño	100
3.4.1 Arquitectura candidata	100
3.4.2 Interfaz de inicio de sesión.....	102
CAPÍTULO IV FASE DE ELABORACIÓN	104
4.1 Introducción	104
4.2 Flujo de trabajo análisis	105
4.2.1 Especificación de los casos de usos por usuarios	105
4.2.1.1 Actor Administrador	106
4.2.2 Diagrama de clase de análisis	106
4.2.2.1 Diagrama de análisis para el caso de uso Consultar datos de teléfono	107
4.2.2.2 Diagrama de análisis para el caso de uso Actualizar datos de teléfono	108
4.2.2.3 Diagrama de análisis para el caso de uso Gestionar Roles	109
4.2.2.4 Diagrama de análisis para el caso de uso Agregar datos gerenciales	109
4.2.3 Diagrama de colaboración	110
4.2.3.1 Diagrama de colaboración para el caso de uso Consultar datos de teléfono.....	111
4.2.3.2 Diagrama de colaboración para el caso de uso Actualizar datos de teléfono.....	112
4.2.3.3 Diagrama de colaboración para el caso de uso Asignar Roles.....	113
4.2.3.4 Diagrama de colaboración para el caso de uso Agregar datos gerenciales.....	114
4.2.4 Análisis de la arquitectura.....	115
4.2.5 Descripción de paquetes del sistema SAFREST.....	119
4.3 Flujo de trabajo diseño	119



4.3.1	Diseño de la arquitectura.....	120
4.3.2	Diseño de la base de datos	121
4.3.2.1	Modelo de datos	122
4.3.2.2	Descripción de la Base de Datos.....	122
4.3.3	Modelo de Hipertexto	139
4.3.4	Modelo de gestión de contenidos.....	141
4.3.5	Modelo de personalización	143
4.4	Implementación.....	144
4.4.1	Identificación de componentes de la arquitectura.....	145
4.4.2	Implementación de la arquitectura.....	145
CAPÍTULO V FASE DE CONSTRUCCIÓN		158
5.1	Introducción	158
5.2	Flujo de trabajo implementación.....	158
5.2.1	Página principal del sistema SAFREST.....	159
5.2.2	Área de cargos de teléfonos	184
5.2.3	Área de Resumen de Cargos	195
5.2.4	Área de Gestionar Teléfonos.....	200
5.3	Flujo de trabajo Pruebas.....	220
5.3.1	Modelos de pruebas	221
5.3.2	Casos de pruebas	221
5.3.3	Pruebas de integración	224
CAPÍTULO VI FASE DE TRANSICIÓN.....		225
6.1	Introducción	225
6.2	Manual de Usuario	226
CONCLUSIONES.....		227
RECOMENDACIONES.....		229
BIBLIOGRAFÍA.....		230
METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:.....		234

ÍNDICE DE FIGURAS

Figura 2.1: ExtJS 1.0.1.a y las bases de la relaciones de la librería.....	41
Figura 2.2: ExtJS 1.1 y la base de relaciones de la librería.....	41
Figura 2.3: Tecnologías agrupadas bajo el concepto de AJAX.	43
Figura 2.4: Fases del Proceso Unificado de Desarrollo del Software.....	47
Figura 2.5: Enlace de los casos de uso con los flujos de trabajo.	48
Figura 2.6: Logo de UML.	51
Figura 2.7: Relación de dependencia entre dos clases.	53
Figura 2.8: Representación de una relación de asociación.	53
Figura 2.9: Relación de generalización.....	53
Figura 2.10: Representación de un diagrama de caso de uso.....	54
Figura 2.11: Representación de un diagrama de secuencia.....	55
Figura 2.12: Representación de un diagrama de clases.....	56
Figura 2.13: Sitio Web haciendo uso de los conceptos WebML.	61
Figura 3.1: Modelo de Dominio.....	76
Figura 3.2: Diagrama de casos de uso general del sistema.	88
Figura 3.3: Especificación de grupos formalmente descritos.	94
Figura 3.4: Usuario Consultor.....	94
Figura 3.5: Diagrama de análisis para el caso de uso Acceder.	96
Figura 3.6: Diagrama de análisis para el caso de uso Cargar datos de teléfono.	97
Figura 3.7: Diagrama de colaboración para el caso de uso Acceder.	98
Figura 3.8: Diagrama de colaboración para el caso de uso Cargar datos de teléfono.	99
Figura 3.9: Arquitectura Candidata del Sistema.	101
Figura 3.10: Niveles de Configuración de Red.....	102
Figura 3.11: Prototipo de interfaz de inicio de sesión.....	102
Figura 4.1: Actor Administrador.....	106



Figura 4.2: Diagrama de análisis para el caso de uso Consultar datos de teléfono...	107
Figura 4.3: Diagrama de análisis para el caso de uso Actualizar datos de teléfono.	108
Figura 4.4: Diagrama de análisis para el caso de uso Asignar Roles.....	109
Figura 4.5: Diagrama de análisis para el caso de uso Agregar datos gerenciales.	110
Figura 4.6: Diagrama de colaboración para el caso de uso Consultar datos de teléfono.	111
Figura 4.7: Diagrama de colaboración para el caso de uso Actualizar datos de teléfono.....	112
Figura 4.8: Diagrama de colaboración para el caso de uso Asignar Roles.	113
Figura 4.9: Diagrama de colaboración para el caso de uso Agregar datos gerenciales.	114
Figura 4.10: Vista del sitio para actor Consultor.	117
Figura 4.11: Vista del sitio para actor Administrador.....	117
Figura 4.12: Vista detallada del sitio para el actor Administrador.	118
Figura 4.13: Arquitectura del sistema SAFREST.....	121
Figura 4.14: Relación de tablas sistema SAFREST.....	123
Figura 4.15: Imagen de la tabla “tr001_divisiones”.....	124
Figura 4.16: Imagen de la tabla “tr002_distritos”.....	125
Figura 4.17: Imagen de la tabla “tr003_organizaciones”.....	125
Figura 4.18: Imagen de la tabla “tr004_telefonos”.	126
Figura 4.19: Imagen de la tabla “tr005_movilgeneral”.....	127
Figura 4.20: Imagen de la tabla “tr006_detallemovil”.....	128
Figura 4.21: Imagen de la tabla “tr007_usuarios”.	129
Figura 4.22: Imagen de la tabla “tr008_operadoras”.	129
Figura 4.23: Imagen de la tabla “tr009_tipos_servicios”.....	130
Figura 4.24: Imagen de la tabla “tr010_negocio”.	130
Figura 4.25: Imagen de la tabla “tr011_area”.	131
Figura 4.26: Imagen de la tabla “tr012_tipo_organizacion”.....	132
Figura 4.27: Imagen de la tabla “tr013_rela_dist_orga”.....	132



Figura 4.28: Imagen de la tabla “tr014_rela_divi_nego”.....	133
Figura 4.29: Imagen de la tabla “tr015_rol”.	133
Figura 4.30: Imagen de la tabla “tr016_usuario_rol”.....	134
Figura 4.31: Imagen de tabla “tr017_mes_monto”.....	135
Figura 4.32: Imagen de la tabla “tr018_cuenta_general”.....	135
Figura 4.33: Imagen de la tabla “tr019_rela_divi_distri”.	136
Figura 4.34: Imagen de la tabla “tr020_rela_dist_area”.	137
Figura 4.35: Imagen de la tabla “tr021_eventos”.....	137
Figura 4.36: Imagen de la tabla “tr022_historial_eventos”.	138
Figura 4.37: Imagen de la tabla “tr023_responsables”.	139
Figura 4.38: Modelo de Hipertexto de la Aplicación “SAFREST”.....	140
Figura 4.39: Modelo de Gestión de Contenidos de la Aplicación “SAFREST”.....	142
Figura 4.40: Símbolos para la personalización del sistema.	144
Figura 4.41: Autenticación de Usuarios del SAFREST.	144
Figura 4.42: Finalización de Sesión.....	145
Figura 4.43: Diagrama de Despliegue para el sistema SAFREST.....	146
Figura 4.44: Pantalla de inicio de sesión Sistema SAFREST.....	146
Figura 4.45: Página index.html usuario no válido.	156
Figura 5.1: Pantalla Principal de la aplicación SAFREST.....	159
Figura 5.2: Pantalla de Cargos de Teléfonos.	185
Figura 5.3: Pantalla Resumen de Cargos.	196
Figura 5.4: Pantalla Listar Teléfonos.....	201

ÍNDICE DE TABLAS

Tabla 2.1: Elementos del Modelo de Hipertexto WebML (1/5).	62
Tabla 2.1: Elementos del Modelo de Hipertexto WebML (2/5).	63
Tabla 2.1: Elementos del Modelo de Hipertexto WebML (3/5).	64
Tabla 2.1: Elementos del Modelo de Hipertexto WebML (4/5).	65
Tabla 2.1: Elementos del Modelo de Hipertexto WebML (5/5).	66
Tabla 2.2: Desarrollo Web Vs. Desarrollo Tradicional (1/2).	69
Tabla 2.2: Desarrollo Web Vs. Desarrollo Tradicional (2/2).	70
Tabla 3.1: Riesgos del Sistema (1/3).	82
Tabla 3.1: Riesgos del Sistema (2/3).	83
Tabla 3.1: Riesgos del Sistema (3/3).	84
Tabla 4.1: Descripción de la tabla “tr001_divisiones”.	124
Tabla 4.2: Descripción de la tabla “tr002_distritos”.	124
Tabla 4.3: Descripción de la tabla “tr003_organizaciones”.	125
Tabla 4.4: Descripción de la tabla “tr004_telefonos”.	126
Tabla 4.5: Descripción de la tabla “tr005_movilgeneral”.	127
Tabla 4.6: Descripción de la tabla “tr006_detallemovil” (1/2).	127
Tabla 4.6: Descripción de la tabla “tr006_detallemovil” (2/2).	128
Tabla 4.7: Descripción de la tabla “tr007_usuarios”.	128
Tabla 4.8: Descripción de la tabla “tr008_operadoras”.	129
Tabla 4.9: Descripción de la tabla “tr009_tipos_servicios”.	130
Tabla 4.10: Descripción de la tabla “tr010_negocio”.	130
Tabla 4.11: Descripción de la tabla “tr011_area”.	131
Tabla 4.12: Descripción de la tabla “tr012_tipo_organizacion”.	131
Tabla 4.13: Descripción de la tabla “tr013_rela_dist_orga”.	132
Tabla 4.14: Descripción de la tabla “tr014_rela_divi_nego”.	132



Tabla 4.15: Descripción de la tabla “tr015_rol”.....	133
Tabla 4.16: Descripción de la tabla “tr016_usuario_rol”.....	134
Tabla 4.17: Descripción de la tabla “tr017_mes_monto”.....	134
Tabla 4.19: Descripción de la tabla “tr019_rela_divi_distri”	136
Tabla 4.20: Descripción de la tabla “tr020_rela_dist_area”.....	136
Tabla 4.21: Descripción de la tabla “tr021_eventos”	137
Tabla 4.22: Descripción de la tabla “tr022_historial_eventos”. (1/2)	137
Tabla 4.22: Descripción de la tabla “tr022_historial_eventos”. (2/2)	138
Tabla 4.23: Descripción de la tabla “tr023_responsables”.....	138

CAPÍTULO I

INTRODUCCIÓN

1.1 Planteamiento del Problema

Hoy día, el mundo de las empresas avanza a pasos agigantados, y este movimiento arrollador va de la mano con los cambios que surgen en la tecnología, las nuevas demandas de información, los cambios sociales, culturales y económicos existentes en este nuevo entorno. Todo esto pone de manifiesto el nuevo horizonte que deben seguir las Aplicaciones Web, como herramientas utilizadas por las empresas/organizaciones para administrar sus actividades internas por medio de las redes. Por lo tanto, las empresas han requerido el procesamiento de la información en línea haciendo uso de la Web, con el fin de automatizar sus procesos, centralizar la información en un único ambiente y de acceder a ella desde cualquier lugar y en cualquier momento, el uso de estas tecnologías ayudan a facilitar los objetivos de las organizaciones ya que permiten realizar cambios radicales en sus procesos, logrando así alcanzar una mejor y mas eficiente comunicación con los individuos, por medio del incremento del rendimiento en áreas como calidad, rapidez y servicios.

La empresa, Petróleos de Venezuela, S.A. (PDVSA) es una de las corporaciones que demanda sistemas informáticos en el ámbito Web, sistemas que son específicamente gestionados por la Gerencia de Automatización, Informática y Telecomunicaciones (AIT) de la Corporación PDVSA, cuyo fin de ésta gerencia es la de apalancar su visión de que la empresa alcance la soberanía tecnológica plena, haciendo uso del Software libre, en el área de la Informática.

Actualmente dentro de la Gerencia de AIT Distrito Social Norte, se encuentra en la cadena de valor un proceso llamado CADENA DE SUMINSTROS (CDS), este



proceso tiene inmerso de manera modular otro proceso llamado PROVISIÓN DE BIENES Y SERVICIOS este tiene como objetivo satisfacer las necesidades de Provisión de Bienes/Servicios tecnológicos de Automatización, Informática y Telecomunicaciones (AIT) requeridas por las regiones y negocios de la Corporación, y su alcance comprende dotación de bienes, servicios y servicios profesionales tecnológicos, que incluye: materiales, activos fijos y activos móviles tecnológicos, activos de propiedad intelectual, servicios de Consultoría, Asesoría, Mantenimiento de equipos y Soporte técnico, garantizando los mejores niveles de calidad. Seguimiento de los compromisos contraídos por los proveedores, cumplimiento de las fechas de entrega, calidad de lo entregado, valores agregados y compromisos sociales, incluidos en los contratos, reportando el comportamiento en cada uno de ellos.

Desde al año 1990 aproximadamente en Lagoven se llevaba de manera manual el control de los consumos telefónicos, la empresas que proveían este tipo de servicio entregaba facturas físicas las cuales eran llevadas a una hoja de calculo para realizar el recobro de los servicios a las diferentes gerencias. Para esas fechas solo se manejaba 49 celulares.

La facturación interna de las filiales de PDVSA en el año 1985 también se controlaba con sistemas parecidos al actual SISTEL (es un sistema desarrollado en PDVSA que permite la gestión administración en cuanto a la asignación de líneas telefónicas en oficinas y viviendas. Su objetivo es recobrar y reintegrar los consumos telefónicos por línea) versiones anteriores de este programa.

A partir del año 2003 se comenzó a realizar el control de los consumos y facturación telefónica de todo Oriente desde el Distrito Social Norte (PLC, ANACO, SAN TOME, SUCRE, MORICHAL, PUNTA DE MATA y MATURIN). A partir de finales del 2005 las empresas proveedoras de servicios telefónicos comenzaron a



enviar la información de la facturación en digital (CD, con la información detallada de los teléfonos, que en el caso de movilnet solo es posible acceder a los datos por la aplicación que este trae), pero el formato de esta data no permite en algunos casos importar los datos para que se puedan estructurar según las necesidades del analista que hace el trabajo de facturación y recobro.

Los procedimientos manuales que continúan llevándose a cabo, originan retrasos considerables que perjudican el desempeño de la Gerencia ante los clientes, produciendo paradas en la correcta circulación de la información en los distintos departamentos, así como también pérdidas que afectan las metas anuales trazadas por la Gerencia.

El proceso de pasar la data del formato de cada uno de los proveedores a la hoja de calculo donde se lleva el control de los consumos telefónicos tarda aproximadamente un mes ya que hay que hacer este proceso de manera manual factura por factura.

Con el propósito de apalancar la gestión de CDS y poder hacer seguimiento a los compromisos adquiridos con los proveedores de servicio telefónico, se debe desarrollar una aplicación bajo estándares abiertos que genere de manera automática por gerencia, región y distrito la facturación telefónica correspondiente al consumo de las mismas además de ser un mecanismo de control para el recobro de estos servicios.

El sistema que se propone es una aplicación vía WEB que permita a través de la interconectividad de las diferentes bases de datos de los proveedores de servicio telefónico hacer el seguimiento del consumo y de la facturación de estos servicios, tanto para las gerencias como para los proveedores.



Es por tal situación que se hizo necesaria la automatización de estos procesos por medio del desarrollo de un sistema automatizado para facturación y recobro de servicios telefónicos “SAFREST” realizado bajo la filosofía de software libre que proveerá de una base de datos única que permitirá el manejo de la información de forma eficiente, reducir los riesgos de pérdida de información, de mayor seguridad para la data y genere con mayor rapidez reportes de gestión para la toma de decisiones gerenciales efectivas, con mínimos porcentajes de error.

Con esta aplicación se busca generar respuestas a cada una de las gerencias pertenecientes a la Corporación de manera rápida y segura, ya que el tiempo de respuestas de las aplicaciones informáticas es mucho menor que el de cualquier procedimiento manual, teniendo en cuenta los niveles de seguridad que estas otorgan esta será una herramienta que redundara en beneficios para la Corporación.

1.2 Objetivos

1.2.1 Objetivo General

Desarrollar un sistema automatizado para facturación y recobro de servicios telefónicos asociado a la gerencia AIT, PDVSA Maturín.

1.2.2 Objetivos Específicos

- ❖ Describir el funcionamiento del sistema de recobro de facturación telefónica actual de las actividades que se realizan en la Gerencia AIT, PDVSA Maturín.
- ❖ Determinar los requerimientos del sistema, considerando las necesidades y prioridades de los usuarios de la Corporación.
- ❖ Diseñar el modelo de la base de datos a desarrollar.



- ❖ Diseñar la interfaz con el usuario tomando en cuenta los requerimientos del personal que va a ser uso de la aplicación.

- ❖ Codificar los módulos del sistema con su documentación.

- ❖ Realizar las pruebas finales al sistema para la detección de fallas.

CAPÍTULO II

MARCO TEÓRICO

2.1 Antecedentes

En la actualidad las empresas proveedoras de servicios telefónicos continúan enviando la información de la facturación en digital, pero el proceso de pasar la data del formato de cada uno de los proveedores a la hoja de calculo donde se lleva el control de los consumos telefónicos se hace de manera manual factura por factura.

Es por ello que a continuación se mencionan todos aquellos trabajos de investigación que permitieron respaldar de manera eficiente el desarrollo de este proyecto, exponiendo temas relacionados al uso de los fundamentos del Proceso Unificado de Desarrollo de Software, manipulación de bases de datos, etc.

Entre los trabajos de investigación que sirvieron como fundamentos teóricos para la mejor realización de este proyecto citamos los siguientes:

“Desarrollo de un software para la automatización de las actividades de mantenimiento de inventario de la infraestructura tecnológica de información de una empresa petrolera” (2005). Trabajo de Grado presentado en la Universidad de Oriente, Núcleo de Anzoátegui, realizado por Seijas R., para optar al título de Ingeniero en Computación. Este trabajo se ejecuto con la finalidad de desarrollar un sistema que alcanzará la automatización de las actividades de mantenimiento de inventario de los equipos de informática del Departamento 1M (Gerencia de Información) de Petrozuata C.A. Para lograr este objetivo se realizo la integración de varios sistemas informáticos con el principio fundamental de proporcionar data confiable, objetiva y veraz. En este trabajo se siguieron los lineamientos de la



metodología “Proceso Unificado de Desarrollo de Software”, el sistema fue codificado utilizando el lenguaje de programación Visual Basic 6.0. Se utilizó el sistema manejador de base de datos Microsoft Access 2000, para la implementación de la base de datos y la gestión de la información. (Seijas R., 2005)

“Desarrollo de un Sistema de Información para la Automatización de los procesos Realizados en el Departamento de Ciencias de la Universidad de Oriente, Núcleo de Anzoátegui” (2.004). Trabajo de Grado presentado en la Universidad de Oriente, Núcleo de Anzoátegui, realizado por Miguel E. Simoni G., para optar al título de Ingeniero en Computación. Se empleó el lenguaje unificado de modelado (UML) para la presentación de la arquitectura del software mostrando las vistas de los modelos estructurados durante la evolución del proyecto, tales como: modelos de caso de usos, modelos de análisis, modelo de diseño, modelo de despliegue, modelo de implementación y modelo de prueba. Se utilizó el sistema manejador de base de datos (DBMS) Microsoft SQLServer2000 para la codificación de la base de datos y la gestión de la información del sistema. (Simoni M., 2004)

“Desarrollo de un Software que Permita la Integración de las Bases de Datos de Activos Pertenecientes a las Unidades de Explotación de PDVSA Punta de Mata” (2.007). Trabajo de Grado presentado a la Universidad de Oriente, Núcleo de Anzoátegui, realizado por René D. Suniaga A., para optar al título de Ingeniero en Computación. Este trabajo se ejecuto para permitir el manejo automatizado de toda la información relativa a los activos de una o varias estaciones de flujo. Se denomina Sistema Integrador de Bases de Datos de Activos porque dicho sistema se encarga de agrupar en una sola colección de datos, la información de los activos que se encuentran distribuidos en las bases de datos corporativas o que provienen de los computadores industriales ubicados en campo. (Suniaga R., 2007)



2.2 Marco Teórico

2.2.1 Software

Se denomina software (palabra de origen anglosajón, pronunciada "sóft-uer"), programa, equipamiento lógico o soporte lógico a todos los componentes intangibles de una computadora, es decir, al conjunto de programas y procedimientos necesarios para hacer posible la realización de una tarea específica, en contraposición a los componentes físicos del sistema (hardware). (Preessman, R., 2002)

Probablemente la definición más formal de software es la atribuida a la IEEE en su estándar 729: «la suma total de los programas de cómputo, procedimientos, reglas documentación y datos asociados que forman parte de las operaciones de un sistema de cómputo». (Preessman, R., 2002)

El término «software» fue usado por primera vez en este sentido por John W. Tukey en 1957. En las ciencias de la computación y la ingeniería de software, el software es toda la información procesada por los sistemas informáticos: programas y datos. (Preessman, R., 2002)

2.2.2 Objetivos de la ingeniería de software

En la construcción y desarrollo de proyectos se aplican métodos y técnicas para resolver los problemas, la informática aporta herramientas y procedimientos sobre los que se apoya la ingeniería de software.

- ❖ Mejorar la calidad de los productos de software.

- ❖ Aumentar la productividad y trabajo de los ingenieros del software.



- ❖ Facilitar el control del proceso de desarrollo de software.
- ❖ Suministrar a los desarrolladores las bases para construir software de alta calidad en una forma eficiente.
- ❖ Definir una disciplina que garantice la producción y el mantenimiento de los productos software desarrollados en el plazo fijado y dentro del costo estimado. (Solange, G., 2004)

2.2.3 Base de Datos

Una base de datos es un conjunto de datos almacenados entre los que existen relaciones lógicas y ha sido diseñada para satisfacer los requerimientos de información de una empresa u organización. En una base de datos, además de los datos, también se almacena su descripción. Además, la base de datos no sólo contiene los datos de la organización, también almacena una descripción de dichos datos. Esta descripción es lo que se denomina metadatos, se almacena en el diccionario de datos o catálogo y es lo que permite que exista independencia de datos lógica-física. (Marqués, M., 2001)

2.2.3.1 Arquitectura de un Sistema de Base de Datos

Posee tres niveles con el objetivo de separar el software de aplicación con la base de datos física. (Trejo, J., 2002)

Nivel Físico es el nivel real de los datos almacenados. Es decir como se almacenan los datos, ya sea en registros, o como sea. Este nivel es usado por muy pocas personas que deben estar calificadas para ello. Este nivel lleva asociada una representación de los datos, que es lo que denominamos Esquema Físico. (Trejo, J., 2002)



Nivel Conceptual es el correspondiente a una visión de la base de datos desde el punto de vista del mundo real. Es decir tratamos con la entidad u objeto representado, sin importarnos como está representado o almacenado. Este nivel lleva asociado el Esquema Conceptual. (Trejo, J., 2002)

Nivel Visión son partes del esquema conceptual. El nivel conceptual presenta toda la base de datos, mientras que los usuarios por lo general sólo tienen acceso a pequeñas parcelas de ésta. El nivel visión es el encargado de dividir estas parcelas. Un ejemplo sería el caso del empleado que no tiene por qué tener acceso al sueldo de sus compañeros o de sus superiores. El esquema asociado a éste nivel es el Esquema de Visión. (Trejo, J., 2002)

Los 3 niveles vistos, componen lo que conocemos como arquitectura de base de datos a 3 niveles. (Trejo, J., 2002)

2.2.3.2 Sistema de gestión de base de datos

Los Sistemas de gestión de base de datos son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. En los textos que tratan este tema, o temas relacionados, se mencionan los términos SGBD y DBMS, siendo ambos equivalentes, y acrónimos, respectivamente, de Sistema Gestor de Bases de Datos y DataBase Management System, su expresión inglesa. (Barzanallana, R., 2006)

El propósito general de los sistemas de gestión de base de datos es el de manejar de manera clara, sencilla y ordenada un conjunto de datos que



posteriormente se convertirán en información relevante, para un buen manejo de datos. (Barzanallana, R., 2006)

2.2.3.3 Objetivos de los SGBD

Abstracción de la información los SGBD ahorran a los usuarios detalles acerca del almacenamiento físico de los datos. (Barzanallana, R., 2006)

Independencia es la capacidad de modificar el esquema (físico o lógico) de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella. (Barzanallana, R., 2006)

Redundancia Mínima un buen diseño de una base de datos logrará evitar la aparición de información repetida o redundante. (Barzanallana, R., 2006)

Consistencia en aquellos casos en los que no se ha logrado esta redundancia nula, será necesario vigilar que aquella información que aparece repetida se actualice de forma coherente, es decir, que todos los datos repetidos se actualicen de forma simultánea. (Barzanallana, R., 2006)

Seguridad la información almacenada en una base de datos puede llegar a tener un gran valor. Los SGBD deben garantizar que esta información se encuentra segura frente a usuarios malintencionados, que intenten leer información privilegiada, por lo cual disponen de un complejo sistema de permisos a usuarios y grupos de usuarios, que permiten otorgar diversas categorías de permisos. (Barzanallana, R., 2006)



Integridad se trata de proteger los datos ante fallos de hardware, datos introducidos por usuarios descuidados, o cualquier otra circunstancia capaz de corromper la información almacenada. (Barzanallana, R., 2006)

Respaldo y recuperación los SGBD deben proporcionar una forma eficiente de realizar copias de respaldo de la información almacenada en ellos, y de restaurar a partir de estas copias los datos que se hayan podido perder. (Barzanallana, R., 2006)

Control de la concurrencia un SGBD debe controlar este acceso concurrente a la información, que podría derivar en inconsistencias. (Barzanallana, R., 2006)

Tiempo de respuesta lógicamente, es deseable minimizar el tiempo que el SGBD tarda en darnos la información solicitada y en almacenar los cambios realizados. (Barzanallana, R., 2006)

2.2.3.4 Diseño de Bases de Datos

Para almacenar los datos de la aplicación, es necesario también diseñar la estructura de la base de datos que va a contener esos datos. El diseño de la base de datos es importante para optimizar el tamaño ocupado por los datos y la agilidad de consulta de los mismos. (Date, C., 2001)

El Diseño de la Base de Datos es fundamental para obtener cualidades como la Integridad de los datos, la seguridad, el tiempo de respuesta, la concurrencia. Cualidades que deben ser mantenidas mediante la Evaluación y el Análisis, una vez que la Base de Datos entra en funcionamiento. (Date, C., 2001)



Un buen diseño inicial es el pilar básico de una Base de Datos eficiente. Las optimizaciones en el rendimiento son menos costosas de implementar si se plantean en la fase de diseño que no si se plantean en fases posteriores. (Date, C., 2001)

El diseño de bases de datos se compone de dos fases altamente diferenciadas: diseño lógico y diseño físico.

Diseño Lógico: Consiste en analizar los requerimientos de la empresa y cuáles van a ser los componentes de la Base de Datos, como por ejemplo las tablas y las restricciones. El diseño lógico no tiene en cuenta dónde o cómo van a estar almacenados físicamente los datos. (Date, C., 2001)

Diseño Físico: Consiste en implementar el diseño lógico sobre los recursos físicos que disponemos, aprovechando las capacidades de nuestro sistema software/hardware. Y de esta forma conseguir un acceso eficiente a los datos. (Date, C., 2001)

2.2.3.5 Modelos de Bases de Datos

Un modelo de datos es básicamente una "descripción" de algo conocido como contenedor de datos (algo en donde se guarda la información), así como de los métodos para almacenar y recuperar información de esos contenedores. (Date, C., 2001)

Los modelos de datos no son cosas físicas: son abstracciones que permiten la implementación de un sistema eficiente de base de datos; por lo general se refieren a algoritmos, y conceptos matemáticos. (Date, C., 2001)



2.2.3.6 Modelos utilizados con frecuencia en las Bases de Datos

Bases de Datos Jerárquicas. (Barzanallana, R., 2006)

Éstas son bases de datos que, como su nombre indica, almacenan su información en una estructura jerárquica. En este modelo los datos se organizan en una forma similar a un árbol, en donde un nodo padre de información puede tener varios hijos. El nodo que no tiene padres es llamado raíz, y a los nodos que no tienen hijos se los conoce como hojas.

Las bases de datos jerárquicas son especialmente útiles en el caso de aplicaciones que manejan un gran volumen de información y datos muy compartidos permitiendo crear estructuras estables y de gran rendimiento. Una de las principales limitaciones de este modelo es su incapacidad de representar eficientemente la redundancia de datos.

Base de Datos de Red. (Barzanallana, R., 2006)

Éste es un modelo ligeramente distinto del jerárquico; su diferencia fundamental es la modificación del concepto de nodo: se permite que un mismo nodo tenga varios padres.

Fue una gran mejora con respecto al modelo jerárquico, ya que ofrecía una solución eficiente al problema de redundancia de datos; pero, aun así, la dificultad que significa administrar la información en una base de datos de red ha significado que sea un modelo utilizado en su mayoría por programadores más que por usuarios finales.

Base de Datos Relacional. (Barzanallana, R., 2006)



Éste es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Su idea fundamental es el uso de "relaciones".

Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados "tuplas". Esto es pensando en cada relación como si fuese una tabla que está compuesta por registros (las filas de una tabla), que representarían las tuplas, y campos (las columnas de una tabla).

En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia (a diferencia de otros modelos como el jerárquico y el de red). Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario esporádico de la base de datos.

La información puede ser recuperada o almacenada mediante "consultas" que ofrecen una amplia flexibilidad y poder para administrar la información.

2.2.3.7 Requerimientos de las bases de datos

El análisis de requerimientos para una base de datos incorpora las mismas tareas que el análisis de requerimientos del software. Es necesario un contacto estrecho con el cliente; es esencial la identificación de las funciones e interfaces; se requiere la especificación del flujo, estructura y asociatividad de la información y debe desarrollarse un documento formal de los requerimientos. (Date, C., 2001)

2.2.4 Navegador web

Un Navegador Web, es una aplicación que permite al usuario recuperar y visualizar documentos de hipertexto, comúnmente descritos en HTML, desde servidores Web de todo el mundo a través de Internet o localmente a través de una Intranet. Esta red



de documentos es denominada World Wide Web (WWW) o Telaraña Mundial. Los navegadores actuales permiten mostrar y/o ejecutar: gráficos, secuencias de vídeo, sonido, animaciones y programas diversos además del texto y los hipervínculos o enlaces. (Mateu, C., 2004)

La funcionalidad básica de un navegador Web es permitir la visualización de documentos de texto, posiblemente con recursos multimedia incrustados. Los documentos pueden estar ubicados en la computadora en donde está el usuario, pero también pueden estar en cualquier otro dispositivo que esté conectado a la computadora del usuario o a través de Internet, y que tenga los recursos necesarios para la transmisión de los documentos (un software servidor Web). Tales documentos, comúnmente denominados páginas Web, poseen hipervínculos que enlazan una porción de texto o una imagen a otro documento, normalmente relacionado con el texto o la imagen. (Mateu, C., 2004)

2.2.5 Internet

Internet son millones de ordenadores conectados entre sí independientes unos de otros. Para que todos estos ordenadores puedan coexistir y comunicarse entre ellos, deben ponerse de acuerdo. Con este motivo fueron creados los Protocolos, que son reglas de comunicación que han de adoptarse para ser entendidos por los otros ordenadores de la red. Los dos protocolos más importantes son Protocolo de Control de Transmisión (Transfer Control Protocol) y el Protocolo de Internet (Internet Protocol). Un ordenador, si maneja estos dos protocolos, no tendrá ningún problema para ser entendido por los demás ordenadores de Internet. Uno de los servicios de Internet es el World Wide Web a menudo llamado Web, es una herramienta basada en hipertextos que permite recuperar y mostrar información. (Castells, M., 2001)



2.2.6 Sistema web

Se denominan sistemas web a aquellas aplicaciones cuya interfaz se construye a partir de páginas web. Las páginas web no son más que ficheros de texto en un formato estandar denominado HTML (Hiptertext Markup Language). Estos ficheros se almacenan en un servidor web al cual se accede utilizando el protocolo HTTP (Hypertext Transfer Protocol), uno de los protocolos de internet. Para utilizar una aplicación web desde una maquina concreta basta con tener instalado un navegador web en esa maquina, ya sea este el Internet Explorer de Microsoft, el Netscape Navigator o cualquier otro navegador. (Berzal, F y Cubero, J. 2005, p.187)

2.2.7 Interfaz web

Las interfaces web tienen ciertas limitaciones en la funcionalidad del cliente. Métodos comunes en las aplicaciones de escritorio como dibujar en la pantalla o arrastrar-y-soltar no están soportadas por las tecnologías web estándar. Los desarrolladores web comúnmente utilizan lenguajes interpretados del lado del cliente para añadir más funcionalidad, especialmente para crear una experiencia interactiva que no requiera recargar la página cada vez (cosa que suele molestar a los usuarios). Recientemente se han desarrollado tecnologías para coordinar estos lenguajes con tecnologías del lado del servidor, como por ejemplo PHP. AJAX, es una técnica de desarrollo web que usa una combinación de varias tecnologías. (Mateu, C., 2004)

2.2.8 Estructura web

Aunque muchas variaciones son posibles, una aplicación web está comúnmente estructurada como una aplicación de tres-capas. En su forma más común, el navegador web es la primera capa, un motor usando alguna tecnología web dinámica (ejemplo: CGI, PHP, Java Servlets o ASP) es la capa de en medio, y una base de datos como última capa. El navegador web manda peticiones a la capa media, que la



entrega valiéndose de consultas y actualizaciones a la base de datos generando una interfaz de usuario.

En tiempos recientes se ha usado la estrategia de generalizar esta arquitectura mediante la adición de piezas de hardware que permiten balancear la carga de los servidores web y de aplicación. (Mateu, C., 2004)

2.2.9 Lenguajes de programación

Un lenguaje de programación es un sistema notacional para describir computaciones de una forma legible tanto para la maquina como para el ser humano. (Louden, K. 2002, p.3)

Según se aproximen al lenguaje nativo del ordenador, llamado código máquina, o a la forma humana de expresar órdenes, se distinguen distintos niveles de lenguajes de programación:

Lenguajes de Bajo nivel: son los más rápidos y ponen al servicio del programador todo el equipo físico. Se utilizan para el desarrollo de programas básicos (otros lenguajes, control de procesos, etc.). Como ejemplo puede citarse el ensamblador.

Lenguajes de alto nivel: en este grupo se encuentran los más conocidos. La elaboración de programas es más sencilla y cada lenguaje suele estar enfocado a un determinado campo. Entre ellos los más habituales son: COBOL, FORTRAN, BASIC, PASCAL, C, PROLOG, MUMPS.

Lenguajes de cuarta generación: son las herramientas de desarrollo de programas. En este caso las órdenes existentes están mucho más próximas a la forma



de pensar humana y por ello resulta más sencillo, flexible y productivo trabajar con estos lenguajes. Estas herramientas suelen constar de generadores de pantallas, generadores de informes, generadores de aplicaciones y utilidades. Como ejemplos mas habituales pueden citarse: Oracle, Transtool, Clipper, Informix, DBase IV, etc.

Otros lenguajes: el gran auge de los entornos Windows a conducido al desarrollo de lenguajes visuales que utilizan la programación orientada a objetos. Entre estos lenguajes esta el Visual Basic de Microsoft, el Delphi de Borland, el C++ y el SQL. (Castañeiras, M y Fuentes, X. 1999, p.113-114)

2.2.9.1 PHP

PHP es un lenguaje de desarrollo web escrito por y para los desarrolladores web. PHP significa: Hypertext Preprocessor. El producto fue originalmente llamado Personal Home Page Tools, Actualmente se encuentra en su quinta reescritura, llamado PHP5 o simplemente PHP.

Es un lenguaje de scripts del lado del servidor, que puede ser embebido en HTML o usado únicamente como binario (aunque el uso anterior es mucho más común). (Converse, T. y Park, J, 2004, p3)

PHP es un lenguaje ideal tanto para aprender a desarrollar aplicaciones web como para desarrollar aplicaciones web complejas. Añade a todo eso la ventaja de que el intérprete de PHP, los diversos módulos y gran cantidad de librerías desarrolladas para PHP son de código libre, con lo que el programador, dispone de un impresionante arsenal de herramientas libres para desarrollar aplicaciones.

PHP suele ser utilizado conjuntamente con Perl, Apache, MySQL o PostgreSQL en sistemas Linux, formando una combinación barata (todos los



componentes son de código libre), potente y versátil. Tal ha sido la expansión de esta combinación que incluso ha merecido conocerse con un nombre propio LAMP (formado por las iniciales de los diversos productos).

Apache, así como algunos otros servidores web, Roxen entre ellos, pueden incorporar PHP como un módulo propio del servidor, lo cual permite que las aplicaciones escritas en PHP resulten mucho más rápidas que las aplicaciones CGI habituales. (Mateu, C. 2004, p.187)

2.2.9.2 JavaScript

Netscape creó el lenguaje JavaScript en 1996 y lo incluyó en su Netscape Navigator (NN) 2,0 a través de un intérprete que lee y ejecuta el código JavaScript añadido en páginas Html. El lenguaje ha crecido en popularidad de forma constante desde entonces, y ahora está apoyado por los navegadores más populares. (Heilmann, C. 2006, p.4)

Javascript es un lenguaje de programación interpretado (un lenguaje de tipo script). A pesar de que existen intérpretes no dependientes de ningún navegador, es un lenguaje de script que suele encontrarse vinculado a páginas web. Javascript y Java son dos lenguajes de programación distintos con filosofías muy diferentes. El único punto en común es la sintaxis, ya que cuando Netscape diseñó Javascript, se inspiró en la sintaxis de Java. (Mateu, C. 2004, p.94)

2.2.9.3 Frameworks

En el desarrollo de software, un framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado



entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

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

Los frameworks son diseñados con el intento de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software que tratando con los tediosos detalles de bajo nivel de proveer un sistema funcional. Por ejemplo, un equipo que usa Apache Struts para desarrollar un sitio web de un banco puede enfocarse en cómo los retiros de ahorros van a funcionar en lugar de preocuparse de cómo se controla la navegación entre las páginas en una forma libre de errores. Sin embargo, hay quejas comunes acerca de que el uso de frameworks añade código innecesario y que la preponderancia de frameworks competitivos y complementarios significa que el tiempo que se pasaba programando y diseñando ahora se gasta en aprender a usar frameworks.

Fuera de las aplicaciones en la informática, un framework puede ser considerado como el conjunto de procesos y tecnologías usados para resolver un problema complejo. Es el esqueleto sobre el cual varios objetos son integrados para una solución dada. (<http://es.wikipedia.org/wiki/Framework>)

2.2.9.4 Ext-js (Extend JS)

Ext JS es una librería Javascript para construir aplicaciones (RIA), la cual se comenzó a desarrollar a principios del 2006 por Jack Slocum como un conjunto de extensiones



para la librería Yahoo! User Interface (YUI), estas extensiones recibieron el nombre de yui-ext.

El otoño del 2006 la librería ganó tanta popularidad (con su versión 0.33) que cambió su nombre a Ext (con licencia BSD) como un reflejo de su madurez e independencia como framework. (http://extjs.com/learn/Ext_FAQ)

En 2007 se forma una empresa y se liberan las versiones 1, 1.1, 2.0 (actualmente se encuentra en construcción la 3.0).

Incluye:

- ❖ Alto rendimiento, widgets personalizables en entorno de usuario (UI).
- ❖ Bien diseñado y modelo de Componentes extensibles.
- ❖ Intuitivo, API fácil de utilizar.
- ❖ Licencias Comerciales y Open Source disponibles.

Ext JS soporta y es compatible con la mayoría de navegadores actuales.

- ❖ Internet Explorer 6+.
- ❖ Firefox 1.5+ (PC, Mac).
- ❖ Safari 3+.
- ❖ Opera 9+ (PC, Mac).

Ext Js trabaja conjuntamente con las librerías:

- ❖ Yahoo! UI (.12+).
- ❖ jQuery (1.1+).
- ❖ Prototype (1.5+) / Scriptaculous (1.7+).



En la Figura 2.1 y Figura 2.2 se puede observar la interacción de dichas librerías:

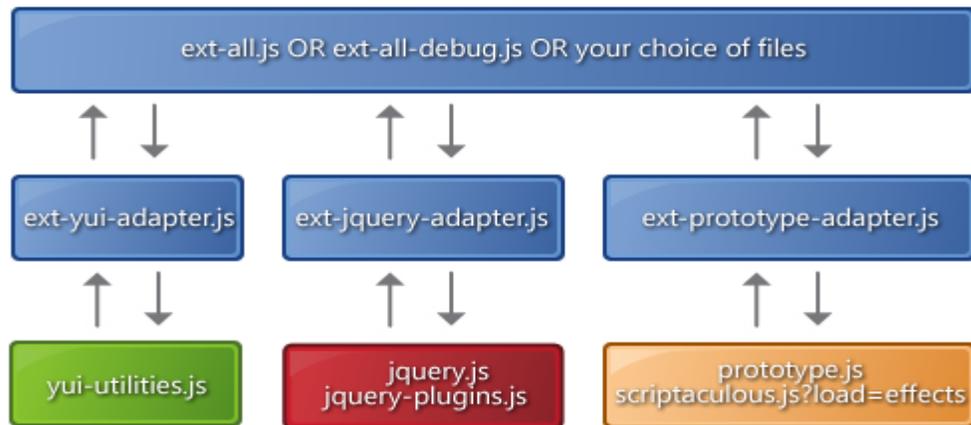


Figura 2.1: ExtJS 1.0.1.a y las bases de la relaciones de la librería.

Fuente: http://extjs.com/learn/Ext_Getting_Started, 2008.

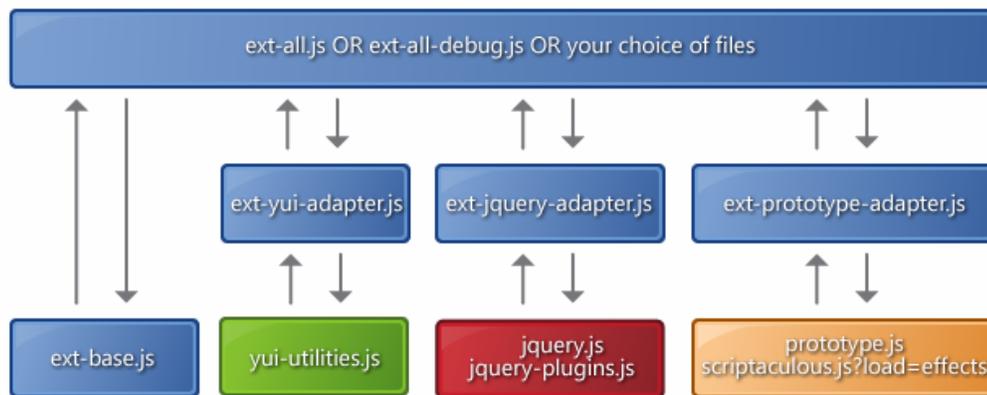


Figura 2.2: ExtJS 1.1 y la base de relaciones de la librería.

Fuente: http://extjs.com/learn/Ext_Getting_Started, 2008.

2.2.9.5 Ajax

El término AJAX se presentó por primera vez en el artículo "Ajax: A New Approach to Web Applications" publicado por Jesse James Garrett el 18 de Febrero de 2005. Hasta ese momento, no existía un término normalizado que hiciera referencia a un nuevo tipo de aplicación web que estaba apareciendo.



En realidad, el término AJAX es un acrónimo de *Asynchronous JavaScript + XML*, que se puede traducir como "JavaScript asíncrono + XML".

Ajax no es una tecnología en sí mismo. En realidad, se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes. Es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Éstas se ejecutan en el cliente, es decir, en el navegador de los usuarios y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la interactividad, velocidad y usabilidad en la misma. (Eguíluz, J. 2008. p.5)

Las tecnologías que forman AJAX son:

- ❖ XHTML y CSS, para crear una presentación basada en estándares.
- ❖ DOM, para la interacción y manipulación dinámica de la presentación.
- ❖ XML, XSLT y JSON, para el intercambio y la manipulación de información.
- ❖ XMLHttpRequest, para el intercambio asíncrono de información.
- ❖ JavaScript, para unir todas las demás tecnologías.

2.2.9.6 Protocolo http

El protocolo HTTP (hypertext transfer protocol, por sus siglas en inglés) es el protocolo base de la WWW. Se trata de un protocolo simple, orientado a conexión y sin estado. (Mateu, C., 2004)

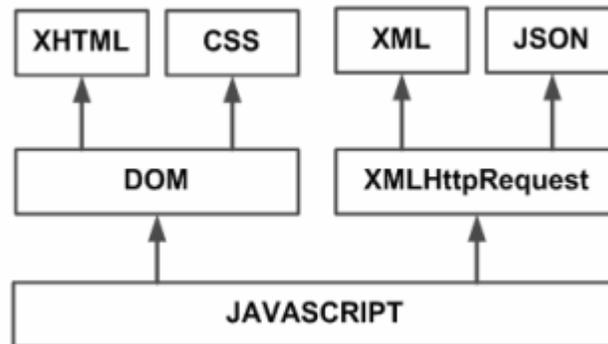


Figura 2.3: Tecnologías agrupadas bajo el concepto de AJAX.
Fuente: Eguíluz, J. 2008.

La razón de que esté orientado a conexión es que emplea para su funcionamiento un protocolo de comunicaciones (TCP, transport control protocol, por sus siglas en inglés) de modo conectado, un protocolo que establece un canal de comunicaciones de extremo a extremo (entre el cliente y el servidor) por el que pasa el flujo de bytes que constituyen los datos que hay que transferir, en contraposición a los protocolos de datagrama o no orientados a conexión que dividen los datos en pequeños paquetes (datagramas) y los envían, pudiendo llegar por vías diferentes del servidor al cliente. (Mateu, C., 2004)

El protocolo no mantiene estado, es decir, cada transferencia de datos es una conexión independiente de la anterior, sin relación alguna entre ellas, hasta el punto de que para transferir una página Web tenemos que enviar el código HTML del texto, así como las imágenes que la componen, pues en la especificación inicial de HTTP, la 1.0, se abrían y usaban tantas conexiones como componentes tenía la página, transfiriéndose por cada conexión un componente (el texto de la página o cada una de las imágenes). (Mateu, C., 2004)



2.2.10 MySQL

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario, es un sistema de administración relacional de bases de datos. Una base de datos relacional archiva datos en tablas separadas en vez de colocar todos los datos en un gran archivo. Esto permite velocidad y flexibilidad. Las tablas están conectadas por relaciones definidas que hacen posible combinar datos de diferentes tablas sobre pedido. (Vaswani, V., 2005)

MySQL es software de fuente abierta. Fuente abierta significa que es posible para cualquier persona usarlo y modificarlo. Cualquier persona puede bajar el código fuente de MySQL y usarlo sin pagar. Cualquier interesado puede estudiar el código fuente y ajustarlo a sus necesidades. MySQL usa el GPL (GNU General Public License) para definir que puede hacer y que no puede hacer con el software en diferentes situaciones. (Vaswani, V., 2005)

2.2.11 Proceso Unificado de Desarrollo de Software

El Proceso Unificado de Desarrollo Software o simplemente Proceso Unificado, es un marco de desarrollo de software iterativo e incremental. El refinamiento más conocido y documentado del Proceso Unificado es el Proceso Unificado Rational o simplemente RUP. (Casallas, R., 2006)

El Proceso Unificado no es simplemente un proceso, sino un marco de trabajo extensible que puede ser adaptado a organizaciones o proyectos específicos. De la misma forma, el Proceso Unificado de Rational, también es un marco de trabajo extensible, por lo que muchas veces resulta imposible decir si un refinamiento particular del proceso ha sido derivado del Proceso Unificado o del RUP. Por dicho



motivo, los dos nombres suelen utilizarse para referirse a un mismo concepto. (Casallas, R., 2006)

El nombre Proceso Unificado se usa para describir el proceso genérico que incluye aquellos elementos que son comunes a la mayoría de los refinamientos existentes. También permite evitar problemas legales ya que Proceso Unificado de Rational o RUP son marcas registradas por IBM (desde su compra de Rational Software Corporation en 2003). El primer libro conocido sobre el tema se denominó, “El Proceso Unificado de Desarrollo de Software” y fue publicado en 1999 por Iván Jacobson, Grady Booch y James Rumbaugh, conocidos también por ser los desarrolladores del UML, el Lenguaje Unificado de Modelado. Desde entonces los autores que publican libros sobre el tema y que no están afiliados a Rational utilizan el término Proceso Unificado (PU), mientras que los autores que pertenecen a Rational favorecen el nombre de Proceso Unificado de Rational. (Casallas, R., 2006)

El PU crea un modelo de proceso enfocado desde la perspectiva de desarrollo de software en el campo industrial y empresarial. En ese sentido, el proceso involucra tanto a las personas (sus habilidades o roles) pertenecientes a los diferentes niveles de organización de las empresas, como a las tecnologías (lenguajes de programación, sistemas operativos, ordenadores, entornos de desarrollo, estructuras de red, etc.) disponibles en el momento en que se va a emplear el proceso, y a las herramientas de software que se utilizan para automatizar las actividades definidas en el proceso. (Casallas, R., 2006)

Esta metodología es el producto final de tres décadas de desarrollo y uso práctico en el que han estado involucrados otros productos de proceso anteriores y que ha recibido aportaciones de muchas otras fuentes. Es un proceso con gran influencia actualmente en la industria del desarrollo de software, gracias a que se encuentra adaptado a las dificultades que afrontan los desarrolladores para coordinar



las múltiples cadenas de trabajo de un gran proyecto de software. Se trata de un método común, unificado, ante la necesidad de integrar las múltiples facetas del desarrollo. Proporciona una guía para ordenar las actividades de un equipo, es un proceso que dirige las tareas de cada desarrollador por separado y del equipo como un todo, especifica además los “artefactos” (piezas de información tangible) que deben desarrollarse y ofrece criterios para el control y la medición de los productos y actividades del proyecto. (Casallas, R., 2006)

El PU se define como un proceso dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental. Está basado en componentes (el sistema software en construcción está formado por componentes software) y utiliza el nuevo estándar de modelado visual UML para describir los distintos modelos generados durante el desarrollo del proceso, es decir, para preparar todos los esquemas de un sistema software. En este proceso, se distinguen cuatro fases fundamentales, dentro de cada una de las cuales se plantean una serie de flujos de trabajo iterativos que permiten la evolución del proyecto de manera incremental (ver Figura 2.4).

Se diferencian cuatro fases en el proceso de desarrollo de software, atendiendo al momento en que se realizan: inicio, elaboración, construcción y transición. Cada una de estas fases se divide entonces en una o más iteraciones. En cada iteración, el proceso se detiene en mayor o menor grado en cada uno de los distintos flujos de trabajo. Se diferencian cinco flujos de trabajo fundamentales en este proceso, atendiendo al estado de desarrollo del sistema: requisitos, análisis, diseño, implementación y prueba. (Jiménez, R, 2003)

2.2.11.1 Fase de Inicio

En la fase de inicio se establece una visión del proyecto y su alcance, es cuando la idea inicial para el desarrollo se lleva al punto de estar suficientemente fundamentada



para garantizar la entrada en la fase de elaboración. En esta fase de inicio se evalúa la viabilidad del proyecto, sobre todo cuando está en juego una gran inversión de recursos humanos y económicos. En este sentido, esta fase apenas consumirá dedicación cuando esos proyectos sean de pequeña escala. (Jiménez, R., 2003)

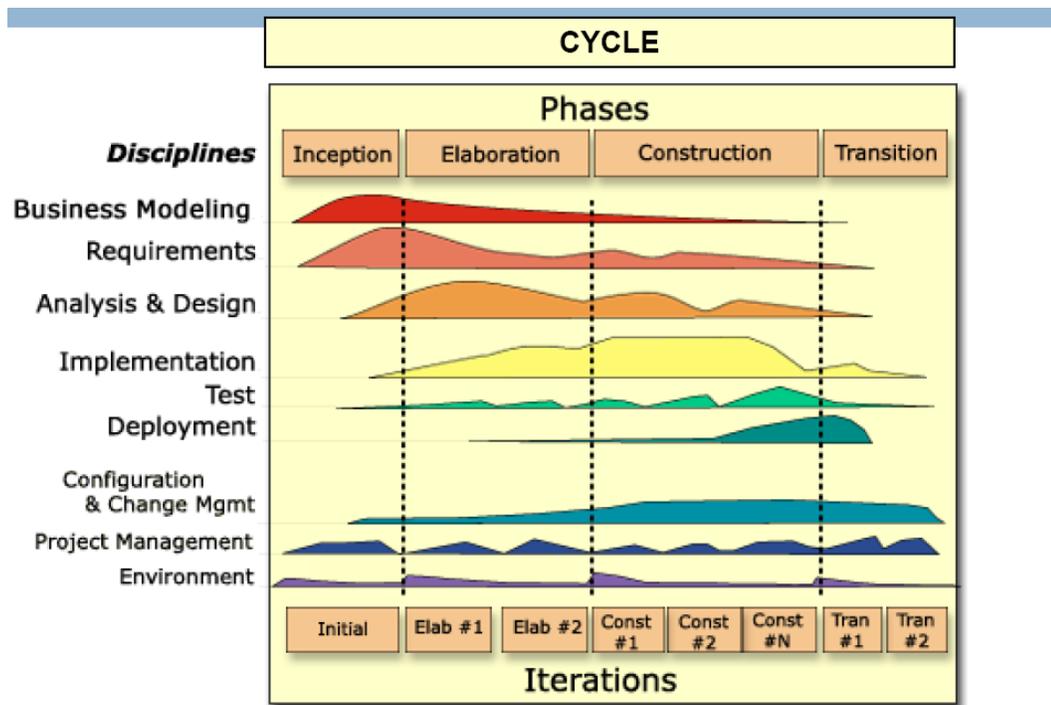


Figura 2.4: Fases del Proceso Unificado de Desarrollo del Software.
Fuente: Casallas, R. 2006.

La captura de requisitos comienza en esta fase, que consiste en un flujo de trabajo que pretende modelar la funcionalidad del sistema acudiendo a un nivel de abstracción elevado. Para conseguir ese esquema de funcionalidad utiliza los llamados casos de uso. Un caso de uso es una secuencia de acciones que el sistema lleva a cabo para ofrecer algún resultado de valor para un actor. Un actor puede ser una persona humana, un dispositivo de hardware, u otro sistema. Los actores utilizan el sistema interactuando con los casos de uso. (Jiménez, R., 2003)



2.2.11.2 Fase de Elaboración

La elaboración es la segunda fase del proceso, cuando se definen la visión del producto y su arquitectura. En esta fase se expresan con claridad los requisitos del sistema, proporcionando una arquitectura estable para guiar el sistema a lo largo del ciclo de vida. Esta arquitectura es la estructura central del sistema, la línea base, el armazón a partir del cual evolucionará el sistema hacia el producto final. (Jiménez, R, 2003)

Se dice que el PU está dirigido por los casos de uso (producto de la captura de requisitos), que otorgan esa funcionalidad necesaria para que el sistema evolucione. En consecuencia, la forma del sistema corresponde a la arquitectura y la función a los casos de uso (ver Figura 2.5). En esta misma fase, se lleva a cabo el análisis de los casos de uso capturados, una vista más detallada de la funcionalidad del sistema y que sirve como abstracción o simplificación del diseño del modelo, flujo de trabajo que se inicia también en esta fase. (Jiménez, R, 2003)



Figura 2.5: Enlace de los casos de uso con los flujos de trabajo.

Fuente: Jiménez, R, 2003.

En la última iteración dentro de la fase de elaboración empieza a adquirir importancia la implementación del diseño asociado a los casos de uso considerados relevante durante la captura de requisitos en dicha iteración. A su vez, los componentes obtenidos con la implementación del diseño son sometidos a un período de pruebas; en esta fase se prueba la línea base ejecutable de la arquitectura. (Jiménez, R, 2003)



En cada iteración se identifican e implementan unos cuantos casos de uso. Cada iteración, excepto quizás la primera de todas de un proyecto, se dirige por los casos de uso a través de todos los flujos de trabajo, de los requisitos al diseño y a la prueba, añadiéndose un incremento más en el desarrollo del sistema. Cada incremento del desarrollo es, por tanto, una realización funcional de un conjunto de casos de uso. En cada iteración, se toma otro conjunto de casos de uso para desarrollar, y se añaden a los de la iteración anterior. (Jiménez, R, 2003)

En consecuencia, la arquitectura se desarrolla mediante iteraciones, principalmente durante la fase de elaboración. Cada iteración evoluciona comenzando con los requisitos y siguiendo con el análisis, diseño, implementación y pruebas, pero centrándose en los casos de uso relevantes desde el punto de vista de la arquitectura y de otros requisitos. El resultado al final de la fase de elaboración es una línea base de la arquitectura. (Jiménez, R, 2003)

2.2.11.3 Fase de Construcción

Durante la fase de construcción es cuando se desarrolla, también de forma iterativa e incremental, un producto completo que está preparado para la transición hacia la comunidad de usuarios. Esto significa describir los requisitos restantes, refinando el diseño y completando la implementación y las pruebas de software. En esta fase, por tanto, los distintos modelos del sistema van creciendo hasta completarse. La descripción de la arquitectura, sin embargo, no crece significativamente debido a que la mayor parte de esta arquitectura se definió durante la fase de elaboración. (Jiménez, R, 2003)

2.2.11.4 Fase de Transición

Finalmente, durante la fase de transición, el software se despliega en la comunidad de usuarios. Una vez que el sistema ha sido puesto en manos de los usuarios finales, a



menudo aparecen cuestiones que requieren un desarrollo adicional para ajustar el sistema, corregir algunos problemas no detectados o finalizar algunas características que habían sido postpuestas. Esta fase comienza normalmente con una versión beta del sistema, que luego será reemplazada por la versión definitiva del producto. (Jiménez, R, 2003)

2.2.12 Modelados de Páginas Web

En la actualidad el diseño del manejo intensivo de datos que soporta un sitio Web es fundado sobre metodologías adquiridas de diferentes sectores, entre los cuales esta la ingeniería de software y las bases de datos, es por ello que la falta de un modelo que permita controlar al desarrollador aspectos como: la igualdad en la estructura de las bases de datos para la navegación y la cantidad de códigos manuscritos se traducen en grandes esfuerzos hasta para la realización de un prototipo.

Algunos factores incrementan la complejidad de los sitios Web actuales, entre ellos esta: los múltiples dispositivos de salida para la información. Estos factores o requerimientos impactan sobre el costo de desarrollo de los sitios Web y la evolución de los sitios ya creados.

Por todo esto se crea un lenguaje para modelar la información a manejar durante el desarrollo de un sitio Web permitiendo así reducir el trabajo de los diseñadores gráficos, incrementar los niveles de abstracción, hacer un mejor uso de las destrezas disponibles para el análisis y diseño de alto nivel a pesar de la inmensa perdida de tiempo en la codificación de las páginas ASP o PHP, las técnicas a implementar pueden enfocarse en el análisis de la ejecución y optimización a conseguir y por ultimo los creativos del sitio pueden enfocarse en la “creación”. (Brito, R., 2007)



Entre las ventajas del modelado de las aplicaciones Web se encuentran:

- ❖ Puede reducir esfuerzos en el desarrollo (costo y tiempo).
- ❖ Permite un proceso de desarrollo más estructurado.
- ❖ Produce resultados finales más coherentes y utilizables.
- ❖ Los modelos de diseño son siempre actualizados y auto-documentados.
- ❖ El desarrollo del prototipo puede ser inmediatamente alcanzado.

2.2.12.1 Lenguaje Unificado De Modelado (UML)

El UML es un sistema de notación que se ha convertido en estandar en el mundo de desarrollo de sistemas (ver logo en Figura 2.6). Es el resultado del trabajo hecho por Grady Booch, James Rumbaugh e Ivar Jacobson. El UML esta constituido por un conjunto de diagramas, y proporciona un estandar que permite al analista de sistemas generar un anteproyecto de varias facetas que sean comprensibles a los clientes, desarrolladores y todos aquellos que esten involucrados en el proceso de desarrollo. (Schmuller, J.)



Figura 2.6: Logo de UML.
Fuente: Schmuller, J.

En UML, un sistema viene representado por cinco vistas diferentes que lo describen desde diferentes perspectivas. Cada vista se representa mediante un conjunto de diagramas. En UML están presentes las siguientes vistas:



- ❖ **Vista del usuario:** Representa el sistema (producto) desde la perspectiva de los usuarios (llamados actores en UML). El caso de uso es el enfoque elegido para modelar esta vista.
- ❖ **Vista estructural:** los datos y la funcionalidad se muestran desde dentro del sistema, es decir, modela la estructura estática (clases, objetos y relaciones).
- ❖ **Vista del comportamiento:** esta parte del modelo del análisis representa los aspectos dinámicos o de comportamiento del sistema. También muestra las interacciones o colaboraciones entre los diversos elementos estructurales descritos en las vistas anteriores.
- ❖ **Vista de implementación:** los aspectos estructurales y de comportamiento se representan aquí tal y como van a ser implementados.
- ❖ **Vista del entorno:** aspectos estructurales y de comportamiento en el que el sistema a implementar se representa. (Pressman, R. 2002, p.362,363)

Relaciones:

Dependencia: Establece una relación entre una clase dependiente y otra independiente. No establece un tipo específico de dependencia, simplemente se indica que hay una dependencia entre dos clases. (Montiva, J y Besembel I. 2007, pg.26). (Ver Figura 2.7).

Asociación: Establece una relación funcional y bidireccional entre dos o más clases, cada instancia de una clase se asocia a cero, uno o mas instancias de la otra clase asociada. (Montiva, J y Besembel I. 2007, pg.21). (Ver Figura 2.8).

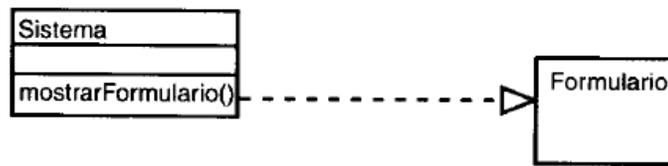


Figura 2.7: Relación de dependencia entre dos clases.
Fuente: Smuller, J.

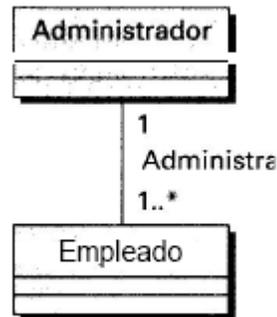


Figura 2.8: Representación de una relación de asociación.
Fuente: Pressman, R. 2002.

Generalización: Establece una relación del tipo “es un” entre dos o más clases. Una o más clases específicas, denominadas subclases, heredan la estructura y comportamiento de una clase genérica (superclase). Las subclases tienen (heredan) los mismos atributos y operaciones que tiene su superclase. (Montiva, J y Besembel, I. 2007, pg.20) (Ver Figura 2.9).

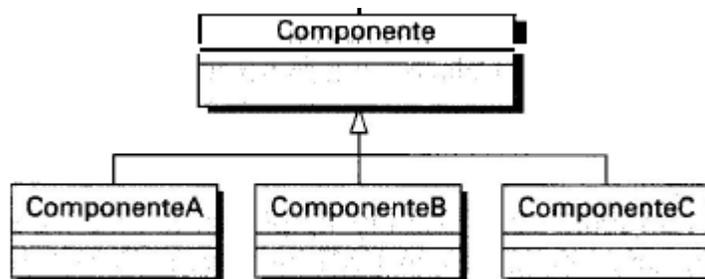


Figura 2.9: Relación de generalización.
Fuente: Pressman, R. 2002.



Diagramas:

Diagramas de casos de uso: es una descripción de las acciones de un sistema desde el punto de vista del usuario. Para los desarrolladores del sistema, ésta es una herramienta valiosa, ya que es una técnica de aciertos y errores para obtener los requerimientos del sistema desde el punto de vista del usuario. Esto es importante si la finalidad es crear un sistema que pueda ser utilizado por la gente en general (no sólo por expertos en computación). (Smuller, J. sf, p.75) (Ver Figura 2.10).

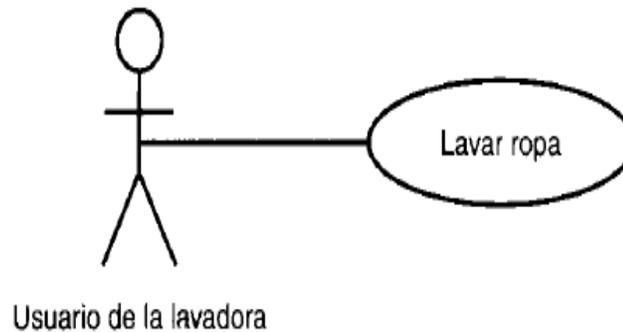


Figura 2.10: Representación de un diagrama de caso de uso.
Fuente: Smuller, J.

Los elementos implicados en un diagrama de casos de uso son los casos de uso, las relaciones y los actores. Un actor es un rol que interactúa con el sistema. Lo definimos como el rol, porque un actor puede ser tanto un usuario de la aplicación como otro sistema o dispositivos externos.

Diagramas de secuencia: Estos diagramas muestran la secuencia de mensajes que se van lanzando los objetos implicados en una determinada operación del programa. Dentro del diagrama los objetos se alinean en el eje X respetando su orden de aparición. En el eje Y se van mostrando los mensajes que se envían, también



respetando su orden temporal, es decir muestra la mecánica de la interacción con base en tiempos. (Smuller, J. sf, p.103) (Ver Figura 2.11).

Cada objeto tiene una línea de vida donde se sitúa su foco de control. El foco de control es un rectángulo que representa el tiempo durante el que un objeto está activo ejecutando una acción. Con este sencillo esquema podemos visualizar la comunicación y sincronización bajo un estricto orden temporal de los objetos implicados en las distintas funcionalidades de un sistema.

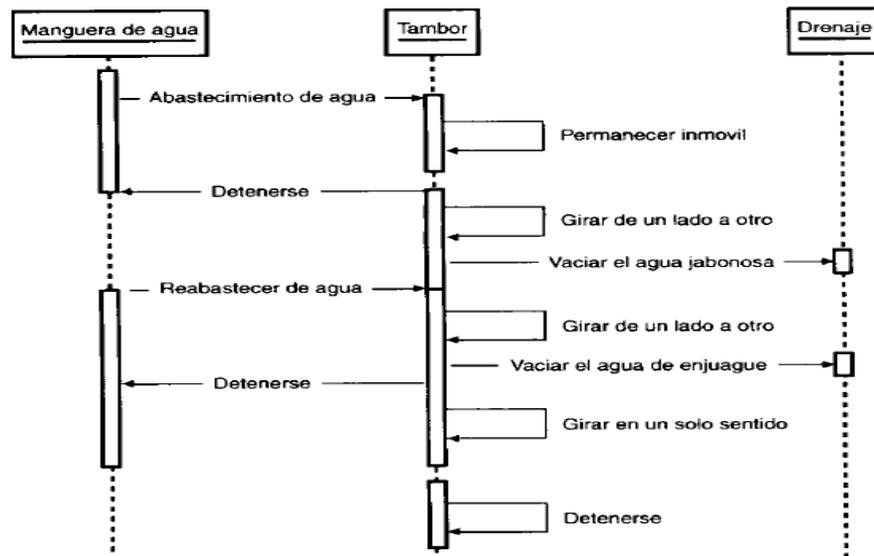


Figura 2.11: Representación de un diagrama de secuencia.

Fuente: Smuller, J.

Diagramas de clases: es una descripción de las clases en un sistema y sus relaciones. No describe el comportamiento dinámico del sistema, por ejemplo el comportamiento de objetos individuales. El primer elemento de un diagrama de clases es una descripción de clases individuales. La Figura 2.12 muestra como se describe una clase. La clase describe al cliente de un banco.



Cada cuadro que representa una clase contiene el nombre de la clase, una sección que enumera los atributos de los objetos definidos por la clase, y una sección que describe las operaciones asociadas con tales objetos. (Pressman, R.2002. p.393)

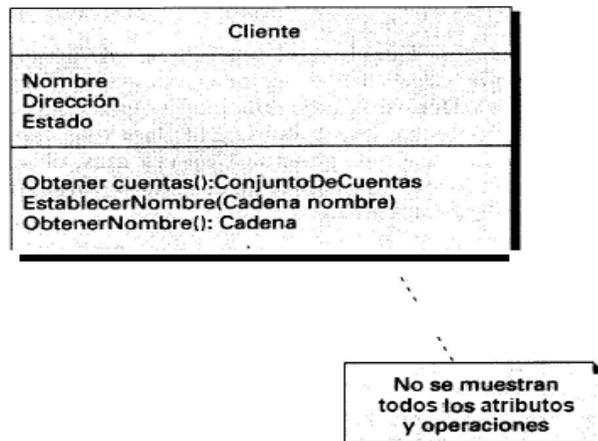


Figura 2.12: Representación de un diagrama de clases.
Fuente: Pressman, R. 2002.

2.2.12.2 WebML (Lenguaje de Modelado Web)

Hasta el día de hoy, lo más importante en el desarrollo de aplicaciones Web han sido las herramientas, pero muy poco se ha dicho y escrito sobre el proceso de desarrollo. La fácil creación de páginas HTML y en general de sitios Web, usando herramientas simples, ha hecho que el desarrollo de éste tipo de aplicaciones se haga sin un trabajo serio de análisis y diseño. Cualquier sistema de complejidad no trivial, necesita ser analizado y modelado. Las aplicaciones Web, al igual que otras aplicaciones, necesitan métodos y técnicas formales de análisis y diseño. (Stefano, C. y Fraternali, P., 2003)

Todas las metodologías propuestas para las aplicaciones Web desde mediados de los años noventa presentan su propia notación para casi todos sus diagramas. Una recopilación excelente de varios de estos métodos fue presentada en el primer taller



internacional sobre tecnologías de software orientadas a Web. En ella se describen cada una de estas propuestas en base a un mismo caso de estudio. (Stefano, C. y Fraternali, P., 2003)

Jim Conallen, a finales de los noventas, presentó una extensión de UML basada en las técnicas de implementación actuales de ese tiempo. Esta extensión propuesta por Conallen es bastante útil como técnica de diseño para cuando la implementación de la aplicación que está siendo modelada se encuentra próxima a realizarse o se está realizando, pero no cuando se trata de modelar una aplicación como parte de la planificación de un proyecto. Estas metodologías, y otras muchas que se han propuesto, contribuyen con ideas importantes para el diseño de software orientado a Web. Entre estas aportaciones mencionamos a continuación algunas de las más significativas:

El método Hipermedia Orientado a Objetos (Object-oriented Hypermedia, OOH) proporciona un conjunto de nuevas vistas que extienden el UML para modelar únicamente la interfaz de aplicaciones Web.

En este contexto, la Ingeniería Web Basada en UML (UML-Based Web Engineering, UWE) es una propuesta de metodología que recupera muchos de los conceptos descritos y utilizados en otras metodologías e incorpora nuevos elementos que permiten describir de mejor manera una aplicación Web.

Otra propuesta interesante además de ser motivo de estudio e implementación en este trabajo es el lenguaje de modelado Web (Web Modeling Language, WebML). El WebML es una notación visual para el diseño de aplicaciones Web complejas y con gran volumen de datos, ya que provee especificaciones gráficas formales envueltas en un completo proceso de diseño que puede ser asistido por herramientas visuales de diseño; a su vez permite la descripción de los sitios Web desde distintos



puntos de vista entre los que podemos mencionar el conceptual, el navegacional y el de presentación. (Stefano, C. y Fraternali, P., 2003)

WebML apunta a proveer un acercamiento en la estructura del diseño intensivo de datos que se manejan en un sitio Web. Un conjunto de modelos integrados ayudan a los diseñadores en la producción del sitio de alta calidad ya que todas las facetas del diseño deben ser y son dirigidas. El uso de metodologías viejas se vuelve despreciado. (Stefano, C. y Fraternali, P., 2003)

Los principales objetivos del proceso de diseño WebML son:

- ❖ Expresar la estructura de una aplicación Web con un alto nivel de descripción que pueda ser usado para consultas, evolución y mantenimiento futuro de la aplicación.
- ❖ Provee múltiples vistas del mismo contenido.
- ❖ Descompone la información en páginas, navegación y presentación, que pueden ser definidas de manera independiente.
- ❖ Permite almacenar la meta - data recolectada durante el diseño para emplearlos durante toda la vida de la aplicación para generar páginas Web de forma dinámica.
- ❖ Sitios Web que manejen gran cantidad de datos haciendo uso de interfaces dirigidas al público en general.

El desarrollo de aplicaciones Web con WebML consiste en diferentes fases que deben ser aplicadas de manera iterativa e incremental. El proceso involucra varios ciclos, cada uno de los cuales produce un prototipo o versión parcial de la aplicación, lo que permite realizar evaluaciones y pruebas desde las fases iniciales de desarrollo.



El proceso de desarrollo comienza con el Modelo de Datos del sistema en el que mediante algún lenguaje de modelado como UML (WebML no exige ninguno en concreto), se representa la estructura estática del mismo. Tras esto, se realiza el Modelo de Hipertexto donde se describen uno o más hipertextos que pueden ser publicados en el sitio Web. Cada uno de estos hipertextos define una vista del sitio. La descripción de los hipertextos se realiza mediante dos modelos: el Modelo de Composición, que define las páginas que componen el sistema, y el Modelo de Navegación, que describe cómo se podrá navegar a través de ellas. Y por último, se describe el Modelo de Presentación que define la apariencia física de las páginas. (Stefano, C. y Fraternali, P., 2003)

Fuera del proceso, el diseño de los datos y el hipertexto son las actividades más afectadas por la adopción de WebML. Incluso, aplicaciones para el manejo de contenidos y publicación de datos tienen ciertas peculiaridades que pueden ser explotadas en el diseño de datos. Reconocerlas puede ayudar al diseñador a organizar su trabajo de una manera más sistémica, lo que normalmente resultaría en esquemas de datos más consistentes. Por lo tanto el método refuerza los distintos roles desempeñados por los objetos y utiliza esto para proponer una secuencia de pasos para ensamblar el esquema de datos de una aplicación Web. (Stefano, C. y Fraternali, P., 2003)

Los principales modelos incluidos en la metodología de diseño WebML son:

- ❖ Modelo de Datos.
- ❖ Modelo de Hipertexto.
- ❖ Modelo de Presentación.

Modelo de Datos (Modelo de Estructura + Modelo de Derivación) (Stefano, C. y Fraternali, P., 2003)



El modelo de datos de WebML es una adaptación conveniente de los modelos conceptuales de diseño de datos que se emplea en otras disciplinas como diseño de bases de datos, ingeniería de software y representación del conocimiento. El modelo de datos de WebML es compatible con el modelo de datos Entidad – Relación usado en el diseño conceptual de bases de datos, también es compatible con los diagramas de clase UML empleados en el modelado orientado a objetos.

El elemento fundamental del modelo de datos son las entidades, definidas como contenedores de elementos de datos, y sus relaciones definidas como las conexiones semánticas entre entidades. Las entidades tienen propiedades, llamadas atributos, con un tipo asociado. Las entidades pueden ser organizadas de manera jerárquica y sus relaciones pueden restringirse por medio de la cardinalidad.

Modelo de Hipertexto (Navegación + Composición) (Stefano, C. y Fraternali, P., 2003)

El modelo de hipertexto especifica como se compone el sitio y la navegación en el sitio. La composición del sitio describe las páginas que forman parte del hipertexto y las unidades de contenido que constituyen cada página. Las páginas de un sitio Web son catalogadas como contenedores de información que es enviada al usuario. Las unidades son elementos atómicos de contenido empleadas para publicar información descrita en el modelo de datos. WebML contiene siete (7) tipos de unidades predefinidas para desarrollar páginas Web: data, multi-data, index (y sus variantes jerárquicas y selección múltiple), entry, scroller. Cada unidad está asociada a una entidad subyacente, de la cual se obtiene el contenido. La especificación de una entidad subyacente determina el tipo de objeto del cual se deriva el contenido de la unidad.



La navegación del sitio se realiza a través de enlaces, los cuales se definen entre unidades que se encuentran en una misma página, en diferentes páginas o entre páginas completas. La información transportada a través de los enlaces se conoce como contexto de navegación o simplemente contexto. Los enlaces que transportan información contextual se denominan enlaces contextuales mientras que los que no transportan información son conocidos como enlaces no contextuales. La información contextual generalmente es necesaria para asegurar las operaciones de computación de las unidades.

Modelo de Presentación

Define como lucirá la vista del sitio. WebML incluye un modelo simple de presentación que permite colocar contenidos dinámicos en la página además de aplicar estilos distintos para cada uno.

A continuación la Figura 2.13 muestra la estructura de un sitio Web modelado haciendo uso de WebML.

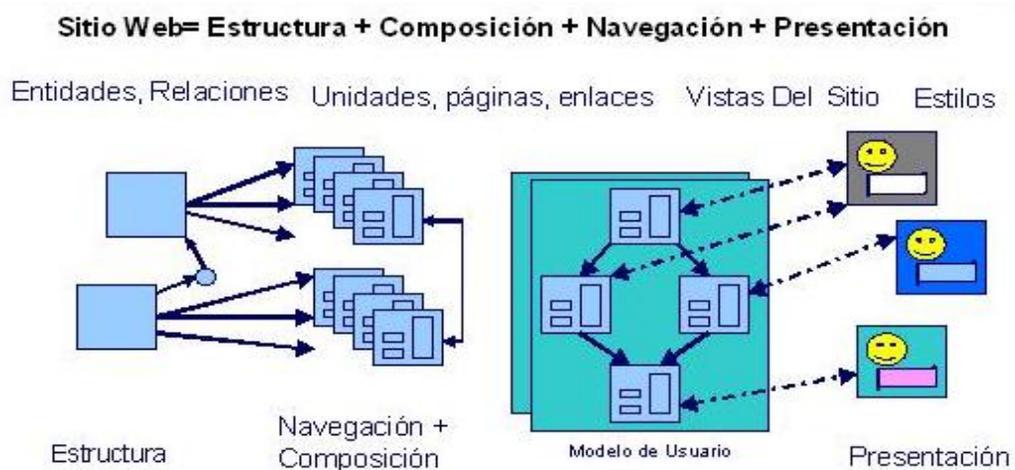


Figura 2.13: Sitio Web haciendo uso de los conceptos WebML.

Fuente: http://www.webml.org/webml/upload/webml_training1_introduction.pdf

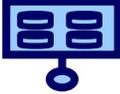


Una observación muy importante es el hecho de que WebML no es el mejor enfoque para sitios Web estáticos o pequeños. (Stefano, C y Fraternali, P. 2003. p.259)

Elementos del Modelo de Hipertexto WebML

La siguiente tabla muestra la simbología utilizada por los diseñadores para realizar el diseño de hipertexto durante el proceso de modelado del sistema.

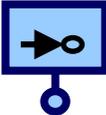
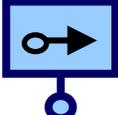
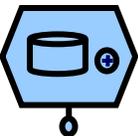
Tabla 2.1: Elementos del Modelo de Hipertexto WebML (1/5).

Elementos WebML	Descripción Breve	Propiedades
Data Unit (Unidad de Datos) 	La unidad de datos publica un solo objeto obtenido de una entidad determinada.	<ul style="list-style-type: none">▪ Nombre.▪ Entidad Fuente.▪ Selector (opcional).▪ Atributos Incluidos.
Multidata Unit (Unidad de Datos Múltiple) 	La unidad de datos múltiples presenta múltiples objetos de una entidad juntos, repitiendo la presentación de algunas unidades de datos.	<ul style="list-style-type: none">▪ Nombre.▪ Entidad Fuente.▪ Selector (opcional).▪ Atributos Incluidos.▪ Cláusula de Orden (opcional).
Index Unit (Unidad Índice) 	Una unidad índice presenta objetos múltiples de una entidad como una lista.	<ul style="list-style-type: none">▪ Nombre.▪ Entidad Fuente.▪ Selector (opcional).▪ Atributos Incluidos.▪ Cláusula de Orden (opcional).
Multi-choice index Unit (Unidad índice de múltiples elecciones) 	Es una variante de la unidad de índice, donde cada elemento de la lista esta asociado con un checkbox; permitiendo a los usuarios seleccionar múltiples objetos.	<ul style="list-style-type: none">▪ Nombre.▪ Entidad Fuente.▪ Selector (opcional).▪ Atributos Incluidos.▪ Cláusula de Orden (opcional).
Scroller-Unit (Unidad de desplazamiento) 	Una unidad de desplazamiento provee comandos para desplazar los objetos en un escenario.	<ul style="list-style-type: none">▪ Nombre.▪ Entidad Fuente.▪ Selector (opcional).▪ Bloque de factores.▪ Cláusula de Orden (opcional).

Fuente: http://www.webml.org/webml/upload/ent17/1/webml_elements.pdf



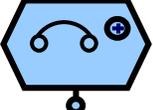
Tabla 2.1: Elementos del Modelo de Hipertexto WebML (2/5).

Elementos WebML	Descripción Breve	Propiedades
Hierarchical index Unit (Unidad índice jerárquica) 	Es una variante de la unidad de índice, en la cual las anotaciones de índice están organizadas en un árbol multinivel.	<ul style="list-style-type: none">▪ Nombre.▪ Para cada nivel:<ul style="list-style-type: none">▪ Entidad Fuente.▪ Selector (opcional).▪ Atributos Incluidos.▪ Cláusula de Orden (opcional).
Entry Unit (Unidad de entrada) 	Esta unidad de entrada soporta el ingreso de datos basado en un formulario.	<ul style="list-style-type: none">▪ Nombre.▪ Para cada campo:<ul style="list-style-type: none">▪ Nombre.▪ Tipo.▪ Valor inicial.▪ Predicado de validez.
Global Parameter (Parámetro Global)	Almacena información disponible para múltiples paginas.	<ul style="list-style-type: none">▪ Nombre.▪ Tipo.▪ Valor por defecto.
Set Unit (Unidad de Asignación) 	Asigna valores a un parámetro global.	<ul style="list-style-type: none">▪ Parámetro Global.
Get Unit (Unidad de Extracción) 	Recupera el valor de un parámetro global.	<ul style="list-style-type: none">▪ Parámetro Global.
Create Unit (Unidad de creación) 	Permite la creación de un nuevo ejemplo de entidad (instancias).	<ul style="list-style-type: none">▪ Nombre.▪ Entidad Fuente.▪ Conjunto de valores asignados.
Delete Unit (Unidad de borrado) 	Permite eliminar uno más objetos de una entidad determinada.	<ul style="list-style-type: none">▪ Nombre.▪ Entidad Fuente.▪ Selector.

Fuente: http://www.webml.org/webml/upload/ent17/1/webml_elements.pdf



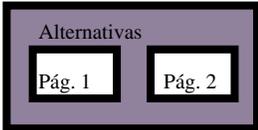
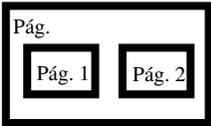
Tabla 2.1: Elementos del Modelo de Hipertexto WebML (3/5).

Elementos WebML	Descripción Breve	Propiedades
Modify Unit (Unidad de modificación) 	Actualiza uno o más objetos de una entidad dada.	<ul style="list-style-type: none">▪ Nombre.▪ Entidad Fuente.▪ Selector.▪ Conjunto de valores asignados.
Connect Unit (Unidad de conexión) 	Crea nuevas instancias de una relación.	<ul style="list-style-type: none">▪ Nombre.▪ Papel de la relación.▪ Selector de entidad fuente.▪ Selector de entidad objetivo.
Disconnect Unit (Unidad de Desconexión) 	Elimina instancias de una relación.	<ul style="list-style-type: none">▪ Nombre.▪ Papel de la relación.▪ Selector de entidad fuente.▪ Selector de entidad objetivo.
Login Unit (Unidad de entrada al sistema) 	Verifica la entidad de un usuario accediendo al sitio.	Parámetro: <ul style="list-style-type: none">▪ Nombre de usuario.▪ Clave.
Logout Unit (Unidad de cierre de sesión) 	La unidad de salida del sistema envía el usuario a una página por defecto, sin control de acceso.	<ul style="list-style-type: none">▪ Nada.
Change Group Unit (Unidad de cambio de grupo) 	Verifica la entidad de un usuario accediendo al sitio por medio de otro grupo de usuarios.	Parámetro: <ul style="list-style-type: none">▪ Nombre de usuario.▪ Clave.
Sendmail Unit (Unidad de envío de correo o mensajes) 	Suministra la capacidad enviar los mensajes de correo.	Parámetros: <ul style="list-style-type: none">▪ Remitente.▪ Receptor.▪ Asunto.▪ Cuerpo del mensaje.▪ Adjuntos.

Fuente: http://www.webml.org/webml/upload/ent17/1/webml_elements.pdf



Tabla 2.1: Elementos del Modelo de Hipertexto WebML (4/5).

Elementos WebML	Descripción Breve	Propiedades
Generic Operation Unit (Unidad de Operación Genérica) 	Define una operación genérica: la contribución y el producto. Por el que los parámetros deben ser definidos el diseñador.	Definidos por el diseñador.
Transaction (Transacción) 	Una transacción es una secuencia de las operaciones ejecutadas automáticamente. Las operaciones individuales se ejecutan con éxito, o la secuencia entera está pendiente.	<ul style="list-style-type: none">Nada.
Page (Pagina) 	Representan la actual interfaz buscada por el usuario. Esta contiene sub-paginas AND/OR.	<ul style="list-style-type: none">Nombre.Marca.Contenido: unidades, paginas and/or.
OR sub-pages (Sub-Páginas OR (alternativo)) 	Cierta porción de la pantalla puede contener piezas alternativas de contenido, cada una modelada como una página distinta. Las sub-páginas OR están contenidas en una página o en una sub-página.	<ul style="list-style-type: none">Animación de páginas.Paginas anidadas por defecto.
AND Sub-Pages (Sub-Páginas AND) 	Son usadas para dividir el contenido de la página en una pantalla en partes. Las sub-páginas AND están contenidas en una página o en una sub-página.	<ul style="list-style-type: none">Anidación de Páginas.
Área 	Es un contenedor de páginas o recursivamente de sub-áreas, la cual cada una puede ser usada para obtener una organización jerárquica del hipertexto.	<ul style="list-style-type: none">Nombre.MarcaContenido: páginas, sub-áreas.Página por defecto o subárea.
Vista del Sitio (site view) 	Representa una vista del hipertexto.	<ul style="list-style-type: none">Nombre.Contenido: páginas, áreas.Página de inicio.

Fuente: http://www.webml.org/webml/upload/ent17/1/webml_elements.pdf



Tabla 2.1: Elementos del Modelo de Hipertexto WebML (5/5).

Elementos WebML	Descripción Breve	Propiedades
<p>Link (Enlace)</p> <p>Parámetros →</p> <p>- Automático</p> <p>Parámetros →</p> <p>- Transporte</p> <p>Parámetros →</p> <p>- OK Link (enlace)</p> <p>Parámetros →</p> <p>- KO Link (enlace)</p> <p>Parámetros →</p> <p>- KO Link (enlace)</p>	<p>Un enlace es orientado a la conexión entre dos unidades o paginas. Permite transportar información por medio de sus parámetros. Pueden ser definidos como:</p> <ul style="list-style-type: none">Automáticos: se ejecutan sin necesidad de la intervención del usuario.Transporte: permiten el paso de un parámetro. <p>Enlaces en los que existen operaciones distinguidas:</p> <ul style="list-style-type: none">Enlaces OK: se ejecutan en caso de que la operación haya sido exitosa.Enlaces KO: se ejecutan en caso de que ocurra una falla.	<p>Enlaces normales, automáticos y de transporte:</p> <ul style="list-style-type: none">Nombre.Fuente de Elemento (unidad o página).Elemento destino.Tipo de enlace (normal, automático y transporte).Parámetros de enlace. <p>Parámetros de enlace:</p> <ul style="list-style-type: none">Nombre.Valor Fuente. <p>Enlaces OK/KO:</p> <ul style="list-style-type: none">Nombre.Elemento Fuente (unidad de operación).Elemento destino.Parámetros de enlace.

Fuente: http://www.webml.org/webml/upload/ent17/1/webml_elements.pdf

2.2.12.3 WebML Vs. UML

Desde el punto de vista del desarrollo de proyectos, existen claras diferencias entre una aplicación Web y una tradicional. (Ver Tabla 2.2)

Las diferencias entre estos tipos de proyectos son de vital importancia en la selección de los perfiles del equipo de trabajo, en la estimación del tiempo de desarrollo y de costo. Es muy difícil, o casi imposible, utilizar las mismas métricas en ambos tipos de proyectos.

Un enfoque que parece razonable, desde el punto de vista de nuevas tendencias de desarrollo, parece ser la separación de un proyecto Web en dos sub-proyectos: uno referido a la funcionalidad de la aplicación y otro referido al diseño gráfico y al contenido. (Stefano, C. y Fraternali, P., 2003)



El primer sub-proyecto se puede atacar utilizando la experiencia y metodología del desarrollo tradicional de aplicaciones (UML). El segundo proyecto, enfocado en la interfaz de la aplicación (diseño, gráfico y contenido), debe ser realizado utilizando paradigmas y metodologías no tradicionales de la ingeniería de software (WebML). El tipo de personal, así como la estimación de costos y tiempo, varía para cada sub-proyecto. (Stefano, C. y Fraternali, P., 2003)

UML es aceptado como lenguaje estándar de modelado para sistemas de software y por consiguiente como mejor opción para el modelado de diseños de aplicaciones Web. Sin embargo, el gran consumo de tiempo en el proceso de modelado, unido a la gran experiencia que requiere el profesional en esta área junto a la necesidad de mostrar una visión detallada para páginas Web más complejas, despertó el interés de los desarrolladores en crear e implementar nuevas opciones que permitan un modelado igualmente eficiente y eficaz como el que brinda UML, pero de una manera más fácil y rápida, es por ello que surge el WebML como una herramienta especializada en el modelado de aplicaciones Web. (Stefano, C. y Fraternali, P., 2003)

WebML es un lenguaje completamente nuevo, más fácil de comprender que UML y por lo tanto más sencillo de implementar, consta de solamente cuatro modelos mientras UML cuenta con doce. WebML fue desarrollado específicamente para el modelado de aplicaciones Web al contrario que UML, el cual se desarrollo para ser utilizado en el más amplio rango de aplicaciones de acuerdo con las Especificaciones de Lenguaje Unificado de Modelado de OMG (de sus siglas en inglés, Grupo de Gestión de Objetos, consorcio dedicado al cuidado y el establecimiento de diversos estándares de tecnologías orientadas a objetos). (Stefano, C. y Fraternali, P., 2003)



Al realizar una comparación entre ambos lenguajes se determinó que el modelo estructural de WebML es similar y compatible al modelo conceptual y al diagrama de clases de UML, esto es un aspecto positivo, ya que si se está familiarizado con la notación de UML es más natural comprender cómo trabaja la notación de WebML. Igualmente se concluyó que los modelos más importantes de WebML son el Modelo de Datos y el Modelo de Hipertexto; este último permite describir la estructura y diseño de una página Web mejor que cualquier modelo del lenguaje UML, mostrando de cuáles páginas consta la aplicación, qué contienen y cómo están vinculadas entre sí. (Stefano, C. y Fraternali, P., 2003)

Otra característica resaltante en el modelo de hipertexto es que muestra cuándo los datos son enviados entre páginas facilitando la visualización de su contenido. Dichas características dan una visión general de la aplicación Web, logrando una mayor eficiencia en el trabajo de los desarrolladores pues cuentan con una buena estructura a seguir. WebML proporciona habilidad de dirigir aplicaciones Web cuando crecen en tamaño y complejidad. (Stefano, C. y Fraternali, P., 2003)

Sin embargo, UML tiene una característica que lo ha mantenido vigente en la actualidad, esta es la capacidad de hacer un modelado de la aplicación Web desde la perspectiva de los usuarios, mediante diagramas de casos de uso. Estos son muy útiles para indicar la interacción entre el usuario y el sistema, por lo tanto, juegan un papel muy importante cuando se debe explicar al cliente cómo trabaja la aplicación Web deseada. (Stefano, C. y Fraternali, P., 2003)

En un principio WebML carecía de esa capacidad, actualmente WebML adoptó los diagramas de casos de uso UML como parte principal y primordial para lograr una correcta recolección y definición de los requisitos y requerimientos necesarios para el desarrollo de la aplicación, entre los cuales se encuentran: la identificación de los usuarios, los diferentes niveles de acceso con los que contarán y los tipos de



mantenimiento a realizarse a la aplicación para lograr su correcto funcionamiento. (Stefano, C. y Fraternali, P., 2003)

Finalmente otra de las ventajas de WebML esta en el modelo de hipertexto el cual da una visión detallada de la estructura y diseño de la(s) aplicación(es) Web a desarrollar, por ello y por lo anteriormente expuesto, WebML se a convertido en una de las mejores opciones de modelado para una empresa de desarrollo de software ya que al lograr una buena perspectiva general de la(s) aplicación(es) Web a desarrollar se obtiene una excelente comunicación con el clientes. A continuación se muestra una tabla comparativa entre un desarrollo tradicional y un desarrollo Web para una aplicación. (Stefano, C. y Fraternali, P., 2003)

Tabla 2.2: Desarrollo Web Vs. Desarrollo Tradicional (1/2).

Características	Desarrollo Tradicional	Desarrollo Web
Objetivo Primario	Productos de calidad (mín. costo)	Productos de calidad, al mercado lo mas rápido posible.
Tamaño típico del producto	Mediano a grande (equipos de cientos de miembros).	Pequeños (equipos de tres a cinco miembros).
Tiempo de desarrollo	De diez a dieciocho meses	De tres a seis meses.
Enfoque de desarrollo	Clásico basado en requisitos, entregas incrementales, casos de uso y documentación.	Desarrollo rápido de aplicaciones (RAD), agrupar bloques de construcción, prototipos, RUP.
Tecnologías de Ingeniería usadas	Orientación a objetos, lenguajes modernos, etc.	Métodos basados en componentes, lenguajes de cuarta y quinta generación, visualización, etc.
Procesos	Basados en CMM (Modelo de Madurez para la Capacidad de desarrollo de software).	AD HOC (específicamente para cada proceso).
Desarrollo de productos	Sistemas basados en código, re-uso, muchas interfaces externas, algunas aplicaciones complejas.	Sistema basado en objetos, componentes reutilizables, pocas interfaces externas, aplicaciones relativamente simples.

Fuente: Reifer, D. 2000.



Tabla 2.2: Desarrollo Web Vs. Desarrollo Tradicional (2/2).

Características	Desarrollo Tradicional	Desarrollo Web
Personal involucrado	Ingenieros de software profesionales, con cinco o más años de experiencia en al menos dos dominios de aplicaciones.	Diseñadores gráficos, ingenieros con poca experiencia (dos o mas años), ingenieros recién graduados
Tecnologías de estimación	Uso de datos históricos, modelos basados en puntos por función, Estructura de Composición de Trabajos para proyectos pequeños.	Uso de la actual experiencia, diseño ajustable basado en recursos disponibles, WBS para proyectos pequeños.

Fuente: Reifer, D. 2000.

2.2.13 Servidor Web Apache

Apache es un servidor web de código libre robusto cuya implementación se realiza de forma colaborativa, con prestaciones y funcionalidades equivalentes a las de los servidores comerciales. El proyecto está dirigido y controlado por un grupo de voluntarios de todo el mundo que, usando Internet y la web para comunicarse, planifican y desarrollan el servidor y la documentación relacionada.

Estos voluntarios se conocen como el Apache Group. Además del Apache Group, cientos de personas han contribuido al proyecto con código, ideas y documentación. (Mateu, C. 2004, p.29)

2.2.14 Software Libre (SL)

Es aquel software, producto o desarrollo a medida, que se distribuye bajo una licencia, según la cual el autor cede una serie de libertades básicas al usuario en el marco de un acuerdo de concesión. Se trata de cuatro libertades de los usuarios del software recogidas en la filosofía de la Fundación para el Software Libre (Free Software Foundation), en particular: la libertad de usar el programa con cualquier propósito; la libertad de estudiar como funciona el programa y adaptarlo a sus necesidades; la libertad de distribuir copias; y la libertad de mejorar el programa y



hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie.
(Martínez, J.2007. pg. 35-36)

CAPÍTULO III

FASE DE INICIO

3.1 Introducción

El Proceso Unificado de desarrollo de software tiene como meta asegurar la producción de software de muy alta calidad que satisfaga las necesidades de los usuarios finales, en esta fase el propósito es realizar el análisis de los requerimientos solicitados por la empresa hasta lograr justificar la realización del proyecto.

Para un correcto desarrollo del proyecto, denominado **“DESARROLLO DE UN SISTEMA AUTOMATIZADO PARA FACTURACIÓN Y RECOBRO DE SERVICIOS TELEFÓNICOS ASOCIADO A LA GERENCIA AIT, PDVSA MATURÍN, BASADO EN TECNOLOGÍA WEB Y ESTÁNDARES ABIERTOS”**, es necesario seguir una serie de pautas y procedimientos, los cuales de no realizarse de manera organizada, pueden traer errores con un alto impacto en el producto final. Por estas razones se ha elaborado una planificación dividida en varias etapas basadas en el modelo del Proceso Unificado de Desarrollo de Software: Inicio, Elaboración, Construcción y Transición. En cada una de ellas se realizan los procedimientos y técnicas correspondientes de tal manera que su ejecución ofrece las herramientas necesarias para proceder con la siguiente fase.

La fase de inicio no es un estudio completo del sistema propuesto, sino que, en ella se busca el porcentaje de casos de uso necesarios para fundamentar el análisis inicial. Para realizar este estudio se siguen cuatro pasos:

- ❖ Delimitar el ámbito del sistema propuesto, es decir, definir los límites del sistema, definir los requerimientos e identificar las interfaces con sistemas relacionados que están fuera de los límites.



- ❖ Describir una propuesta de la arquitectura del sistema (en especial en aquellas partes que son nuevas, arriesgadas o difíciles).
- ❖ Identificar riesgos críticos (los que afectan la capacidad de construir el sistema) y determinar si podemos encontrar una forma de mitigarlos, quizás en una etapa posterior.
- ❖ Demostrar a usuarios o clientes potenciales que el sistema propuesto es capaz de solventar sus problemas o de mejorar sus objetivos de negocio construyendo un prototipo.

3.2 Flujo de trabajo requisito

La captura de requisitos es el proceso de averiguar, normalmente en circunstancias difíciles, lo que se debe construir. El flujo de trabajo requisitos tiene como propósito guiar el desarrollo del proyecto hacia el sistema correcto hasta obtener una arquitectura sólida. Se basa en la comprensión del contexto del sistema usando el diagrama de modelo de dominio o el de negocio, requerimientos funcionales, lista de riesgos, modelos de casos de uso, identificación de actores, descripción de casos de uso y requerimientos adicionales. Todo esto conlleva a que se pueda llegar a un acuerdo entre el cliente (incluyendo los usuarios) y los desarrolladores del sistema, acerca de lo que el sistema debe hacer y lo que no.

3.2.1 Comprender el contexto del sistema

Para el estudio del contexto del sistema es necesario comprender las relaciones entre el software que se está diseñando y el entorno externo. Comprender esto ayuda a decidir cómo suministrar la funcionalidad requerida al sistema y cómo estructurar éste para que se comunique efectivamente con su entorno.



Para comprender el contexto del sistema fue necesario conocer, estudiar y analizar las actividades relacionadas por el proceso **Cadena de Suministros** específicamente por **Provisión de Bienes y Servicios** perteneciente a la Gerencia de AIT., distrito norte de PDVSA S.A., para ello fue necesario realizar lo siguiente:

- ❖ La observación es la más común de las técnicas de recolección de datos, esta se fundamenta en el reconocimiento visual de los acontecimientos o fenómenos correspondientes al sistema actual objeto de estudio. Esta técnica se utilizó para conocer el funcionamiento de los procesos relacionados con el recobro de la facturación telefónica de la gerencia.
- ❖ Se realizaron entrevistas de tipo no estructuradas con el fin de conocer el funcionamiento y necesidades de los procesos de recobro de facturación telefónica, así como también de propuestas para la mejora de dichos procesos, todo esto con el fin de canalizar el flujo de requerimientos para el desarrollo de la aplicación.

Todo esto para identificar las posibles debilidades del sistema actual y así tratar de corregirlas en el sistema en desarrollo, para generar una aplicación que cubra con toda la información obtenida y minimizar las posibles debilidades que el sistema pudiera presentar a futuro. Una vez estructuradas las informaciones necesarias se consiguió una aproximación del contexto del sistema representado mediante del modelo de dominio, describiendo los conceptos importantes del contexto como objeto del dominio y enlazando estos objetos entre sí.

3.2.1.1 Modelo de dominio

El modelo del dominio captura los aspectos más importantes en el contexto del sistema y los representa como objetos o clases del negocio. Estos objetos representan



las “cosas” que existen actualmente o los eventos que tienen lugar en el entorno donde se desenvuelve el sistema.

El modelo de dominio puede utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis, dicho análisis es utilizado por el analista como un medio para comprender el sector industrial o de negocios al cual el sistema va a servir.

El modelo de dominio puede ser tomado como el punto de partida para el diseño del sistema. Esto es así ya que cuando se realiza la programación, se supone que el funcionamiento interno del software va a imitar en alguna medida a la realidad, por lo que el mapa de conceptos del modelo de domino constituye una primera versión del sistema.

Para describir el modelo de dominio nos apoyaremos en una herramienta de diseño que nos permite su fácil entendimiento, esta es los diagramas de clases basado en Lenguaje de Modelado UML.

En la Figura 3.1 se muestra el diagrama de modelo de dominio parcial del sistema actual, cabe destacar que éste no representa ninguna funcionalidad del sistema, sino los objetos más importantes y las asociaciones entre estos objetos.

Este modelo de dominio representa el funcionamiento actual del proceso de recobro, veamos el funcionamiento de este proceso manual: la gerencia respectiva realiza una transacción de servicio, ya sea para cancelar o solicitar un teléfono o una tarjeta, el analista de bienes y servicios recibe esta transacción para su estudio, luego de su estudio envía el servicio a prestar al gerente de la gerencia que recibe el servicio.

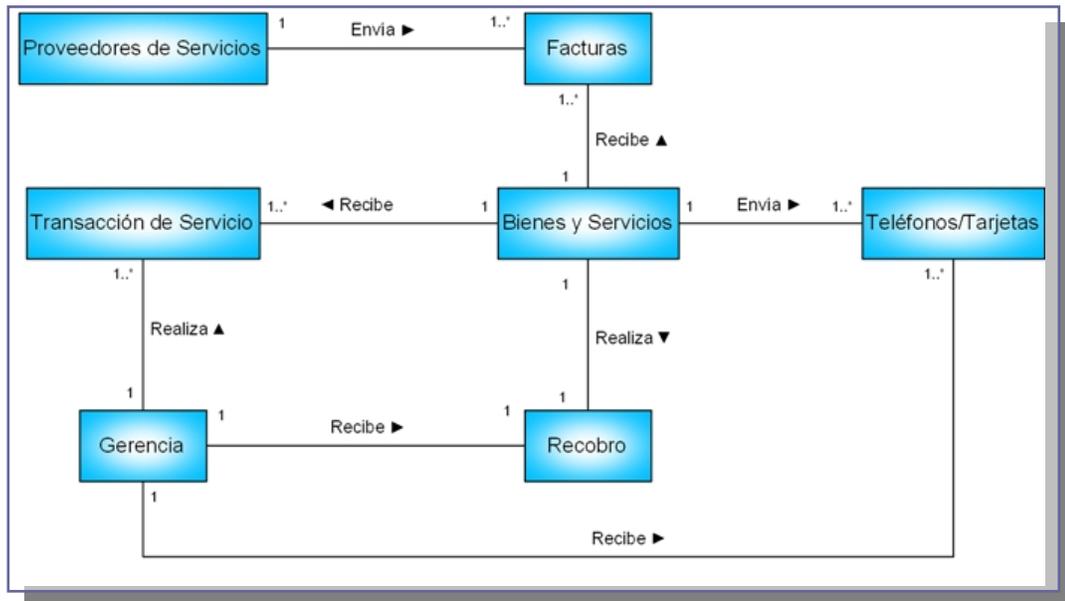


Figura 3.1: Modelo de Dominio.
Fuente: Elaboración Propia.

Los proveedores de servicios envían las facturas, el analista de bienes y servicios las recibe y realiza el proceso de distribución de montos para cada gerencia esto lo logra por medio de archivos de Excel, una vez finalizado este proceso envía el recobro que recibe el gerente de la gerencia respectiva.

Así concluye el proceso manual que conlleva a retardos en los tiempos de respuesta para el determinado recobro mensual de la gerencia.

Glosario de Terminos

- ❖ **Proveedores de Servicios:** Empresas que se encargan de brindar los servicios telefónicos a la corporación.
- ❖ **Facturas:** En esta se especifica toda la información referente a los números telefónicos asociados a cada *Proveedor de Servicio*.
- ❖ **Bienes y Servicios:** Es el sub. proceso de Cadenas de Suministro que se encarga de satisfacer las necesidades de bienes y servicios tecnológicos.



- ❖ **Teléfonos / Tarjetas:** Son los equipos que suministra bienes y servicios a las diferentes organizaciones de la corporación.
- ❖ **Recobro:** Son los montos que bienes y servicios le pasa a cada una de las organizaciones a las que le presta el servicio telefónico.
- ❖ **Gerencia:** Son todas las organizaciones que forman parte de la corporación a las que se la suministra el servicio telefónico.
- ❖ **Transacción de Servicio:** Son las ordenes que generan las organizaciones para la solicitud del servicio telefónico.

3.2.2 Colección de requisitos

El punto central de la aplicación Web a desarrollar debe estar orientado a mejorar las actividades de recobro telefónico de la gerencia de AIT PDVSA distrito norte, y a disminuir el tiempo empleado para la realización de éste, además de su eficiencia, Confiabilidad y seguridad.

El punto inicial en el desarrollo de software es la descripción de los Requerimientos del Sistema, los cuales moldean las funcionalidades que demandan los futuros usuarios de la aplicación. Por ello es de interés establecer desde un principio el alcance, punto de partida y la información a manipular en él. Por otra parte es importante estudiar las posibles estrategias a implementar, para así escoger la que permita a la aplicación Web disponer de un ambiente de trabajo rápido y eficaz.

La obtención de los requisitos es de vital importancia para poder llevar a cabo la creación de cualquier sistema, bien sea en ambiente Web o tradicional, se realiza haciendo uso de los siguientes diagramas o sub-etapas correspondientes. Estos son:

- ❖ Especificación de los riesgos del sistema.
- ❖ Identificación de Usuarios (Actores del Sistema).



- ❖ Colección de Requisitos Funcionales (Modelos de caso de usos).
- ❖ Requisitos de dispositivos específicos.
- ❖ Requisitos no Funcionales.

3.2.2.1 Alcance

Como se menciona anteriormente para concretar una buena base para el inicio de cualquier sistema es necesario conocer cual será el alcance de éste, por tal sentido es necesario conocer el alcance del Sistema de Recobro de Facturación Telefónica (SAFREST).

El SAFREST, será desarrollado utilizando estándares abiertos y herramientas de software libre cumpliendo con los lineamientos del Decreto Presidencial 3390.

Se basará en una herramienta automatizada que permitirá llevar el control del recobro de servicio telefónico asociado a la Gerencia de AIT Dtto. Norte y que contará con las funcionalidades de Monitoreo de la Información, Carga y actualización de datos, Validación de Datos, Generación de Reportes, Búsqueda de datos y Administración del Sistema.

3.2.2.2 Posicionamiento

Para comprender el posicionamiento del producto vamos a desglosarlo en dos partes que son las oportunidades del negocio y la declaración de posición del producto.

Oportunidades de negocio, el desarrollo del SAFREST presenta las siguientes oportunidades de negocio:

- ❖ Reducir los tiempos de procesamiento de información.
- ❖ Monitorear continuamente la información de interés.



- ❖ Reducir los riesgos de pérdida de información.
- ❖ Aumentar la confiabilidad y consistencia de la data procesada.
- ❖ Mayor disponibilidad, seguridad y fácil acceso a la información.
- ❖ Incrementar el nivel de productividad del personal, al disponer éste de menos tiempo para el análisis de la información, la cual se obtiene de manera oportuna y con calidad.
- ❖ Contribución con el proceso de migración hacia el software libre basado en el Decreto Presidencial 3390.
- ❖ Actualización en la utilización de herramientas de desarrollo de software, adaptándose a las necesidades que exigen los nuevos tiempos.

Declaración de posición del producto, define la importancia de implementación de la aplicación dentro de la organización, realizando un bosquejo detallado de los beneficios que aportaría:

Para: Empleados de la gerencia de AIT distrito norte de PDVSA.

Quienes: Necesitan una herramienta Web que permita el control y gestión del recobro de servicios telefónicos.

El Software: Es una aplicación Web desarrollada usando estándares abiertos bajo el enfoque de software libre.

Que: Permitirá llevar el control del recobro de servicios telefónicos de manera eficiente y confiable facilitando la generación de reportes.

No como: El manejo actual del recobro de servicios telefónicos que se lleva de forma convencional por medio de Archivos Excel.



Nuestro producto: Permitirá contribuir con el proceso de recobro de servicios telefónicos llevado a cabo en la Gerencia de AIT Dtto. Norte, además de contribuir con el proceso de migración hacia el software libre establecido en el Decreto Presidencial 3390.

3.2.3 Riesgos del sistema

Todo desarrollo de software tiende a presentar una serie de riesgos que de no ser solucionados a tiempo, trae como consecuencia problemas posteriores que con el transcurrir del tiempo son más difíciles de solucionar, razón por la cual es necesario tomar en cuenta dichos riesgos, desde los inicios del desarrollo del sistema.

Por tal razón es necesario la ejecución de un plan de administración de riesgos que será el mecanismo de prevención de riesgos que permitirá identificar, estudiar y eliminar las fuentes de riesgos antes de que empiecen a amenazar la finalización satisfactoria del proyecto SAFREST.

El propósito de este plan es determinar las estrategias para detección, análisis y jerarquización de riesgos bajo los siguientes pilares:

- ❖ Planificación de la Gestión de Riesgos: decidir cómo abordar y planificar las actividades de gestión de riesgos en el proyecto.
- ❖ Identificación de Riesgos: determinar cuales son los riesgos que pueden afectar al proyecto y documentar sus características.
- ❖ Análisis Cualitativo de Riesgos: realizar un análisis cualitativo de los riesgos y sus condiciones para priorizar sus efectos sobre los objetivos del proyecto.



- ❖ Planificación de Respuestas a los Riesgos: elaborar procedimientos y técnicas para oportunidades de mejora y reducir las amenazas a los objetivos del proyecto.
- ❖ Supervisión y Control de Riesgos: supervisar los riesgos residuales, identificar nuevos riesgos, ejecutar planes de reducción de riesgos, y evaluar su efectividad a lo largo del ciclo de vida del proyecto.

El plan tendrá influencia sobre el proyecto general, además de establecer los riesgos individuales que pudiesen atentar contra el éxito del proyecto. Estos riesgos podrían ser técnicos, de conocimiento, de organización, etc.

La gestión de riesgos se realizará cumpliendo con las siguientes actividades:

- ❖ Mediante entrevistas con los involucrados (usuarios y equipo de proyecto) Análisis de los requerimientos verificando las variables que atentarían contra la salud del proyecto.
- ❖ Deducción de los riesgos para cada uno de los requerimientos en el contexto de cada uno de los subsistemas.
- ❖ Categorización de los riesgos en cuanto a probabilidad de ocurrencia y expectativas de pérdida en caso de ocurrencia.
- ❖ Establecimiento de las estrategias de administración de los riesgos (mitigación, esquivamiento o prevención) para cada uno de los riesgos más importantes.



Las tareas mencionadas deben ser llevadas principalmente por el Líder del Proyecto. En caso de materializarse alguno de los riesgos el Líder del Proyecto invoca los planes de gestión para el riesgo o grupo de riesgos con la finalidad de incluir las tareas de tratamiento dentro de las actividades de los equipos.

Se asume que la organización del proyecto tendrá previsto los recursos que haya que invertir para la mitigación de los riesgos. Los costos asociados dependerán del tipo de riesgo.

Teniendo en cuenta estos riesgos y con la finalidad de disminuirlos se desarrollan los casos de uso y diagramas de especificación del sistema subsiguientes, ya que a través de ellos se permite llegar a un acuerdo sobre los requisitos a utilizar y sobre las condiciones que debe cumplir el sistema.

Los posibles riesgos del sistema son los mostrados en la Tabla 3.1.

Tabla 3.1: Riesgos del Sistema (1/3).

Módulo	Descripción	Nivel	Responsable	Ejecución
H A R D W A R E	Bajo rendimiento del PC por insuficiencia de memoria RAM y/o poca capacidad de procesador	Crítico	PDVSA	Instalar memorias RAM y procesador de más capacidad.
	Tarjeta de red no funciona o no está instalada.			Instalar nueva tarjeta de red.
	Poco espacio en disco duro			Instalar disco duro de mayor capacidad.

Fuente: Elaboración Propia.



Tabla 3.1: Riesgos del Sistema (2/3).

Módulo	Descripción	Nivel	Responsable	Ejecución
D E S A R R O L L O	El cliente e involucrados no participan en los ciclos de revisión de los planes y especificaciones, resultando unos requisitos inestables	Crítico	Líder del Proyecto	Hacer reuniones periódicas referentes al proyecto, con el fin de incrementar al máximo la retroalimentación.
	Crecimiento no controlado de requerimientos y alcance – proyecto fuera de calendario y requerimientos.	No Crítico		Cualquier nuevo requerimiento que se constituya en un subsistema no indispensable para los ya previstos, debe considerarse para un nuevo proyecto.
	Requerimientos no capturados en forma clara y concisa, procesos con alto número de incrementos por corrección, lo que genera un estiramiento no deseado del calendario.	Crítico		Se deben establecer mecanismos de supervisión de requerimientos por parte de los Analistas y expertos del negocio, cuyas funciones se centrarían en ejecutar pruebas de desempeño funcional y aceptación
S O F T W A R E	Fallas al generar reportes	Crítico	Desarrollador	<ul style="list-style-type: none">▪ Revisar código del sistema.▪ Verificar configuración E/S.
	No se puede implementar la funcionalidad deseada con el lenguaje o bibliotecas utilizados	No Crítico		El desarrollador debe utilizar otras bibliotecas, o crearlas el mismo para conseguir la funcionalidad deseada

Fuente: Elaboración Propia.



Tabla 3.1: Riesgos del Sistema (3/3).

Módulo	Descripción	Nivel	Responsable	Ejecución
S O F T W A R E	Incorrecta manipulación del sistema	No Crítico	Desarrollador	<ul style="list-style-type: none">Entrenar a los usuarios del sistema.Crear menú de ayuda a usuario.
	No se establece comunicación entre el sistema y la base de datos	Muy Crítico	Desarrollador	<ul style="list-style-type: none">Revisar código del sistema.Verificar bases de datos.
	Insuficiente capacidad para almacenar datos	Crítico		Verificar espacio en HDD Probar otros manejadores de B.D.
	Menú de ayuda incomprensible.	No Crítico		Realizar ayuda con interfaz amigable y fácil de entender para el usuario.
I N T E R F A Z	Incompatibilidad con navegadores Web.	Crítico	Desarrollador	Probar la interfaz en distintos navegadores y buscar la mejor configuración que se adapte a ellos.
	No se comunican correctamente la interfaz y el sistema.	Muy Crítico		<ul style="list-style-type: none">Revisar Código del sistema.Revisar Código de interfaz.
S E R V I D O R	Imposibilidad de subir archivo grandes; necesarios para el correcto funcionamiento del sistema	Crítico	PDVSA	Otorgar permisos en el servidor para permitir el normal funcionamiento del sistema.

Fuente: Elaboración Propia.



Considerando estos riesgos y con la finalidad de disminuirlos se desarrollan los casos de uso y diagramas de especificación del sistema subsiguientes, ya que a través de ellos se permite llegar a un acuerdo sobre los requisitos a utilizar y sobre las condiciones que debe cumplir el sistema.

3.2.4 Identificación de usuarios (actores del sistema)

Se especifican los usuarios del sistema y su correspondiente nivel de acceso. Cada nivel está asociado a las actividades que pueden desarrollar dentro del sistema y hasta donde pueden llegar (mientras más alto sea el nivel, mayor acceso tienen), lo que será especificado en los diagramas subsiguientes. Los actores son:

Administrador: Actor genérico que modela aquellos actores que tienen la función de administrar el Sistema. La Administración abarca el manejo de los usuarios, los teléfonos y la actualización de la base de datos mensual, entre otras. El Administrador además de sus funciones específicas, también puede realizar las funciones que desempeña el actor Consultor, este actor puede ser representado por dos personas, por la persona que se encarga de todo el proceso de recobro y por el encargado del mantenimiento de la aplicación que le brinda siempre apoyo al anterior.

Consultor: Actor genérico que modela la línea de clientes a quienes prestan sus servicios de tecnologías. Los privilegios de este es la visualización de cada uno de los teléfonos pertenecientes a la gerencia asociada.

BD_SAFREST: Representa donde se almacenará la información más importante para el sistema y que se despliega en la aplicación. A través de su implementación se solventan muchos problemas existentes en la empresa además de suavizar los riesgos estudiados anteriormente.



LDAP: (Lightweight Directory Access Protocol) es un protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red. LDAP también es considerado una base de datos (aunque su sistema de almacenamiento puede ser diferente) al que pueden realizarse consultas. Habitualmente, almacena la información de login (usuario y contraseña) y es utilizado para autenticarse aunque es posible almacenar otra información (datos de contacto del usuario, ubicación de diversos recursos de la red, permisos, certificados, etc.).

3.2.5 Modelo de casos de uso

Los casos de uso han sido adoptados casi universalmente para la captura de requisitos de software en general, durante el desarrollo de este trabajo de grado tomarán una gran relevancia ya que esta herramienta se utilizará para dirigir el proceso de desarrollo en su totalidad.

Normalmente, un sistema posee muchos tipos de usuario. Cada usuario es representado por un actor, los actores utilizan el sistema interactuando con los casos de usos. Un caso de uso es una secuencia de acciones que el sistema lleva a cabo para ofrecer un resultado observable para un actor.

En lo sucesivo se dará un enfoque del sistema, por medio de la esquematización del proceso que constituye el sistema y los usuarios que con él interactúan.

La elipse representa el caso de uso, el actor, puede ser una persona u otro sistema, en nuestro caso sólo interactúan personas, el LDAP y BD_SAFREST. Ver Figura 3.2.

La gerencia AIT se ve beneficiada, puesto que a través del sistema se lleva a cabo el proceso de recobro de servicios telefónicos, de un modo más eficiente y



eficaz, lo cual se ve reflejado en una disminución de tiempo, proporcionando así una rápida respuesta en el reintegro de los cargos por motivo de servicios telefónicos, que ayudarán a la gerencia a reinvertir esos recursos en proyectos que ayuden a exaltar la imagen de la organización y sus trabajadores dentro de la empresa PDVSA y los Distritos que la conforman, aportando mayor valor agregado.

3.2.6 Descripción de casos de uso del sistema SAFREST

3.2.6.1 Caso de uso: Acceder al sistema

Actores: Consultor, Administrador, LDAP y BD_SAFREST.

Descripción: El caso de uso inicia cuando el usuario requiere ingresar a la aplicación SAFREST. Este ingresa su indicador y contraseña de intranet, los cuales son validados, con lo cual si resulta positiva tal validación este accede a dicha aplicación dentro de los roles que le corresponda o puede que le niegue la entrada al sistema, bien sea por no pertenecer a la base de datos del sistema o no ser empleado de PDVSA.

3.2.6.2 Caso de uso: Generar reporte

Actores: Consultor, Administrador y BD_SAFREST.

Descripción: El caso de uso inicia cuando los usuarios requieren visualizar alguno de los reportes del SAFREST. Dichos reportes pueden ser organizados por gerencias, teléfono, persona responsable y montos según la opción seleccionada.

3.2.6.3 Caso de uso: Consultar datos de teléfono

Actores: Consultor, Administrador y BD_SAFREST.



Descripción: El caso de uso inicia cuando los usuarios requieren consultar los datos asociados a una gerencia y los datos generales de un teléfono en particular, para esto solo seleccionan el teléfono del cual desean conocer sus detalles y automáticamente estos deben mostrarse.

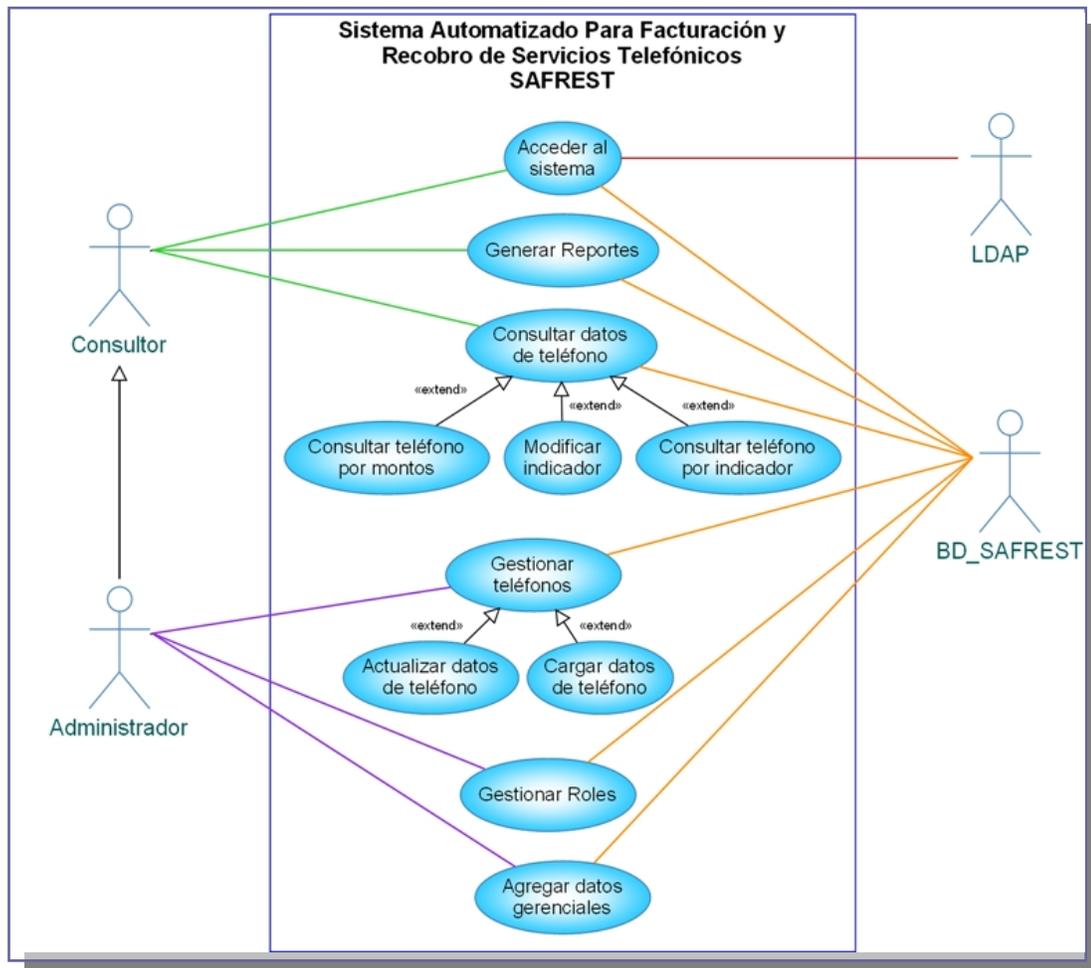


Figura 3.2: Diagrama de casos de uso general del sistema.

Fuente: Elaboración Propia.

3.2.6.4 Caso de uso: Consultar datos de teléfono

Actores: Consultor, Administrador y BD_SAFREST.



Descripción: El caso de uso inicia cuando los usuarios requieren consultar los datos asociados a una gerencia y los datos generales de un teléfono en particular, para esto solo seleccionan el teléfono del cual desean conocer sus detalles y automáticamente estos deben mostrarse.

3.2.6.5 Caso de uso: Consultar teléfono por montos

Actores: Consultor, Administrador y BD_SEFREST.

Descripción: El caso de uso inicia cuando el usuario requiere consultar los teléfonos por un monto en específico, por un rango de montos y por montos mayores y menores que, para esto selecciona consultar montos de teléfono y luego en la columna de montos seleccionan el sub menú filtro para realizar la consulta.

3.2.6.6 Caso de uso: Modificar indicador

Actores: Consultor, Administrador y BD_SEFREST.

Descripción: El caso de uso inicia cuando el usuario requiere modificar el indicador asociado a un teléfono, para esto se selecciona consultar datos de teléfono y luego en el campo de texto indicador se coloca el indicador a cambiar y se presiona guardar.

3.2.6.7 Caso de uso: Consultar teléfono por indicador

Actores: Consultor, Administrador y BD_SEFREST.

Descripción: El caso de uso inicia cuando el usuario requiere consultar los teléfonos asociados a un indicador específico, para esto se selecciona consultar datos de teléfono y luego en la columna indicador se selecciona el sub menú filtro se coloca el indicador para realizar la consulta.



3.2.6.8 Caso de uso: Gestionar teléfonos

Actores: Administrador y BD_SAFREST.

Descripción: El caso de uso inicia cuando el administrador requiere gestionar la información concerniente a los teléfonos, este debe seleccionar listar teléfono y luego se mostrará la interfaz con toda la información asociada a un teléfono.

3.2.6.9 Caso de uso: Cargar datos de teléfono

Actores: Administrador y BD_SAFREST.

Descripción: El caso de uso inicia cuando el administrador requiere añadir un teléfono, este debe seleccionar listar teléfono y luego agregar para después cargar toda la información asociada a un teléfono.

3.2.6.10 Caso de uso: Actualizar datos de teléfono

Actor: Administrador y BD_SAFREST.

Descripción: El caso de uso inicia cuando el administrador desea añadir o cambiar información referente a un teléfono, para realizar esta operación el administrador debe seleccionar el teléfono que desee luego en actualizar para introducir la información faltante.

3.2.6.11 Caso de uso: Gestionar roles

Actores: Administrador y BD_SAFREST.

Descripción: El caso de uso inicia cuando el administrador desea añadir un usuario o actualizar el rol existente de uno que ya se encuentre en el sistema, para



esto debe seleccionar roles de usuario y agregar para añadir un usuario nuevo con el rol que le corresponda.

3.2.6.12 Caso de uso: Agregar datos gerenciales

Actores: Administrador y BD_SAFREST.

Descripción: El caso de uso inicia cuando el administrador desea agregar datos relacionados con el área gerencial de la empresa tales como división, gerencias, distritos y localidades que no se encuentran en el sistema.

3.2.7 Requisitos de dispositivos específicos

Servidor: se recomienda, Procesador Intel Xeon secuencia 3000. Memoria RAM mayor o igual a 2 GB, 250 GB de Disco Duro SATA y Sistema Operativo Windows 2000 service pack 4.

Clientes: se recomienda, un procesador Pentium 4, 1,60 GHz o más, 512 MB de memoria de acceso aleatorio (RAM), un monitor de 14" mínimo, con capacidad para mostrar una resolución de 800 x 600 píxeles y una calidad de color de 32 bits.

El sistema operativo establecido es Ubuntu 7 y el software para Internet Mozilla Firefox 3.0.

3.2.8 Requisitos no funcionales

Los requerimientos no funcionales responden a las características en cuanto a funcionabilidad, usabilidad, confiabilidad, compatibilidad con hardware y software, especificaciones del producto, etc. Este tipo de requisitos tienen que ver con características que de una u otra forma puedan limitar el sistema, como por ejemplo,



el rendimiento (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema, disponibilidad de equipo), mantenimiento, seguridad, portabilidad, estándares.

Rendimiento: el tiempo de respuesta de la aplicación depende de las características físicas de los equipos y va de la velocidad de procesamiento de los computadores.

Usabilidad: interfaz amigable para las personas autorizadas que usarán la aplicación, la cual debe generar comodidad durante la navegación, siempre enfocado en las actividades del negocio empresarial.

Disponibilidad: la aplicación estará disponible en todo momento para los usuarios autorizados.

Seguridad: se implementan claves de acceso para el control de ingreso al del sistema logrando así la protección de la información.

Mantenimiento: el administrador del sistema con su clave de acceso deberá poder insertar, modificar, eliminar y probar módulos y procesos del sistema lo cual generará un correcto mantenimiento del mismo.

Estándares: el sistema es flexible y se puede acceder desde distintas plataformas computacionales con diferentes sistemas operativos y navegadores webs.

3.3 Flujo de trabajo análisis

Durante este flujo se analizan, refinan y estructuran los requerimientos que han sido identificados.



Luego de este proceso descrito se obtiene una primera impresión del modelo de diseño y así se logra estructurar el sistema entero incluyendo su arquitectura, lo que facilita una mejor comprensión, preparación, modificación, y en general, un mejor mantenimiento de estos requerimientos.

3.3.1 Análisis de requerimientos y/o requisitos

La revisión y formalización de requisitos recopilados es de suma importancia para obtener el conjunto de especificaciones necesarias para desarrollar la aplicación. Generalmente la revisión se realiza en términos de sub-etapas y sus diagramas correspondientes. Estas son:

3.3.2 Especificación de los grupos de usuarios formalmente descritos

En esta sub-etapa se identifica los usuarios que intervienen en cada uno de los procesos que forman los distintos casos de uso. La especificación en grupos de usuarios determina su ubicación dentro de la aplicación, permitiendo saber su nivel de acceso e importancia dentro de la misma, esto ayuda a disminuir los riesgos del sistema, así como también detalla que datos deben manejar cada uno de ellos y en que procesos están involucrados.

El actor Administrador posee los mismos beneficios del actor Consultor, además de otros únicos para él, esta herencia se puede observar en la Figura 3.3.

3.3.3 Especificación de los casos de usos por usuarios

En este punto se realiza la descripción formal de las unidades de interacción con la aplicación, de los usuarios o actores obtenidos en los grupos ya definidos, por medio de los casos de uso, tablas o diagramas de actividades.

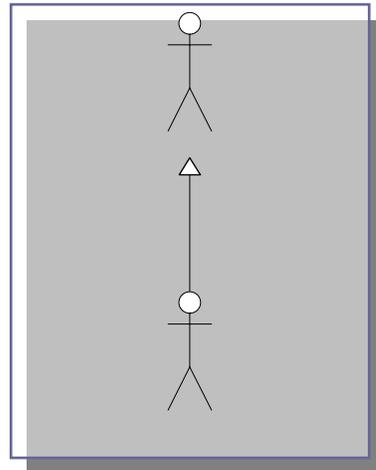


Figura 3.3: Especificación de grupos formalmente descritos.
Fuente: Elaboración Propia.

En este caso se escogió los diagramas de casos de uso establecidos por el lenguaje unificado de modelado, UML.

3.3.3.1 Actor Consultor

La herramienta Webml permite definir los casos de uso por usuario, en este caso se detalla al actor consultor, el cual tiene las opciones de: Acceder, Generar reportes, Consultar datos de teléfono y Salir. Ver Figura 3.4.

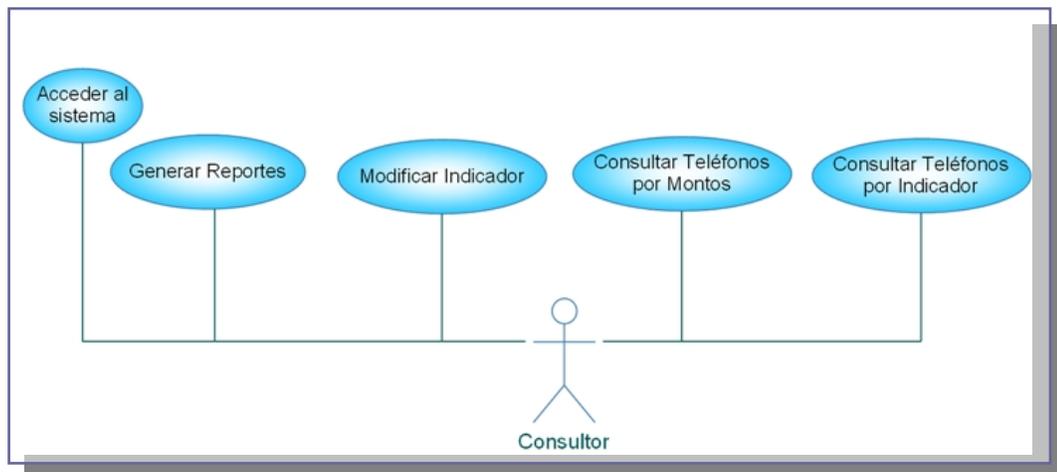


Figura 3.4: Usuario Consultor.
Fuente: Elaboración Propia.



3.3.4 Diagramas de análisis

Las clases de análisis identifican y describen los casos de uso más críticos del sistema, lo que permite que el sistema crezca incrementalmente a medida que se analicen los requisitos. Una clase de análisis se centra en el tratamiento de los requisitos funcionales y define atributos generalmente conceptuales y reconocibles en el dominio del problema.

Las clases de análisis siempre encajan en uno de tres estereotipos básicos: De interfaz, para modelar la interacción entre el sistema y sus actores, de control para representar coordinación, secuenciamiento, transacciones y control entre objetos o de entidad para modelar información de larga vida o permanente.

Cada estereotipo implica una semántica específica, lo cual constituye un método potente y consistente de identificar y describir las clases de análisis.

En este capítulo sólo se esboza una parte del análisis, por tanto se presentarán los diagramas de clase de análisis y de colaboración para los casos de uso más importantes dentro del sistema.

Entre los diagramas de clases de análisis que tocaremos en este capítulo están, *las clases interfaz* que son Acceder y Cargar datos de teléfonos, estas nos ayudan a modelar la interacción entre los actores y el sistema, *las clases de control* que son Autenticar datos y Gestor cargar datos de teléfono, estas representan la coordinación y secuenciamiento de las transacciones y para controlar las clases de entidad y *las clases entidad* que son LDAP, Usuario-Rol y Teléfonos, las cuales representan la información de larga vida dentro del sistema.



3.3.4.1 Diagrama de análisis para el caso de uso Acceder

Debido a que los actores que deben autenticarse para entrar al sistema son: Consultor y Administrador, se consideró el nombre “*Usuario*” para que de manera general se represente que puede ser cualquiera de los actores antes mencionado.

La Figura 3.5 muestra el diagrama de clases de análisis para el caso de uso Acceder, que expone la clase interfaz *Acceder*, utilizada por el *Usuario* para introducir los datos necesarios, los cuales serán validados por la clase de control *Autenticar Datos*, que consulta en el *LDAP* de PDVSA si el usuario es personal de la empresa y en la tabla *Usuario-Rol* de la base de datos del sistema, la existencia del mismo.

3.3.4.2 Diagrama de análisis para el caso de uso Cargar datos de teléfono

La Figura 3.6 muestra el diagrama de clases de análisis para el caso de uso Cargar datos de teléfono, el cual describe el proceso de como el *Administrador* ingresa el archivo de una operadora telefónica a través de la clase interfaz *Cargar datos de teléfono*, que es la encargada de presentar al Administrador la pantalla de interacción necesaria para llevar a cabo la inserción del archivo.

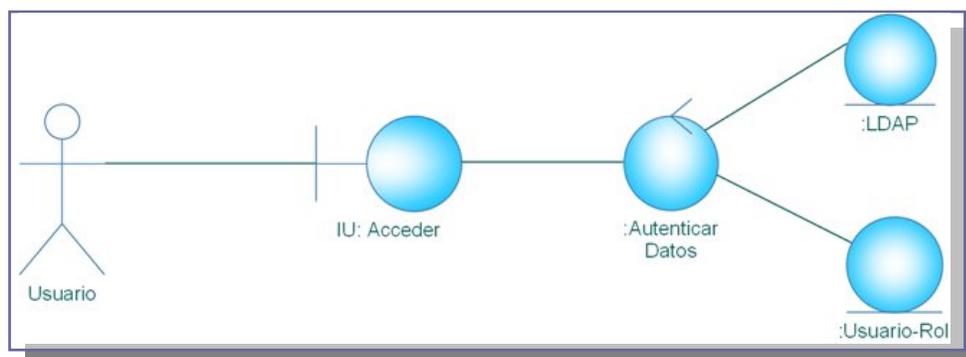


Figura 3.5: Diagrama de análisis para el caso de uso Acceder.
Fuente: Elaboración Propia.



Luego esta interfaz se conecta con la clase de control *Gestor cargar datos de teléfono*, el cual se encarga de procesar la información de la solicitud, e inmediatamente es almacenada en las clases entidad *Teléfonos*, que en este caso será una tabla de la base de datos del sistema donde se almacenan todos los datos concernientes a cada solicitud que se ingrese.

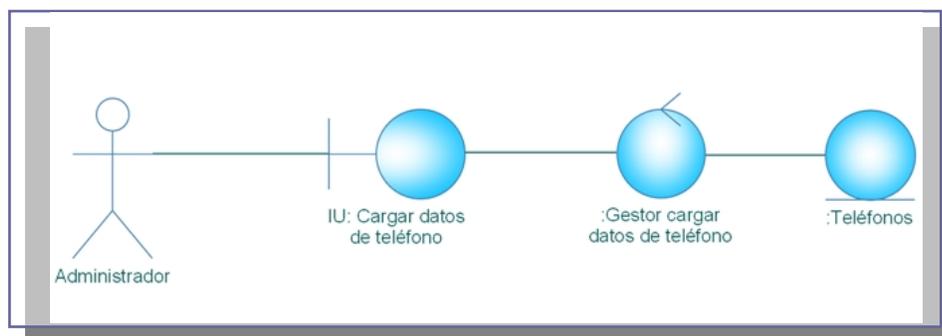


Figura 3.6: Diagrama de análisis para el caso de uso Cargar datos de teléfono.
Fuente: Elaboración Propia.

3.3.5 Diagramas de colaboración

Muestra la interacción entre varios objetos y los enlaces que existen entre ellos. Representa las interacciones entre objetos organizadas alrededor de los objetos y sus vinculaciones. A diferencia de un diagrama de secuencias, un diagrama de colaboraciones muestra las relaciones entre los objetos, no la secuencia en el tiempo en que se producen los mensajes. Como se podrán observar los diagramas de colaboración poseen la misma estructura de los diagramas de clase de análisis, solo que ahora se representa la interacción entre estos componentes, y para ello se utilizan los mensajes, mediante un número se pretende organizar dicho mensajes y las flechas indican la dirección del flujo.



Describen el proceso a detalle señalando la interacción y comunicación entre las clases, actores y entidades. El nombre de un mensaje debería denotar el propósito del objeto invocante en la interacción con el objeto invocado.

3.3.5.1 Diagrama de colaboración para el caso de uso Acceder

Debido a que los actores que deben autenticarse para entrar al sistema son: Consultor y Administrador, se consideró el nombre “*Usuario*” para que de manera general se represente que puede ser cualquiera de los actores antes mencionado. Ver Figura 3.7.

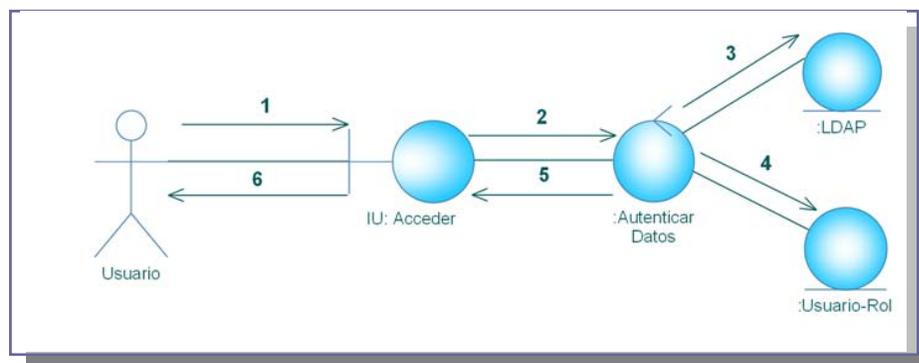


Figura 3.7: Diagrama de colaboración para el caso de uso Acceder.
Fuente: Elaboración Propia.

Leyenda:

1. Ingresar datos del usuario (Usuario y Contraseña).
2. Validar datos del usuario.
3. Consultar registro de usuario, en la clase entidad LDAP (Usuario y contraseña).
4. Consultar registro de usuario, en la clase entidad Usuario-Rol (Usuario y Rol).
5. Respuesta de acceso de usuario si es valido regresa el Rol del usuario.
6. Muestra interfaz dependiendo de la respuesta anterior.

El Usuario decide entrar al sistema y activa el objeto interfaz IU Acceder donde éste debe ingresar los datos de acceso requeridos para su autenticación (1). IU



Acceder valida si algún campo usuario y/o contraseña se encuentra vacío y si es el caso se muestra un mensaje de error y actualiza la interfaz requiriendo nuevamente los datos de autenticación; al estar ambos campos con sus respectivos datos esta interfaz solicita al objeto Autenticar datos que valide el acceso del mismo al sistema (2). Para ello el objeto Autenticar datos solicita al objeto LDAP, que representa el directorio activo de PDVSA, la búsqueda en su registro de la existencia del actor como empleado de PDVSA (3). Y al objeto Usuario, que representa una tabla de la base de datos del sistema, la búsqueda en su registro de la existencia del Actor como usuario autorizado para el uso del sistema, y si es el caso verifica el rol con el cual puede acceder al mismo (4). Finalmente después de validar la autenticación del usuario, éste recibe una respuesta de acceso (5), bien sea de acceso válido, mostrando la bienvenida al sistema, o de acceso no válido, prohibiendo su entrada e indicándole, clave o contraseña inválida. (6).

3.3.5.2 Diagrama de colaboración para el caso de uso Cargar datos de teléfono

En la Figura 3.8 se puede observar las interacciones asociadas al caso de uso Cargar datos de teléfono y sus respectivos pases de mensajes de un objeto a otro que nos ayuda a mostrar la implementación de esta operación.

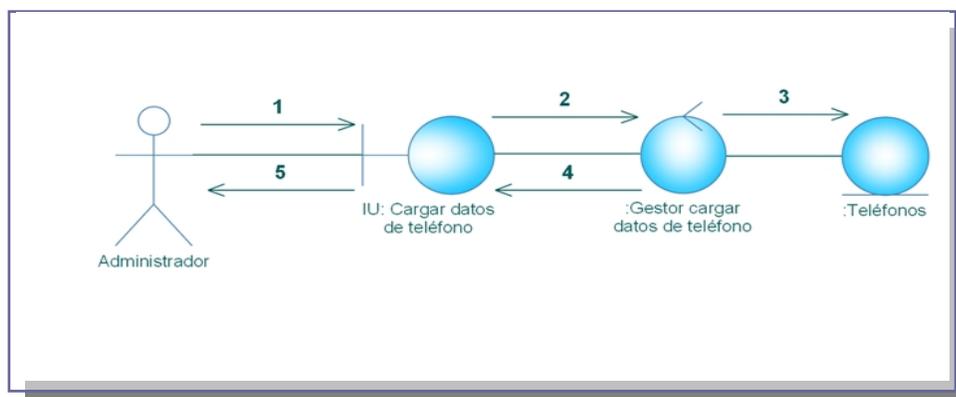


Figura 3.8: Diagrama de colaboración para el caso de uso Cargar datos de teléfono.

Fuente: Elaboración Propia.



Leyenda:

1. Ingresar Archivo de solicitud.
2. Validar el archivo de solicitud.
3. Ingresar registro del archivo de solicitud en la clase entidad Teléfonos.
4. Respuesta de la solicitud de ingreso del registro.
5. Confirmación de guardado.

Esta operación es permitida únicamente a aquellos usuarios registrados y autenticados y que además su rol sea de administrador. Se realiza de la siguiente manera, el Administrador ingresa el archivo solicitado correspondiente a los datos de los teléfonos (1). Estos archivos pasan por un proceso de validación (2) que lo hace el gestor cargar datos de teléfono. El archivo ingresado una vez que se le extrajeron los datos de los teléfonos y estos fueron validados se almacenan en la base de datos (3) se envía la respuesta de solicitud valida o invalida (4) y por último, se efectúa la confirmación de guardado de datos (5).

3.4 lujo de trabajo diseño

En el diseño se modela el sistema y se encuentra su forma (incluida la arquitectura) para que soporte todos los requisitos. Como esta es la fase de inicio, para este flujo de trabajo solo se presentará un bosquejo de las interfaces principales que debería presentar el sistema.

3.4.1 Arquitectura candidata

La arquitectura propuesta para esta aplicación se muestra en la Figura 3.9, está dividida en cuatro partes, cada una representando algunas de las funcionalidades del sistema.



La interfaz de usuario, comprende los componentes relacionados con el diseño, captura de datos y validación de datos.

En controlador de interfaz, se encuentran todos los componentes que toman decisión sobre el flujo de eventos, que depende de las solicitudes que los usuarios pidan al sistema. Estas solicitudes no son más que tareas que el sistema debe realizar de manera automática o no. Tareas como consultas y actualizaciones.

La parte lógica de la aplicación contiene las entidades necesarias para la funcionalidad del sistema.

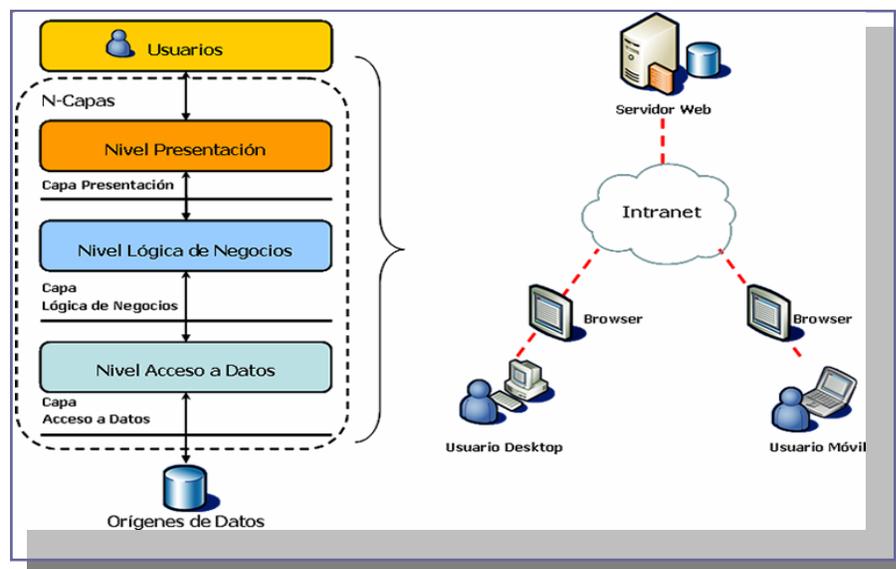


Figura 3.9: Arquitectura Candidata del Sistema.

Fuente: Elaboración Propia.

El acceso a datos contiene los componentes dedicados a establecer conexión con la base de datos del sistema, de establecer privilegios a los usuarios que intentan acceder a la información y comprende también la base de datos del sistema.



A continuación, en la Figura 3.10, se pueden observar cada uno de los niveles de la configuración de red con sus características correspondientes.

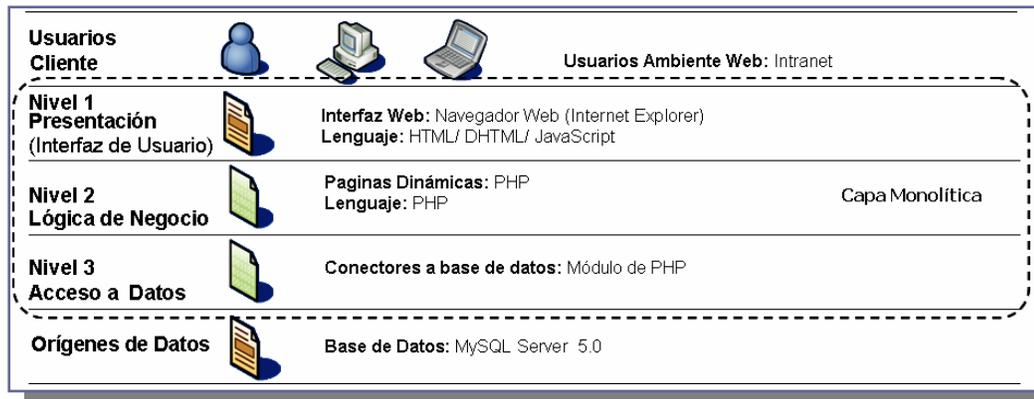


Figura 3.10: Niveles de Configuración de Red.
Fuente: Elaboración Propia.

3.4.2 Interfaz de inicio de sesión

La interfaz de inicio de sesión, mostrada en la Figura 3.11, es la primera a la que tiene acceso cualquiera de los usuarios, en esta interfaz es donde inicia el intercambio de información con cada uno de los usuarios del sistema.

Logo y Nombre de la Empresa

Nombre de la Aplicación

Usuario:

Clave:

Entrar

Símbolos Relacionados

Figura 3.11: Prototipo de interfaz de inicio de sesión.
Fuente: Elaboración Propia.



Se puede observar en la parte superior un área para colocar el logo de la empresa y el nombre de esta, un área central donde se colocarían el nombre de la aplicación y un formulario para el ingreso de datos de identificación de usuario y en la parte inferior un área para símbolos relacionados con la empresa.

CAPÍTULO IV

FASE DE ELABORACIÓN

4.1 Introducción

La meta de la fase de elaboración es definir y establecer la base de la arquitectura del sistema, brindando así una base estable para la mayor parte del esfuerzo de diseño e implementación en la fase de construcción.

En esta fase se van a transformar o refinar los modelos de la fase de inicio en otra serie de modelos que vayan perfilando una solución más cercana al mundo real. La fase de elaboración hace énfasis en establecer la arquitectura del sistema software, esto es obtener la jerarquía de módulos y la estructura de datos del sistema software, aunque también recaen en esta fase tareas tan importantes como son el diseño de la interfaz de usuario y el diseño de la base de datos.

La fase de elaboración construye la línea base de la arquitectura del sistema. Los principales objetivos para esta fase:

- ❖ Recopilar la mayor parte de los requisitos que aún queden pendientes, formulando los requisitos funcionales como casos de uso.
- ❖ Establecer una base de la arquitectura sólida -la línea base de la arquitectura- para guiar el trabajo durante las fases de construcción y transición, así como en las posteriores generaciones del sistema.



- ❖ Continuar la observación y control de los riesgos críticos que aún queden, e identificar riesgos significativos hasta el punto de que se pueda estimar su impacto en el análisis negocio, y en particular en la apuesta económica.

- ❖ Completar los detalles del plan del proyecto.

Para esta fase se debe desarrollar alrededor del ochenta por ciento (80%) de los casos de uso y abordar los riesgos que interfieran en la consecución de este objetivo.

En esta fase se trabajará a partir del flujo de trabajo que se refiere al análisis, debido a que no se identificaron nuevos actores, casos de uso, ni requisitos que se pudiesen agregar a los establecidos en la fase de inicio.

4.2 Flujo de trabajo análisis

En la fase de inicio se desarrolló el análisis de la arquitectura sólo hasta el punto de determinar que había una arquitectura del sistema factible. Ahora en esta fase se extiende el análisis hasta el punto de que pueda servir como base a la línea principal de la arquitectura ejecutable.

4.2.1 Especificación de los casos de usos por usuarios

En este punto se realiza la descripción formal de las unidades de interacción con la aplicación, de los usuarios o actores obtenidos en los grupos ya definidos, por medio de los casos de uso, tablas o diagramas de actividades. En este caso se escogió los diagramas de casos de uso establecidos por el lenguaje unificado de modelado, UML.



4.2.1.1 Actor Administrador

El actor Administrador, es el que posee privilegios para entrar en cualquier modulo del sistema, él es el encargado de administrar la base de datos de la que depende el sistema para su funcionamiento, esto abarca desde modificar y asignar roles para nuevos usuarios hasta agregar datos de gerencias, distritos, áreas y localidades a demás de poder modificar la información relacionada a los números telefónicos.

En la Figura 4.1, se puede observar todos los casos de uso a los que el actor Administrador tiene acceso.

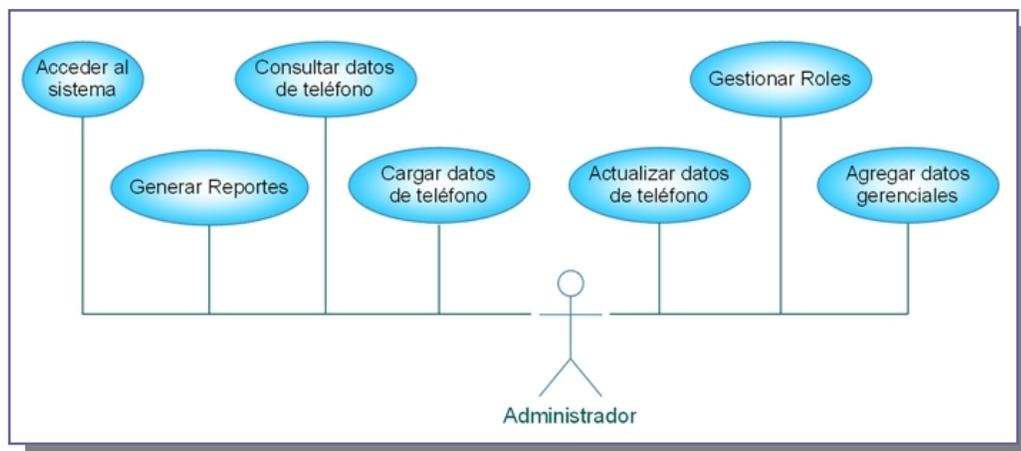


Figura 4.1: Actor Administrador.
Fuente: Elaboración Propia.

4.2.2 Diagrama de clase de análisis

Ahora en la fase de elaboración se extiende el análisis de la arquitectura hasta el extremo de que pueda servir de base a una línea base de la arquitectura ejecutable. Para esta fase se presentan los diagramas de clases de análisis para cada uno de los casos de uso del sistema en su totalidad.



4.2.2.1 Diagrama de análisis para el caso de uso Consultar datos de teléfono

Actualizar datos de teléfono es la encargada de presentar al actor, la pantalla de interacción necesaria para visualizar la información relacionada con un teléfono, para ello se toman los datos de la clase entidad Teléfonos y la clase entidad Detalles, para poder listar en la interfaz los datos de los teléfonos.

En esta interfaz se lleva a cabo la selección del teléfono del cual se desea conocer de manera minuciosa su información, la cual puede ser necesitada por el actor Consultor o el Administrador para el estudio de consumo respectivo, luego esta interfaz se conecta con la clase de control Gestor actualizar datos de teléfono, encargada de procesar la información del teléfono proveniente desde la clase entidad Teléfonos y de la clase entidad Detalles.

En la Figura 4.2, se muestra el actor representado por Usuario, para generalizar que puede ser cualquiera de los actores del sistema, tanto el Consultor como el Administrador.

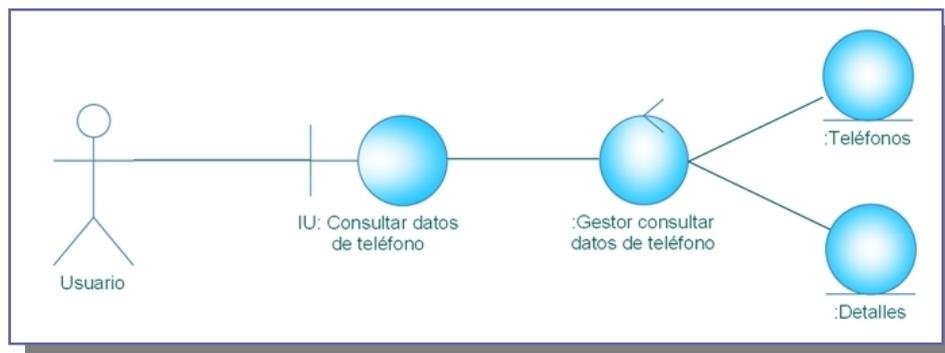


Figura 4.2: Diagrama de análisis para el caso de uso Consultar datos de teléfono.
Fuente: Elaboración Propia.



4.2.2.2 Diagrama de análisis para el caso de uso Actualizar datos de teléfono

Actualizar datos de teléfono es la encargada de presentar al actor la pantalla de interacción necesaria para enviar la información relacionada al teléfono, para ello se toman los datos de la clase entidad Teléfonos y de la clase entidad Responsables, para poder listar en la interfaz los teléfonos al cargarse la misma.

En esta interfaz se lleva a cabo la selección del teléfono que desea actualizar, luego esta interfaz se conecta con la clase de control Gestor actualizar datos de teléfono, encargada de procesar los cambios realizados al teléfono, que inmediatamente será almacenado en la clase entidad Teléfonos y en la clase entidad Responsables.

En la Figura 4.3, puede observarse el diagrama de análisis para el caso de uso Actualizar datos de teléfono, siendo el Administrador el que tiene acceso a este caso de uso.

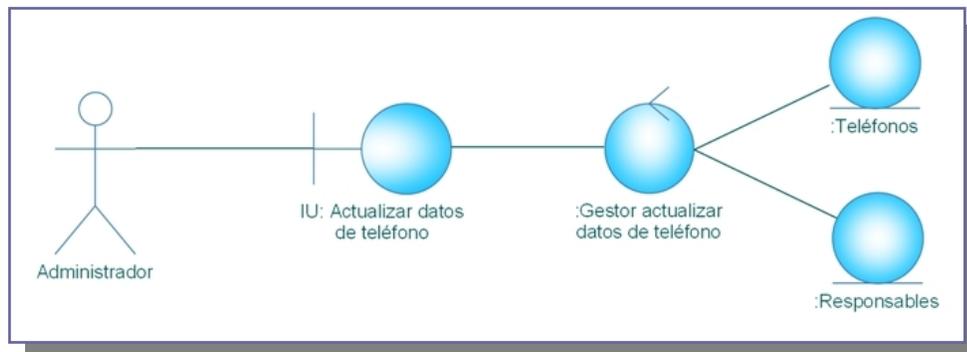


Figura 4.3: Diagrama de análisis para el caso de uso Actualizar datos de teléfono.
Fuente: Elaboración Propia.



4.2.2.3 Diagrama de análisis para el caso de uso Gestionar Roles

Gestionar Roles es la encargada de presentar al actor la pantalla de interacción necesaria para la asignación o actualización del rol de un usuario en el sistema, para ello se toman los datos de la clase entidad Roles, para poder listar en la interfaz los roles actuales de los usuarios del sistema.

En esta interfaz se lleva a cabo la asignación del rol específico a un usuario, luego esta interfaz se conecta con la clase de control Gestor asignar roles, que es la encargada de procesar la asignación que luego se almacenará en la clase entidad Roles, en la Figura 4.4, se observa el diagrama correspondiente.

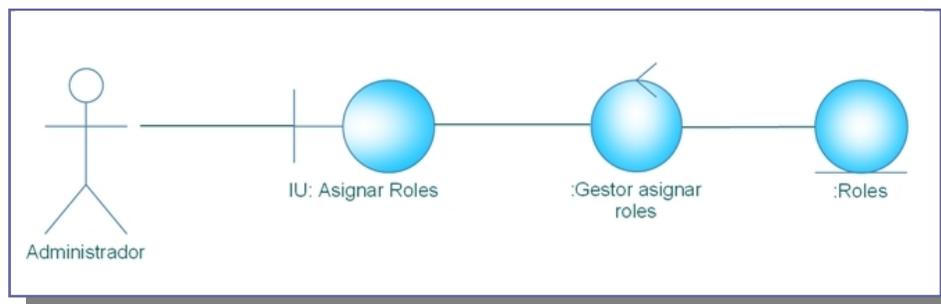


Figura 4.4: Diagrama de análisis para el caso de uso Asignar Roles.
Fuente: Elaboración Propia.

4.2.2.4 Diagrama de análisis para el caso de uso Agregar datos

gerenciales

Agregar datos gerenciales es la encargada de presentar al actor la pantalla de interacción necesaria para la agregación de datos gerenciales como, Distritos, Gerencias, Áreas y Localidades en el sistema, para ello se toman los datos de la clase entidad Datos Gerenciales, para poder mantener actualizados los datos corporativos dentro del sistema.



En esta interfaz se lleva a cabo la agregación de datos corporativos, luego esta interfaz se conecta con la clase de control Gestor agregar datos gerenciales, que es la encargada de procesar dicha operación que luego se almacenará en la clase entidad Datos Gerenciales, en la Figura 4.5, se observa el diagrama correspondiente.

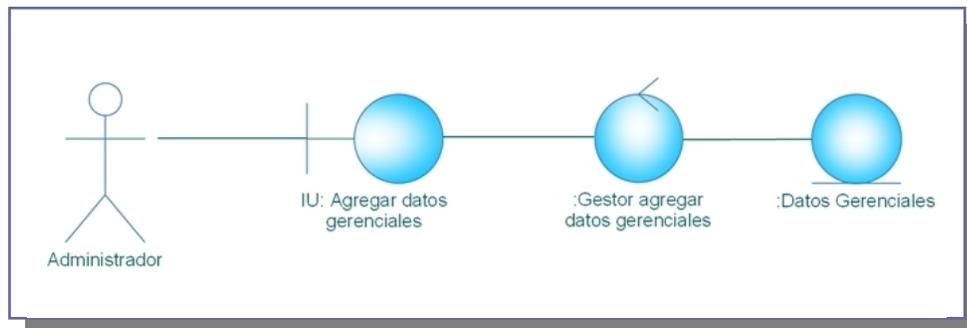


Figura 4.5: Diagrama de análisis para el caso de uso Agregar datos gerenciales.
Fuente: Elaboración Propia.

4.2.3 Diagrama de colaboración

Un diagrama de colaboración es una forma alternativa al diagrama de secuencia de mostrar un escenario. Este tipo de diagrama muestra las interacciones entre objetos organizadas entorno a los objetos y los enlaces entre ellos. Proporcionan la representación principal de un escenario, ya que las colaboraciones se organizan entorno a los enlaces de unos objetos con otros.

Los diagrama de colaboración, describen el proceso a detalle señalando la interacción y comunicación entre las clases, actores y entidades. El nombre de un mensaje debería denotar el propósito del objeto invocante en la interacción con el objeto invocado.



4.2.3.1 Diagrama de colaboración para el caso de uso Consultar datos de teléfono

El diagrama de colaboración para el caso de uso Consultar datos de teléfono se muestra en la Figura 4.6, como este caso de uso puede ser accedido tanto por el actor Consultor como por el Administrador, se generalizo a estos actores por medio del actor Usuario.

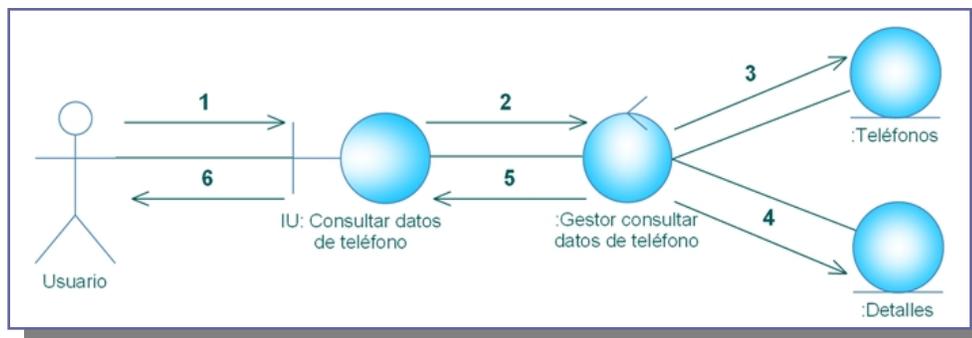


Figura 4.6: Diagrama de colaboración para el caso de uso Consultar datos de teléfono.
Fuente: Elaboración Propia.

Leyenda:

1. Verificar datos.
2. Gestionar datos.
3. Consultar teléfono.
4. Consultar detalles de teléfono.
5. Respuesta de solicitud de datos telefónicos.
6. Mostrar datos de teléfono.

El actor Usuario desea verificar la información relacionada con un teléfono y para ello activa la IU Consultar datos de teléfono (1).



La IU Consultar datos de teléfono solicita al objeto Gestor consultar datos de teléfono la información asociada al número de teléfono (2).

El objeto Gestor consultar datos de teléfono, solicita la información general del teléfono a la entidad Teléfonos (3) y la información detallada a la entidad Detalles (4), para que esta a su vez maneje la respuesta de solicitud de datos de teléfono (5) y así esta información sea mostrada al Usuario (6).

4.2.3.2 Diagrama de colaboración para el caso de uso Actualizar datos de teléfono

El diagrama de colaboración para el caso de uso Actualizar datos de teléfono se muestra en la Figura 4.7.

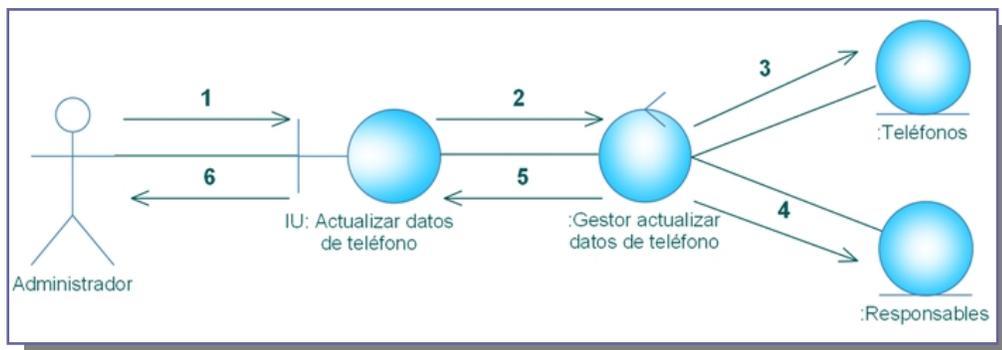


Figura 4.7: Diagrama de colaboración para el caso de uso Actualizar datos de teléfono.

Fuente: Elaboración Propia.

Leyenda:

1. Solicitar actualizar teléfono.
2. Gestionar actualización.
3. Guardar cambios en teléfonos.
4. Guardar cambios en responsables.



5. Respuesta a solicitud de actualización.
6. Mostrar mensaje de respuesta a solicitud de actualización.

El actor Administrador es el único con los permisos necesarios para actualizar datos de teléfono y por medio de la IU Actualizar datos de teléfono, solicita al sistema, insertar o modificar los datos de un teléfono (1). IU Actualizar datos de teléfono solicita al Gestor actualizar datos de teléfono que se encargue de procesar los datos del teléfono seleccionado y validar que acción se va hacer con la misma (2). Es así como el gestor solicita al objeto Teléfono actualizar su registro, guardando los cambios que se hayan realizados (3). De la misma forma al proceso anterior el gestor solicita al objeto Responsables que guarde los cambios ocurridos (4). Luego el gestor le envía a la clase interfaz la respuesta de la operación ya sea válida o invalida (5). Finalmente el Administrador recibe un mensaje de respuesta por la operación de actualización, ya sea por haber insertado o modificado los datos de un teléfono (6).

4.2.3.3 Diagrama de colaboración para el caso de uso Asignar Roles

El diagrama de colaboración para el caso de uso Asignar Roles se muestra en la Figura 4.8.

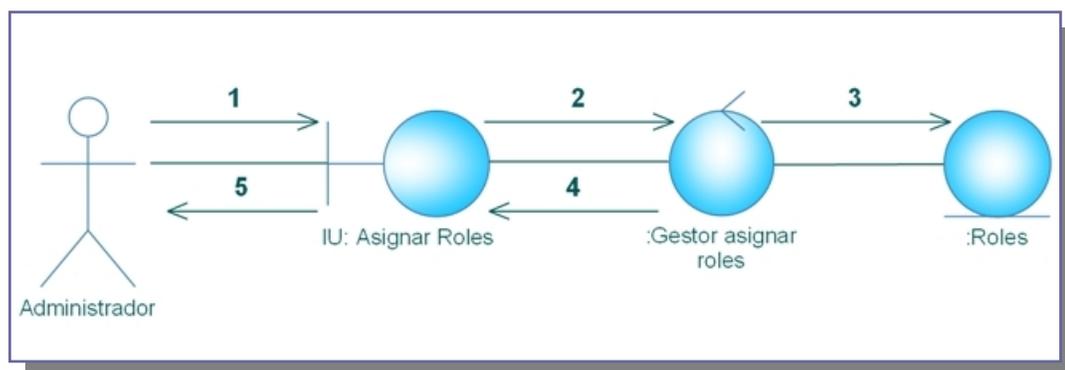


Figura 4.8: Diagrama de colaboración para el caso de uso Asignar Roles.
Fuente: Elaboración Propia.



Leyenda:

1. Solicitar asignar rol.
2. Gestionar asignación.
3. Guardar cambio.
4. Respuesta de asignación de rol.
5. Mostrar mensaje de respuesta de asignación de rol.

El actor administrador es el único con derechos para la asignación de roles y por medio de la IU Asignar Roles, realiza la asignación de roles para los usuarios que tendrán acceso al sistema (1). IU Asignar Roles solicita al gestor asignar roles que se encargue de procesar la asignación de roles de los usuarios (2). Es así como el gestor solicita al objeto Roles actualizar su registro, guardando el cambio que haya hecho el Administrador (3). Luego el gestor le envía a la clase interfaz la respuesta de la operación ya sea válida o invalida (4). Finalmente el Administrador recibe un mensaje de respuesta por la operación de asignación de rol (5).

4.2.3.4 Diagrama de colaboración para el caso de uso Agregar datos gerenciales

El diagrama de colaboración para el caso de uso Agregar datos gerenciales se muestra en la Figura 4.9.

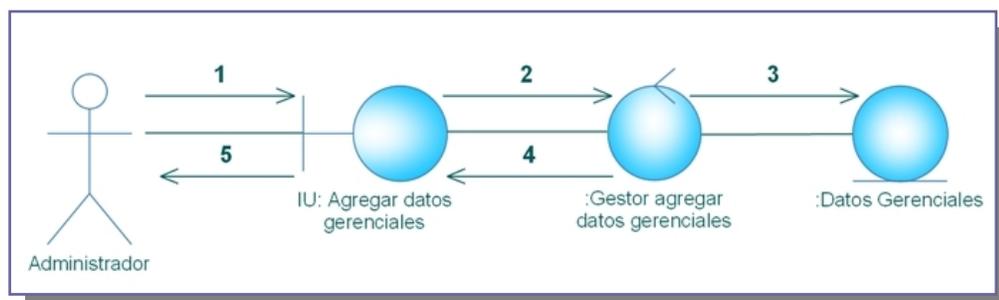


Figura 4.9: Diagrama de colaboración para el caso de uso Agregar datos gerenciales.
Fuente: Elaboración Propia.



Leyenda:

1. Solicitar agregar datos gerenciales.
2. Gestionar agregación.
3. Guardar cambio.
4. Respuesta de agregación.
5. Mensaje de respuesta de agregación de datos gerenciales.

El actor Administrador es el que posee privilegios para agregar datos gerenciales y lo realiza por medio de la IU Agregar datos gerenciales, solicita al sistema agregar un Distrito, Gerencia, Área o una Localidad (1). IU Agregar datos gerenciales solicita al Gestor agregar datos gerenciales que procese estas solicitudes realizadas por el Administrador (2). Es así como el Gestor solicita al objeto Datos Gerenciales actualizar su registro, guardando los cambios que se hayan realizados (3). Luego el gestor le envía a la clase interfaz la respuesta de la operación ya sea válida o invalida (4). Finalmente el Administrador recibe un mensaje con la respuesta obtenida de la agregación (5).

4.2.4 Análisis de la arquitectura

La definición de la arquitectura usando WebML se encuentra basada en la definición de las áreas (módulos de información) que constituyen a la aplicación Web y de las páginas que conforman dichas áreas, pues esto permite establecer de forma efectiva las vistas del sitio como producto final y a su vez concretar cómo estará compuesto el mismo y cómo será su navegación.

La visibilidad de las áreas esta constituida de la siguiente manera:



- ❖ Área por defecto (Default area “D”): se dice que un área es por defecto cuando esta adjunta a la vista del sitio que es accedido.
- ❖ Landmark área “L”: cuando es globalmente accesible desde cualquier otra área dentro del sitio Web.
- ❖ Área Interna (internal área “I”): cuando es visible solo por medio de enlaces explícitos.

La visibilidad de las páginas se diferencia de la siguiente manera:

- ❖ Pagina Principal (home page “H”): se presenta por defecto cuando el usuario ingresa al sitio Web.
- ❖ Página por defecto (default page “D”): se presenta por defecto cuando se accede al área.
- ❖ Landmark area “L”: tiene un alcance global desde todas las páginas encerradas en un mismo módulo (vista del sitio o área).
- ❖ Página interna (internal page “I”): implementado dependiendo del contenido.

La Aplicación **SAFREST** presenta dos tipos de vistas del sitio (site views) diferentes de acuerdo al tipo de rol de usuario. La vista de sitio para Consultores se encuentra estructurada en 2 áreas y 6 páginas principales. Ver Figura 4.10.

La vista del sitio para el actor Administrador se encuentra estructurada por 3 áreas y 6 páginas principales. Ver Figura 4.11.

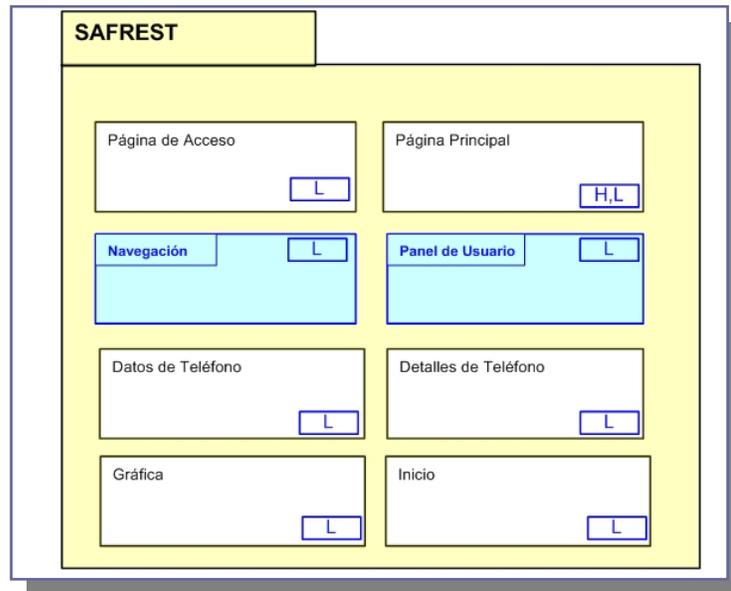


Figura 4.10: Vista del sitio para actor Consultor.
Fuente: Elaboración Propia.

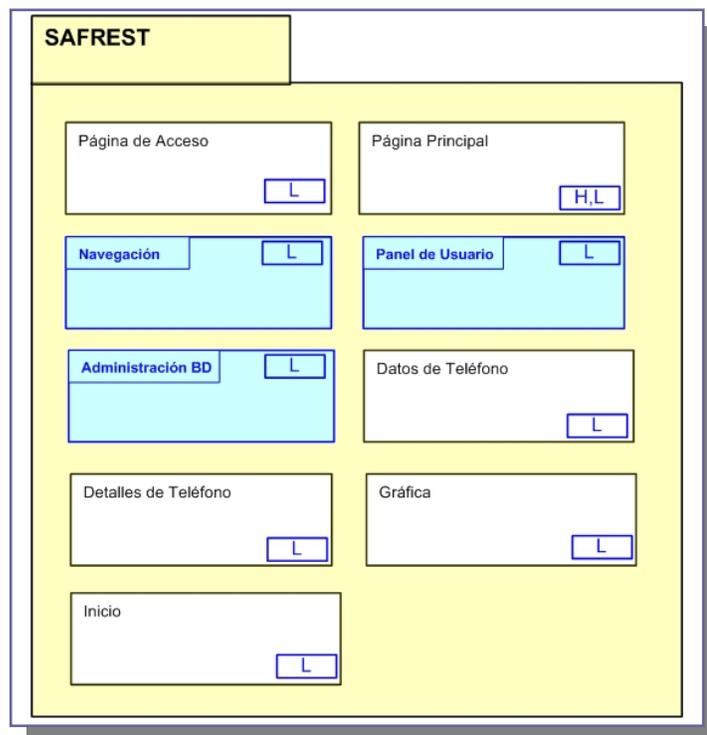


Figura 4.11: Vista del sitio para actor Administrador.
Fuente: Elaboración Propia.



Como muestra la figura anterior el Administrador tiene acceso a todas las áreas y sub-áreas del sistema, así como también a todas las páginas que forman parte de la aplicación.

La Figura 4.12, modela la estructura de la aplicación del área de Administración BD, perteneciente al site view para Administrador, mostrando sus elementos y componentes a un nivel más detallado. Esta área será utilizada para estudiar los diagramas y modelos WebML a través de su implementación en los caso de uso del sistema. Del mismo modo se obtiene la estructura definida de todas las páginas y sub-áreas que constituyen a la misma.

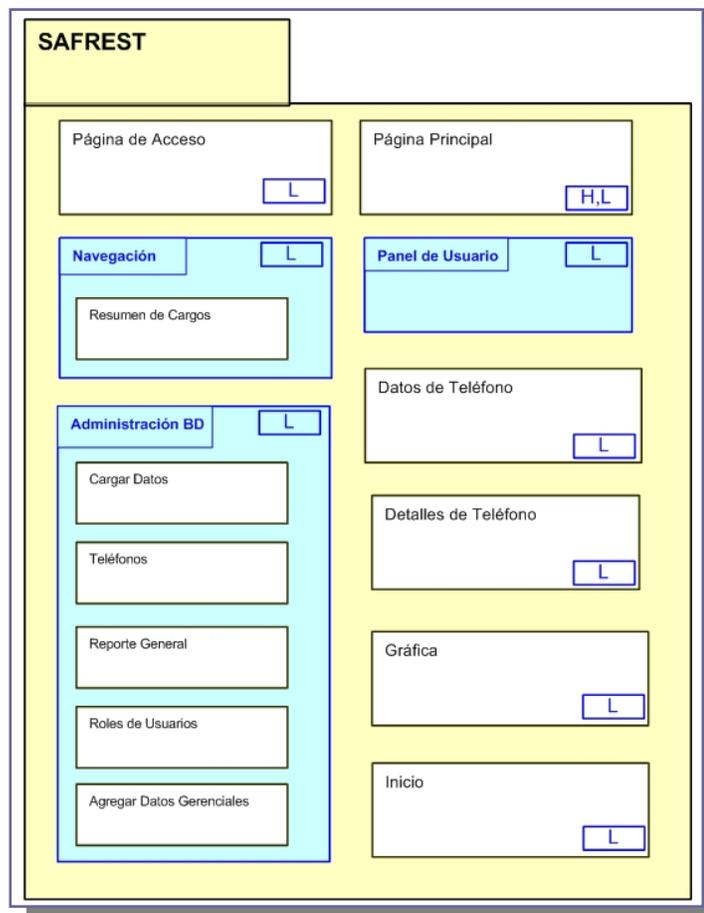


Figura 4.12: Vista detallada del sitio para el actor Administrador.
Fuente: Elaboración Propia.



4.2.5 Descripción de paquetes del sistema SAFREST

Los paquetes se representan mediante rectángulos con pestañas y las relaciones de dependencias se muestran como flechas con líneas discontinuas.

Para el sistema desarrollado se identificaron una serie de paquetes, los cuales encapsulan los diferentes casos de usos que fueron definidos al realizar el análisis del sistema, los paquetes definidos fueron los siguientes: “Gestión de autenticar”, “Gestión de visualización”, “Gestión de administración”, “Gestión de teléfonos.

Gestión de Autenticar: Este paquete se encarga del proceso de autenticar a los usuarios que desean hacer uso de la aplicación.

Gestión de Visualización: Es el encargado de cómo mostrar a los usuarios toda la información referente a la aplicación.

Gestión de Administración: Este paquete es el encargado de administrar y organizar toda la información de entrada y salida de la aplicación.

Gestión de Teléfonos: Este paquete es el encargado de la relación de todos los datos referente a los teléfonos, tales como sus detalles y responsables.

4.3 Flujo de trabajo diseño

En esta fase los casos de uso significativos se diseñan en términos de subsistemas de diseño.

Los propósitos del diseño son:



- ❖ Adquirir una comprensión en profundidad de los aspectos relacionados con los requisitos no funcionales.
- ❖ Crear una entrada apropiada y un punto de partida para actividades de implementación subsiguientes capturando los requisitos o subsistemas individuales e interfaces.
- ❖ Ser capaces de descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo, teniendo en cuenta la posible concurrencia.

4.3.1 Diseño de la arquitectura

El sistema SAFREST está basado en el uso de páginas Web HTML y scripts de servidor. Este subsistema conforma lo que sería la capa específica de la aplicación. Las páginas que forman parte de SAFREST son desplegadas mediante el subsistema navegador Web que en este caso puede ser representado por Internet Explorer, Firefox, Opera, Netscape, Mozilla, o algún otro navegador, todo esto siendo la capa general de la aplicación. El navegador para poder procesar las peticiones del cliente utiliza el subsistema servidor Web que consiste del servidor Apache, y este a su vez usa el motor de scripts PHP para generar las páginas web mediante el subsistema MySQL que es el encargado del manejo de la base de datos del sistema. Estos cuatro subsistemas establecen la capa intermedia.

Finalmente tenemos la capa de software del sistema formada por el subsistema sistema operativo que sería una instancia de Microsoft Windows o en su defecto Linux, y también formada por el subsistema TCP/IP que representa el protocolo de comunicación entre el servidor y los clientes. El diseño de la arquitectura del sistema se puede observar en la Figura 4.13.

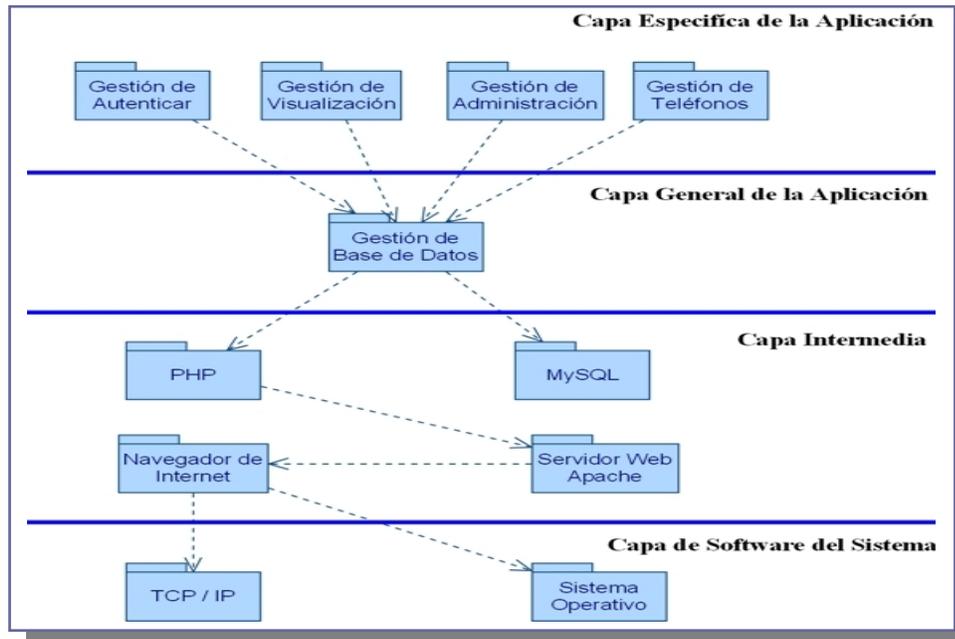


Figura 4.13: Arquitectura del sistema SAFREST.

Fuente: Elaboración Propia.

4.3.2 Diseño de la base de datos

Para el diseño de la base de datos se utilizarán el modelo de datos de la especificación de WEBML y la descripción de las tablas de la base de datos.

Conociendo el volumen de datos que podrá manejar el sistema SAFREST, se determinó el diseño de una Base de Datos relacional BD_SAFREST para el control y gestión de los datos.

Una base de datos relacional es un conjunto de dos o más tablas estructuradas en registros (líneas) y campos (columnas), que se vinculan entre sí por un campo en común, en ambos casos posee las mismas características como por ejemplo el nombre de campo, tipo y longitud; a este campo generalmente se le denomina ID, identificador o clave.



A esta manera de construir bases de datos se le denomina modelo relacional, es un modelo de datos basado en la lógica de predicado y en la teoría de conjuntos.

Éste es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente.

Para el diseño de una Base de Datos relacional tienen que tomarse en cuenta las siguientes leyes básicas:

1. Una tabla sólo contiene un número fijo de campos.
2. El nombre de los campos de una tabla es distinto.
3. Cada registro de la tabla es único.
4. El orden de los registros y de los campos no está determinados.
5. Para cada campo existe un conjunto de valores posible.

4.3.2.1 Modelo de datos

El modelo de datos de WebML es una adaptación de los modelos conceptuales de diseño de base datos, es compatible con el modelo de datos Entidad – Relación usado en el diseño conceptual de bases de datos, y también es compatible con los diagramas de clases UML empleados en el modelado orientado a objetos.

El modelo de datos define la estructura de la base de datos. En la Figura 4.14 pueden verse los atributos de cada entidad así como su relación con otras entidades.

4.3.2.2 Descripción de la Base de Datos

Tomando el modelo de datos y realizando la normalización de las entidades, se pudo diseñar la base de datos del sistema, utilizando el administrador de base de datos MySQL como herramienta.

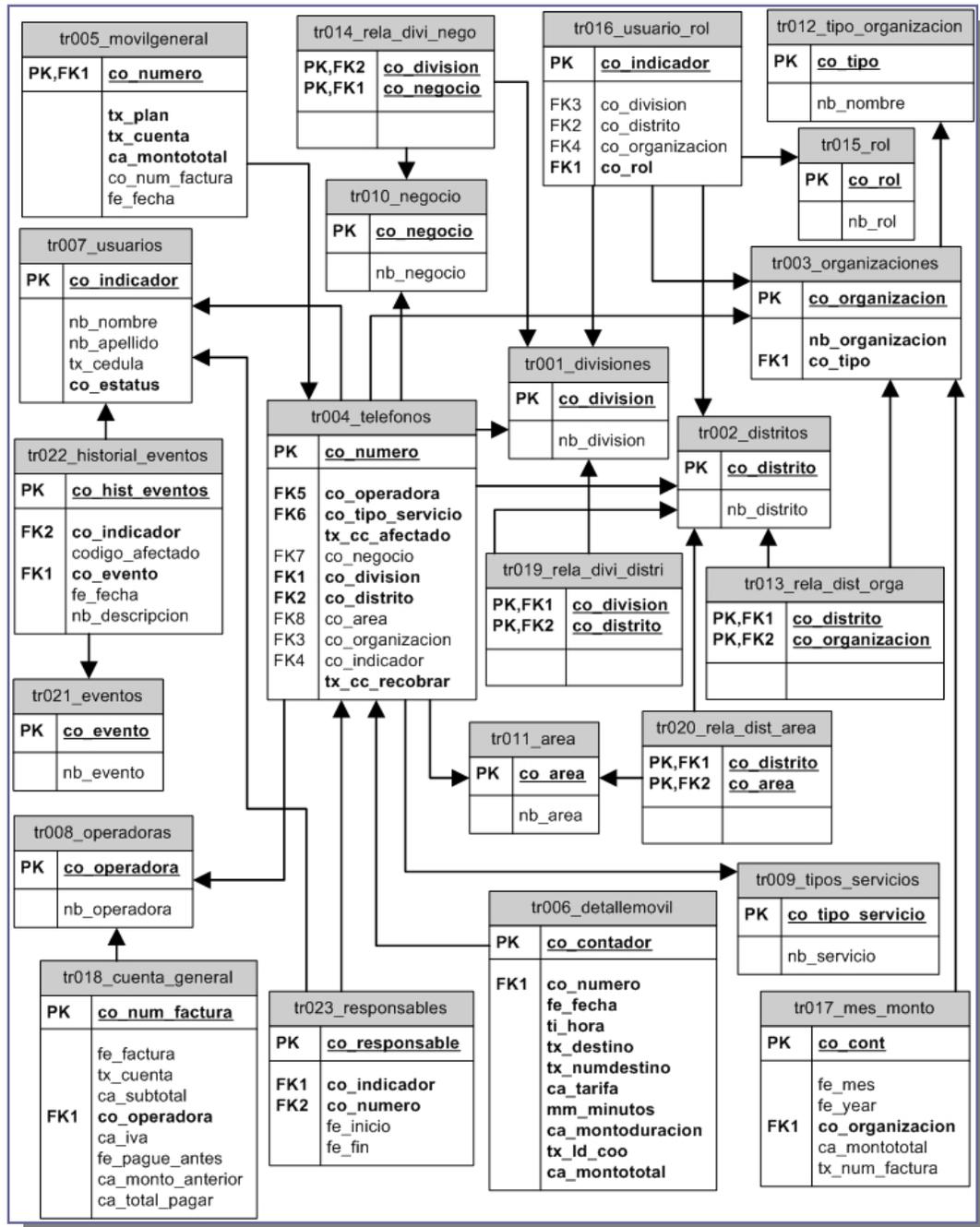


Figura 4.14: Relación de tablas sistema SAFREST.

Fuente: Elaboración Propia.



La base de datos del sistema SAFREST (denominada “BD_SAFREST”) está formada por un conjunto de dieciocho (18) tablas que serán descritas a continuación.

Tabla “tr001_divisiones”: registra las diferentes divisiones en las que se encuentra dividida PDVSA. Ver Tabla 4.1.

Tabla 4.1: Descripción de la tabla “tr001_divisiones”.

tr001_divisiones		
Columna	Tipo	Descripción
CO_DIVISION	int(11)	Código único de la división.
NB_DIVISION	varchar(60)	Nombre de la división.

Fuente: Elaboración Propia.

tr001_divisiones	
Columna	Tipo
CO_DIVISION	int(11)
NB_DIVISION	varchar(60)

Figura 4.15: Imagen de la tabla “tr001_divisiones”.

Fuente: Elaboración Propia.

Tabla “tr002_distritos”: Registra los diferentes distritos en los que se encuentra dividida PDVSA. Ver

Tabla 4.2.

Tabla 4.2: Descripción de la tabla “tr002_distritos”.

tr002_distritos		
Columna	Tipo	Descripción
CO_DISTRITO	int(11)	Código único de distrito.
NB_DISTRITO	varchar(60)	Nombre del distrito.

Fuente: Elaboración Propia.



A placeholder image for the table 'tr002_distritos'. It consists of a grey header bar at the top, followed by a table with two columns and two rows. The top row has a horizontal line in the right column. The bottom row is empty.

Figura 4.16: Imagen de la tabla “tr002_distritos”.
Fuente: Elaboración Propia.

Tabla “tr003_organizaciones”: Registra las diferentes organizaciones en las que se encuentra dividida PDVSA. Ver Tabla 4.3.

Tabla 4.3: Descripción de la tabla “tr003_organizaciones”.

tr003_organizaciones		
<i>Columna</i>	<i>Tipo</i>	<i>Descripción</i>
CO_ORGANIZACION	int(11)	Código único de organización.
NB_ORGANIZACION	varchar(60)	Nombre de la organización.
CO_TIPO	int(11)	Código único del tipo de organización.

Fuente: Elaboración Propia.

A placeholder image for the table 'tr003_organizaciones'. It consists of a grey header bar at the top, followed by a table with two columns and two rows. The top row has a horizontal line in the right column. The bottom row is empty.

Figura 4.17: Imagen de la tabla “tr003_organizaciones”.
Fuente: Elaboración Propia.

Tabla “tr004_telefonos”: Registra los diferentes teléfonos que se encuentran distribuidos entre las diferentes organizaciones que forman parte en PDVSA. Ver Tabla 4.4.



Tabla 4.4: Descripción de la tabla “tr004_telefonos”.

tr004_telefonos		
Columna	Tipo	Descripción
CO_NUMERO	varchar(60)	Código único de teléfono (número telefónico).
CO_OPERADORA	int(11)	Código único de la operadora del teléfono.
CO_TIPO_SERVICIO	int(11)	Código único del tipo de servicio prestado.
TX_CC_AFECTADO	varchar(60)	Código del centro de costo afectado por cobro.
CO_NEGOCIO	int(11)	Código único de negocio.
CO_DIVISION	int(11)	Código único de división.
CO_DISTRITO	int(11)	Código único de distrito.
CO_AREA	int(11)	Código único del área.
CO_ORGANIZACION	int(11)	Código único de organización.
CO_INDICADOR	varchar(60)	Código único de indicador del usuario.
TX_CC_RECOBRAR	varchar(60)	Código del centro de costo afectado por recobro.

Fuente: Elaboración Propia.

Figura 4.18: Imagen de la tabla “tr004_telefonos”.

Fuente: Elaboración Propia.

Tabla “tr005_movilgeneral”: Registra la información general respectiva a un teléfono, que se encuentra asignado a una organización perteneciente a PDVSA. Ver Tabla 4.5.



Tabla 4.5: Descripción de la tabla “tr005_movilgeneral”.

tr005_movilgeneral		
<i>Columna</i>	<i>Tipo</i>	<i>Descripción</i>
CO_NUMERO	varchar(60)	Código único de teléfono (número telefónico).
TX_PLAN	varchar(60)	Nombre del plan que posee el teléfono.
TX_CUENTA	varchar(60)	Código de la cuenta a la que pertenece el teléfono.
CA_MONTOTOTAL	float	Cantidad del monto facturado al teléfono.
CO_NUM_FACTURA	varchar(60)	Código del número de factura del mes.
FE_FECHA	varchar(60)	Fecha de la factura.

Fuente: Elaboración Propia.

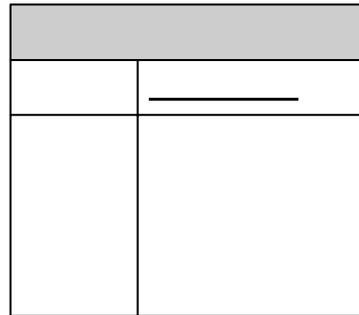


Figura 4.19: Imagen de la tabla “tr005_movilgeneral”.

Fuente: Elaboración Propia.

Tabla “tr006_detallemovil”: Registra toda la información referente al detalle de los teléfonos asociados a una organización perteneciente a PDVSA. Ver Tabla 4.6.

Tabla 4.6: Descripción de la tabla “tr006_detallemovil” (1/2).

tr006_detallemovil		
<i>Columna</i>	<i>Tipo</i>	<i>Descripción</i>
CO_CONTADOR	int(11)	Código único para control.
CO_NUMERO	varchar(60)	Código único de teléfono (número telefónico).
FE_FECHA	varchar(60)	Fecha de la llamada.
TI_HORA	varchar(60)	Hora de la llamada.
TX_DESTINO	varchar(60)	Destino de la llamada.
TX_NUMDESTINO	varchar(60)	Número de teléfono de destino de la llamada.

Fuente: Elaboración Propia.

tr005_

PK,FK1



Tabla 4.6: Descripción de la tabla “tr006_detallemovil” (2/2).

tr006_detallemovil		
Columna	Tipo	Descripción
CA_TARIFA	varchar(60)	Cantidad de la tarifa por minutos.
MM_MINUTOS	varchar(60)	Minutos de la llamada.
CA_MONTODURACION	float	Cantidad del monto por duración.
TX_LD_COO	float	Cargos por larga distancia o a otras operadoras.
CA_MONTOTOTAL	float	Cantidad del monto facturado por la llamada.

Fuente: Elaboración Propia.

Figura 4.20: Imagen de la tabla “tr006_detallemovil”.

Fuente: Elaboración Propia.

Tabla “tr007_usuarios”: Registra la información concerniente a los usuarios que utilizan el sistema y los que son responsables de teléfonos. Ver

Tabla 4.7.

Tabla 4.7: Descripción de la tabla “tr007_usuarios”.

tr007_usuarios		
Columna	Tipo	Descripción
CO_INDICADOR	varchar(60)	Código único de indicador del usuario.
NB_NOMBRE	varchar(60)	Nombre del usuario o responsable de teléfono.
NB_APELLIDO	varchar(60)	Apellido del usuario o responsable de teléfono.
TX_CEDULA	varchar(60)	Cedula de identidad del usuario o responsable.

tr006_

PK co_

FK1 co_

fe_

ti_h

tx_

tx_

ca_

mm



CO_ESTATUS	int(11)	Código de estatus del usuario o responsable.
------------	---------	--

Fuente: Elaboración Propia.

Figura 4.21: Imagen de la tabla “tr007_usuarios”.

Fuente: Elaboración Propia.

Tabla “tr008_operadoras”: Registra todas las operadoras que prestan los servicios de telefonía para PDVSA. Ver Tabla 4.8.

Tabla 4.8: Descripción de la tabla “tr008_operadoras”.

tr008_operadoras		
Columna	Tipo	Descripción
CO_OPERADORA	int(11)	Código único de la operadora.
NB_OPERADORA	varchar(60)	Nombre de la operadora.

Fuente: Elaboración Propia.

Figura 4.22: Imagen de la tabla “tr008_operadoras”.

Fuente: Elaboración Propia.

Tabla “tr009_tipos_servicios”: Registra la información concerniente al tipo de servicio que presta el número particular, este puede ser tarjeta, celular o fijos. Ver Tabla 4.9.



Tabla 4.9: Descripción de la tabla “tr009_tipos_servicios”.

tr009_tipos_servicios		
<i>Columna</i>	<i>Tipo</i>	<i>Descripción</i>
CO_TIPO_SERVICIO	int(11)	Código único del tipo de servicio.
NB_SERVICIO	varchar(60)	Nombre del servicio (tarjeta, celular o fijos).

Fuente: Elaboración Propia.

tr009_tipos_servicios	
<i>Columna</i>	<i>Tipo</i>
CO_TIPO_SERVICIO	int(11)
NB_SERVICIO	varchar(60)

Figura 4.23: Imagen de la tabla “tr009_tipos_servicios”.

Fuente: Elaboración Propia.

Tabla “tr010_negocio”: Registra los negocios que forman parte de PDVSA, como por ejemplo E.Y.P., Refinación, Gas, etc. Ver Tabla 4.10.

Tabla 4.10: Descripción de la tabla “tr010_negocio”.

tr010_negocio		
<i>Columna</i>	<i>Tipo</i>	<i>Descripción</i>
CO_NEGOCIO	int(11)	Código único del tipo de negocio de la corporación.
NB_NEGOCIO	varchar(60)	Nombre del negocio (E.Y.P., Gas, Refinación, etc.).

Fuente: Elaboración Propia.

tr010_negocio	
<i>Columna</i>	<i>Tipo</i>
CO_NEGOCIO	int(11)
NB_NEGOCIO	varchar(60)

Figura 4.24: Imagen de la tabla “tr010_negocio”.

Fuente: Elaboración Propia.

tr009_tip

PK co_

nb_s



Tabla “tr011_area”: Registra todas las áreas que forman parte de la corporación, como por ejemplo PDM, Jusepín, Morichal, PLC, etc. Ver Tabla 4.11.

Tabla 4.11: Descripción de la tabla “tr011_area”.

tr011_area		
<i>Columna</i>	<i>Tipo</i>	<i>Descripción</i>
CO_AREA	int(11)	Código único del área de la corporación.
NB_AREA	varchar(60)	Nombre del área (PDM, Jusepín, PLC, etc.).

Fuente: Elaboración Propia.

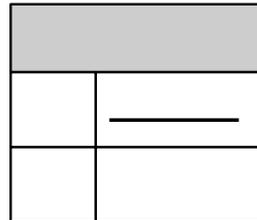


Figura 4.25: Imagen de la tabla “tr011_area”.

Fuente: Elaboración Propia.

Tabla “tr012_tipo_organizacion”: Registra la información relacionada con el tipo de organización a la que se le presta el servicio, como por ejemplo: misiones, Norte, etc. Ver Tabla 4.12.

Tabla 4.12: Descripción de la tabla “tr012_tipo_organizacion”.

tr012_tipo_organizacion		
<i>Columna</i>	<i>Tipo</i>	<i>Descripción</i>
CO_TIPO	int(11)	Código único del tipo de organización.
NB_NOMBRE	varchar(60)	Nombre del tipo de organización (Misiones, Norte).

Fuente: Elaboración Propia.

Tabla “tr013_rela_dist_orga”: Esta tabla relaciona los diferentes Distritos que conforman a PDVSA con sus respectivas Organizaciones. Ver Tabla 4.13.

tr01

PK



tr012_tipo_organizacion	

Figura 4.26: Imagen de la tabla “tr012_tipo_organizacion”.
Fuente: Elaboración Propia.

Tabla 4.13: Descripción de la tabla “tr013_rela_dist_orga”.

tr013_rela_dist_orga		
Columna	Tipo	Descripción
CO_DISTRITO	int(11)	Código único del Distrito.
CO_ORGANIZACION	int(11)	Código único de la Organización.

Fuente: Elaboración Propia.

tr013_rela_dist_orga	

Figura 4.27: Imagen de la tabla “tr013_rela_dist_orga”.
Fuente: Elaboración Propia.

Tabla “tr014_rela_divi_nego”: Esta tabla relaciona las diferentes Divisiones que conforman a PDVSA con sus respectivos Negocios. Ver Tabla 4.14.

Tabla 4.14: Descripción de la tabla “tr014_rela_divi_nego”.

tr014_rela_divi_nego		
Columna	Tipo	Descripción
CO_DIVISION	int(11)	Código único de la División.
CO_NEGOCIO	int(11)	Código único del Negocio.

Fuente: Elaboración Propia.

tr012_tipo

PK co_

nb_

tr013_re



tr014_rela_divi_nego	

Figura 4.28: Imagen de la tabla “tr014_rela_divi_nego”.
Fuente: Elaboración Propia.

Tabla “tr015_rol”: Registra los roles con los cuales los usuarios del sistema pueden ingresar al mismo. Ver Tabla 4.15.

Tabla 4.15: Descripción de la tabla “tr015_rol”.

tr015_rol		
Columna	Tipo	Descripción
CO_ROL	int(11)	Código único del rol de usuario.
NB_ROL	varchar(60)	Nombre del tipo de rol (Consultor, Administrador).

Fuente: Elaboración Propia.

tr015_rol	

Figura 4.29: Imagen de la tabla “tr015_rol”.
Fuente: Elaboración Propia.

Tabla “tr016_usuario_rol”: Registra la información general del usuario con su respectivo rol dentro del sistema. Ver Tabla 4.16.

Tabla “tr017_mes_monto”: Registra la información referente al monto a cancelar por cada organización mensualmente. Ver Tabla 4.17.

tr014_re

PK,FK2
PK,FK1



Tabla 4.16: Descripción de la tabla “tr016_usuario_rol”.

tr016_usuario_rol		
<i>Columna</i>	<i>Tipo</i>	<i>Descripción</i>
CO_INDICADOR	varchar(60)	Código único de indicador del usuario.
CO_DIVISION	int(11)	Código único de división.
CO_DISTRITO	int(11)	Código único de distrito.
CO_ORGANIZACION	int(11)	Código único de organización.
CO_ROL	int(11)	Código único de rol del usuario.

Fuente: Elaboración Propia.

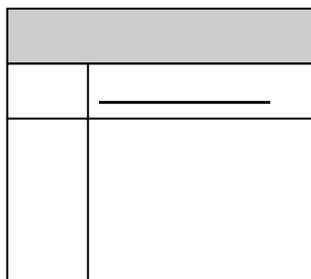


Figura 4.30: Imagen de la tabla “tr016_usuario_rol”.

Fuente: Elaboración Propia.

Tabla 4.17: Descripción de la tabla “tr017_mes_monto”.

tr017_mes_monto		
<i>Columna</i>	<i>Tipo</i>	<i>Descripción</i>
CO_CONT	int(11)	Código único de control.
FE_MES	varchar(60)	Mes de la factura.
FE_YEAR	varchar(60)	Año de la factura.
CO_ORGANIZACION	int(11)	Código único de organización.
CA_MONTOTOTAL	float	Monto total a cancelar por la organización.
TX_NUM_FACTURA	varchar(60)	Código de la factura del cobro.

Fuente: Elaboración Propia.

Tabla “tr018_cuenta_general”: Registra la información concerniente a la factura recibida por PDVSA de parte de las prestadoras de servicios. Ver Tabla 4.18.

tr016_

PK **co**

FK3 **co**

FK2 **co**

FK4 **co**

FK1 **co**



Figura 4.31: Imagen de tabla “tr017_mes_monto”.
Fuente: Elaboración Propia.

Tabla 4.18: Descripción de la tabla “tr018_cuenta_general”.

tr018_cuenta_general		
<i>Columna</i>	<i>Tipo</i>	<i>Descripción</i>
CO_NUM_FACTURA	varchar(60)	Código único de la factura.
FE_FACTURA	varchar(60)	Fecha de la factura.
TX_CUENTA	varchar(60)	Cuenta a cobrar
CA_SUBTOTAL	double	Cantidad del sub-total a cobrar.
CO_OPERADORA	int(11)	Código único de operadora.
CA_IVA	double	Cantidad del impuesto al valor agregado.
FE_PAGUE_ANES	varchar(60)	Fecha tope para el pago de la factura.
CA_MONTO_ANTERIOR	double	Cantidad del monto anterior por pagar.
CA_TOTAL_PAGAR	double	Cantidad del total a pagar de la factura.

Fuente: Elaboración Propia.

Figura 4.32: Imagen de la tabla “tr018_cuenta_general”.
Fuente: Elaboración Propia.

tr017_

PK co_

fe_

fe_

FK1 co_

ca_

tx_



Tabla “tr019_rela_divi_distri”: Esta tabla relaciona las diferentes Divisiones que conforman a PDVSA con sus respectivos Distritos. Ver Tabla 4.19.

Tabla 4.19: Descripción de la tabla “tr019_rela_divi_distri”

tr019_rela_divi_distri		
Columna	Tipo	Descripción
CO_DIVISION	int(11)	Código único de la División.
CO_DISTRITO	int(11)	Código único de Distrito.

Fuente: Elaboración Propia.

tr019_rela_divi_distri	
Columna	Tipo
CO_DIVISION	int(11)
CO_DISTRITO	int(11)

Figura 4.33: Imagen de la tabla “tr019_rela_divi_distri”.

Fuente: Elaboración Propia.

Tabla “tr020_rela_dist_area”: Esta tabla relaciona los diferentes Distritos que conforman a PDVSA con sus respectivas Áreas. Ver Tabla 4.20.

Tabla 4.20: Descripción de la tabla “tr020_rela_dist_area”.

tr020_rela_dist_area		
Columna	Tipo	Descripción
CO_DISTRITO	int(11)	Código único de Distrito.
CO_AREA	int(11)	Código único de Área.

Fuente: Elaboración Propia.

Tabla “tr021_eventos”: Registra los eventos que se pueden producir en el sistema, los cuales pueden ser inserción, actualización y eliminación. Ver Tabla 4.21.

tr019_re

**PK,FK1
PK,FK2**



tr020_rela_dist_area	

Figura 4.34: Imagen de la tabla “tr020_rela_dist_area”.
Fuente: Elaboración Propia.

Tabla 4.21: Descripción de la tabla “tr021_eventos”

tr021_eventos		
Columna	Tipo	Descripción
CO_EVENTO	int(11)	Código único del Evento.
NB_EVENTO	varchar(60)	Nombre del tipo de evento (Ingresar, Eliminar, etc).

Fuente: Elaboración Propia.

tr021_eventos	

Figura 4.35: Imagen de la tabla “tr021_eventos”.
Fuente: Elaboración Propia.

Tabla “tr022_historial_eventos”: Registra el historial de los eventos producidos en el sistema, el código afectado por el evento ya que puede ser un teléfono o un indicador y una breve descripción del mismo. Ver Tabla 4.22.

Tabla 4.22: Descripción de la tabla “tr022_historial_eventos”. (1/2)

tr022_historial_eventos		
Columna	Tipo	Descripción
CO_HIST_EVENTOS	int(11)	Código único del histórico de evento.
CO_INDICADOR	varchar(60)	Código de usuario que produce el evento.
CODIGO_AFECTADO	varchar(60)	Código afectado por el evento.
CO_EVENTO	int(11)	Código único del evento.

Fuente: Elaboración Propia.

tr020_r

PK,FK1

PK,FK2

tr02



Tabla 4.22: Descripción de la tabla “tr022_historial_eventos”. (2/2)

tr022_historial_eventos		
Columna	Tipo	Descripción
FE_FECHA	datetime	Fecha del Evento.
NB_DESCRIPCION	varchar(300)	Breve descripción del Evento.

Fuente: Elaboración Propia.

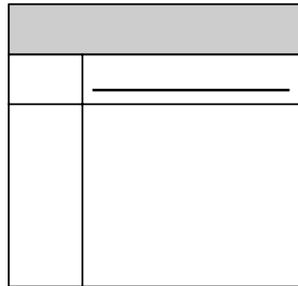


Figura 4.36: Imagen de la tabla “tr022_historial_eventos”.

Fuente: Elaboración Propia.

Tabla “tr023_responsables”: Registra la información referente a los responsables de los teléfonos que forman parte del sistema, guardando la fecha de inicio y de fin de un responsable de teléfono, así como el número de teléfono y el indicador del responsable de ese número. Ver Tabla 4.23

Tabla 4.23: Descripción de la tabla “tr023_responsables”.

tr023_responsables		
Columna	Tipo	Descripción
CO_RESPONSABLE	int(11)	Código único de Responsable.
CO_INDICADOR	int(11)	Código de usuario responsable del teléfono.
CO_NUMERO	varchar(60)	Código de teléfono.
FE_INICIO	datetime	Fecha de inicio del responsable.
FE_FIN	datetime	Fecha de fin del responsable.

Fuente: Elaboración Propia.

tr022_hi

PK co

FK2 co

co

FK1 co

fe

nb



Figura 4.37: Imagen de la tabla “tr023_responsables”.
Fuente: Elaboración Propia.

4.3.3 Modelo de Hipertexto

El objetivo del modelo de hipertexto es modelar a un alto nivel la interfaz de usuario de una aplicación Web, usando una simple, todavía formal, notación visual permitiendo la generación automática de plantillas de páginas dinámicas y acceso a datos y manipulación de consultas.

El modelo de hipertexto especifica cómo se compone el software en función de las páginas que forman la vista para el usuario, y de la navegación por cada una de estas, a continuación se muestra la Figura 4.38 donde se visualiza el modelo de hipertexto del sistema SAFREST enfocado a la vista Administrador ya que en esta se observa el funcionamiento completo del sistema.

En el modelo de hipertexto se observan Data Unit como, **Acceso** que involucra las tablas de *usuario_rol* y *LDAP* de PDVSA, **Datos Gerenciales** que involucra las tablas *divisiones*, *distritos* y *organizaciones*, **DatosCarga** que involucra las tablas *cuenta_general*, *movilGeneral* y *detalleMovil* y finalmente el Data Unit **Responsable** que involucra las tablas *teléfonos* y *responsables*. El resto de los Data Unit involucra las tablas de su respectivo nombre.

tr023_

PK co

FK1 co

FK2 co

fe_
fe_

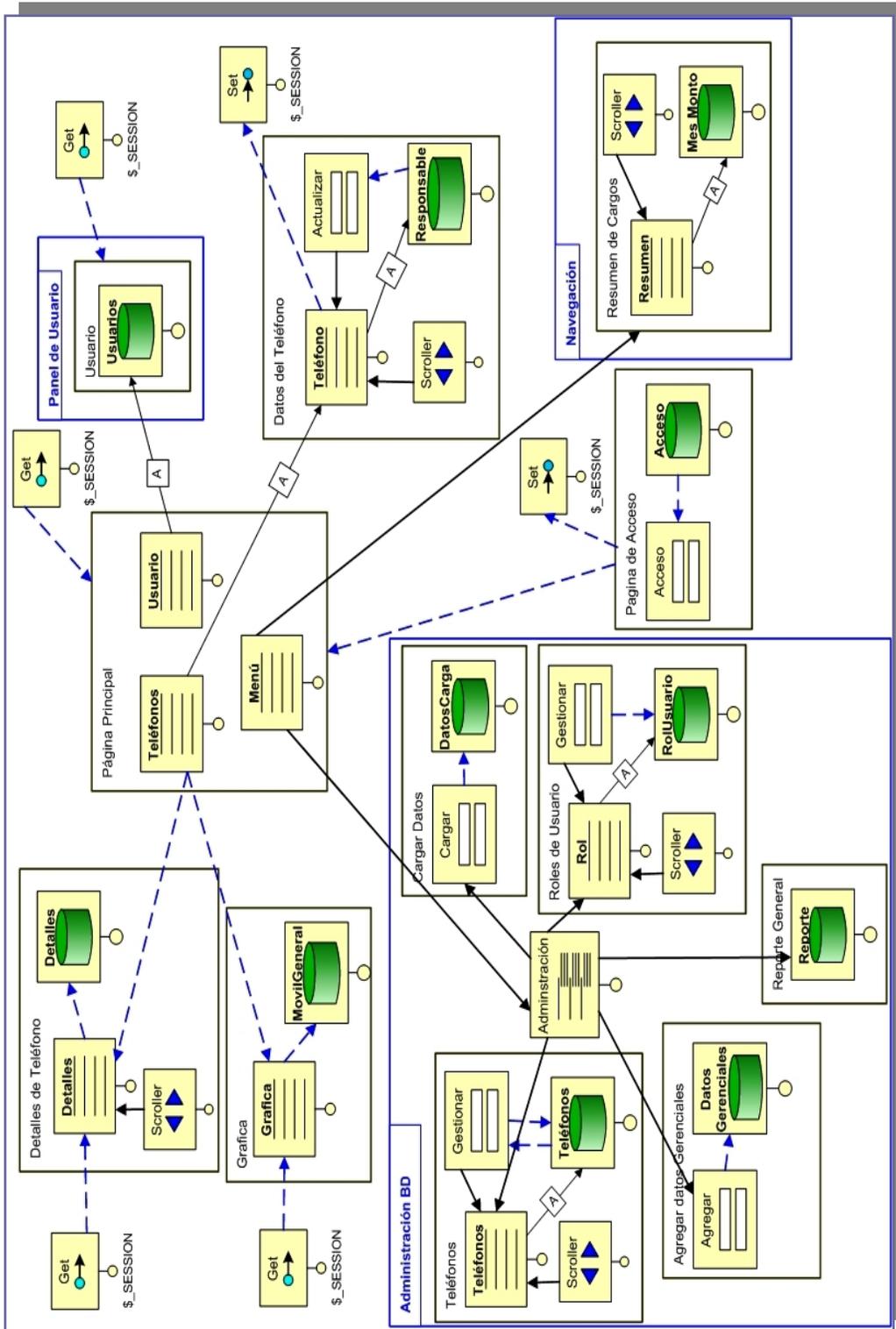


Figura 4.38: Modelo de Hipertexto de la Aplicación “SAFREST”.
Fuente: Elaboración Propia.



4.3.4 Modelo de gestión de contenidos

Las aplicaciones web realizan con frecuencia operaciones en los datos, ejemplo de ello son las adiciones de elementos en un carrito de compra virtual o la actualización de contenidos publicados en la web. En todos estos casos, las acciones realizadas a través de la interfaz web tienen efectos secundarios como el de cambiar el contenido de algunos datos fuentes conectados al sitio web.

Además de actualizar los datos, las aplicaciones web pueden invocar programas definidos externamente, dotándoles de entradas que dependen del contenido de la página actual o de las selecciones de los usuarios, ejemplo de esto son las operaciones de acceso de un usuario, el envío de correos electrónicos y así sucesivamente

El modelo de Gestión de Contenidos (MGC) viene a ser una extensión del modelo de Hipertexto, que se constituye en dos ampliaciones: La primera añade unidades para la manipulación de datos y unidades para la ejecución de servicios externos y la segunda se refiere a los enlaces salientes de las unidades de operaciones (OK-KO) para la captura del éxito o fracaso de las operaciones permitiendo al diseñador plasmar cursos alternativos dependiendo de los resultados de las operaciones.

A continuación se presenta el modelo de Gestión de Contenidos de la Aplicación SAFREST. Ver Figura 4.39, la cual esta centrado a los casos de uso Consultar datos de teléfono, Cargar datos de teléfono, Actualizar datos de teléfono, Asignar Roles, Asignar datos gerenciales.

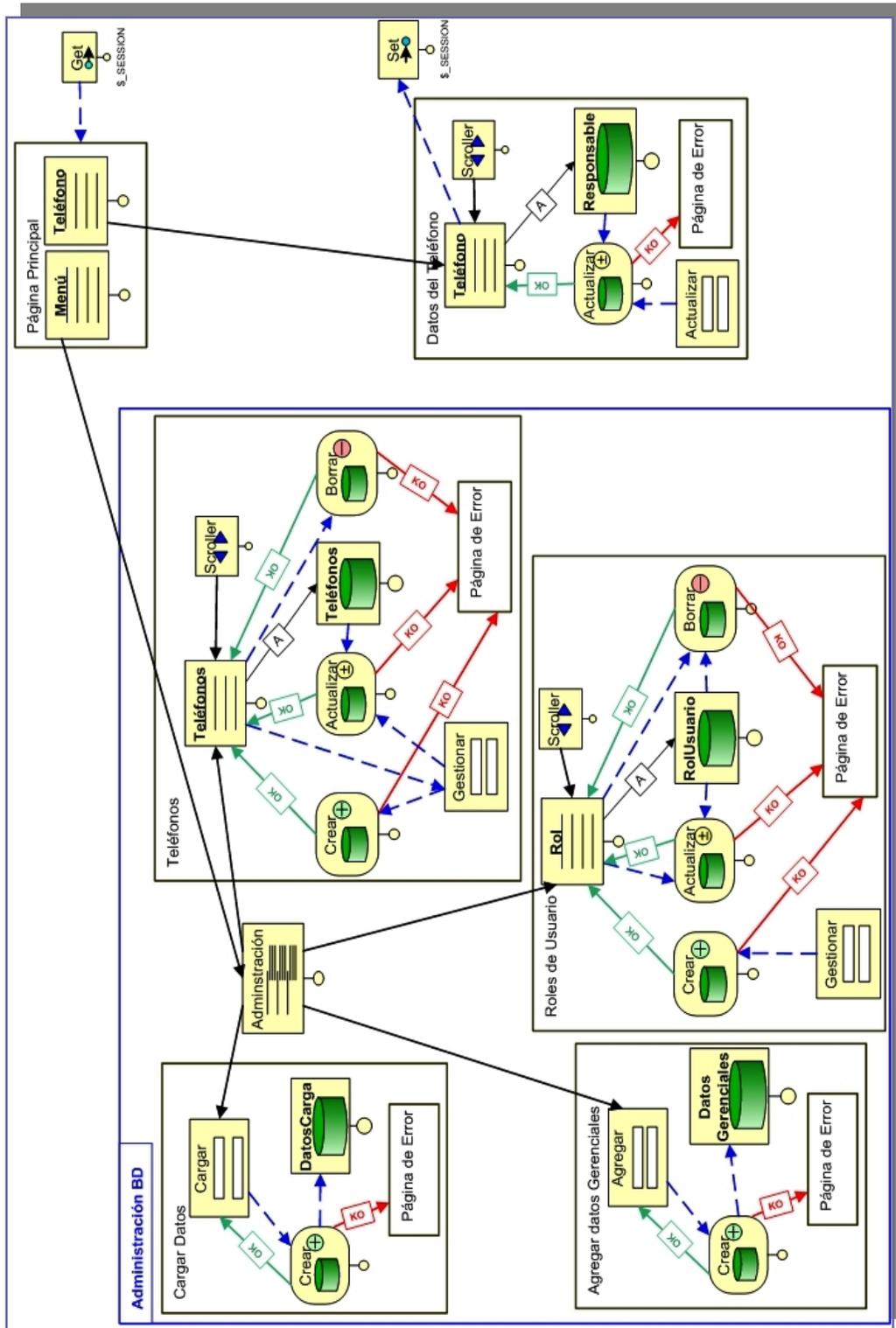


Figura 4.39: Modelo de Gestión de Contenidos de la Aplicación “SAFREST”.
Fuente: Elaboración Propia



4.3.5 Modelo de personalización

La personalización tiene tres factores que el diseñador debe tomar en cuenta a la hora de llevarla a cabo durante el proceso de modelado de la aplicación. Es importante destacar que la personalización no es más que la definición de los filtros a los que la información será sometida para mantener su integridad y seguridad dentro del sistema según los usuarios que tengan acceso a ella. Los factores antes mencionados serán descritos a continuación.

Control de Acceso: se refiere a la implementación de claves de inicio y cierre de sesión por usuario para de esta manera delimitar las operaciones correspondientes y permitidas para cada usuario.

Vistas del Sitio Asignadas: dependiendo del grupo al que el usuario pertenece, algunas vistas del sitio son accesibles, una o más vistas del sitio por grupo.

Personalización de Páginas: el contenido de las páginas depende del usuario o del grupo al cual pertenecen.

Claves de Inicio y Cierre de Sesión para los usuarios

Un sitio puede contener una página que permita al usuario conectarse a un área restringida del sistema mediante un inicio de sesión. Cada sitio por seguridad debe permitir el cierre de sesión para los usuarios, de esta forma se evita la entrada de usuarios no autorizados. En caso de que el usuario pertenezca a más de un grupo se debe incluir la opción de cambio de grupo sin necesidad de salir del sistema.

En el proceso de modelado de la operación de inicio de sesión (login), cierre de sesión (logout) y cambio de grupo de usuario, se representan mediante símbolos los elementos que permiten la ejecución de cada una de estas actividades en el sistema (elementos de validación de usuarios) con el



fin de proporcionar un mejor entendimiento de ellas. La simbología utilizada se muestra a continuación. (Ver Figura 4.40).

La

Figura 4.41 y Figura 4.42 muestran la implementación del modelo de personalización utilizado en el sistema para lograr la autenticación de los usuarios y la finalización de sesión.



Figura 4.40: Símbolos para la personalización del sistema.
Fuente: Elaboración Propia.

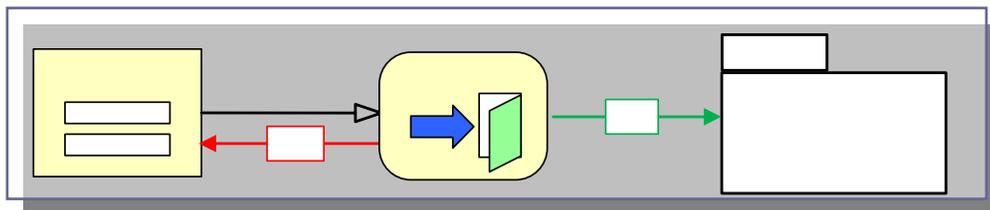


Figura 4.41: Autenticación de Usuarios del SAFREST.
Fuente: Elaboración Propia.

4.4 Implementación

En la implementación se comenzará con el resultado del diseño y se implementa el sistema en términos de componentes. Para esta fase solo se creará la línea base ejecutable de la arquitectura. Se implementarán las interfaces de validación de usuarios y bienvenida a la aplicación, debido a que la implementación es el centro durante las iteraciones de la fase de Construcción.

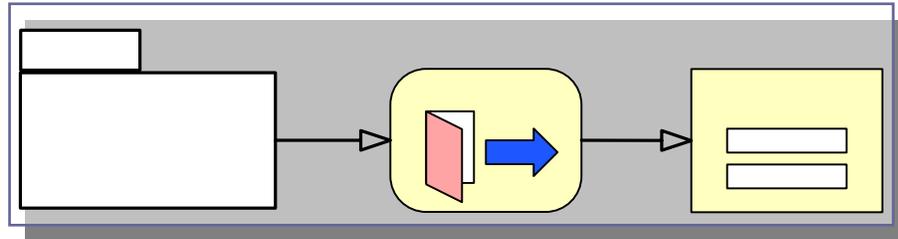


Figura 4.42: Finalización de Sesión.

Fuente: Elaboración Propia.

4.4.1 Identificación de componentes de la arquitectura

Para poder implementar la arquitectura de este sistema, se identificaron los componentes mostrados en la Figura 4.43, ésta muestra el servidor web que permite la distribución de las páginas, el motor de PHP que ejecuta los scripts que manejan la configuración del sistema, y el sistema manejador de base de datos (DBMS) MySQL. Se muestra también un cliente que contiene el navegador web el cual genera un documento HTML, permitiendo a los usuarios visualizar e interactuar con la aplicación. El cliente y el servidor se conectan a través de la Intranet mediante el protocolo de transmisión http.

4.4.2 Implementación de la arquitectura

En esta fase se implementa la arquitectura a través de la construcción de un prototipo de la página principal de inicio de sesión denominada index, la cual se puede ver en la Figura 4.44, la cual conforma el caso de uso Acceder, y se ejecutan los componentes que conforman esta arquitectura. Esta página está realizada utilizando HTML, mostrando la interfaz de inicio de sesión y conteniendo las instrucciones necesarias para conectar y consultar la base de datos.

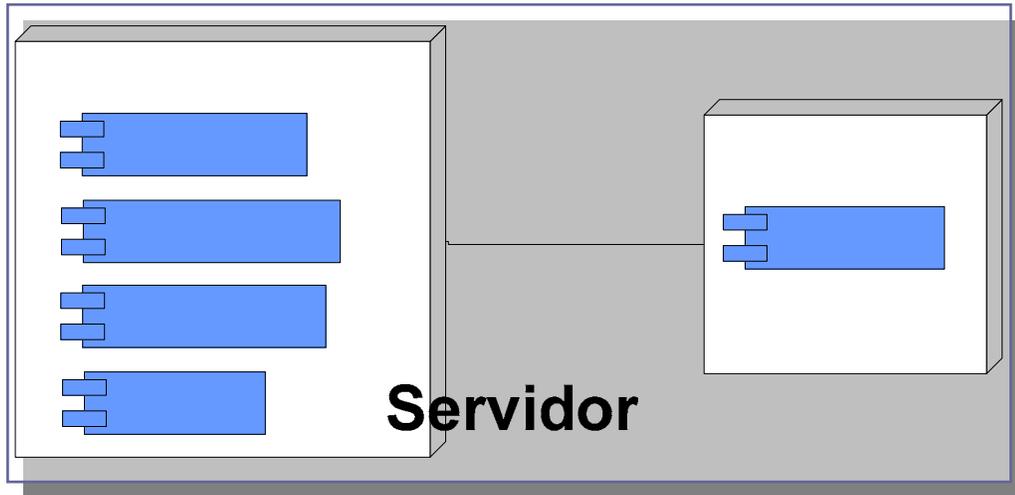


Figura 4.43: Diagrama de Despliegue para el sistema SAFREST.

Fuente: Elaboración Propia.

Sistema SAFREST

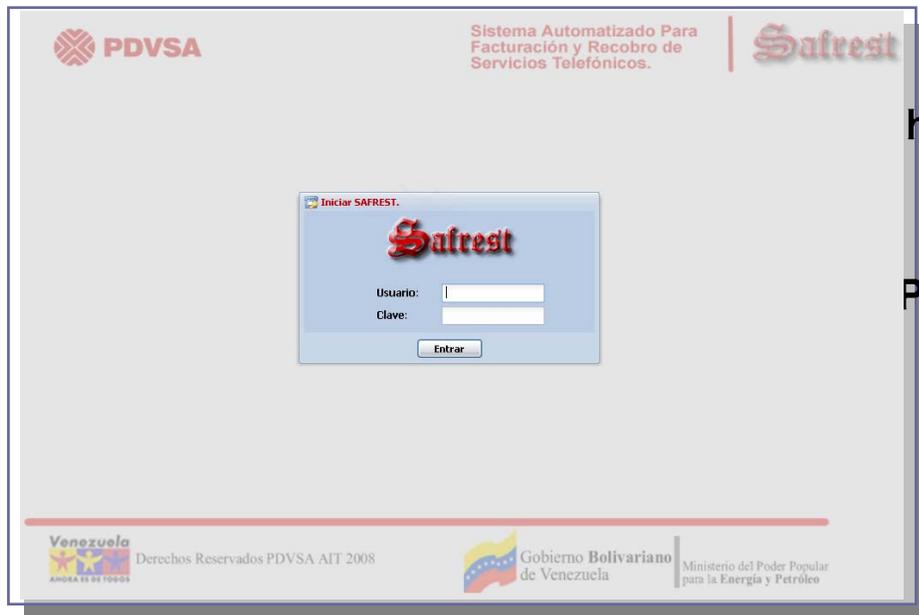


Figura 4.44: Pantalla de inicio de sesión Sistema SAFREST.

Fuente: Elaboración Propia.



Código Fuente: Pantalla de inicio de sesión

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Safrest</title>
<link rel="stylesheet" type="text/css" href="css/ext-all.css" />
<!-- GC -->
<!-- LIBS -->
<script type="text/javascript" src="js/ext-base.js"></script>
<!-- ENDLIBS -->
<script type="text/javascript" src="js/ext-all-debug.js"></script>
<script language="javascript" type="text/javascript"
src="js/ObjetoAjax.js"></script>
<link rel="shortcut icon" href="images/Ima/itel2.gif" />
<style type="text/css">
html, body {
font:normal 12px verdana;
margin:0;
padding:0;
border:0 none;
overflow:hidden;
height:100%;
}
.telefonoKey {
background-image:url(images/icons/fam/telephone_key.png)
!important;
```



```
    }
</style>
<script language="JavaScript" type="text/javascript">
    Ext.namespace("Ext.ux.Login");

    Ext.SSL_SECURE_URL = 'images/default/s.gif';
    Ext.BLANK_IMAGE_URL = 'images/default/s.gif';

    Ext.ux.Login = function(){
        var win,
            FormPanel,
            Win,
            indexUrl = 'principal.html',
            submitUrl = 'Prueba/llevarCombos.php?f=login';

        function onSubmit(){

            if(Ext.isEmpty(Ext.util.Format.trim(Ext.get('Login').getValue()))) {
                Ext.MessageBox.show({
                    title: 'Error',
                    msg: '<b>Escriba un Usuario</b>',
                    buttons: Ext.MessageBox.OK,
                    animEl: 'Botao',
                    closable:false,
                    icon: Ext.MessageBox.ERROR,
                    fn: function(){
                        Ext.getCmp('Login').setValue("");
                        Ext.getCmp('Login').focus();
                    }
                });
            }
        }
    }
}
```



```
});  
return false;  
}
```

```
if(Ext.isEmpty(Ext.util.Format.trim(Ext.get('Senha').getValue()))){  
    Ext.MessageBox.show({  
        title: 'Error',  
        msg: '<b>Escriba su Clave</b>',  
        buttons: Ext.MessageBox.OK,  
        animEl: 'Botao',  
        closable:false,  
        icon: Ext.MessageBox.ERROR,  
        fn: function(){  
            Ext.getCmp('Senha').setValue("");  
            Ext.getCmp('Senha').focus();  
        }  
    });  
return false;  
}
```

```
if (FormPanel.form.isValid()) {  
    FormPanel.form.submit({  
        waitTitle: 'Por favor aguarde',  
        waitMsg: 'Validando usuario...',  
        reset: false,  
        success: Ext.ux.Login.Success,  
        failure: Ext.ux.Login.Failure,  
        scope: Ext.ux.Login
```



```
    });  
  }  
}  
  
return{  
  Init:function(){  
    Ext.QuickTips.init();  
    Ext.form.Field.prototype.msgTarget = 'side';  
  
    var LogoPanel = new Ext.Panel({  
      baseCls: 'x-plain',  
      id: 'login-logo',  
      region: 'center'  
    });  
  
    if(!(FormPanel instanceof Ext.form.FormPanel)){  
      FormPanel = new Ext.form.FormPanel({  
        labelWidth:150,  
        region: 'south',  
        method: 'POST',  
        url: submitUrl,  
        baseCls: 'x-plain',  
        bodyStyle: (Ext.isIE? 'padding-left: 3px;' :  
'padding:85px      5px      5px      5px;border:none;background:#bacce4  
url(images/logo_inicio.png) no-repeat center top;'),  
        baseParams: {module: 'Login'},  
        onSubmit: Ext.emptyFn,  
        waitMsgTarget: false,
```




```
        tabIndex:2
    }],
    buttonAlign:'center',
    buttons: [{
        id:'Botao',
        text:'<b>Entrar</b>',
        type: 'submit',
        handler: onSubmit,
        scope: Ext.ux.Login
    }]
});
}

if(!(Win instanceof Ext.Window)){
    Win = new Ext.Window({
        title: 'Iniciar SAFREST.',
        iconCls: 'telefonoKey',
        width:350,
        height:(Ext.isIE? 207 : 200),

        plain: true,
        collapsible: false,
        resizable: false,
        closable: false,
        modal: true,
        border: false,
        keys: [{
            key: Ext.EventObject.ENTER,
```




```
configuraci&oacute;n</b>",
    msg: " &nbsp;<b>Cargando su
progressText: 'Accesando al sistema...',
width:300,
wait:true,
waitConfig:{interval:1500}
});
setTimeout(function(){
    Ext.MessageBox.hide();
    window.location = indexUrl;
}, 1200);
Win.destroy(true);
}
},

Failure: function(formPanel, action) {
    Ext.MessageBox.show({
        title: 'Error',
        msg: '<b>'+ action.result.msg.text +'</b>',
        buttons: Ext.MessageBox.OK,
        animEl: 'Botao',
        closable:false,
        icon: Ext.MessageBox.ERROR,
        fn:function(){
            if( action.result.msg.code == '2' ){
                Ext.getCmp('Senha').setValue("");
                Ext.getCmp('Login').focus();
            } else if(action.result.msg.code == '3') {
```



```
        Ext.getCmp('Senha').focus();
    }
}
});

}
};
}0;
```

```
Ext.BasicForm.prototype.afterAction=function(action, success){
    this.activeAction = null;
    var o = action.options;
    if(o.waitMsg){
        Ext.MessageBox.updateProgress(1);
        Ext.MessageBox.hide();
    }
    if(success){
        if(o.reset){
            this.reset();
        }
        Ext.callback(o.success, o.scope, [this, action]);
        this.fireEvent('actioncompleted', this, action);
    }else{
        Ext.callback(o.failure, o.scope, [this, action]);
        this.fireEvent('actionfailed', this, action);
    }
}
}
```

```
Ext.onReady(Ext.ux.Login.Init, Ext.ux.Login, true);
```



```
</script>

</head>

<body>
<script type="text/javascript" src="js/examples.js"></script><!-- EXAMPLES -->
<script type="text/javascript" src="js/ext-lang-es.js"></script>

</body>
</html>
```

Como prueba se introdujeron los datos de usuario Luisa y contraseña 123456, la cual no está contenida en la tabla usuarios obteniendo el resultado de la Figura 4.45.

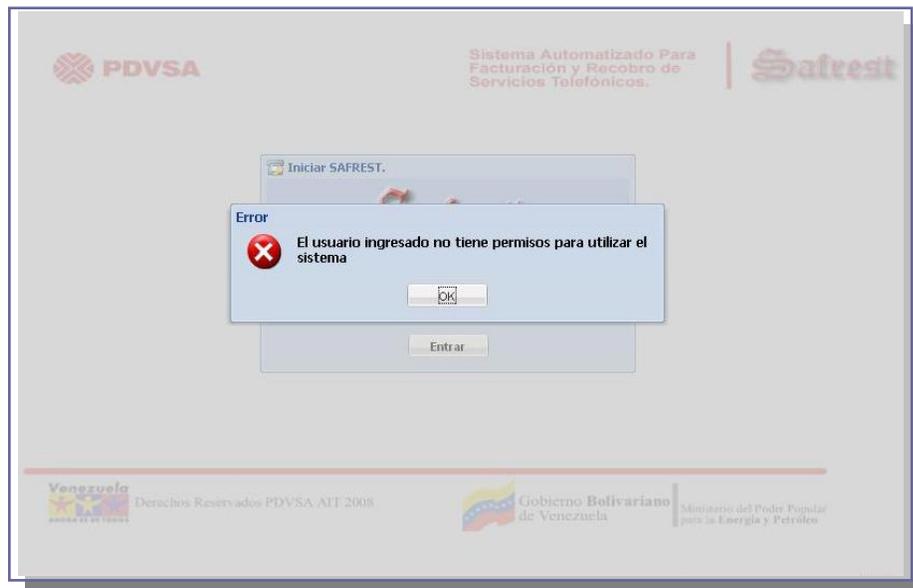


Figura 4.45: Página index.html usuario no válido.
Fuente: Elaboración Propia.



Con esto se pudo comprobar el funcionamiento de la arquitectura, ejecutándose los componentes que la conforman, el servidor web, el motor de scripts PHP y el manejador de base de datos MySQL, así como el servidor web que permite la interacción entre el cliente y el servidor utilizando el protocolo de Internet http.

CAPÍTULO V

FASE DE CONSTRUCCIÓN

5.1 Introducción

La fase de construcción tiene como propósito primordial dejar producto de software en su versión operativa inicial en el entorno de usuario, comúnmente llamada “versión beta”. El producto debería tener la calidad adecuada para su aplicación y asegurarse de cumplir con los requisitos, esta fase es análoga al desarrollo, por tanto el énfasis se traslada de la acumulación del conocimiento básico necesario para construir el proyecto a la construcción propiamente dicha de un sistema o producto.

Las fases anteriores han reducido los riesgos críticos y significativos a niveles rutinarios que pueden ser gestionados mediante el plan de construcción. En esta fase se integran los subsistemas y se prueban, luego se integra todo el sistema y se prueba.

Los flujos trabajados en esta fase son: Implementación y prueba, en implementación se podrán observar los subsistemas con los componentes que los integran y las interfaces con sus respectivos bloques de código, y en prueba todas las pruebas realizadas para obtener el correcto funcionamiento del sistema SAFREST.

5.2 Flujo de trabajo implementación

A continuación mostraremos la descripción de dos áreas principales dentro de la aplicación Web luego de su implementación. Se llevara a cabo bajo el siguiente formato:

1. Breve explicación del área (en qué consiste).



2. Identificación de los datos a manejar por el área. (datos que debe introducir el usuario durante su visita).
3. Imagen referencial de la pantalla del sistema.
4. Código Fuente.

5.2.1 Página principal del sistema SAFREST

En la Figura 5.1, se observa la página principal de la aplicación SAFREST para el tipo de vista “Administrador”, en donde podemos apreciar los módulos de Navegación y Administración BD representando estos la funcionalidad total de la aplicación.

PDVSA

Sistema Automatizado Para Facturación y Recobro de Servicios Telefónicos. **Safrest**

Menú Inicio Teléfono Detalles Gráfica

Navegación
Resumen de Cargos
Salir

Datos Usuario
Usuario

Cargos de Teléfonos

Reporte

Número	Operadora	Organización	Indicador	Total
4165002093	Movinet	A.I.T	SIN ASIGNAR	1487591.38
4165003247	Movinet	A.I.T	SIN ASIGNAR	522641.59
4165001716	Movinet	A.I.T	SIN ASIGNAR	1160842.38
4165001785	Movinet	A.I.T	SIN ASIGNAR	721538.81
4165001788	Movinet	A.I.T	SIN ASIGNAR	285822.25
4165001791	Movinet	A.I.T	GENED	323417.84
4165001794	Movinet	A.I.T	SIN ASIGNAR	326328.25
4165001795	Movinet	A.I.T	SIN ASIGNAR	206070.41
4165001797	Movinet	A.I.T	SIN ASIGNAR	99350.00
4165001905	Movinet	A.I.T	SIN ASIGNAR	99350.00
4165001907	Movinet	A.I.T	SIN ASIGNAR	54450.00
4165001909	Movinet	A.I.T	SIN ASIGNAR	106152.31
4168124478	Movinet	A.I.T	SIN ASIGNAR	351085.88
4168801254	Movinet	A.I.T	SIN ASIGNAR	101467.55
4169001431	Movinet	A.I.T	SIN ASIGNAR	197552.00
4169000709	Movinet	MESON BARRIO ADETRTO	SIN ASIGNAR	1267062.75
4169210059	Movinet	MESON BARRIO ADETRTO	SIN ASIGNAR	845469.81
4169000576	Movinet	MESON BARRIO ADETRTO	SIN ASIGNAR	1832335.25
4168800678	Movinet	MESON BARRIO ADETRTO	SIN ASIGNAR	1772629.88

Página 1 de 17

Mostrando datos del 1 - 25 de 421

NUMERO: INDICADOR: NOMBRE: APELLIDO:

Actualizar Teléfonos Buscar

Administración BD

Apellidos: CABELLO
Nombre: JOSE
Indicador: CABELLOIN

Figura 5.1: Pantalla Principal de la aplicación SAFREST.

Fuente: Elaboración Propia.



Como se pudo ver en la figura anterior la información pertinente a los datos de los servicios telefónicos prestados por PDVSA, se visualizan en la parte central de la aplicación, del lado izquierdo, se observa un menú en forma de acordeón, dentro de este se puede visualizar el árbol de ítems de las diferentes opciones que tenemos en el sistema y del lado derecho se observa la foto de la persona que esta usando la aplicación, es decir la persona que inició sesión.

Código Fuente Pantalla Principal:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Safrest</title>
<link rel="stylesheet" type="text/css" href="css/ext-all.css" />
<link rel="stylesheet" type="text/css"
href="SwfUploadPanel/SwfUploadPanel.css" />
<!-- GC -->
<!-- LIBS -->
<script type="text/javascript" src="js/ext-base.js"></script>
<!-- ENDLIBS -->
<script type="text/javascript" src="js/ext-all-debug.js"></script>
<script language="javascript" type="text/javascript"
src="js/ObjetoAjax.js"></script>
<script language="javascript" type="text/javascript"
src="js/datosGerenciales.js"></script>
<script type="text/javascript" src="SwfUploadPanel/SwfUpload.js"></script>
```



```
<script type="text/javascript"
src="SwfUploadPanel/SwfUploadPanel.js"></script>
<link rel="shortcut icon" href="images/Ima/itel2.gif" />
<style type="text/css">
html, body {
font:normal 12px verdana;
margin:0;
padding:0;
border:0 none;
overflow:hidden;
height:100%;
}
p {
margin:5px;
}
select{
height:20px;
margin:0px 0px 0px 5px;
display:inline;
border-style:double;
border-color:#CCCCCC;
border-width:1px;
color:#333333;
width:120px;
}
opacidad{
}
option{
```



```
        color:#333333;
        width:110px;
    }
    .agregar {
        background-image:url(images/gray/add.gif) !important;
    }
    .salir {
        background-image:url(images/icons/fam/telephone.png) !important;
    }
    .desconectar {
        background-image:url(images/icons/fam/desconectar.jpg) !important;
    }
    .agregarTelefono {
        background-image:url(images/icons/fam/telephone_add.png) !important;
    }
    .celular {
        background-image:url(images/icons/fam/phone.png) !important;
    }
    .eliminar {
        background-image:url(images/gray/delete.gif) !important;
    }
    .actualizar {
        background-image:url(images/icons/fam/actualizar.png) !important;
    }
    .refrescar {
        background-image:url(images/gray/drop-no.gif) !important;
    }
    .buscar {
        background-image:url(images/icons/fam/magnifier.png) !important;
```



```
    }
.settings {
    background-image:url(images/icons/fam/folder_wrench.png);
}
.nav {
    background-image:url(images/icons/fam/folder_go.png);
}
.imgUsuario {
    background-image:url(images/icons/fam/user_gray.png);
}
.agregarUsuario {
    background-image:url(images/icons/fam/user_add.png);
}
.tabs {
    background-image:url(images/icons/fam/grid.png) !important;
}
.torta {
    background-image:url(images/gray/torta1.png) !important;
}
.inicio {
    background-image:url(images/icons/fam/house.png) !important;
}
.icoacordion {
    background-image:url(images/icons/fam/accordion.gif) !important;
}
.icopdf {
    background-image:url(images/icons/fam/pdf.gif) !important;
}
.cargarBD {
```



```
        background-image:url(images/icons/fam/telephone_go.png) !important;
    }
    .cargarMS {
        background-image:url(images/icons/fam/logo_movistar.jpg) !important;
    }
    .telefono {
        background-image:url(images/icons/fam/telephone.png) !important;
    }
    .listarTlf {
        background-image:url(images/icons/fam/application_go.png) !important;
    }
    .subirArch {
        background-image:url(images/icons/arrow_up.png) !important;
    }
    .contentInner
    {
        position:absolute;
        font-family: Arial, Helvetica, sans-serif;
        font-size: 12px;
        border: none;
        width: 100%;
        height: 100%;
        top:0%;
        left:0%;
    }
    body {
        background-color: #FFFFFFF;
    }
</style>
```



```
<script type="text/javascript">
Ext.SSL_SECURE_URL = 'images/default/s.gif';
Ext.BLANK_IMAGE_URL = 'images/default/s.gif';
var count=0;
var ventanaSubirArch;
Ext.QuickTips.init();
Ext.form.Field.prototype.msgTarget = 'side';

var tree1 = new Ext.tree.TreePanel({
animate:true,
enableDD:true,
containerScroll: true,
ddGroup: 'organizerDD',
rootVisible:false,
border:false,
width:200,
split:true,
autoScroll:true
});

var root1 = new Ext.tree.TreeNode({
text: 'Navegar Sistema',
allowDrag:false,
allowDrop:false
});
tree1.setRootNode(root1);

var tree2 = new Ext.tree.TreePanel({
animate:true,
```



```
enableDD:true,
containerScroll: true,
ddGroup: 'organizerDD',
rootVisible:false,
border:false,
width:200,
split:true,
autoScroll:true
});

var root2 = new Ext.tree.TreeNode({
text: 'Agregar Tel&eacute;fono',
allowDrag:false,
allowDrop:false
});
tree2.setRootNode(root2);

var buscar1 = new Ext.tree.TreeNode({
    text:'Listar Tel&eacute;fonos',
    allowDrag:false,
    iconCls:'listarTlf',
    listeners :{
        click: function(){
            mostrarTab('Prueba/anadirTelefono.html', 'Agregar
Tel&eacute;fono', 'agregar02');
        }
    }
});
```



```
var cBaseDatos = new Ext.tree.TreeNode({text:'Cargar Base Datos',
allowDrag:false});
```

```
var buscar2 = new Ext.tree.TreeNode({
    text:'Movilnet',
    allowDrag:false,
    iconCls:'cargarBD',
    listeners :{
        click: function(){
            Ext.MessageBox.confirm('CONFIRMACION',
'&iquest;'+ 'Seguro que desea realizar la carga de la Base de Datos?',
cargar_bd_movilnet);
        }
    }
});
```

```
var buscar3 = new Ext.tree.TreeNode({
    text:'Movistar',
    allowDrag:false,
    iconCls:'cargarBD',
    listeners :{
        click: function(){
            Ext.MessageBox.confirm('CONFIRMACION',
'&iquest;'+ 'Seguro que desea realizar la carga de la Base de Datos?',
cargar_bd_movistar);
        }
    }
});
```

```
var buscar4 = new Ext.tree.TreeNode({
```



```
        text:'Resumen de Cargos',
        allowDrag:false,
        iconCls:'listarTlf',
        listeners :{
            click: function(){
                mostrarTab('Prueba/cargosGerencias.html', 'Resumen de
Cargos', 'cargosGerencias02');
            }
        }
    });
    var buscar5 = new Ext.tree.TreeNode({
        text:'Reporte General',
        allowDrag:false,
        iconCls:'icopdf',
        listeners :{
            click: function(){

                mostrarVentanaPDF('pdfReporteGeneral','ArchivosControl/reportesPDF.php?f
=reporteGeneral','Reporte General');
            }
        }
    });
    var rolesUsuarios = new Ext.tree.TreeNode({
        text:'Roles de Usuarios',
        allowDrag:false,
        iconCls:'listarTlf',
        listeners :{
            click: function(){
```



```
        mostrarTab('Prueba/roles.html', 'Lista de Roles',
'listaRolUsua');
    }
}
});
```

```
var agregarDatosGeren = new Ext.tree.TreeNode({
    text:'Agregar Datos Gerenciales',
    allowDrag:false,
    iconCls:'listarTlf',
    listeners :{
        click: function(){
            ventanaGerenTabs();
        }
    }
});
```

```
var salir = new Ext.tree.TreeNode({
    text:'Salir',
    allowDrag:false,
    iconCls:'desconectar',
    listeners : {
        click: function(){
            cargando ();
            var objAjax = new objetoWeb.CargarAjax
('Prueba/lleñarCombos.php?f=cerrarSesion',false,null,'POST',false,false);
            cerrarLoading();
            location.href='index.html'
        }
    }
});
```



```
});

cBaseDatos.appendChild(buscar2,buscar3);
root2.appendChild(cBaseDatos,buscar1,buscar5,rolesUsuarios,agregarDatosGer
en);
root1.appendChild(buscar4,salir);

var a = new Ext.Panel({
    autoLoad:{url:'Prueba/cargosTelefono.html',scripts:true,scope:this },
    id:'telefono',
    title: 'Teléfono',
    iconCls: 'telefono',
    autoScroll:true,
    closable:false
});

var b = new Ext.Panel({
    autoLoad:{url:'Prueba/detalletelef.html',scripts:true,scope:this },
    id: 'detalle',
    disabled:true,
    title: 'Detalles',
    iconCls: 'tabs',
    autoScroll:true,
    closable:false
});

var d = new Ext.Panel({
    html:'',
```



```
        title: 'Inicio',
        iconCls: 'inicio',
        autoScroll: true,
        closable:true
    });

var c = new Ext.Panel({
    html:<iframe id="iframe02" src="Modificacion/grafica.html"
class="contentInner" scrolling="no" frameborder="0"></iframe>',
    id:'grafica',
    disabled:true,
    title: 'Gráfica',
    iconCls: 'torta',
    autoScroll:true,
    closable:false
});

var tabs = new Ext.TabPanel({
    region:'center',
    id:'centro',
    deferredRender:false, //Para que las tabs no se carguen al inicio
    activeTab:0,
    resizeTabs:true,
    minTabWidth: 115,
    tabWidth:135,
    enableTabScroll:true,
    defaults: { autoScroll:true },
    items:[d]
});
```



```
var tabsUsuario = new Ext.TabPanel({
    border:false,
    id: 'usuarioTab',
    hidden: true,
    activeTab:0,
    tabPosition:'top',
    items:[{
        autoLoad:'ArchivosControl/foto.php',
        title: 'Usuario',
        iconCls:'imgUsuario',
        autoScroll:true
    }]
});
```

```
var panelAdminiAgregarTele = new Ext.Panel({
    title: 'Administrar Tel&eacute;fonos',
    layout:'column',
    border: false,
    items:tree2
});
```

```
var panelDerechaUsu = new Ext.Panel({
    region:'east',
    title: 'Datos Usuarios',
    collapsible: true,
    split:true,
    width: 225,
    minSize: 175,
```



```
        maxSize: 400,
        layout:'fit',
        margins:'0 5 0 0',
        items:tabsUsuario
    });

var panelAcorNavega = new Ext.Panel({
    contentEl: 'west',
    title:'Navegación',
    border:false,
    iconCls:'nav',
    items:tree1
});

var panelAcorAdmini = new Ext.Panel({
    title:'Administración BD',
    html:'<p>&nbsp;&nbsp;&nbsp;</p>',
    border:false,
    items: [panelAdminiAgregarTele],
    iconCls:'settings'
});

var panelAcorDion = new Ext.Panel({
    region:'west',
    id:'west-panel',
    title:'Menú',
    split:true,
    width: 200,
    minSize: 175,
```



```
    maxSize: 400,  
    collapsible: true,  
    margins:'0 0 0 5',  
    layout:'accordion',  
    layoutConfig:{  
        animate:true  
    },  
    items: [panelAcorNavega,panelAcorAdmini]  
});
```

```
Ext.onReady(function(){  
  
    Ext.QuickTips.init();  
    Ext.form.Field.prototype.msgTarget = 'side';  
  
    Ext.state.Manager.setProvider(new Ext.state.CookieProvider());  
  
    var viewport = new Ext.Viewport({  
        layout:'border',  
        items:[  
            new Ext.BoxComponent({  
                region:'north',  
                el: 'north',  
                height:70  
            }  
        ),{  
            region:'south',  
            contentEl: 'south',
```



```
        split:true,
        height: 100,
        minSize: 100,
        maxSize: 200,
        collapsible: true,
        title:'AIT Dtto. Norte PDVSA',
        margins:'0 0 0 0'
    }, panelDerechaUsu,panelAcordion,tabs]
});

verificarRol();
function verificarRol(){
    cargando ();
    var    objAjax    =    new    objetoWeb.CargarAjax
('Prueba/lleñarCombos.php?f=revisarRol',false,null,'POST',false,false);
    cerrarLoading();
    var objeto = objAjax.respuesta;
    if (objeto == '3'){
        panelAcordion.remove(panelAcorAdmini);
    }
}

});

//Parte que forma parte del div que cargo con autoLoad en una de las tabs.
function $(id){
    return document.getElementById(id);
}

function cargando (){
```



```
var capa = $('#bigLoadingPanel');
capa.style.visibility="visible";
}

function cerrarLoading (){
    var capa = $('#bigLoadingPanel');
    capa.style.visibility="hidden";
}

function revisarSesion (){
    var objAjax = new objetoWeb.CargarAjax
('ArchivosControl/LLenarElementos.php',redireccion,'bloque='+5,'POST',false,true);
}

function buscar_dir_act(indicador,fn){
    campos = "f=buscarIndicador&indicador=" + indicador;
    var objAjax = new objetoWeb.CargarAjax
('Prueba/lLenarCombos.php',fn,campos,'POST',false,true);
}

function actualizarMesMonto(){ //Función para actualizar tabla de mes monto por
gerencia.
    cargando ();
    var objAjax = new objetoWeb.CargarAjax
('Prueba/lLenarCombos.php?f=lLenarTablaMesOrg',false,null,'POST',false,false);
    dsCargosTelefonos.load({ params: { start:0, limit:25 } });
    cerrarLoading();
    Ext.MessageBox.alert("MENSAJE",objAjax.ajax.responseText);
}
```



```
}
```

```
function redireccion (){  
    var obj = this.ajax.responseText;  
    var objJSON = eval ('(' + obj + ')');  
    var valido = objJSON.valida;  
    if (valido == 1){  
        tabs.add(a);  
        tabs.add(b);  
        tabs.add(c);  
        tabs.activate(a);  
        tabsUsuario.setVisible(true);  
        cerrarLoading();  
    }else{  
        document.location.href='index.html';  
    }  
}
```

```
//Funcion para cargar la base de datos de movilnet
```

```
function cargar_bd_movilnet (btn){  
    var campos = "f=vaciarTabla";  
    if(btn=='yes'){  
        if(confirm('¿Desea remplazar los datos del mes actual de la base de  
datos?')){  
            cargando ();  
            var objAjax = new objetoWeb.CargarAjax  
( 'Prueba/llevarCombos.php',false,campos,'POST',false,false);  
            cerrarLoading();  
        }  
    }  
}
```



```
        mostrarVentanaNormal('Subir Archivo
Movilnet','Prueba/subirArchivo3.php?nombreArchivo=factura.txt&funcion=M');
    }
}
```

```
//Funcion para el almacenamiento en la base de datos
```

```
function continuar_carga_bd_movilnet (){
    cargando ();
    var objAjax = new objetoWeb.CargarAjax
('ArchivosControl/DetalleGeneral.php',false,null,'POST',false,false);
    cerrarLoading();
    var campos = "f=eliminarArchivo&p=M";
    cargando ();
    var objAjax1 = new objetoWeb.CargarAjax
('Prueba/lleñarCombos.php',false,campos,'POST',false,false);
    cerrarLoading();
    actualizarMesMonto();
}
```

```
//Funcion para cargar la base de datos de movistar
```

```
function cargar_bd_movistar (btn){
    var campos = "f=vaciarTabla";
    if(btn=='yes'){
        if(confirm('¿Desea remplazar los datos del mes actual de la base de
datos?')){
            cargando ();
            var objAjax = new objetoWeb.CargarAjax
('Prueba/lleñarCombos.php',false,campos,'POST',false,false);
            cerrarLoading();
```



```
    }
    mostrarVentanaNormal('Subir Archivo
Movistar',"Prueba/subirArchivo3.php?nombreArchivo=movistar.mdb&funcion=T");
    }
}
```

//Funcion para el almacenamiento en la base de datos

```
function continuar_carga_bd_movistar (){
    cargando ();
    var objAjax = new objetoWeb.CargarAjax
('Prueba/llevarCombos.php?f=llevarMovistar',false,null,'POST',false,false);
    cerrarLoading();
    actualizarMesMonto();
}
```

//Funcion para actualizar contenido de un tab

```
function actualizarTab (tabId/*,title, url*/) {
var tab = tabs.getItem(tabId);
if(tab){
    tab.body.update('<iframe id="iframe02"
src="Modificacion/grafica.html" class="contentInner" scrolling="no"
frameborder="0"></iframe>');
    b.setDisabled(false);
    c.setDisabled(false);
    tabs.activate(a);
}
}
```

//Funcion para crear o mostrar tab.



```
function mostrarTab(nombre, titulo, id){
    if(titulo==null)
        titulo=nombre;
    var open = !tabs.getItem(id);
    if(open){
        tab = new Ext.Panel({
            id: id,
            title: titulo,
            autoScroll:true,
            iconCls: 'tabs',
            closable:true,
            autoScroll:true,
            autoLoad: {url:nombre, scripts: true, scope: this }
        });
        tabs.add(tab);
        tab.show();
        return;
    }
    tab = tabs.getItem(id);
    tab.show();
    return;
}
```

```
function mostrarVentanaNormal (nombre,url2){
    if (ventanaSubirArch){
        ventanaSubirArch.destroy();
    }
    ventanaSubirArch = new Ext.Window({
```

```
        layout:'fit',
```



```
width: 514,
height:230,
title: nombre,
closeAction:'hide',
modal:true,
plain: true,
autoLoad: {url:url2,
scripts: true, scope: this }
});

ventanaSubirArch.show(this);
}

//Función para crear una ventana para los iframe de pdf
function mostrarVentanaPDF (id,url,nombre){
var pdf = !Ext.getCmp(id);
if (pdf){
var stringHtml = "<iframe id='" + id + "' src='" + url + "' width=880
height=660 scrolling='no' frameborder='0'></iframe>";
ventana0123= new Ext.Window({
layout:'fit',
width: 900,
height:700,
title: nombre,
closeAction:'hide',
modal:true,
plain: true,
html: stringHtml
});
}
```




```
<table width="100%" border="0" cellpadding="0" cellspacing="0">
  <tr>
    <td align="center" valign="middle">
      </img>
    </td>
  </tr>
</table>
</div>

</body>
</html>
```

5.2.2 Área de cargos de teléfonos

Entre las actividades que conforman el área de cargos de teléfonos se encuentra el asociar un indicador válido de PDVSA, a un número de teléfono específico, generar un reporte en PDF con los teléfonos mostrados y el de mostrar los montos asociados a cada uno de los teléfonos, dependiendo los permisos que tenga el usuario para ingresar a la aplicación se mostrarán todos los números telefónicos si es Administrador o los de su organización si es Consultor. Los datos que son mostrados también se pueden filtrar de diversas formas, para así tener una visualización personalizada de los datos que más le importen al usuario que este usando el sistema. Se puede visualizar esta interfaz en la Figura 5.2.



Número	Operadora	Organización	Indicador	Total
4165802093	Movinet	A.I.T	SIN ASIGNAR	1487591.38
4165803247	Movinet	A.I.T	SIN ASIGNAR	522641.59
4165861716	Movinet	A.I.T	SIN ASIGNAR	1160842.38
4165861785	Movinet	A.I.T	SIN ASIGNAR	721538.81
4165861789	Movinet	A.I.T	SIN ASIGNAR	285622.25
4165861791	Movinet	A.I.T	GENED	323417.84
4165861794	Movinet	A.I.T	SIN ASIGNAR	326328.25
4165861795	Movinet	A.I.T	SIN ASIGNAR	286078.41
4165861797	Movinet	A.I.T	SIN ASIGNAR	99350.00
4165861865	Movinet	A.I.T	SIN ASIGNAR	99350.00
4165861867	Movinet	A.I.T	SIN ASIGNAR	54450.00
4165861869	Movinet	A.I.T	SIN ASIGNAR	186152.31
4168124478	Movinet	A.I.T	SIN ASIGNAR	351085.66
4166801254	Movinet	A.I.T	SIN ASIGNAR	101467.55
4166801431	Movinet	A.I.T	SIN ASIGNAR	197552.06
4166068709	Movinet	MISION BARRIO ADENTRO	SIN ASIGNAR	1267862.75
4166218059	Movinet	MISION BARRIO ADENTRO	SIN ASIGNAR	845469.81
4166806576	Movinet	MISION BARRIO ADENTRO	SIN ASIGNAR	1832935.25
4166806578	Movinet	MISION BARRIO ADENTRO	SIN ASIGNAR	1772629.88

Figura 5.2: Pantalla de Cargos de Teléfonos.
Fuente: Elaboración Propia.

Código Fuente de Cargos de Teléfonos:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Documento sin título</title>
```



```
<script language="javascript" type="text/javascript">

var dsCargosTelefonos;
Ext.onReady(function(){
    dsCargosTelefonos = new Ext.data.Store({
        proxy: new Ext.data.HttpProxy({
            url: 'Prueba/llevarCombos.php?f=cargosTelefonos',
            method: 'POST'
        }),
        // Crear la variable de lectura de tipo JSON
        reader: new Ext.data.JsonReader({
            root: 'datos',
            totalProperty: 'maximo',
            id: 'rCargos',
        }, [
            {name: 'numero'},
            {name: 'operadora'},
            {name: 'organizacion'},
            {name: 'indicador'},
            {name: 'nombre'},
            {name: 'apellido'},
            {name: 'total'}
        ])
    });
    dsCargosTelefonos.setDefaultSort('organizacion', 'asc');

    var cmCargosTelefonos = new Ext.grid.ColumnModel([
```



```
    header: "Número",
    dataIndex: 'numero'
  },{
    header: "Operadora",
    dataIndex: 'operadora'
  },{
    header: "Organización",
    dataIndex: 'organizacion',
    width: 180,
    align: 'left'
  },{
    header: "Indicador",
    dataIndex: 'indicador',
    align: 'right'
  },{
    header: "Nombre",
    dataIndex: 'nombre',
    hidden: true,
    align: 'right'
  },{
    header: "Apellido",
    dataIndex: 'apellido',
    hidden: true,
    align: 'right'
  },{
    header: "Total",
    dataIndex: 'total',
    align: 'right'
  }
}
```



```
]);  
cmCargosTelefonos.defaultSortable = true;  
  
var filters02 = new Ext.ux.grid.GridFilters({filters:[  
    {  
        type: 'list',  
        dataIndex: 'operadora',  
        options: ['Movilnet', 'Movistar', 'Cantv'],  
        phpMode: true  
    },  
    {type: 'string', dataIndex: 'numero'},  
    {type: 'string', dataIndex: 'indicador'},  
    {type: 'string', dataIndex: 'organizacion'},  
    {type: 'numeric', dataIndex: 'total'}  
]);  
  
var gridCargosTelefonos = new Ext.grid.GridPanel({  
title: 'Teléfonos',  
store: dsCargosTelefonos,  
    plugins: filters02,  
cm: cmCargosTelefonos,  
    iconCls: 'telefono',  
    stripeRows: true,  
    collapsible: true,  
animCollapse: true,  
    autoExpandColumn: 'organizacion',  
    height: 500,  
    width: 600,  
    loadMask: true,
```



```
sm: new Ext.grid.RowSelectionModel({
    singleSelect: true,
    listeners: {
        rowselect: function(sm, row, rec) {
            llenarForma(rec);
            var linea = rec;
            cargando ();
            var objAjax = new objetoWeb.CargarAjax
('ArchivosControl/LLenarElementos.php',false,'tlf='+linea['data']['numero']+'&bloque
='+2,'POST',false,false);

Ext.getCmp("detalleTelefono01").setText(rec['data']['numero']);

Ext.getCmp("detalleOperadora01").setText(rec['data']['operadora']);

Ext.getCmp("detalleOrganizacion01").setText(rec['data']['organizacion']);

Ext.getCmp("detalleTotal01").setText(rec['data']['total']);
        actualizarTab('grafica');
        store.load({ params: { start:0, limit:25 } });
        cerrarLoading();
        }
    }
}),
tbar:[{
    text: 'Reporte',
    iconCls: 'icopdf',
    listeners :{
        click: function(){
```



```
    mostrarVentanaPDF('pdfCargosTel','ArchivosControl/reportesPDF.php?f=reporteGridCarTel','Reporte Cargos Por Telefonos');
    }
}
}],
viewConfig: {
    forceFit:true,
    enableRowBody:true,
showPreview:true
},
bbar: new Ext.PagingToolbar({
    pageSize: 25,
    store: dsCargosTelefonos,
    plugins: filters02,
    displayInfo: true,
    displayMsg: 'Mostrando datos del {0} - {1} de {2}',
    emptyMsg: "No Hay Datos Para Mostrar"
})
});
dsCargosTelefonos.load({params:{start:0, limit:25}});

var gridFormCargosTel = new Ext.FormPanel({
    id: 'formCargosTel',
    formId:'formCargosTelefono',
    frame: true,
    labelAlign: 'top', //alineación del texto, left, top
    title: 'Cargos de Teléfonos',
    bodyStyle:'padding:5px',
```



```
width: 625,
items: [gridCargosTelefenos,
        {
            layout:'column',
            items:[{
                columnWidth:.25,
                layout: 'form',
                defaults: {width: 130},
                items: [{
                    xtype:'textfield',
                    fieldLabel: 'N&Uacute;MERO',
                    id:'cargoNumero01',
                    allowBlank:false,
                    disabled: true,
                    name: 'numero'
                }]
            },{
                columnWidth:.25,
                defaults: {width: 130},
                layout: 'form',
                items: [{
                    xtype:'textfield',
                    fieldLabel: 'INDICADOR',
                    id:'cargosIndicador01',
                    allowBlank:false,
                    name: 'indicador'
                }]
            },{
                columnWidth:.25,
```



```
        layout: 'form',
        defaults: { width: 130},
        items: [{
            xtype:'textfield',
            fieldLabel: 'NOMBRE',
            allowBlank:false,
            id:'cargoNombre01',
            name: 'nombre'
        }
    ],{
        columnWidth:.25,
        defaults: { width: 130},
        layout: 'form',
        items: [{
            xtype:'textfield',
            fieldLabel: 'APELLIDO',
            id:'cargoApellido01',
            allowBlank:false,
            name: 'apellido'
        }
    ]
    }
],
buttons: [{
    text:'Actualizar Tel&eacute;fono',
    id: 'botonCargoActua',
    iconCls: 'actualizar',
    handler:function() {if (gridFormCargosTel.form.isValid()){var
```



```
Ext.getCmp('cargoNumero01');Ext.MessageBox.confirm('CONFIRMACION',
'¿'+Seguro que desea Actualizar los datos del telefono:
'+numero.getValue()+?', actualizarAsignaTel);} }
    },{
        text: 'Buscar',
        id: 'botonCargoBuscar',
        iconCls: 'buscar',
        handler: function()
        {if(Ext.isEmpty(Ext.util.Format.trim(Ext.get('cargosIndicador01').getValue()))
        {Ext.MessageBox.show({
            title: 'Error',
            msg: '<b>Escriba el Indicador a buscar</b>',
            buttons: Ext.MessageBox.OK,
            animEl: 'Botao',
            closable:false,
            icon: Ext.MessageBox.ERROR,
            fn: function(){
                Ext.getCmp('cargosIndicador01').setValue("");
                Ext.getCmp('cargosIndicador01').focus();
            }
        });}else{var indicador =
Ext.getCmp('cargosIndicador01');llenarFormaCargo(indicador.getValue());}
        }},
        renderTo: 'container'
    });
});

function llenarFormaCargo (indicador){
```



```
campos = "f=buscarIndicador&indicador=" + indicador;
cargando ();
var objAjax = new objetoWeb.CargarAjax
('Prueba/llevarCombos.php',false,campos,'POST',false,false);
var obj = objAjax.respuesta;
var respuesta = eval ('(' + obj + ')');
if (respuesta.valido == 0){
    Ext.MessageBox.alert("ERROR","El indicador no se encuentra en el
sistema.");
} else {
    var apellido = Ext.getCmp('cargoApellido01');
    var nombre = Ext.getCmp('cargoNombre01');
    apellido.setValue(respuesta.apellido);
    nombre.setValue(respuesta.nombre);
}
cerrarLoading();
}

function eliminar_usuario () {
}

function actualizarAsignaTel () {
    var numero = Ext.getCmp('cargoNumero01');
    var indicador = Ext.getCmp('cargosIndicador01');
    var nombre = Ext.getCmp('cargoNombre01');
    var apellido = Ext.getCmp('cargoApellido01');
    var campos = "f=actualizarDatosNumero&numero=" + numero.getValue() +
"&indicador=" + indicador.getValue().toUpperCase() + "&nombre=" +
```



```
nombre.getValue().toUpperCase() + "&apellido=" +
apellido.getValue().toUpperCase();
    cargando ();
    var objAjax = new objetoWeb.CargarAjax
('Prueba/llevarCombos.php',false,campos,'POST',false,false);
    cerrarLoading();
    dsCargosTelefonos.load({ params:{ start:0, limit:25 } });
    Ext.MessageBox.alert("MENSAJE",objAjax.respuesta);
}

function llenarForma (data){
    Ext.getCmp("formCargosTel").getForm().loadRecord(data);
}
</script>

</head>

<body>
    <div id="container" style="position:absolute; top:50%; left:50%; margin-top:-
312px; margin-left:-300px;">
        </div>
</body>
</html>
```

5.2.3 Área de Resumen de Cargos

Entre las actividades que conforman el área de Resumen de Cargos se encuentran el realizar reportes con los datos mostrados y el de mostrar los mostos asociados a cada una de las organizaciones que forman parte de PDVSA, dependiendo de los permisos



que tenga el usuario para ingresar a la aplicación se mostraran todas las organizaciones, si es Administrador o solo la organización a la que pertenece el usuario si es Consultor. Los datos mostrados se pueden filtrar para tener una visualización personalizada. Se puede ver esta interfaz en la Figura 5.3.

Mes	Año	Organización	Total
Agosto	2007		63347300
Agosto	2007		447026.53
Agosto	2007		285622.25
Agosto	2007		111700
Agosto	2007		
Agosto	2007	FINANZAS	980029.38
Agosto	2007	GERENCIA DTTO NORTE	463283.75
Agosto	2007	I.P.S	354648.59
Agosto	2007	MANEJO DE GAS Y AGUA	582446.31
Agosto	2007	MANTENIMIENTO OPERACIONAL	347217.88
Agosto	2007	OPERACIONES DE PRODUCCION	756907.5
Agosto	2007	P.P.Y.G	1995303.75
Agosto	2007	P.C.P	3355408.75
Agosto	2007	PERFORACION	638098.63
Agosto	2007	PROPIEDADES Y CATASTRO	477022.69
Agosto	2007	PROYECTO CARIPITO	183867.84
Agosto	2007	RECURSOS HUMANOS	1828035
Agosto	2007	S.H.A	1164916.25

Figura 5.3: Pantalla Resumen de Cargos.

Fuente: Elaboración Propia.

Código Fuente Resumen de Cargos:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```



```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Documento sin título</title>
```

```
<script language="javascript" type="text/javascript">
```

```
Ext.onReady(function(){
    var dsCargosGerencia = new Ext.data.Store({
        proxy: new Ext.data.HttpProxy({
            url: 'Prueba/llevarCombos.php?f=cargosGerencia',
            method: 'POST'
        }),
        // Crear la variable de lectura de tipo JSON
        reader: new Ext.data.JsonReader({
            root: 'datos',
            totalProperty: 'maximo',
            id: 'rCargosGerencia',
        }, [
            {name: 'mes'},
            {name: 'year'},
            {name: 'organizacion'},
            {name: 'total'}
        ])
    });

    var cmCargosGerencia = new Ext.grid.ColumnModel([
        {header: "Mes",
            width: 50,
```



```
        dataIndex: 'mes'
    },{
        header: "A&ntilde;o",
        width: 50,
        dataIndex: 'year'
    },{
        header: "Organizaci&oacute;n",
        dataIndex: 'organizacion',
        align: 'left'
    },{
        header: "Total",
        dataIndex: 'total',
        width: 50,
        align: 'right'
    }
    ]);
cmCargosGerencia.defaultSortable = true;

var filtersGerencia = new Ext.ux.grid.GridFilters({ filters:[
    {type: 'string', dataIndex: 'mes'},
    {type: 'string', dataIndex: 'year'},
    {type: 'string', dataIndex: 'organizacion'},
    {type: 'numeric', dataIndex: 'total'}
]});

var gridCargosGerencia = new Ext.grid.GridPanel({
title:'Resumen de Cargos',
store: dsCargosGerencia,
plugins: filtersGerencia,
```



```
cm: cmCargosGerencia,
    iconCls: 'celular',
    stripeRows: true,
    collapsible: true,
animCollapse: true,
    autoExpandColumn: 'organizacion',
    height:500,
    width:600,
    loadMask: true,
sm: new Ext.grid.RowSelectionModel({
    selectRow:Ext.emptyFn
}),
viewConfig: {
    forceFit:true,
    enableRowBody:true,
showPreview:true
},
tbar:[{
    text: 'Reporte',
    iconCls: 'icopdf',
    listeners :{
        click: function(){
            mostrarVentanaPDF('pdfCargosGer','ArchivosControl/reportesPDF.php?f=repo
rteGridCarGer','Reporte Cargos Por Gerencia');
        }
    }
}],
bbar: new Ext.PagingToolbar({
```



```
    pageSize: 25,
    store: dsCargosGerencia,
        plugins: filtersGerencia,
    displayInfo: true,
    displayMsg: 'Mostrando datos del {0} - {1} de {2}',
    emptyMsg: "No Hay Datos Para Mostrar"
  }),
    renderTo: 'containerGerencia'
  });
  dsCargosGerencia.load({params:{start:0, limit:25}});
});
</script>

</head>

<body>
  <div id="containerGerencia" style="position:absolute; top:50%; left:50%;
margin-top:-250px; margin-left:-300px;">
    </div>
</body>
</html>
```

5.2.4 Área de Gestionar Teléfonos

Entre las actividades que conforman el área de Listar Teléfonos se encuentran el insertar, actualizar y mostrar los distintos teléfonos que forman parte del sistema SAFREST. Existe la opción de realizar reportes de los datos mostrados y se pueden mostrar los datos de manera personalizada por medio de los diferentes filtros que



posee el grid de visualización de la información. Se puede ver esta interfaz en la Figura 5.4.

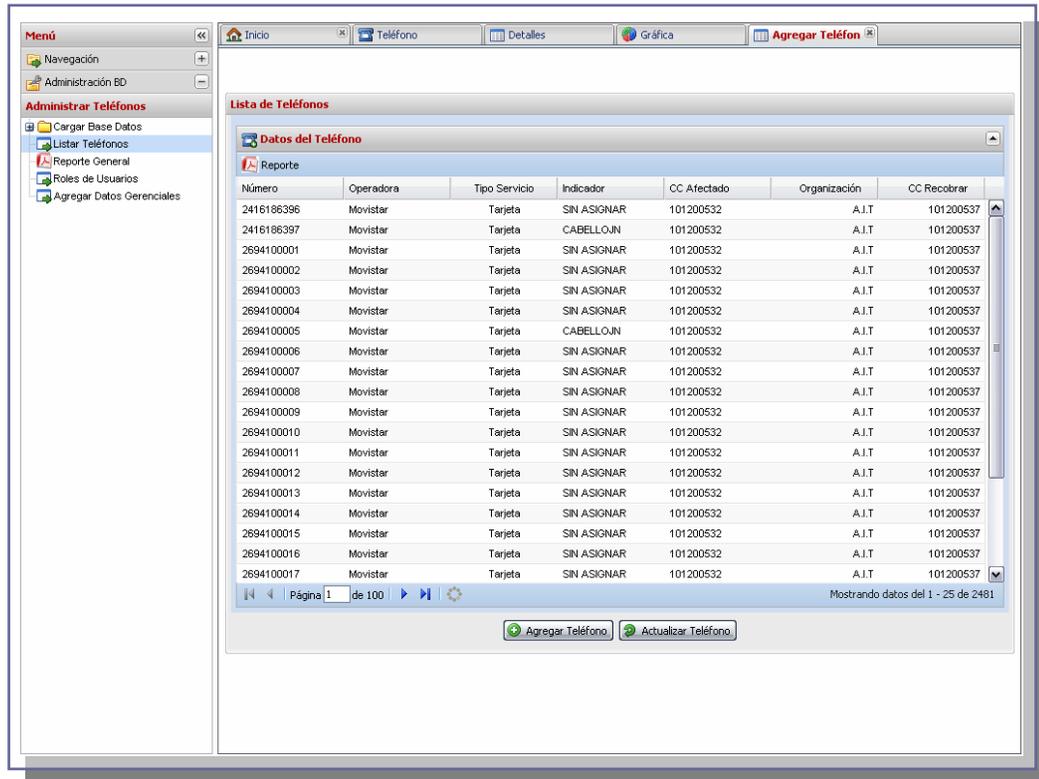


Figura 5.4: Pantalla Listar Teléfonos.

Fuente: Elaboración Propia.

Código Fuente Listar Teléfonos:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>A&ntilde;adir Tel&eacute;fono</title>
```



```
<script type="text/javascript" language="javascript">
```

```
Ext.QuickTips.init();
```

```
Ext.onReady(function(){
```

```
    Ext.QuickTips.init();
```

```
    var ventana01;
```

```
    var linea="";
```

```
    var numeroTelefono="";
```

```
    var dsOperadora02 = llenarCombo (5);
```

```
    var dsTServicio02 = llenarCombo (6);
```

```
    var dsDivision02 = llenarCombo (1);
```

```
    var dsDistrito02 = llenarCombo (2);
```

```
    var dsGerencia02 = llenarCombo (3);
```

```
    var dsLocalidad02 = llenarCombo (4);
```

```
    var dsNegocio02 = llenarCombo (7);
```

```
    var formulario01 = new Ext.FormPanel({
```

```
        labelWidth: 80,
```

```
        id: 'formuInsertElimin',
```

```
        frame:true,
```

```
        title: 'A&ntilde;adir Tel&eacute;fono',
```

```
        bodyStyle:'padding:5px 5px 0',
```



```
width: 380,
defaults: {width: 200},
defaultType: 'textfield',

items: [{

    fieldLabel: 'Indicador',
    id: 'indicador02',
    allowBlank:false,
    name: 'indicador',
    fireKey : function(e){
        if(e.getKey() == e.ENTER){
            buscar_dir_act();
        }
    }
}, {

    fieldLabel: 'Apellido',
    id: 'apellido02',
    allowBlank:false,
    name: 'apellido'

}, {

    fieldLabel: 'Nombre',
    id: 'nombre02',
    allowBlank:false,
    name: 'nombre'

}, {

    fieldLabel: 'Número',
    id: 'numero02',
    allowBlank:false,
    name: 'numero'
```



```
},{
    xtype: 'combo',
    name: 'operadora',
    id:'operadora02',
    valueField:'id_fun',
    fieldLabel: 'Operadora',
    allowBlank:false,
    store: dsOperadora02,
    lazyRender:true,
    displayField:'nombre',
    typeAhead: true,
    mode: 'local',
    editable : false,
    triggerAction: 'all',
    emptyText:'-- Seleccionar --'
},{
    xtype: 'combo',
    name: 'tipo_servicio',
    id:'tServicio02',
    valueField:'id_fun',
    fieldLabel: 'Tipo Servicio',
    allowBlank:false,
    store: dsTServicio02,
    lazyRender:true,
    displayField:'nombre',
    typeAhead: true,
    mode: 'local',
    editable : false,
    triggerAction: 'all',
```



```
        emptyText:'-- Seleccionar --'
    },{
        xtype: 'combo',
        name: 'division',
        id:'division02',
        valueField:'id_fun',
        fieldLabel: 'Division',
        allowBlank:false,
        store: dsDivision02,
        lazyRender:true,
        displayField:'nombre',
        typeAhead: true,
        mode: 'local',
        editable : false,
        listeners :{
            select: function(){
                Ext.getCmp('distrito02').clearValue();
            }
        }
    },
    triggerAction: 'all',
    emptyText:'-- Seleccionar --'
},{
    xtype: 'combo',
    name: 'distrito',
    id:'distrito02',
    valueField:'id_fun',
    fieldLabel: 'Distrito',
```



```
allowBlank:false,
store: dsDistrito02,
lazyRender:true,
displayField:'nombre',
typeAhead: true,
mode: 'local',
editable : false,
triggerAction: 'all',
emptyText:'-- Seleccionar --'
},{
xtype: 'combo',
name: 'organizacion',
id:'organizacion02',
valueField:'id_fun',
fieldLabel: 'Organizaci&oacute;n',
allowBlank:false,
store: dsGerencia02,
lazyRender:true,
displayField:'nombre',
typeAhead: true,
mode: 'local',
editable : false,
triggerAction: 'all',
emptyText:'-- Seleccionar --'
},{
xtype: 'combo',
name: 'area',
id:'area02',
valueField:'id_fun',
```



```
        fieldLabel: 'Localidad',
        allowBlank:false,
        store: dsLocalidad02,
        lazyRender:true,
        displayField:'nombre',
        typeAhead: true,
        mode: 'local',
        editable : false,
        triggerAction: 'all',
        emptyText:'-- Seleccionar --'
    },{
        xtype: 'combo',
        name: 'negocio',
        id:'negocio02',
        valueField:'id_fun',
        fieldLabel: 'Negocio',
        allowBlank:false,
        store: dsNegocio02,
        lazyRender:true,
        displayField:'nombre',
        typeAhead: true,
        mode: 'local',
        editable : false,
        triggerAction: 'all',
        emptyText:'-- Seleccionar --'
    },{
        fieldLabel: 'C/C Afectado',
        id: 'ccAfectado02',
        allowBlank:false,
```



```
        name: 'cc_afectado'
    },{
        fieldLabel: 'C/C Recobrar',
        id: 'ccRecobrar02',
        itemCls: 'required',
        allowBlank:false,
        name: 'cc_recobrar'
    }
],
    buttons: [{
        text:'Agregar Tel&eacute;fono',
        id: 'botonAgregar01',
        iconCls: 'agregar',
        handler:function() {if
(formulario01.form.isValid()){Ext.MessageBox.confirm('CONFIRMACION',
'¿'+ 'Seguro que desea realizar la siguiente operacion con el Telefono: '+
Ext.getCmp('numero02').getValue() +'?', cargar_nuevo_usuario);}}
    },{
        text: 'Reestablecer',
        iconCls: 'refrescar',
        handler: reestablecer
    },{
        text: 'Buscar',
        iconCls: 'buscar',
        handler: buscar_dir_act
    }
    ]
});
function mostrarVentana01 (indicar){
    if (!ventana01){
```



```
ventana01 = new Ext.Window({
    layout:'fit',
    width: 450,
    height:500,
    closeAction:'hide',
    modal:true,
    plain: true,
    items: formulario01 });
}
ventana01.show(this);
if (indicar == 2){
    Ext.getCmp("formuInsertElimin").setTitle('Actualizar
Tel&eacute;fono');
    Ext.getCmp("botonAgregar01").setText('Actualizar
Tel&eacute;fono');
    Ext.getCmp("botonAgregar01").setIconClass('actualizar');
    if (linea != ""){

        Ext.getCmp("formuInsertElimin").getForm().loadRecord(linea);
            numeroTelefono = Ext.getCmp("numero02").value;
            asignarValorCombo
('operadora02',Ext.getCmp("operadora02").value);
            asignarValorCombo
('tServicio02',Ext.getCmp("tServicio02").value);
            asignarValorCombo
('division02',Ext.getCmp("division02").value);
            asignarValorCombo
('distrito02',Ext.getCmp("distrito02").value);
```



```
        asignarValorCombo
('organizacion02',Ext.getCmp("organizacion02").value);
        asignarValorCombo ('area02',Ext.getCmp("area02").value);
        asignarValorCombo
('negocio02',Ext.getCmp("negocio02").value);
    }
} else{
    formulario01.form.reset();
    Ext.getCmp("formuInsertElimin").setTitle('A&ntilde;adir
Tel&eacute;fono');
    Ext.getCmp("botonAgregar01").setText('Agregar
Tel&eacute;fono');
    Ext.getCmp("botonAgregar01").setIconClass('agregar');
}
}

// Creando el Data Store
var ds01 = new Ext.data.Store({
    proxy: new Ext.data.HttpProxy({
        url: 'Prueba/llevarCombos.php?f=mostrarTelefonos',
        method: 'POST'
    }),

    // Crear la variable de lectura de tipo JSON
    reader: new Ext.data.JsonReader({
        root: 'datos',
        totalProperty: 'maximo',
        id: 'lectura',
    }, [
```



```
{ name: 'numero'},
{ name: 'operadora'},
    { name: 'tipo_servicio'},
{ name: 'indicador'},
    { name: 'nombre'},
{ name: 'apellido'},
    { name: 'cc_afectado'},
    { name: 'negocio'},
    { name: 'division'},
    { name: 'distrito'},
    { name: 'area'},
    { name: 'organizacion'},
    { name: 'cc_recobrar'}
]
});

var dsDivision01 = llenarCombo (1);

var dsDistrito01 = llenarCombo (2);

var dsGerencia01 = llenarCombo (3);

var dsLocalidad01 = llenarCombo (4);

var cm01 = new Ext.grid.ColumnModel([
    {
        header: "Número",
        dataIndex: 'numero'
    },
    {
        header: "Operadora",
```



```
    dataIndex: 'operadora',
    hidden: false
  },{
    header: "Tipo Servicio",
    dataIndex: 'tipo_servicio',
    align: 'center'
  },{
    header: "Indicador",
    dataIndex: 'indicador'
  },{
    header: "Nombre",
    dataIndex: 'nombre',
    hidden: true,
    align: 'right'
  },{
    header: "Apellido",
    dataIndex: 'apellido',
    hidden: true,
    align: 'right'
  },{
    header: "CC Afectado",
    dataIndex: 'cc_afectado',
    align: 'left'
  },{
    header: "Negocio",
    dataIndex: 'negocio',
    hidden: true,
    align: 'right'
  },{
```



```
        header: "Divisi&oacute;n",
        dataIndex: 'division',
        hidden: true,
        align: 'right'
    },{
        header: "Distrito",
        dataIndex: 'distrito',
        hidden: true,
        align: 'right'
    },{
        header: "&Aacute;rea",
        dataIndex: 'area',
        hidden: true,
        align: 'right'
    },{
        header: "Organizaci&oacute;n",
        dataIndex: 'organizacion',
        hidden: true,
        align: 'right'
    },{
        header: "CC Recobrar",
        dataIndex: 'cc_recobrar',
        width: 100,
        align: 'right'
    }
    ]);
cm01.defaultSortable = true;

var filters01 = new Ext.ux.grid.GridFilters({ filters:[
```



```
        {
            type: 'list',
            dataIndex: 'operadora',
            options: ['Movilnet', 'Movistar', 'Cantv'],
            phpMode: true
        },
        {type: 'string', dataIndex: 'numero'},
        {type: 'string', dataIndex: 'indicador'},
        {type: 'string', dataIndex: 'organizacion'}
    ]});

var grid01 = new Ext.grid.GridPanel({
    height:500,
    title:'Datos del Tel&eacute;fono',
    store: ds01,
    plugins: filters01,
    cm: cm01,
    iconCls: 'agregarTelefono',
    stripeRows: true,
    collapsible: true,
    animCollapse: true,
    autoExpandColumn: 'organizacion',
    sm: new Ext.grid.RowSelectionModel({
        singleSelect: true,
        listeners: {
            rowselect: function(sm, row, rec) {
                linea = rec;
            }
        }
    })
});
```



```
    }),
    loadMask: true,
    viewConfig: {
        forceFit:true,
        enableRowBody:true,
        showPreview:true
    },
    tbar:[{
        text: 'Reporte',
        iconCls: 'icopdf',
        listeners :{
            click: function(){
                mostrarVentanaPDF('pdfDatoTel','ArchivosControl/reportesPDF.php?f=reporte
GridDatoTel','Reporte de Datos Telefonicos');
            }
        }
    }],
    bbar: new Ext.PagingToolbar({
        pageSize: 25,
        store: ds01,
        plugins: filters01,
        displayInfo: true,
        displayMsg: 'Mostrando datos del {0} - {1} de {2}',
        emptyMsg: "No Hay Datos Para Mostrar"
    })
});

var gridForm = new Ext.FormPanel({
```



```
id: 'listaTelefono',
formId:'formListaTelefono',
frame: true,
labelAlign: 'top', //alineación del texto, left, top
title: 'Lista de Tel&eacute;fonos',
bodyStyle:'padding:5px',
width: 820,
items: [grid01],
buttons: [{
    text:'Agregar Tel&eacute;fono',
    id: 'botonAgregar',
    iconCls: 'agregar',
    handler:function() {mostrarVentana01(1);}
},{
    text: 'Actualizar Tel&eacute;fono',
    id: 'botonActualizar',
    iconCls: 'actualizar',
    handler: function() {mostrarVentana01(2);}
}],
renderTo: 'gridTelefono'
});
```

```
function cargar_nuevo_usuario(btn){
    if(btn=='yes'){
        if (formulario01.getForm().getEl().dom.elements[0].value!=""){
            var valores = new Array();
            var campos="";
```



```
valores =
['indicador02','apellido02','nombre02','numero02','operadora02','tServicio02','division
02','distrito02','organizacion02','area02','negocio02','ccAfectado02','ccRecobrar02'];
for (i=0;i<13;i++){
    var dato = Ext.getCmp(valores[i]);
    var valorDato = dato.getValue().toUpperCase();
    if (i==0){
        campos += "campo"+i+"="+valorDato+"";
    }else{
        campos += "&campo"+i+"="+valorDato+"";
    }
}
if (Ext.getCmp("botonAgregar01").getText() == 'Agregar
Tel&eacute;fono'){
    campos += "&f=agregarTelefono";
    cargando ();
    var objAjax = new objetoWeb.CargarAjax
('Prueba/llenarCombos.php',cargarTelefono,campos,'POST',false,true);
}else{
    campos += "&telefono="+numeroTelefono+"";
    campos += "&f=actualizarTelefono";
    cargando ();
    var objAjax = new objetoWeb.CargarAjax
('Prueba/llenarCombos.php',cargarTelefono,campos,'POST',false,true);
}
}
}
```



```
function cargarTelefono (){
    ventana01.hide();
    var respuesta = this.ajax.responseText;
    Ext.MessageBox.alert("Aviso",respuesta);
    ds01.load({params:{start:0, limit:25 }});
    cerrarLoading();
}

function reestablecer(){
    gridForm.form.reset();
    formulario01.form.reset();
}

function buscar_dir_act(){
    if(Ext.isEmpty(Ext.util.Format.trim(Ext.get('indicador02').getValue()))){
        Ext.MessageBox.show({
            title: 'Error',
            msg: '<b>Escriba el Indicador a buscar</b>',
            buttons: Ext.MessageBox.OK,
            animEl: 'Botao',
            closable:false,
            icon: Ext.MessageBox.ERROR,
            fn: function(){
                Ext.getCmp('indicador02').setValue("");
                Ext.getCmp('indicador02').focus();
            }
        });
    }else{
        var indicador = Ext.getCmp('indicador02').getValue();
    }
}
```



```
        campos = "f=buscarIndicador&indicador=" + indicador;
        var      objAjax      =      new      objetoWeb.CargarAjax
('Prueba/llevarCombos.php',completarCampos,campos,'POST',false,true);
    }
}
```

```
function completarCampos(){
    var obj = this.ajax.responseText;
    var respuesta = eval ('(' + obj + ')');
    if (respuesta.valido == 0){
        Ext.MessageBox.alert("ERROR","El indicador no se encuentra en
el sistema.");
    }else{
        var apellido = Ext.getCmp('apellido02');
        var nombre = Ext.getCmp('nombre02');
        apellido.setValue(respuesta.apellido);
        nombre.setValue(respuesta.nombre);
    }
    cerrarLoading();
}
```

```
function asignarValorCombo (id,valor){
    var comboPrueba = Ext.getCmp(id);
    var r = comboPrueba.findRecord (comboPrueba.displayField,valor);
    if (r){
        var indice = comboPrueba.store.indexOf(r);
        comboPrueba.setValue(r.data['id_fun']);
    }
}
```



```
    ds01.load({params:{start:0, limit:25}});
  });
</script>

</head>
<body>
<div id="gridTelefono" style="position:absolute; left:50%; top:50%; margin-left:-
410px; margin-top:-320px;"></div>
</body>
</html>
```

5.3 Flujo de trabajo Pruebas

La fase de prueba consiste básicamente en comprobar que el software realiza correctamente las tareas indicadas en la especificación. Una técnica de prueba es probar por separado cada módulo del software y luego probarlo de forma integral. Entre los principales propósitos de las pruebas tenemos:

1. Verificar la interacción entre los componentes.
2. Verificar la integración apropiada de los módulos.
3. Verificar que se satisfacen los requerimientos.
4. Identificar los defectos y corregirlos antes de la instalación.

La realización de pruebas se centra en esta fase, ya que el grueso del sistema está implementado. El resultado de la implementación se verifica probando las construcciones realizadas, las construcciones en las que se encuentren fallas deben ser



probadas nuevamente y es posible que sean devueltas a flujos de trabajos como diseño o implementación, de forma tal que sean corregidos, para así obtener un software de calidad.

5.3.1 Modelos de pruebas

Un modelo de prueba describe como se evalúan los componentes ejecutables del sistema y se realiza con la finalidad de encontrar errores a tiempo para su corrección. Para ello se implementaran un tipo de prueba muy común, la cual representa un caso de prueba que detalla cómo probar un caso de uso o un escenario específico y que incluye la verificación del resultado de la interacción entre actores y el sistema, representa el caso de prueba del sistema conocido como “caja negra”. Es importante especificar que esta prueba posee dos tipos: caja negra en lo pequeño y caja negra en lo grande.

5.3.2 Casos de pruebas

Un caso de prueba específica una forma de probar el sistema, incluyendo la entrada resultado con la que se ha de probar y las condiciones bajo las cuales ha de probarse. Existen dos casos de prueba comunes.

Un caso de prueba que especifica como probar un caso de uso o un escenario específico de un caso de uso. Un caso de prueba de este tipo incluye la verificación del resultado de la interacción entre los actores y el sistema, conocido como prueba de “caja negra”.

Un caso de prueba que especifica como probar una realización de un caso de uso diseño o un escenario específico de la realización. Un caso de prueba de este tipo



puede incluir la verificación de la interacción entre los componentes que implementan dicho caso de uso, conocido como prueba de “caja blanca”.

En el caso del sistema SAFREST se trabajó realizando el caso de prueba especificado como caja negra, y se realizaron solo para los casos de uso: Insertar datos de teléfono, Actualizar datos de teléfono, Cargar datos de teléfono, Asignar rol.

Caso de prueba del caso de uso Insertar datos de teléfono.

Entrada: hacer clic en Administrar BD, en Listar Teléfonos, en el botón Agregar Teléfono y nuevamente clic en botón Agregar Teléfono de la ventana nueva.

Salida: se deben llenar todos los campos con los datos del teléfono, se debe recibir un mensaje que indique que han sido guardados los datos del teléfono, y si falta por llenar algún campo de carácter obligatorio, se debe arrojar un mensaje indicando el o los campos que faltan por llenar.

Resultado: el sistema arroja un mensaje que los datos han sido guardados con éxito y muestra la interfaz de Listar Teléfonos con los cambios realizados.

Caso de prueba del caso de uso Actualizar datos de teléfono.

Entrada: hacer clic en Administrar BD, en Listar Teléfonos, en cualquier teléfono luego en el botón Actualizar Teléfono y nuevamente en el botón Actualizar Teléfono de la ventana nueva.

Salida: se debe cambiar el o los datos que desee con el valor nuevo a guardar, se debe recibir un mensaje que indique que ha sido actualizado el teléfono, en caso de



dejar un campo obligatorio vacío, se debe arrojar un mensaje indicando el o los campos que faltan por llenar.

Resultado: el sistema arroja un mensaje que el teléfono ha sido actualizado con éxito y muestra la interfaz de Listar Teléfono con los cambios realizados.

Caso de prueba del caso de uso Cargar datos de teléfono.

Entrada: hacer clic en Administrar BD, en Cargar Base de Datos, en Movilnet o Movistar, en el botón Si, en el botón Aceptar o Cancelar, en Agregar Archivo y en Subir Archivo.

Salida: se debe buscar el archivo que se desea cargar, los formatos permitidos son .txt para movilnet y .mdb para movistar, se debe recibir un mensaje de confirmación para realizar la carga de la base de datos, en caso de intercambiar los tipos de archivos que le corresponden a cada operadora, se debe recibir un mensaje de error indicando que el tipo de archivo no corresponde a la operadora.

Resultado: el sistema arroja un mensaje de que la Base de Datos se ha cargado de manera correcta y muestra la interfaz principal de la aplicación.

Caso de prueba del caso de uso Asignar Rol.

Entrada: hacer clic en Administrar BD, en Roles de Usuario, en Agregar y luego en Agregar Usuario en la ventana nueva.

Salida: se deben llenar todos los campos obligatorios, se debe recibir un mensaje que el usuario va a ser agregado, en caso de dejar un campo obligatorio vacío, se debe arrojar un mensaje indicando el o los campos que faltan por llenar.



Resultado: el sistema arroja un mensaje que el usuario ha sido agregado con éxito y muestra la interfaz de Roles de Usuarios con los cambios realizados.

5.3.3 Pruebas de integración

Las pruebas de integración se utilizan para verificar que los componentes interaccionan entre sí de la forma apropiada después de haber sido integrados.

El sistema SAFREST está basado en un conjunto de archivos HTML, PHP, MySQL, JavaScript y CSS, entrelazados entre si, por tal motivo, las pruebas de integración se realizaron probando todos y cada uno de los enlaces y simuladores presentes en el software.

Se navegó por todas las unidades haciendo link en cada una de las partes y procesos automatizados probando así su enlace y respuesta con las demás páginas, de igual manera se probaron los enlaces entre las páginas por medio del botón salir del menú de la aplicación.

Después de haber realizado todas las pruebas, tanto de unidad como de integración, se hizo una revisión de los resultados de estas y se determinó que el software ha superado las pruebas del sistema y que ha alcanzado la capacidad operativa inicial, por lo tanto no es necesario incluir otra iteración.

CAPÍTULO VI

FASE DE TRANSICIÓN

6.1 Introducción

La transición, completa la versión del producto, ya que se centra en implantar el producto en su entorno de operación. Una vez que el proyecto entra en esta fase, significa que ha alcanzado la capacidad operativa inicial, aunque no sea necesariamente perfecto, puesto que aun en esta etapa se pueden evidenciar fallas que no se detectaron en la fase de construcción.

Los objetivos básicos de esta fase son:

- ❖ Cumplir los requisitos establecidos en las fases anteriores, hasta la satisfacción de todos los usuarios.
- ❖ Gestionar todos los aspectos relativos a la operación en el entorno del usuario, incluyendo la corrección de los defectos remitidos por los usuarios de la versión beta o por los encargados de las pruebas de aceptación.

En pocas palabras, el proyecto recibirá información de los usuarios para:

- ❖ Determinar si el sistema hace lo que demandan sus usuarios y el negocio.
- ❖ Descubrir riesgos inesperados.
- ❖ Anotar problemas no resueltos.



- ❖ Encontrar fallos.

- ❖ Eliminar ambigüedades y lagunas en la documentación de usuario.

Centrarse en áreas en las que los usuarios muestren deficiencias y necesiten información o formación.

Debido a las políticas de seguridad de la Empresa esta fase quedará a cargo del equipo de Implantación de la Gerencia de AIT Dtto. Norte, por lo cual no será incluida en la investigación, centrandose esta fase en la construcción del Manual de Usuario.

6.2 Manual de Usuario



CONCLUSIONES

1. SAFREST es un novedoso sistema que permite almacenar, seleccionar y generar rápida y efectiva respuesta a las solicitudes de recobros de servicios telefónicos de las diferentes gerencias de PDVSA Distrito Norte, lo que permite que el reembolso por motivo del servicio prestado, se realice en un tiempo muy corto.
2. Logrando una correcta implementación del Proceso Unificado Racional (RUP) se obtuvo la definición, diseño y modelado del sistema SAFREST. Las iteraciones que incluye esta metodología junto al flujo de trabajo planteado por la misma permitieron el desarrollo de un proyecto que diera con la solución a los problemas del cliente. Los distintos diagramas considerados tanto de WebML como de UML, como herramienta conjunta de desarrollo, hicieron posible la realización de las diferentes fases establecidas por la metodología del proceso unificado.
3. El análisis del sistema actual a través de los diferentes diagramas, facilitó la determinación de los requerimientos ya que a través de ellos se realizó la identificación, definición y recolección de los verdaderos requisitos necesarios para el desarrollo de la aplicación, que se centran en el usuario y sus necesidades.
4. Se logró establecer una arquitectura del sistema sólida y lista para ser puesta en práctica debido a que los diagramas WebML permitieron concretar los flujos y vías con la que se trabajaron los datos para generar una información veraz al usuario. Por lo anteriormente descrito y junto a los diagramas de casos de uso implementados se llevó a cabo la fase de elaboración.



5. El adecuado diseño del modelo de la base de datos resultó fundamental en el desarrollo del software, ya que el mismo sirvió de base para erigir la arquitectura del sistema.
6. La utilización de software libre, cumpliendo con el decreto 3.390, es de gran importancia, ya que se obtiene software de calidad sin necesidad de comprar licencias. Además la utilización de PHP como lenguaje de programación para páginas Web, permitió el desarrollo de la aplicación de una manera más simple y efectiva.
7. Con la utilización de las tecnologías AJAX para la programación de páginas web, junto con la librería Ext-JS y el manejador de bases de datos MySQL, permitieron la creación del sistema de una manera mas sencilla, clara y efectiva, debido a que estas tecnologías son de fácil entendimiento e implementación y se adaptan a cualquier entorno computacional.
8. La correcta codificación del SAFREST dependió en gran parte de un correcto diseño de su arquitectura.
9. Por medio de la aplicación de pruebas como las de Caja Negra, se pudo corroborar la efectividad del sistema, obteniéndose un funcionamiento eficiente de cada uno de sus módulos.
10. El poder tener todo lo referente a la conexión con la base de datos separado del resto del código permitió que el desarrollo de la aplicación fuera más fácil, incluyendo el mantenimiento de la misma.



RECOMENDACIONES

1. Realizar todas las actividades pertinentes a la implantación del sistema en los servidores de producción de la Gerencia de AIT Distrito Norte.
2. La aplicación SAFREST fue creada pensando en una posible masificación, por tanto se recomienda que realmente se implemente en los demás distritos que conforman la empresa PDVSA.
3. Llevar a cabo la debida promoción del sistema, para la participación total de los usuarios y obtener de esta forma una data siempre actualizada.
4. Planificar jornadas de adiestramiento al personal destinado a la utilización del sistema SAFREST.
5. Se recomienda realizar periódicamente un respaldo de la base de datos del sistema, así como también el mantenimiento respectivo, debido a que con este software se maneja gran cantidad de información.
6. Se recomienda que, en caso de realizarse la actualización de SAFREST, inspeccionar el código de los componentes y, de ser posible, optimizar los mismos.



BIBLIOGRAFÍA

Barzanallana, R (2006). **“Sistemas de Bases de Datos”**. Universidad de Murcia, España. [Página Web en Línea]. Disponible en:

<http://www.um.es/docencia/barzana/IAGP/Iagp06.pdf>. [Consulta: 2008, Julio 07].

Berzal, F y Cubero, J. (2005). **“Desarrollo Profesional de Aplicaciones Web con ASP.NET”**. Editorial iKor Consulting.

Boggs W y Boggs M. (2002) **“UML with Rational Rose 2002”**. Editorial SYBEX. United States of America.

Casallas, R. (2006) **“Ingeniería de Software: Racional Unified Process RUP”**. Universidad de los Andes, Depto de Sistemas y Computación. Mérida, Venezuela.

Castells, M. (2001). **“La Galaxia Internet”** Editorial Areté Madrid, España.

Castiñeiras, M y Fuentes, X. (1999) **“Bioquímica Clínica y Patología Molecular: Volumen 1”**. Editorial Reverté S.A. España

Converse, T y Park, J (2004). **“PHP5 and MySQL Bible”**. Editorial Wiley Publishing, inc. Indiana, United States of America.

Date, C. (2001) **“Introducción a los sistemas de bases de datos”**. Editorial Pearson Educación. España.

Eguíluz, J. (2008) **“Introducción a AJAX”**. Editorial Creative Commons. España



Ext Js, a foundation you can build on. **“EXT Faq”**. [Página Web en Línea]. Disponible en:

http://extjs.com/learn/Ext_FAQ. [Consulta: 2008, Julio 08].

Heilmann, C (2006) **“Beginning JavaScript with DOM Scripting and Ajax From Novice to Professional”**. Editorial Apress. New York, United States of America.

Jacobson, Y, Booch G y Rumbaugh J. (2000) **“El Proceso Unificado de Desarrollo de Software”**. Editorial Addison Wesley Madrid, España

Jiménez, R. (2003). **“Análisis y diseño orientado a objetos de un framework para el modelado estadístico con MLG”**. Universitat de les Illes Balears. [Página Web en Línea]. Disponible en:

<http://www.tdx.cat/TDX-0713104-112553/index.html>. [Consulta: 2008, Enero 08].

Louden, K. (2002) **“Lenguajes de Programación: Principios y Practicas”**. Cengage Learning Editores. Mexico.

Marquez, M (2001) **“Sistemas de Bases de Datos”** [Página Web en Línea]. Disponible en:

<http://www3.uji.es/~mmarques/f47/apun/node4.html> [Consulta: 2008, Julio 07].

Mateu, C (2004). **“Desarrollo de Aplicaciones Web”**. Editorial Eureka Media, SL. Barcelona, España.

Montiva, J y Besembel I. (2007) **“Modelado de Sistemas usando UML 2.0”**. Centro de Excelencia en Ingeniería del Software. Mérida, Venezuela.



Pressman, R (2002). **“Ingeniería del Software, Un Enfoque Práctico”**. Editorial Mc Graw Hill. España.

Schmuller, J, (sf) **“Aprendiendo UML en 24 Horas”**. Editorial Prentice Hall.

Seijas, R. (2005). **“Desarrollo de un software para la automatización de las actividades de mantenimiento de inventario de la infraestructura tecnológica de información de una empresa petrolera”**. Trabajo de Grado De Computación, Universidad de Oriente, Edo. Anzoátegui.

Simoni M. (2.004). **“Desarrollo de un Sistema de Información para la Automatización de los procesos Realizados en el Departamento de Ciencias de la Universidad de Oriente, Núcleo de Anzoátegui”**. Trabajo de Grado De Computación, Universidad de Oriente, Edo. Anzoátegui.

Solange G., (2004). **“Ingeniería de Software”** [Página Web en Línea]. Disponible en:

<http://www.monografias.com/trabajos5/inso/inso.shtml> [Consulta: 2008, Octubre 28].

Stefano, C y Fraternali P (2003). **“Designing Data-Intensive Web Applications”**. Editorial Morgan Kaufmann Publishers. Milano, Italia.

Suniaga R. (2007). **“Desarrollo de un Software que Permita la Integración de las Bases de Datos de Activos Pertenecientes a las Unidades de Explotación de PDVSA Punta de Mata”**. Trabajo de Grado De Computación, Universidad de Oriente, Edo. Anzoátegui.

Trejo J. (2002). **“Bases de Datos”** [Página Web en Línea]. Disponible en:



<http://www.monografias.com/trabajos11/basda/basda.shtml> [Consulta: 2008, Octubre 28].

Vaswani, V (2005). **“PHP and MySQL”**. Editorial Mc Graw Hill. United States of America.

Wikipedia, La Enciclopedia Libre. (s.f.). “WebML” [Página Web en Línea]. Disponible en:

<http://es.wikipedia.org/wiki/Webml>, [Consulta: 2008, Julio 07].

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

TÍTULO	DESARROLLO DE UN SISTEMA AUTOMATIZADO PARA FACTURACIÓN Y RECOBRO DE SERVICIOS TELEFÓNICOS ASOCIADO A LA GERENCIA AIT, PDVSA MATURÍN, BASADO EN TECNOLOGÍA WEB Y ESTÁNDARES ABIERTOS.
SUBTÍTULO	

AUTOR (ES):

APELLIDOS Y NOMBRES	CÓDIGO CULAC / E MAIL
DOUGLAS RAFAEL, GENE PEREIRA	CVLAC: 15.514.925 E MAIL: gpdouglas02@gmail.com
	CVLAC: E MAIL:
	CVLAC: E MAIL:
	CVLAC: E MAIL:

PALÁBRAS O FRASES CLAVES:

UML

RUP

PHP

MySQL

WEBML

APLICACIÓN WEB

SOFTWARE LIBRE

EXTJS

PDVSA

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

ÁREA	SUBÁREA
Ingeniería y Ciencias Aplicadas	Ingeniería en Computación
	Desarrollo de Software
	Base de Datos

RESUMEN (ABSTRACT):

El proceso de recobro de facturación telefónica que lleva a cabo la gerencia AIT (Automatización, Informática y Telecomunicaciones) del Distrito Norte, consiste en realizar el recobro a las diferentes Gerencias que forman parte del Distrito, pero este proceso actualmente se lleva a cabo de forma manual, lo cual genera contratiempos y retrasos en el recobro a las distintas gerencias del Distrito. Esta acción es fundamental para controlar el presupuesto de la Gerencia por medio del reembolso de los servicios prestados, razón por la cual la gerencia de AIT específicamente el proceso de Cadena de Suministros, se vio en la necesidad de solicitar el desarrollo de un software para la automatización del proceso de recobro de facturación telefónica, para obtener beneficios de seguridad de la información, disminución de esfuerzo y tiempo. El desarrollo del proyecto estuvo basado en el Proceso Unificado de Desarrollo de Software, la Técnica UML y su extensión WEBML, para la implementación se utilizaron las herramientas de PHP y MYSQL cumpliendo con las normas de la empresa, en cuanto al desarrollo de aplicaciones con software libre de acuerdo al decreto 3.390.

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

CONTRIBUIDORES:

APELLIDOS Y NOMBRES	ROL / CÓDIGO CVLAC / E_MAIL				
	ROL	CA	AS X	TU	JU
Ing. Gabriela Veracierta	CVLAC:				
	E_MAIL	gabrielaveracierta@yahoo.com			
	E_MAIL				
Ing. Claudio Cortinez	ROL	CA	AS	TU	JU X
	CVLAC:				
	E_MAIL	cl_cortinez@cantv.net			
	E_MAIL				
Ing. José Luis Bastardo	ROL	CA	AS	TU	JU X
	CVLAC:				
	E_MAIL	josebastardo@gmail.com			
	E_MAIL				
	ROL	CA	AS	TU	JU
	CVLAC:				
	E_MAIL				
	E_MAIL				

FECHA DE DISCUSIÓN Y APROBACIÓN:

2010	08	13
AÑO	MES	DÍA

LENGUAJE. SPA

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

ARCHIVO (S):

NOMBRE DE ARCHIVO	TIPO MIME
Tesis_Aplicacion_Web_Gened.doc	application/msword

CARACTERES EN LOS NOMBRES DE LOS ARCHIVOS: A B C D E F G H I
J K L M N O P Q R S T U V W X Y Z. a b c d e f g h i j k l m n o p q r s t u
v w x y z. 0 1 2 3 4 5 6 7 8 9.

ALCANCE

ESPACIAL: _____ (OPCIONAL)

TEMPORAL: _____ (OPCIONAL)

TÍTULO O GRADO ASOCIADO CON EL TRABAJO:

Pre-Grado.

NIVEL ASOCIADO CON EL TRABAJO:

Ingeniero.

ÁREA DE ESTUDIO:

Departamento de Computación y Sistemas.

INSTITUCIÓN:

Universidad de Oriente – Núcleo de Anzoátegui.

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

DERECHOS

De acuerdo al artículo 41 del Reglamento de trabajo de grado:

“Los Trabajos de Grado son de la exclusiva propiedad de la Universidad de Oriente, y sólo podrán ser utilizados para otros fines con el consentimiento del Consejo de Núcleo respectivo, quien deberá participarlo previamente al Consejo Universitario, para su autorización”

Br. Douglas Rafael Gene Pereira.

AUTOR

Ing. Gabriela Veracierta
TUTOR

Ing. José Luis Bastardo
JURADO

Ing. Claudio Cortínez
JURADO

Ing. José Luis Bastardo
COMISION DE TESIS