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

PROYECTO DE TRABAJO DE GRADO:
"DESARROLLO DE UN SOFTWARE EN AMBIENTE WEB DESTINADO A GESTIONAR LOS PROCESOS DEL DEPARTAMENTO DE ADMINISTRACIÓN DE UNA EMPRESA ENCARGADA DEL MANTENIMIENTO DE EQUIPOS DISPENSADORES DE COMBUSTIBLE, UBICADA EN PUERTO LA CRUZ – ESTADO ANZOÁTEGUI"

Presentado por:
Luis Carlos Guevara Franco
C. I.: 13.167.548

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

Barcelona, 2010

UNIVERSIDAD DE ORIENTE

NÚCLEO DE ANZOATEGUI

ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS

DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

PROYECTO DE TRABAJO DE GRADO:

"DESARROLLO DE UN SOFTWARE EN AMBIENTE WEB DESTINADO A GESTIONAR LOS PROCESOS DEL DEPARTAMENTO DE ADMINISTRACIÓN DE UNA EMPRESA ENCARGADA DEL MANTENIMIENTO DE EQUIPOS DISPENSADORES DE COMBUSTIBLE, UBICADA EN PUERTO LA CRUZ – ESTADO ANZOÁTEGUI"

Asesor:

_____

Ing. Aquiles Torrealba

Asesor Académico

Barcelona, 2010

UNIVERSIDAD DE ORIENTE

NÚCLEO DE ANZOATEGUI

ESCUELA DE INGENIERÍA Y CIENCIAS APLICADAS

DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS

PROYECTO DE TRABAJO DE GRADO:

"DESARROLLO DE UN SOFTWARE EN AMBIENTE WEB DESTINADO A GESTIONAR LOS PROCESOS DEL DEPARTAMENTO DE ADMINISTRACIÓN DE UNA EMPRESA ENCARGADA DEL MANTENIMIENTO DE EQUIPOS DISPENSADORES DE COMBUSTIBLE, UBICADA EN PUERTO LA CRUZ – ESTADO ANZOÁTEGUI"

Jurado Calificador:

Ing. Mónica Saettone
(Jurado)

Ing. Gabriela Veracierta
(Jurado)

Ing. Aquiles Torrealba
(Asesor Académico)

Barcelona, 2010

RESOLUCIÓN

De acuerdo con el artículo 44 del reglamento de Trabajo de Grado:

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

RESUMEN

HIDROCONS, C. A. es una compañía contratista cuya principal actividad económica es la construcción de obras civiles en general, electricidad, plomería, pintura, instalación, reconstrucción, mantenimiento y reparación de equipos de estaciones de servicios dispensadoras de combustible. Debido a problemas como el retraso en que llegan los presupuestos a mano de los directivos de las empresas clientes, aunado al desconocimiento del estado de cuenta de la facturación por parte de los mismos, e inclusive de los servicios y repuestos que ofrece la empresa, se propuso el proyecto denominado "Desarrollo De Un Software En Ambiente Web Destinado A Gestionar Los Procesos Del Departamento De Administración De Una Empresa Encargada Del Mantenimiento De Equipos Dispensadores De Combustible, Ubicada En Puerto La Cruz – Estado Anzoátegui", que permite el flujo de información de una manera más rápida, oportuna y segura, además de fidedigna. Para diseñar la aplicación se utilizó el Proceso Unificado Racional de Desarrollo de Software (RUP) en combinación con los lenguajes de modelado UML y WebML. La construcción del mismo fue realizada mediante la herramienta de computación distribuida IBM RATIONAL 6.0. Debido a que es un sistema con entorno Web, la arquitectura se construyó utilizando el patrón cliente-servidor y se implementó IBM DB2 Versión 9 como sistema manejador de Base de Datos Relacionales. Para la codificación se usó Java como lenguaje de programación del lado del servidor, JavaScript (lenguaje interpretado orientado a páginas Web) del lado del cliente, ambos acompañados con lenguaje HTML. La visualización de la aplicación se encuentra a cargo del navegador Web Internet Explorer 6, y el Sistema Operativo a utilizar para el mismo es el Windows XP. La aplicación se denominó SIADCON, por las siglas de "**S**istema **I**ntegral de **AD**ministración para **CON**tratistas".

DEDICATORIA

Este Trabajo de Grado está dedicado primeramente a Dios Todopoderoso, quien ha estado presente en todo momento de mi vida, y que por supuesto, no podía faltar en este período, en esta importante fase de mi existencia, la más reciente por supuesto, que va desde que ingresé a la universidad hasta los actuales momentos, en que en definitiva presento mi tesis.

De igual modo se lo dedico a mis padres y a mi hermano, mi núcleo familiar más cercano, y del cual yo provengo, los cuales siempre han tenido la confianza, la certeza y la convicción de que este día llegaría, el día en que presentaría mi Trabajo de Grado.

También este Trabajo de Grado es dedicado al resto de mis familiares, a mis abuelos (que ya no están físicamente con nosotros), a mis abuelas, a mis tíos y tías, que siempre han estado pendiente de cuando he de culminar mi carrera, a mis primos y primas, en fin!!!, a toda mi familia en general.

Finalmente se la dedico a mi asesor y tutor de Trabajo de Grado, quien para mí es el mejor profesor del Departamento de Computación y Sistemas, y que por supuesto, para él corresponde un tomo de esta tesis.

AGRADECIMIENTO

Ante todo doy gracias a Dios Todopoderoso por darme la oportunidad de llegar hasta este punto de mi existencia en que cumplo una meta tan importante como lo es graduarme de Ingeniero en Computación.

Agradezco a mis padres por supuesto, por todo el apoyo y la confianza que han tenido para conmigo, y su deseo de verme graduado lo más pronto posible. A mi hermano, quien de alguna u otra forma siempre ha sabido estar presente en los trabajos y asignaciones universitarias (como ésta ¡claro está!) que me ha tocado realizar, a pesar de la distancia geográfica que nos separa.

A mis tíos y tías, todos en su conjunto, tanto hermanos de mi madre como hermanos de mi padre, a mis abuelos y abuelas, a mis primos y primas, en fin, a toda mi familia, a todos ellos muchas gracias.

Igualmente me siento agradecido con mi asesor y tutor de Trabajo de Grado, al cual le debo el que esta nueva experiencia de realizar mi primera tesis no haya sido como yo me la imaginaba, es decir, traumática!!!

A todos los profesores que en diversas formas contribuyeron con el fomento de mis conocimientos en la materia de computación y sistemas de información.

Por último, pero no menos importante, agradezco a mis amigos (los que no pertenecen a la universidad) y a mis compañeros de estudios universitarios, quienes nos ayudamos entre sí, y que al igual que yo, espero pronto saber que finalizaron.

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CONTENIDO

CAPÍTULO I

EL PROBLEMA

1.1 Reseña histórica de HIDROCONS, C. A.

HIDROCONS, C. A., empresa fundada en el año de 1985, se inició dedicándose a realizar trabajos en el ramo de mantenimiento y servicio del sistema de iluminación y electromecánico, así como de reemplazo de equipos de despacho de combustible, instalación de elevadores hidráulicos y semihidráulicos, carretes dispensadores y bombas de grasa y aceite en Estaciones de Servicio dispensadoras de combustible.

Posteriormente pasó a ofrecer servicios de reconstrucción general en su taller ubicado en la Calle Cumaná Nº. 21, Sector La Caraqueña en Puerto La Cruz – Estado Anzoátegui, a surtidores tipo succión propia, dispensadores y bombas sumergibles, así como en remodelaciones menores: cambio de sistema de succión propia a control remoto, haciendo todos los trabajos necesarios de obras civiles sobre los tanques de almacenamiento de combustible, islas para dispensadores, todos los trabajos electromecánicos, pintura en general, arranque de los equipos y puesta en marcha, todo dentro de las normas y reglamentos emanados del Ministerio de Energía y Minas y la Resolución 241 para la época.

Con el paso del tiempo y a raíz de la entrada en vigencia de la Liberación del Mercado Interno de Hidrocarburos y de la aparición e ingreso de nuevas Compañías Nacionales e Internacionales especialistas en este ramo, lo que se conoció como Apertura Petrolera, acontecida entre los años 2001 y 2003, la empresa decidió comenzar a prestar servicio de mantenimiento correctivo y preventivo a los equipos en las Estaciones de Servicio, por intermedio de las empresas representante de las casas fabricantes de los equipos, como lo era GILBARCO y de las operadoras como TEXACO, MOBIL, LLANOPETROL, PDVSA, DELTAVEN, SHELL, etc. directamente. Actualmente se dedica a prestar Servicio de Asesorías, remodelación menor a mayor, construcción de Estaciones de Servicio, instalación de todos los equipos de despacho de combustible (dispensadores, bombas sumergibles convencionales e inteligentes, equipos de control de inventario de gasolina y diesel, etc.), trabajos de obras civiles, electromecánicos y pintura en general, reconstrucción de equipos, mantenimiento preventivo y correctivo y proveedor de equipos y repuestos ligados al trabajo que desempeñan.

1.2 Planteamiento del prob lema

Gracias al incesante avance de la tecnología, hoy en día es posible delegar la gestión de relaciones de negocio a sistemas computarizados a través de la Internet, que faciliten el manejo de grandes volúmenes de información y que incrementen la eficacia y la productividad del personal administrativo y técnico de una empresa, optimizando el tiempo y la calidad de respuesta a las solicitudes de servicio e incrementando la eficiencia y productividad de muchas organizaciones, y sobretodo la toma de decisiones oportunas.

En la actualidad, las empresas que deseen fomentar sus utilidades deben considerar la idea de integrarse en el ámbito de los negocios On Line, haciendo uso de esa herramienta tan poderosa y que últimamente está tomando mucho auge como

lo es Internet. Las transacciones y operaciones electrónicas desde la comodidad de la casa u oficina de los usuarios, sobre todo cuando dichos usuarios se encuentran separados a grandes distancias unos de otros, cobran relevancia en el diario vivir cuando los factores tiempo y esfuerzo son, y han sido desde siempre, los que se consideran en cuenta al hacer una solicitud de servicio, ya que ello implica al mismo tiempo, ahorro de dinero. De allí que muchas empresas están planteándose la idea de migrar hacia esta tecnología, para así tener un alcance mayor hacia sus clientes y ofrecerles mejoras y actualizaciones instantáneas. Tal es el caso de la empresa HIDROCONS, C. A, la cual hasta los momentos no cuenta con otro medio de comunicación disponible que no sea el de vía telefónica (fax, telefonía fija y móvil), así como correo electrónico; y ningún otro medio publicitario que el de relaciones personales y comerciales "cara a cara", aunado a sus trabajos que la han llevado a contar con la fama con la que actualmente dispone, realizados a los diversos clientes de su cartera y divulgados entre ellos mismos.

Actualmente, uno de los problemas que presenta la empresa es la lentitud con la que se desplazan los flujos de información que intervienen en la elaboración, aceptación y rechazo o declinación de presupuestos, los cuales son los avalúos que se hacen de un trabajo propuesto, es decir, que aún no se ha llevado a cabo. El traslado de este tipo de documentos entre HIDROCONS y las empresas a las cuales son destinados dichos documentos se hace de manera física en la mayoría de los casos, y ello implica un tiempo considerable que se pierde, además de representar un gasto adicional y hasta excesivo. Con respecto al inventario de repuestos y accesorios que esta contratista posee para la venta, en la actualidad es muy poca la clientela que conoce de la existencia de la misma, e incluso de cuáles y qué tipo de repuestos lo conforman, por lo que es imperativo darlo a conocer a los clientes y al público en general. Este desconocimiento por parte de los clientes del conjunto de repuestos a la venta representa un problema considerable, ya que imposibilita la generación de solicitudes de repuestos y por consiguiente, la salida de los mismos.

Aunado a todo esto, HIDROCONS, C. A. está interesada en pertenecer a la Internet por medio de un sitio Web que le permita una interacción con sus clientes, representada mayormente por una amplia gama de Estaciones de Servicio dispensadoras de combustible dispersas a lo largo y ancho del oriente del país, no sólo para tener un registro de las mismas mediante el uso de una base de datos, sino también para darles a conocer los productos que dicha empresa ofrece en venta, así como una idea en resumen de los servicios que está en capacidad de llevar a cabo. Igualmente, la empresa desea un registro de todas sus solicitudes por parte de dichos clientes en lo referente a pedidos y órdenes de cotización de los equipos y repuestos en cuestión, así como de los presupuestos elaborados para satisfacer alguna demanda de parte de dichos clientes, y los registros de facturación propios de cada servicio, venta realizada, e inclusive un módulo de visitas donde los clientes registrados puedan plasmar sus opiniones y sugerencias para luego ser tomadas en consideración por los directivos de la empresa. Igualmente, los usuarios internos de dicho sistema, constituidos por las diversas Estaciones de Servicio registradas previamente, podrán hacer consultas sobre sus facturaciones, cotizaciones, presupuestos previos, para cotejar con los registros de la empresa contratista y llevar igualmente ellos mismos su control al respecto.

En vista de todo lo anteriormente explicado, la empresa HIDROCONS, C. A. espera prontamente poseer un novedoso sitio Web, que le facilite la realización de las actividades administrativas que ejecuta la empresa por parte de sus empleados. Dado que el sistema de información actual que posee la empresa se presta para ser trasladado sin mayores dificultades hacia una plataforma Web, y debido a las ventajas que representarían para los usuarios el que todas o la gran mayoría de sus solicitudes y transacciones sean efectuadas por medio de la Internet, se impone la necesidad de utilizar este medio eficiente con el fin de obtener mejores resultados en un tiempo significativamente reducido. Esta aplicación será desarrollada completamente en entorno Web siguiendo los lineamientos de RUP (*Rational Unified Process* - Proceso

Unificado Racional), con el uso de la metodología UML (*Unified Modeling Language* - Lenguaje de Modelado Unificado), en conjunto con otra metodología exclusiva para sitios Web denominada WebML (Web Modeling Language – Lenguaje de Modelado para Web). Se utilizará como software de aplicaciones de computación distribuida la denominada herramienta IBM RATIONAL Versión 6.0 para la programación de los módulos, JavaScript como lenguaje interpretado orientado a páginas Web del lado del cliente, JAVA para la capa de negocios del lado del servidor, Java Development Kit Versión 1.5.0 como paquete de clases para soporte de JAVA, y el Sistema Manejador de Base de Datos Relacionales IBM DB2 Versión 9 para el manejo de datos.

Cabe destacar que además, con la posible implantación de un sitio Web de tales referencias se registraría el hecho de que la empresa en cuestión incurriría por primera vez en el uso de la tecnología Web como medio informativo y publicitario, lo cual constituye algo innovador para la misma en lo que a manejo de transacciones comerciales electrónicas se refiere, mejorándose el rendimiento del personal administrativo, e inclusive del personal técnico, al contar con un software para ser dado a conocer y visitado a gran escala por su distinguida clientela a través de un medio tan difundido y ampliamente en expansión como lo es Internet.

1.3 Objetivos del proyecto

1.3.1. Objetivo General:

Desarrollar un software en ambiente Web destinado a la gestión de los procesos del departamento de administración de una empresa encargada del mantenimiento de equipos dispensadores de combustible, ubicada en Puerto La Cruz – Estado Anzoátegui.

1.3.2. Objetivos Específicos:

1. Especificar los requerimientos necesarios para el desarrollo del futuro sistema de información para la empresa HIDROCONS, C. A., identificando los nuevos requisitos para la obtención de la arquitectura final del mismo.

2. Diseñar las bases de datos precisas para el almacenamiento de la información relacionadas con los procesos administrativos a ser manejados por el sistema en cuestión.

3. Diseñar las interfaces del programa destinadas a la presentación al usuario de la información apropiada de una manera legible.

4. Codificar cada uno de los módulos que integran el proyecto.

5. Integrar los módulos que conforman el sistema en combinación con las respectivas bases de datos.

6. Realizar las diversas pruebas necesarias para la certificación de la confiabilidad y efectividad de la aplicación, de manera que se detecten y corrijan las posibles fallas existentes.

7. Elaborar la documentación necesaria para el manejo de la aplicación, dirigida a los usuarios que harán uso del programa en cuestión.

CAPÍTULO II

MARCO TEÓRICO

2.1 Antecedentes de la investigación

Previas a esta investigación se tiene conocimiento de algunos trabajos llevados a cabo en esta casa de estudio, los cuales sirven de buena base y referencia para el desarrollo del actual proyecto. Estos trabajos se citan a continuación:

- José Ruperto Medina Monagas. Su trabajo de grado es conocido como "**Desarrollo de un Sistema basado en Aplicaciones Web para la Automatización del Control de Pedidos asociados al Proceso de Ventas de una empresa cafetalera**" (2007). Este trabajo fue presentado en la Universidad de Oriente – Núcleo de Anzoátegui como requisito parcial para optar por el título de Ingeniero en Computación. Este sistema fue solicitado por la empresa Distribuidora Nacional 2000, C. A., la cual posee sucursales en distintas ciudades del territorio nacional y necesita información en tiempo real para establecer un control en el proceso de toma de pedidos para la venta, utilizando tecnología Web para tal fin. El desarrollo del sistema se llevó a cabo haciendo uso de las cuatro fases del proceso unificado de desarrollo de software (RUP), al igual que los diversos diagramas que conforman el Lenguaje Unificado de Modelado (UML). El sistema se denominó KAFANKA, por mantener relación con la empresa Café Anzoátegui, C. A.,

cuyas siglas son CAFANCA. El mismo fue desarrollado usando lenguaje de programación Web PHP y MySQL como manejador de base de datos, que constituyen tecnología de software libre con lo que se puede disponer en la Web de su código bajo licencia GPL (Global Public License - Licencia Pública General). Dicho sistema muestra a los usuarios diversas funciones, permitiendo optimizar sus labores y mejorar su productividad. [14]

- Juan Carlos Tenias Belmonte. Elaboró el trabajo de grado que tiene por título "**Desarrollo de un software basado en Aplicaciones Web para el Monitoreo de Dispositivos de la Plataforma de Telecomunicaciones de PDVSA - GAS**" (2007). Este trabajo fue presentado en la Universidad de Oriente – Núcleo de Anzoátegui como requisito parcial para optar por el título de Ingeniero en Computación. El desarrollo del sistema se llevó a cabo mediante las fases del proceso unificado de desarrollo de software, y algunos diagramas del Lenguaje de Modelado Unificado (UML). El sistema se denominó SIMREDS (Sistema de Monitoreo de Redes y Servidores), y para el cual se utilizó herramientas y lenguajes de software libre, tal como el lenguaje de programación Web PHP sobre el sistema operativo LINUX, empleó el protocolo de red SNMP y sistema manejador de base de datos MySQL. La importancia del software radica en que permite a los usuarios determinar el estado de los enlaces de servidores, detectar cuellos de botella y colapsos de segmentos de red, así como fallas de equipos. [19]

- Jaísfel Enrique Cedeño Maza. Su trabajo de grado tiene por título "**Desarrollo de una aplicación Web para el registro, manejo y control de eventos organizados por la Unidad de Calidad de Vida del departamento de Recursos Humanos de PDVSA – Refinación en Puerto La Cruz**" (2007). Este trabajo fue presentado en Marzo de 2007 en la Universidad de Oriente – Núcleo de Anzoátegui, como requisito parcial para optar por el

título de Ingeniero en Computación. La unidad de Calidad de Vida de la Gerencia de Recursos Humanos (RRHH) de la sede de PDVSA – Refinación en PLC, cuya finalidad es planificar, organizar y proporcionar eventos para el disfrute de los trabajadores de la empresa y de los familiares afiliados solicitó la creación de una aplicación que fuese desarrollada en ambiente Web, destinada para llevar a cabo la creación de eventos, manejo de la información de los participantes por cada evento, generación de reportes y promoción de nuevos eventos. Para el desarrollo de la misma se empleó como metodología de diseño el "Proceso Unificado Racional (RUP)" junto con la metodología y procedimientos de desarrollo propios de la empresa y la gerencia dedicadas a estos proyectos. Para la codificación se utilizaron herramientas libres (no propietarias) orientadas al desarrollo Web, tales como PHP (Hipertext Processor) como lenguaje de programación del lado del servidor, JavaScript (lenguaje interpretado orientado a las páginas Web) del lado del cliente, ambos incrustados en lenguaje HTML para la visualización de la aplicación a través del navegador Web, ya sea por Internet Explorer, Mozilla Firefox o Netscape Navigator, además de PostgreSQL como manejador de bases de datos. La aplicación fue denominada "SIMECAV", siglas de Sistema Manejador de Eventos de Calidad de Vida. [3]

- Luis Eduardo Millán González y Luis Carlos Garelli Boada. Desarrollaron el trabajo de grado cuyo nombre es **"Desarrollo de un Software que permita la Automatización de las Actividades asociadas al Departamento de Admisión y Control de Estudios de la Extensión Centro / Sur del Núcleo de Anzoátegui de la Universidad de Oriente"** (2007). Este trabajo fue presentado en la Universidad de Oriente – Núcleo de Anzoátegui como requisito parcial para optar por el título de Ingeniero en Computación. El software se elaboró utilizando tecnologías y paradigmas de software libre. Este sistema maneja la información de los estudiantes y sus registros

académicos, registra las solicitudes de documentos estudiantiles y genera reportes de las consultas efectuadas por el personal del departamento. Para su desarrollo se empleó la metodología del Proceso Unificado (RUP) y lenguajes de modelado UML y WebML. Se usó lenguaje PHP para codificar los Scripts del servidor, HTML para las páginas Web del lado del cliente y PostgreSQL como manejador de base de datos. [15]

- Leoncio Enrique Núñez Simancas. Realizó el trabajo de grado cuyo nombre es **"Desarrollo de una Aplicación Web para la Visualización en Tiempo Real de los Parámetros Operacionales de Producción de la empresa PDVSA"** (2007). Este trabajo fue presentado en Julio de 2007 en la Universidad de Oriente – Núcleo de Anzoátegui como requisito parcial para optar por el título de Ingeniero en Computación. Esta aplicación fue desarrollada en ambiente Web aplicando paradigmas de software libre para visualizar en tiempo real los parámetros operacionales de producción de la empresa PDVSA tales como temperatura, presión, voltaje, etc. Se utilizó el proceso RUP así como la herramienta Macromedia DreamWeaver 2004 para la programación de los módulos, JavaScript (lenguaje interpretado orientado a las páginas Web) del lado del cliente, tecnología SVG para la creación en tiempo real de gráficas, PHP para la capa de negocios del servidor, ECMAScript para la lógica de la aplicación y MySQL para el manejo de base de datos. El sistema en cuestión se apodó como TRP, lo que se refiere a las siglas de su nombre completo, el cual es "Tiempo Real de Producción". [16]

2.2. Ingeniería de software [12]

La Ingeniería de Software es una rama de la ingeniería cuyo objetivo principal es proveer metodologías y técnicas que ayuden a desarrollar sistemas de software confiables y a tiempo, a su vez de que aseguren que el desarrollador cumpla con las

expectativas de calidad y permanezca dentro de un presupuesto accesible y justo. La importancia de esta ingeniería radica en fomentar un enfoque sistemático para el desarrollo, la implementación y el mantenimiento del software a través del ciclo de vida del sistema mismo.

2.2.1. Sistemas de Información [12]

Conjunto formal de procesos que operando sobre una colección de datos estructurados de acuerdo a las necesidades de la empresa, recopila, elabora y distribuye la información necesaria para la operación de dicha empresa y para las actividades de dirección y control correspondiente, apoyando en parte, la toma de decisiones necesaria para desempeñar las funciones y procesos del negocio de la empresa de acuerdo con su estrategia.

2.2.2. Software [12]

Procedimientos y reglas lógicas escritas en la forma de programas y aplicaciones, que definen el modo de operación de la computadora. Tienen carácter virtual (en contraposición con el hardware) y están almacenadas en los diferentes tipos de memoria de lectura/escritura. Comprende todo tipo de programas, utilidades, aplicaciones, sistemas operativos, drivers que hacen posible que el usuario pueda trabajar con la máquina. La programación a gran escala se apoya en unos principios sólidos y firmes que facilitan el desarrollo de la actividad de programación.

2.2.3. Diseño de Software [9]

Es una parte fundamental de la ingeniería de software, que tiene por objetivo planear una solución para un problema de acuerdo a especificaciones de requerimientos particulares. Es el primer paso para avanzar del dominio del

problema al dominio de la solución, sin importar cuál proceso de software se utilice. Es la primera de las tres actividades técnicas en el proceso de ingeniería de software, a saber: diseño, programación y pruebas. El proceso de diseño de software comienza después del análisis y la especificación de requerimientos. Existen diversos métodos para realizar el diseño: diseño de datos, diseño arquitectónico, diseño de interfaz y diseño de componentes.

- **Diseño de Datos:** ayuda en la creación de las estructuras de datos requeridas para implementar el software. Los objetos de datos y las relaciones que se definen en el diagrama entidad-relación proveen la base para el diseño de datos.

- **Diseño Arquitectónico:** especifica las diversas entidades estructurales en el sistema y su interacción con otras o la relación entre sí. El diseño arquitectónico considera los elementos estructurales y diversos patrones de diseño.

- **Diseño de Interfaz:** especifica la manera en la cual se comunican e interactúan los diversos componentes entre sí. El diseño de interfaz también especifica las formas en las que los usuarios se comunican con el software y viceversa.

- **Diseño de Componentes:** las entradas principales para el diseño de componentes provienen del diseño arquitectónico, el cual provee los elementos estructurales del sistema. Este diseño especifica los componentes (descripción procedural) que corresponde a la funcionalidad sugerida por los elementos estructurales.

2.3. Fundamentos de pruebas de software [10]

2.3.1. Prueba de Software de Caja Negra

También conocida como prueba funcional, esta prueba considera la selección de los datos de prueba y la interpretación de los resultados de la misma basándose en las características funcionales del software. Dicha técnica se concentra en la funcionalidad global del software y permite la prueba funcional para descubrir fallas referentes a funciones erróneas o faltantes, interfaces, estructura y bases de datos, y desempeño.

2.3.2. Prueba de Software de Integración

Una vez que las diversas unidades del software ya fueron sometidas al proceso de prueba de unidad, los defectos que aparecen durante el proceso habrán sido eliminados y los errores corregidos. Ahora se debe ensamblar las diferentes unidades en el sistema total. Este proceso de ensamblar las diferentes unidades en el sistema completo se llama integración, y probar el sistema como la integración de varias unidades se llama Prueba de Integración.

2.4. Fundamentos de "RUP" [10]

2.4.1. RUP (Rational Unified Process – Proceso Unificado Racional)

Es una metodología bien definida y estructurada de desarrollo de software iterativa e incremental, la cual consiste en desarrollo de aplicaciones que podrían considerarse como mini-proyectos, dirigido a una secuencia de versiones, hasta obtener definitivamente el producto final, centrada en la arquitectura y manejada por casos de usos, es decir, su proceso de desarrollo sigue una secuencia de actividades.

Define claramente quién es responsable de tal o cuál actividad, cómo y cuando deben hacerse las cosas y describe una estructura bien definida para el ciclo de vida de un proyecto, marcando claramente los puntos de decisión esenciales. RUP se basa en la evolución de prototipos ejecutables o versiones finales que se mostrará a los usuarios; de allí se refiere su naturaleza iterativa e incremental.

2.4.2. Características de RUP

Las características esenciales de la metodología RUP son las siguientes: 1.- Dirigido por casos de uso: los casos de uso describen cómo los usuarios interactúan con el sistema a desarrollar. 2.- Iterativo e incremental: en un desarrollo iterativo se tiene un ciclo de vida que consiste de varias iteraciones, cada paso por el ciclo de vida produce una versión del producto que incrementalmente se va refinando en las iteraciones de las diferentes fases. Si llegado el final del ciclo de vida del proceso de desarrollo, el producto no cumple con los objetivos planteado, se puede realizar un ciclo más para refinar, corregir y agregar funcionalidades que lleven al software a cumplir con las expectativas o cancelar el proyecto en base a los resultados obtenidos. 3.- Centrado en la arquitectura: se basa en diseñar una arquitectura base ejecutable; es decir, una organización o estructura de las partes más relevantes dejando de lado los detalles.

El Proceso Unificado divide el proceso de desarrollo en ciclos, teniendo un producto final al finalizar cada ciclo. Dichos ciclos a saber, en líneas generales, son cuatro y los mismos se enumeran a continuación:

- **Inicio:** Se hace un plan de fases, se identifican los actores que harán uso del futuro sistema, se determinan los principales casos de uso y se identifican los riesgos.

- **Elaboración:** se hace un plan de proyecto, se completan los casos de uso faltantes, se desarrollan diagramas explicativos y descriptivos y se eliminan los riesgos.

- **Construcción:** se concentra en la elaboración de un producto totalmente operativo y eficiente, incluyendo el código fuente, las interfaces y las pruebas, además del manual de usuario.

- **Transición:** se implementa el producto en el cliente y se entrena a los usuarios finales del mismo. Como consecuencia de esto suelen surgir nuevos requerimientos a ser analizados.

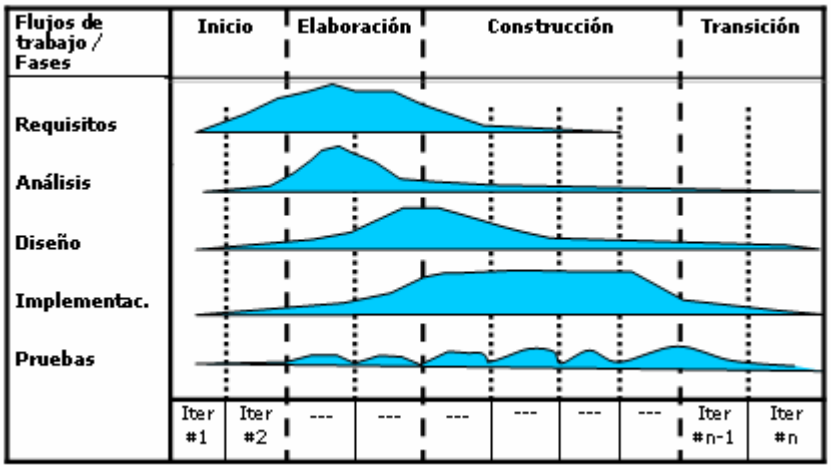


Figura 2.1. Diagrama de Fases de RUP
Fuente: Guía del Estudiante del Curso de Ingeniería de Software (Volumen 2: Métricas, Calidad y Pruebas), 2006

2.4.3. Elementos básicos de RUP

Un proceso de desarrollo de software es representado usando un conjunto de elementos de modelado, tales como roles, actividades, artefactos, disciplinas y flujos de trabajo. Un rol expresa quién (individuo o grupo) hace un trabajo, una actividad describe cómo el trabajo es hecho, un artefacto es una pieza tangible de información que es producida o utilizada por procesos, una disciplina es una colección de actividades relacionadas a una tarea de interés principal dentro del proyecto, y un flujo de trabajo es una secuencia de actividades que producen un resultado de valor.

2.5. Programación orientada a objetos

2.5.1. ¿Qué es la Programación Orientada a Objetos? [20]

Los seres humanos perciben el mundo como si estuviera formado por objetos: mesas, sillas, computadoras, coches, cuentas bancarias, partidos de fútbol, etc. También es un instinto humano intentar organizar estos objetos disponiéndolos de una forma concreta, optando por destacar determinadas características de algunos objetos que los destacan de otros. La programación orientada a objetos se puede describir rápidamente como la identificación de objetos importantes, su organización en jerarquías, la adición de atributos a los objetos que describen las características relevantes en el contexto del problema y de la adición de las funciones (métodos) a los objetos para realizar las tareas necesarias en el objeto.

2.5.2. Características de la Programación Orientada a Objetos [20]

Los lenguajes de programación orientada a objetos (como C++ y Java) se caracterizan por tres conceptos clave: encapsulación, jerarquía y polimorfismo, que son compatibles con este aspecto natural de identificación y clasificación de objetos.

2.5.2.1. Encapsulación

La encapsulación o encapsulamiento es la propiedad que permite asegurar que la información de un objeto no es conocida por ningún otro programador (también se conoce como ocultamiento de la información). Consiste en la combinación de datos y operaciones que se pueden ejecutar sobre esos datos en un objeto que funciona como una unidad completa o caja negra.

2.5.2.2. Jerarquía

Las dos jerarquías más importantes de un sistema complejo son las estructuras de clases (generalización/especialización) y las estructuras de objetos (agregación). Las jerarquías de generalización/especialización se conocen como herencia, y es la propiedad que permite a un objeto transmitir sus propiedades a otros objetos denominados descendientes. Esta propiedad permite la reutilización de objetos previamente definidos. La agregación es el concepto que permite el agrupamiento físico de estructuras relacionadas lógicamente.

2.5.2.3. Polimorfismo

Polimorfismo significa, fundamentalmente, que las clases pueden tener el mismo comportamiento, pero implementarse de distintas maneras. Esto resulta muy útil en términos de programación, ya que permite trabajar con tipos de objetos genéricos cuando lo que interesa no es cómo implementa cada clase las funciones.

2.5.3. Clases y Objetos [20]

Como su propio nombre lo indica, la programación orientada a objetos está relacionada con objetos. Un objeto se compone de datos que describen al objeto y las

operaciones que se pueden realizar en el objeto. No obstante, cuando se crea un programa, en realidad se declaran y definen clases, no objetos. Una clase es un tipo definido por el usuario, encapsula tanto los datos como los métodos que funcionen en esos datos. Una clase es algo muy parecido a una plantilla, que se utiliza para crear (instanciar) objetos.

2.5.4. Abstracción de Objetos [6]

Es una técnica utilizada en el pensamiento orientado a objetos, que permite a un desarrollador concentrarse en los aspectos esenciales del problema a la mano, mientras ignora detalles que tienden a distraer, y que no tienen relevancia para la solución del problema en cuestión. Dicho en pocas palabras, la abstracción es eliminar lo innecesario.

2.5.5. Ventajas de la Programación Orientada a Objetos en el Ciclo de Vida de Desarrollo [20]

Son tres las ventajas clave de la programación orientada a objetos: compresión, reutilización del código y posibilidad de ampliación. La división del código en clases contribuye a imponer una estructura a medida que crecen los programas. La idea es unir los programas orientados a objetos de clases escritas previamente, así como realizar las modificaciones necesarias para admitir los nuevos requisitos mediante la herencia para derivar nuevas clases a partir de las ya existentes. La creación de sistemas a partir de componentes que se pueden volver a utilizar conduce, naturalmente, a una mayor productividad, que suele ser la ventaja más citada de todos los métodos orientados a objetos.

Las características de la programación orientada a objetos también proporcionan varias ventajas adicionales: La encapsulación facilita la escala de pequeños sistemas

hacia grandes sistemas, lo que significa que independientemente del tamaño del sistema, el programador simplemente está creando objetos, que son las partes más sencillas de todo sistema de este tipo. Por último, la posibilidad de ocultar datos también genera sistemas más seguros, lo que conlleva a que el estado de los objetos sólo se puede modificar mediante métodos expuestos públicamente; esto aumenta la posibilidad de predecir el comportamiento del objeto.

2.6. Fundamentos de "UML"

2.6.1. UML (Unified Modeling Language – Lenguaje de Modelado Unificado) [6]

Es un lenguaje usado para especificar, visualizar, construir y documentar las diversas piezas de sistemas de software y también para modelado de negocios y otros sistemas que no sean software. El uso de UML en el desarrollo de sistemas orientados a objetos ganó importancia cuando los tres autores de esta metodología, Grady Booch, James Rumbaugh e Ivar Jacobson llegaron juntos a Rational Software Corporation. Estos autores presentaron un lenguaje de modelado visual que junto con contribuciones de otros metodologistas, líderes, vendedores de software y muchos usuarios, puede considerarse como un estándar para el desarrollo de sistemas Orientados a Objetos.

2.6.2. Objetivos de UML [12]

Los objetivos de UML son muchos, pero se pueden sintetizar sus funciones en estas principales:

- **Visualizar:** UML permite expresar de una forma gráfica un sistema de forma que otro lo puede entender.

- **Especificar:** UML permite especificar cuáles son las características de un sistema antes de su construcción.

- **Construir:** A partir de los modelos especificados se pueden construir los sistemas diseñados.

- **Documentar:** Los propios elementos gráficos sirven como documentación del sistema desarrollado que pueden servir para una futura revisión, y potencial mejoramiento.

2.6.3. Elementos de UML [17]

Un modelo UML esta compuesto por tres clases de bloques de construcción, que son los siguientes:

- **Elementos:** constituidos por los bloques básicos (clases, clase de análisis, componentes).

- **Relaciones:** ligan los elementos (asociación, herencia, agregación, composición, caso de uso, actor).

- **Diagramas:** Agrupan colecciones de elementos ligados aportando un significado adicional.

2.6.4. Diagramas de UML [21]

Algunos de los diagramas que UML describe son:

2.6.4.1. Modelado Estructural [6]

El modelo estructural es el modelo UML básico. El mismo especifica cómo está constituido el sistema, revelando las partes concretas del sistema completo. Simplemente se ocupa de las clases (abstracciones) y objetos (realizaciones concretas de las abstracciones). El mismo se puede ver reflejado indistintamente en cualquiera de los dos diagramas que lo conforman: Diagrama de Clases y Diagrama de Objetos.

- **Diagrama de Clases:** Representa un conjunto de fases, interfases y colaboraciones con sus respectivas relaciones que integran un paquete o contenedor. Un paquete es un mecanismo provisto por Java que ayuda a dividir el espacio de nombre de las clases en bloques manejables. El Diagrama de Clases es el diagrama principal para el análisis y diseño. Un diagrama de clases presenta las clases del sistema con sus relaciones estructurales y de herencia. La definición de clase incluye definiciones para atributos y operaciones. El modelo de casos de uso aporta información para establecer las clases, objetos, atributos y operaciones.

- **Diagrama de Objetos:** Un diagrama de Objetos ilustra un conjunto de objetos y sus relaciones para un contexto o escenario dado. Dicho diagrama contiene, al igual que un diagrama de clases, sólo objetos y enlaces, pero con diferencias específicas. Cada objeto está indicado por el nombre de la clase acompañado del nombre del objeto que representa la instanciación de dicha clase. Un enlace es una conexión semántica entre objetos. También es una instancia de una asociación. Un enlace se muestra como una línea. Generalmente, un diagrama de objetos se diferencia de un diagrama de clases en que es una instancia de un diagrama de clases para un contexto dado.

2.6.4.2. Diagrama de Casos de Uso

Los diagramas de casos de uso describen cómo los usuarios interactúan con el sistema a desarrollar. Representa una funcionalidad puntual del mismo, son la base para el desarrollo del sistema entero, ya que son una forma de capturar los requerimientos funcionales del sistema en cuestión.

2.6.4.3. Diagramas de Interacción [6]

Los diagramas de interacción describen el modo en que grupos de objetos colaboran para realizar un trabajo. Ellos capturan el comportamiento de un solo caso de uso, mostrando el patrón de interacción entre los objetos. Por lo tanto, un diagrama de interacción consiste de un conjunto de objetos, relaciones entre dichos objetos y mensajes entre los mismos. Existen dos tipos de diagramas de interacción que cumplen prácticamente la misma función a saber: Diagramas de Secuencia y Diagramas de Colaboración.

- **Diagramas de Secuencia:** describen el comportamiento del sistema enfatizando el orden o secuencia en el tiempo de los mensajes. Dicho orden se representa gráficamente como una tabla, donde los objetos se disponen sin orden alguno en lo alto del eje X y los mensajes en orden de aparición de acuerdo al tiempo, a lo largo del eje Y.

- **Diagramas de Colaboración:** representan colaboraciones que enfatizan la organización estructural de los objetos que pasan mensajes entre ellos mismo. La notación para un diagrama de colaboración es un conjunto de arcos y vértices.

2.6.5. Lenguajes de modelado [6]

Estos lenguajes consisten típicamente de elementos del modelo, notaciones y directivas, que permiten la representación del mundo real mediante diagramas. Constituyen un conjunto estandarizado de símbolos y de modos de disponer dichos símbolos, con el fin de modelar un diseño de software orientado a objetos. Entre ellos se encuentran UML y WebML.

2.6.6. Análisis y Diseño de Sistemas Orientado a Objetos [6]

Esta metodología se basa en modelar el mundo real y ha ganado importancia significativa en los últimos tiempos. En la orientación a objetos se trabaja con objetos en el sistema que interactúan unos con otros a través de mensajes. La orientación a objetos proporciona los recursos para ocuparse de los objetos de un sistema complejo.

2.6.7. Interfaz de un objeto [6]

Un objeto es una instancia de una clase, son manifestaciones concretas de los marcos de trabajo o clases. Están conformados por propiedades, que son las características del objeto en cuestión, y los métodos, que representan las operaciones que puede realizar un cliente sobre un objeto. La interfaz del objeto es la forma en la cual se presenta la clase al mundo real. Viene representado por el conjunto de atributos y métodos propios del objeto y que da a conocer a los demás objetos sin necesidad de que estos otros conozcan en detalle la implementación de sus métodos. Un objeto se comunica con otros a través de mensajes. Un mensaje es un pedido a un objeto para que realice una tarea a través de un método apropiado.

2.7. Fundamentos de diseño de bases de datos [7]

2.7.1. Modelo de Datos

Es la estructura subyacente de una base de datos. Los modelos de datos más conocidos son los basados en registros, los cuales son llamados así porque la base de datos es estructurada en torno a registros de formato fijo, con varios campos o atributos. Los campos o atributos pueden ser de diferentes tipos de datos y cada campo es usualmente de una longitud fija. Los tres modelos de datos de mayor aceptación basados en registros son el modelo relacional, el modelo de red y el modelo jerárquico. Sin embargo el más empleado es el primero de estos, debido a su sencillez y mayor facilidad de desarrollo.

2.7.2. Modelo de Datos Relacional

En este modelo se representan una o más tablas que contienen los datos y las relaciones entre los diferentes datos. Cada tabla posee las siguientes características: una tabla es una colección de registros en una base de datos, dicha tabla tiene múltiples columnas, cada columna tiene un nombre único y contiene conjuntos de datos, cada conjunto de datos se denomina registro o fila.

2.7.3. Bases de Datos (BDD)

Son conjuntos exhaustivos no redundantes de datos estructurados organizados independientes de su utilización y su implementación en máquina, accesibles en tiempo real y compatibles con usuarios concurrentes con necesidad de información diferente y no predicable en el tiempo.

2.7.4. Diagrama Entidad – Relación

Es una forma de modelar la relación existente entre los objetos del mundo real, denominados entidades. Este modelo expresa entidades relevantes para un sistema de información, así como sus interrelaciones y propiedades. Dicho modelo permite tender un puente sobre la brecha que existe entre la representación del mundo real y la manera como debe ser modelada en un sistema de computación.

2.7.5. Atributos

Todos los datos referentes a una entidad están contenidos en sus atributos. Un atributo es la propiedad de la entidad, cuyo valor tiene que ser almacenado en la base de datos. Cada instancia de una entidad tendrá el mismo conjunto de atributos, pero pueden contener valores diferentes. Los atributos pueden ser simples, como la fecha de nacimiento, o compuestos, como el nombre, conteniendo el primer y segundo nombre. Puede ser de valor único como la fecha de nacimiento (cada persona, por ejemplo, tiene solamente una fecha de nacimiento), o multivaluados, como las calificaciones de una asignatura.

2.7.6. Normalización de Base de Datos

En los modelos de bases de datos relacionales, la normalización es el proceso de organizar los datos para reducir al mínimo la duplicación de los mismos. La normalización generalmente implica el proceso de dividir una base de datos en dos o más tablas y de definir las relaciones entre ellas. El objetivo de este proceso es aislar los datos, de forma tal que la adición, eliminación o modificación del valor a un campo se pueda hacer sobre una sola tabla y luego esta se propague al resto de la base de datos a través de las relaciones definidas.

2.7.7. RDBMS (Relational Data Base Management System)

Es un Sistema Manejador de Bases de Datos (DBMS) basado en el modelo relacional. Organiza los datos y las relaciones entre los datos en tablas que se componen de columnas y filas, define las estructuras de datos, las operaciones de almacenamiento y recuperación, y utiliza restricciones de integridad para mantener la integridad de los datos almacenados.

2.7.8. Pasos para el Diseño de una BDD

Los siguientes son los cuatro pasos esenciales del proceso de diseño de una base de datos: a) Recolección y análisis de requerimientos, b) Diseño conceptual, c) Diseño lógico y d) Diseño físico.

- **Recolección y Análisis de Requerimientos:** antes de desarrollar una base de datos para cualquier sistema, es necesario interactuar estrechamente con el sistema en particular o con los usuarios del sistema actual. Esto ayuda a desarrollar una comprensión completa de los datos que deben ser almacenados en la base de datos, y los procesos involucrados en la captura de esos datos. Esta fase concuerda con la etapa de análisis de requerimientos del sistema en sí, propia del proceso de desarrollo de todo sistema automatizado.

- **Diseño Conceptual:** consiste en formar una descripción concisa de los requerimientos de datos usando un modelo de datos de alto nivel. Esta descripción será independiente de los requerimientos de almacenamiento. Este paso implica identificar las entidades involucradas en el sistema, y entender las relaciones entre estas entidades. Las entidades y relaciones se presentan en forma de diagrama, llamado Diagrama Entidad-Relación (ER).

- **Diseño Lógico:** en la implementación, la mayoría de los sistemas de base de datos tienen un modelo de datos. Cualquier técnica adecuada de diseño conceptual especificará la correspondencia del modelo conceptual a una variedad de modelos de implementación. La traducción de entidades y relaciones a tablas y otros objetos de la base de datos se hace en este nivel. Este proceso implica la normalización de la base de datos.

- **Diseño Físico:** algunos sistemas de base de datos permiten que el administrador de la misma tome decisiones sobre el almacenamiento físico. Estas decisiones se toman generalmente considerando el rendimiento y la disponibilidad de los recursos de hardware. Esta fase implica la selección de un adecuado sistema manejador de base de datos y la construcción de la misma en dicho manejador.

2.7.9. Lenguaje De Consulta Estructurado (Structured Query Language - SQL)

Es el lenguaje usado para comunicarse con una base de datos. El mismo se ha consolidado como el lenguaje estandar de las bases de datos relacionales. Es muy fácil de usar y parece tan simple como el inglés. IBM originalmente desarrolló SQL a comienzos de los setenta, llamado inicialmente “Sequel”, cambió después su nombre a SQL. En 1986, el American Nacional Standard Institute (ANSI) y el Internacional Standards Organización (ISO) presentó un estándar para SQL llamado SQL-86. Desde entonces ha ido evolucionando a medida que se han desarrollado nuevas versiones.

2.7.9.1. Clasificación de los Lenguajes del SQL

En la estructura básica de un Sistema Manejador de BDD se enuncian cuatro lenguajes que permiten trabajar sobre la base de datos. Estos lenguajes estándar son:

- **Lenguaje de Definición de Datos (Data Definition Language - DDL)**: El DDL de SQL proporciona comandos para definir los diferentes objetos de la base de datos. Una tabla, por ejemplo, es un objeto de la base de datos. Otros objetos de la base de datos incluyen vistas, índices y procedimientos almacenados.

- **Lenguaje de Manipulación de Datos (Data Manipulation Language - DML)**: El DML del SQL proporciona comandos para insertar, eliminar y modificar registros en las tablas.

- **Lenguaje de Control de Datos (Data Control Language - DCL)**: El DCL de SQL ayuda al administrador a controlar la seguridad y los accesos a los datos, es decir, ayuda a mantener, administrar y realizar un control ordenado sobre los datos.

- **Lenguaje de Consulta de Datos (Data Query Language – DQL)**: El DQL de SQL proporciona comandos específicos para la recuperación de datos desde las tablas.

2.8. Fundamentos de "WebML"

2.8.1. WebML (Web Modeling Language – Lenguaje de Modelado Web) [5]

Es una notación virtual para diseñar aplicaciones Web complejas. Provee especificaciones gráficas, incluidas en un completo proceso de diseño. Constituye una notación visual para especificar las características de composición y navegación de las aplicaciones con enlace de hipertexto. Este novedoso método posee cinco modelos o diagramas: modelo estructurado, modelo de hipertexto, clasificado en dos:

modelo de composición y modelo de navegación; modelo de presentación y modelo de derivación o de personalización, los cuales son desarrollados en un proceso iterativo.

2.8.2. Modelo estructural [1][5]

Para WebML este expresa el contenido de los datos del sitio Web, en términos de las entidades existentes y las relaciones entre ellas. WebML no propone otro lenguaje para el modelado de datos, sin embargo, es compatible con notaciones clásicas como el modelo Entidad – Relación, el modelado orientado a objetos y los diagramas de clase de UML. Para hacer frente al requisito de expresar la información redundante y calculada, el modelo estructural también ofrece un simplificado lenguaje de consulta, por el cual es posible especificar la información derivada de dicha consulta. Los elementos del modelo estructural son: las entidades, los atributos de estas entidades, las relaciones entre entidades y sus herencias.

2.8.3. Modelo de Hipertexto [1][5]

Describe uno o más enlaces hipertextos que pueden ser publicados en el sitio Web. Cada enlace hipertexto define una vista del sitio Web, y las descripciones de las vistas del sitio Web consiste a su vez en dos sub-modelos: modelo de composición y modelo de navegación. Los ingredientes claves del WebML están constituidos por **páginas**, **unidades** y **enlaces** organizados en construcciones modularizadas llamadas **áreas** y **vistas** del sitio. Las unidades son las piezas atómicas de contenido publicable, que ofrecen formas alternativas de arreglar el contenido dinámicamente extraído de las entidades y relaciones del esquema de datos. Son los bloques de construcción de las páginas, que son la interfaz actual de los elementos presentados al usuario. Las páginas son típicamente construidas ensamblando varias unidades de muchos tipos, para mantener el efecto de

comunicación deseado. Los enlaces representan una vía en el modelo de hipertexto: ellos expresan la posibilidad de navegar de un punto a otro en el hipertexto, y el pasaje de parámetros de una unidad a otra unidad, que es requerida para el adecuado procesamiento del contenido de la página.

Un conjunto de páginas puede ser agrupado en una vista de sitio, conocida como SiteView, la que representa un conjunto bien definido de requisitos, por ejemplo, las necesidades de un grupo específico de usuarios. En aplicaciones mas grandes, puede haber múltiples vistas de sitios definidas sobre el mismo esquema de datos, y grandes vistas de sitios pueden ser jerárquicamente descompuestas en áreas, las cuales son un grupo de páginas con un propósito similar. Algunas propiedades de las paginas y las áreas, como el inicio, predeterminado y puntos de referencia, permiten al diseñador ajustar el nivel de visibilidad de esas construcciones dentro de la estructura jerárquica de la vista del sitio. Este modelo lo constituyen las unidades de contenidos y las páginas Web para el modelo de composición, y los enlaces o hipertextos para el modelo de navegación. Los parámetros globales pueden ser especificados al nivel de una vista de sitio, a denotar pequeñas piezas de información, las cuales pueden ser grabadas durante la navegación del usuario, para ser luego retirada y explotada en el procesamiento de los contenidos de alguna página.

2.8.3.1. Modelo de Composición o de Gestión de Contenido [5]

Este modelo o diagrama especifica cuáles páginas componen el enlace hipertexto, y cuáles entidades de contenido componen una página. Incluye muchas unidades operacionales predefinidas, las cuales ofrecen la mayoría de las primitivas más usadas para actualizar instancias de las entidades y relaciones de la aplicación, creando, modificando, y borrando objetos, y conectándose y desconectándose entre relaciones, además otras operaciones útiles como ingreso de usuario, salida de usuario y envío de E-Mails. Este modelo posee varios tipos de unidades de contenido que

pueden ser utilizadas para conformar las páginas y los procesos que en ellas se llevan a cabo: unidades de entrada de datos, unidades de consulta de datos, unidades de multi-datos, unidades de índice, unidades de creación, modificación y eliminación de datos, transacciones, unidades de desplazamiento entre datos, enlaces contextuales y no contextuales, páginas web, siteviews o vistas, entre otros.

2.8.3.2. Modelo de Navegación [5]

Expresa cómo las páginas, áreas y siteviews son enlazados para formar los hipertextos o referencias cruzadas de textos hacia otros textos. Los enlaces son tanto no contextuales: cuando conectan semánticamente páginas independientes (es decir, no transportan datos de la página de origen a la página destino), como contextuales: cuando el contenido de las unidades de destino del enlace depende del contenido de la unidad de origen (por ejemplo, las páginas Web dinámicas, que extraen e ingresan información de una base de datos). Estos últimos están basados en el esquema estructurado, porque ellos conectan unidades de contenido cuyas entidades subyacentes están asociadas por relaciones en el esquema estructurado.

2.8.4. Modelo de Presentación [5]

Terminología de WebML que expresa el aspecto de la disposición y del gráfico de páginas, independientemente del dispositivo de salida y del lenguaje de interpretación, por medio de una sintaxis abstracta de XML. Las especificaciones de presentación son tanto páginas específicas como páginas genéricas. El primer caso es por ejemplo cuando hace referencia a una página específica y se incluyen referencias específicas al contenido de la página; el segundo caso está basado en modelos predefinidos independientes del contenido específico de la página e incluye referencias a los elementos del contenido genérico.

2.8.5. Modelo de Personalización [5]

En WebML los usuarios y grupos de usuarios son explícitamente modelados en el esquema estructurado en la forma de entidades predefinidas llamadas USUARIO y GRUPOS respectivamente. Las características de estas entidades se pueden utilizar para almacenar el contenido individual o de grupo, como sugerencias de compras o lista de favoritos. Este contenido personalizado se puede utilizar tanto en la composición de unidades como en la definición de las especificaciones de presentación.

2.8.6. SiteViews

Un SiteView es un sub-conjunto de páginas en la que el usuario puede experimentar como un sitio Web completo. Representa un número de páginas y/o áreas que forman una vista coherente del sitio Web. SiteViews diferentes pueden estar definidos para diferentes grupos de usuarios. Siteviews múltiples pueden ser definidos en el mismo modelo de datos. Diferentes SiteViews pueden ser mostrados para diferentes tipos de usuarios y para diferentes tipos de aparatos de salida. Cada SiteView debe contener una página marcada como home, en este caso dichos "Homes" son las interfaces principales para cada grupo de usuario. Los tipos de SiteViews pueden ser:

- **Públicos**: cualquiera puede entrar.

- **Privados**: control de acceso con nombre de usuario y/o password.

Los mapas de SiteViews indican los diferentes sitios que se derivan de un SiteView específico o principal. Es como una distribución en forma de árbol donde el SiteView principal representa el nivel más elevado y los otros sitos que indica el

mapa del respectivo SiteView corresponden a sus derivaciones o ramificaciones. Estos sitios derivados pueden ser accedidos a través de los hiperenlaces, o inclusive de botones especiales o de ”SUBMIT”, que son los que hacen la solicitud de muestreo de una página Web específica haciendo el enlace respectivo.

2.8.7. Unidades de contenido [18]

Una unidad de contenido WebML es un elemento atómico de publicación de información. Existen muchas unidades de contenido, mencionadas en secciones previas, pero las más importantes predefinidas en WebML para componer páginas Web son: unidades de datos, de datos múltiples, unidades de índices (incluyendo sus variantes de índices jerárquicos y de selección múltiple), unidades de entrada o ingreso de datos y unidades de desplazamiento entre datos.

2.8.8. Estereotipos WebML [4]

La herramienta de modelado WebML ofrece un amplio conjunto de notaciones, símbolos y estereotipos que permiten representar la definición, estructuración e interpretación de aplicaciones Web. En la tabla que 2.1, dividida en varias partes debido a su tamaño, se pueden observar algunos de estos estereotipos:

Tabla 2.1. Resumen de Estereotipos de WebML para Aplicaciones Web

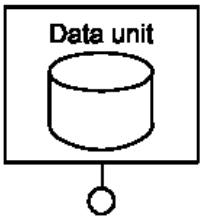
Elemento WebML	Descripción	Propiedades
	Publica un solo objeto de una entidad dada.	» Nombre » Entidad de Origen » Selector (opcional) » Atributos incluidos

Tabla 2.1. Resumen de Estereotipos de WebML para Aplicaciones Web (Continuación)

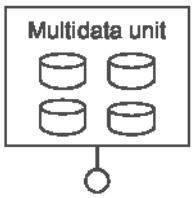
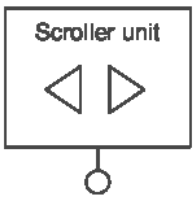
Elemento WebML	Descripción	Propiedades
	Este elemento presenta múltiples objetos de una misma entidad juntos, repitiéndose la presentación de muchas unidades de dato.	» Nombre » Entidad de Origen » Selector (opcional) » Atributos incluidos » Cláusula de orden (opcional)
	Presentan múltiples objetos de una entidad como una lista.	» Nombre » Entidad de Origen » Selector (opcional) » Atributos incluidos » Cláusula de orden (opcional)
	Una variante del anterior, donde cada elemento de la lista está asociado con un "checkbox" permitiendo al usuario seleccionar múltiples objetos de una lista determinada.	» Nombre » Entidad de Origen » Selector (opcional) » Atributos incluidos » Cláusula de orden (opcional)
	Una variante del "index unit", en cual las entradas de los índices están organizadas en un árbol multi – nivel.	» Nombre » Por cada nivel: » Entidad de Origen » Selector (opcional) » Atributos incluidos » Cláusula de orden (opcional)
	Provee comandos para desplazarse a través de los objetos de un conjunto.	» Nombre » Entidad de Origen » Selector (opcional) » Factor de bloqueo (opcional) » Cláusula de orden (opcional)

Tabla 2.1. Resumen de Estereotipos de WebML para Aplicaciones Web (Continuación)

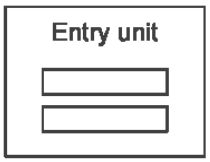
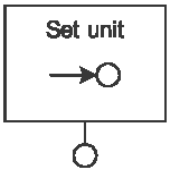
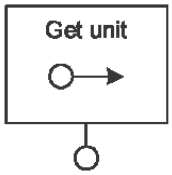
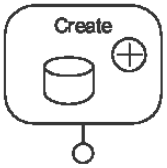
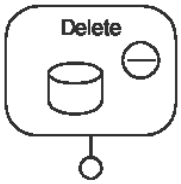
Elemento WebML	Descripción	Propiedades
	Soporta entrada de datos basada en formulario.	» Nombre » Por cada campo: » Nombre » Tipo » Valor inicial » Modificabilidad » Valor predicado
	Define una operación genérica: los parámetros de entrada y salida deben estar definido por el diseñador.	» Definidos por el diseñador
Parámetro Global o Variable de Sesión	Guarda información para usar en múltiples páginas. Sus símbolos son:	» Nombre » Tipo » Valor por defecto
	Asigna un valor a un parámetro global o variable de sesión.	» Parámetro global
	Recupera el valor de un parámetro global o variable de sesión.	» Parámetro global
	Habilita la creación de una nueva instancia de una entidad.	» Nombre » Entidad de origen » Conjunto de agnaciones de valores
	Elimina uno o más objetos de una entidad dada.	» Nombre » Entidad de origen » Selector

Tabla 2.1. Resumen de Estereotipos de WebML para Aplicaciones Web (Continuación)

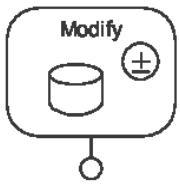
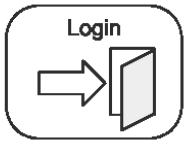
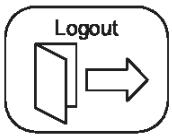
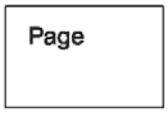
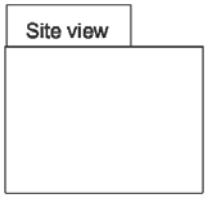
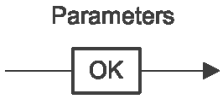
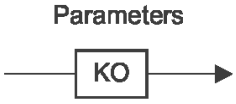
Elemento WebML	Descripción	Propiedades
	Modifica o actualiza uno o más objetos de una entidad dada.	» Nombre » Entidad de origen » Selector » Conjunto de asignaciones de valores
	Verifica la identidad de un usuario que acceda al sitio.	» Nombre de Usuario » Contraseña
	Lleva al usuario a la página principal sin control de acceso.	» Ninguno
	Representa la interfaz actual navegada por el usuario, contiene unidades y/o subpáginas.	» Nombre » Punto de referencia » Contenido: unidades o subpáginas
Transaction 	Es una colección de operaciones que realiza una única unidad lógica de trabajo.	» Ninguno
	Es un contenedor de páginas o, recursivamente, otras subáreas, usadas para darle una organización jerárquica al hipertexto.	» Nombre » Punto de referencia » Contenido: páginas, subáreas y página por defecto o subpágina
	Representa un hipertexto.	» Nombre » Contenido: páginas, áreas y página inicial

Tabla 2.1. Resumen de Estereotipos de WebML para Aplicaciones Web (Continuación)

Elemento WebML	Descripción	Propiedades
	Provee la capacidad de enviar mensajes de E-Mail.	» Remitente » Recipiente » Asunto » Cuerpo » Adjuntos
Enlace 	Es una conexión orientada entre dos unidades o páginas. Abstrae el concepto de ancla y permite portar información (por medio de parámetros entre unidades). Pueden ser definidos como:	Enlaces normales, automáticos y de transporte: » Nombre » Elemento origen » Elemento destino » Tipo de enlace
- Enlace Automático  - Enlace de Transporte 		
- Enlace OK  - Enlace KO 	-OK Link: se ejecuta en caso de éxito en la operación. -KO Link: se ejecuta en caso de que la operación falle.	» Nombre » Elemento origen (unidad de operación) » Elemento destino (Parámetros de enlace)

2.9. Programación WEB

2.9.1. Introducción al Desarrollo de Aplicaciones Web [4]

El desarrollo de una aplicación Web de manejo intenso de datos es una actividad multidisciplinaria, que requiere de una variedad de habilidades, necesariamente para direccionar cada tarea, como el diseño de estructuras de datos para almacenar contenido, la concepción de interfaces de hipertexto para la exploración de información y administración de contenido, la creación de estilos de presentación efectivos, el ensamble de arquitecturas robustas y de alto desempeño. El desarrollo y mantenimiento de aplicaciones Web de manejo intenso de datos requiere todas las herramientas y técnicas de ingeniería de software, incluyendo el proceso de desarrollo de software bien organizado, conceptos de diseño y notaciones apropiadas, y guías sobre como conducir las actividades. La mezcla propuesta une los ingredientes tradicionales conocidos por los desarrolladores, como el diseño conceptual de datos con el modelo Entidad – Relación y especificación de casos de uso con UML, con nuevos conceptos y métodos para el diseño de hipertextos, los cuales se centralizan en el desarrollo Web. Sin embargo, el valor de la propuesta no está en los ingredientes individuales, sino en la definición de una estructura sistemática en la cual las actividades de desarrollo de aplicaciones Web pueden ser organizadas de acuerdo a los principios fundamentales de desarrollo de software, para encontrar el soporte adecuado en los conceptos, notaciones y técnicas apropiadas.

2.9.2. Aplicación Web [2]

Una aplicación Web es una aplicación desarrollada utilizando tecnologías basadas en el entorno Web como HTML, XML, JavaScript, PHP, ASP etc. Una aplicación Web es un conjunto de páginas que interactúan unas con otras y con diversos recursos en un servidor Web, incluidas bases de datos. Esta interacción

permite implementar características en su sitio. Adicionalmente podrá realizar consultas a bases de datos, registrar e ingresar información, solicitudes, pedidos y múltiples tipos de información en línea en tiempo real. Las funcionalidades de los componentes de una aplicación Web son las siguientes:

- **Navegador Web:** este es el componente del lado del cliente, es una aplicación usada para solicitar páginas Web, recuperarlas y mostrar a los usuarios la información solicitada ya procesada.

- **Servidor Web:** Acepta las solicitudes por páginas Web hechas por el cliente y las entrega, puede aceptar solicitudes por scripts en el lado del servidor que generan páginas Web dinámicas, por ejemplo Scripts PHP.

- **Recursos Externos:** Las aplicaciones Web necesitan acceder a la información o solicitar servicios a más de una fuente. Un Sistema de Administración de Base de Datos Relacional (RDBMS) por ejemplo y cualquier otro recurso externo al servidor entran en esta categoría.

Para transmitir información se requiere un medio de comunicación. Uno de los modos de comunicación que usan los computadores es conectarse a Internet o a la WWW. La conversión Estándar que se sigue para esta comunicación es el **H**yper **T**ext **T**ransfer **P**rotocol (HTTP - Protocolo de Transmisión de Hipertexto). En HTTP la información es enviada y recibida en forma de solicitudes y respuesta. La información en Internet o Word Wide Web (WWW) esta guardada en forma de documentos, generalmente como páginas Web, estas páginas están almacenadas en una ubicación central denominada servidor para que múltiples usuarios (clientes Web) acceden y compartan información, las ventanas de acceso a Internet se denomina navegador Web.

2.9.3. Programación Web Cliente / Servidor [4]

El **Cliente a**ctúa como representante del usuario enviando la petición al servidor, y como agente receptor recibiendo la respuesta del servidor. El *Cliente* proporciona la interfaz para que los usuarios ingresen los valores solicitados, establece el enlace de comunicación con el servidor, recibe la respuesta del servidor y la presenta a los usuarios. El **Servidor** es *un programa* que responde la consulta del cliente. Toma los detalles solicitados por el usuario en la base de datos y los devuelve al cliente solicitante.

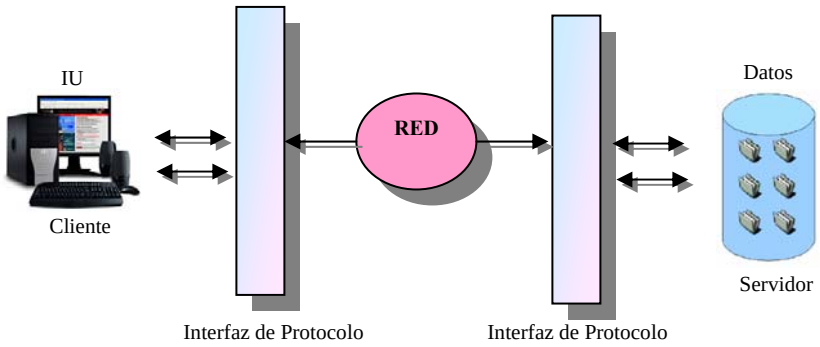


Figura 2.2. Arquitectura Cliente / Servidor
Fuente: Autor desconocido

2.9.4. Sitio Web [4]

Un sitio Web (en inglés: website) es un conjunto de páginas web, típicamente comunes a un dominio de Internet o subdominio en la World Wide Web Internet. Un sitio Web puede ser el trabajo de una persona, una empresa u otra organización y está típicamente dedicada a algún tema particular o propósito. No debemos confundir sitio Web con página Web, esta última es sólo un archivo HTML, y forma parte de un sitio Web. Al ingresar una dirección, como por ejemplo www.wikimedia.org, siempre se está haciendo referencia a un sitio Web, que tiene una página HTML inicial, que es lo primero que se visualiza. La búsqueda en Internet se realiza

asociando el DNS ingresado con la dirección IP del servidor que contenga el sitio Web en el cual está la página HTML buscada.

2.9.5. Interfaz Estática de Usuario [4]

Es aquella interfaz que tiene contenido que no se espera que cambie frecuentemente, y se mantiene manualmente por alguna persona o personas que usan algún tipo de programa editor. Su información no cambia, por ejemplo, según el tipo de usuario que la solicita.

2.9.6. Interfaz Dinámica de Usuario [4]

Es aquella que puede tener cambios frecuentes en la información. Cuando el servidor Web recibe una petición para una determinada página de un sitio Web, la página se genera automáticamente por el software como respuesta directa a la petición de la página y que puede variar de acuerdo a diversos datos y características en un momento determinado.

2.9.7. HTML [2]

Es el acrónimo de HyperText Markup Language (Lenguaje de Marcado de HiperTexto). Constituye el lenguaje de marcas de hipertexto que se utiliza para documentos del World Wide Web. HTML es una aplicación de SGML que utiliza etiquetas para marcar los elementos, como texto y gráficos, en un documento para indicar como deberían visualizar los exploradores Web estos elementos al usuario y como deberían responder a las acciones de este, como la activación de un enlace presionando una tecla o haciendo clic con el ratón. HTML 2.0, definido por el Internet Engineering Task Force (IETF), incluye características del HTML común a

todos los exploradores Web alrededor de 1995 y fue la primera versión de HTML ampliamente utilizado en el WWW. En 1994, se propuso HTML+ para ampliar el HTML 2.0 pero nunca se implementó. HTML 3.0, que nunca se estandarizó ni fue implementado totalmente por ningún desarrollador de exploradores Web, introdujo las tablas. HTML 3.2, el último estándar propuesto, incorpora características ampliamente implementadas en 1996. La mayoría de los exploradores, especialmente Netscape Navigator, Mozilla e Internet Explorer, reconocen más etiquetas de HTML que las incluidas en el estándar actual. HTML 4, la última especificación, soporta hojas de estilo y lenguajes de script e incluye características de internacionalización y de accesibilidad.

El Lenguaje de Marcado de Hipertexto fue desarrollado para que los documentos WWW incluyan texto, imágenes, tablas, hiperenlaces y archivos animados. Los documentos Web están disponibles en el servidor como archivos HTML. Un Archivo HTML tendrá extensión .html o .htm dado que el navegador Web lee el archivo HTML por medio de etiquetas HTML específicas, la pagina Web mostrada como salida al usuario puede variar de un navegador a otro. El conjunto de páginas HTML se pueden enlazar unas con otras usando hiperenlaces, los cuales pueden ser texto o una imagen. Cualquier código HTML comprende dos componentes: Etiquetas y Atributos.

- **Las Etiquetas:** Deciden la naturaleza del formato que se va a aplicar a los documentos HTML, las etiquetas tienen un conjunto de atributos posibles que deciden la extensión y el estilo del formato que se va a aplicar.

- **Atributos:** Se utilizan para manejar requerimientos de los formatos El formato de texto puede incluir elementos como alineación, ancho, tamaño etc.

2.9.7.1. Ventajas de HTML [2]

- Se puede compartir gran cantidad de datos alrededor del mundo.
- El HTML es básicamente formato ASCII, por lo que la posibilidad de corrupción de datos por la red es mucho menor.
- Dado que es un lenguaje de marcado, es fácil de desarrollar y simple para comprender.
- El HTML es fácil de comprender ya que tiene un conjunto de etiquetas.

2.9.8. Lenguaje Java [11]

Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria. Las aplicaciones Java están típicamente compiladas en un bytecode, aunque la compilación en código máquina nativo también es posible. La implementación original y de referencia del compilador, la máquina virtual y las bibliotecas de clases de Java fueron desarrolladas por Sun Microsystems en 1995. Desde entonces, Sun ha controlado las especificaciones, el desarrollo y evolución del lenguaje a través del Java Community Process, si bien otros han desarrollado también implementaciones alternativas de estas tecnologías de Sun, algunas incluso bajo licencias de software libre.

Java es un lenguaje simple de usar, ya que su estilo de programación está basado en C++, de allí que los programadores no encuentran Java muy difícil para

entender y usar; Java proporciona un modelo de seguridad que previene el acceso a los recursos del sistema, y así bloquear cualquier intento malicioso; permite descargar programas y ejecutarlos dinámicamente en diversos tipo de plataformas, mediante un mecanismo para crear código ejecutable de acuerdo a la plataforma empleada; igualmente Java permite la computación distribuida, útil en ambiente de la Internet, donde los módulos de los programas pueden estar en diferentes computadoras alrededor del mundo, lo que permite a un programa que se ejecutan en una computadora invocar un método en una clase Java ejecutándose en otra.

2.10. Fundamentos técnicos de HIDROCONS, C. A. [13]

2.10.1. Estación de Servicio (Establecimiento dispensador de combustible)

Una estación de servicio o gasolinera es un punto de venta de combustible y lubricantes para vehículos de motor; generalmente gasolina o derivados del petróleo. Aunque en teoría pueden establecerse y comprar libremente, las estaciones de servicio normalmente se asocian con las grandes empresas distribuidoras, con contratos de exclusividad. Algunas estaciones proveen combustibles especiales como gas licuado del petróleo (GLP), gas natural, gas natural comprimido, hidrógeno, biodiesel o keroseno. Asimismo, en algunos países también venden bombonas de butano. En la década de 1990, las estaciones de servicio ampliaron su oferta con artículos variados, dando lugar a las llamadas tiendas de conveniencia o minimercados que pasaron a ser habituales en las gasolineras.

2.10.2. Dispensador o surtidor de Combustible

El surtidor o dispensador de combustible es una máquina propia de una estación dispensadora de combustible que se utiliza para abastecer de gasolina a vehículos automotores. El dispensador de combustible también se le conoce como bomba de

gas o dispensador de gasolina. Este dispositivo consiste en dos partes principales: una "cabeza electrónica" que contiene un sistema integrado para controlar la acción de la bomba y que se comunica con un sistema en el interior para indicar las ventas, y en segundo lugar, una sección mecánica que contiene una bomba eléctrica y unas válvulas para bombear físicamente el combustible. El flujo del combustible es medido por unos codificadores rotatorios que generan pulsos eléctricos. En algunos casos el combustible real se puede sellar y sumergir dentro de los depósitos de gasolina en un sitio, en este caso se conoce como bomba sumergible. Los surtidores de combustible son hechos por diversas compañías a través del mundo, entre las cuales están: Tokheim, Gilbarco Veeder-Root y Wayne.

2.10.3. Bomba Sumergible

Una bomba sumergible es una bomba que tiene un motor sellado a la carcasa. El conjunto se sumerge en el líquido a bombear. La ventaja de este tipo de bomba es que puede proporcionar una fuerza de elevación significativa pues no depende de la presión de aire externa para hacer ascender el líquido. Las bombas sumergibles encuentran muchas utilidades: las bombas de etapa simple se utilizan para el drenaje, el bombeo de aguas residuales, el bombeo industrial general y el bombeo de la mezcla. Las bombas sumergibles se colocan habitualmente en la parte inferior de los depósitos de combustible y también se utilizan para la extracción de agua desde algunos pozos.

2.10.4. Tanque de almacenamiento de combustible

Depósito diseñado para almacenar o procesar fluidos, generalmente a presión atmosférica o presión internas relativamente bajas. Los tanques de almacenamiento se usan como depósitos para contener una reserva suficiente de algún producto para su uso posterior y/o comercialización.

2.10.5. Combustible

Es cualquier material capaz de liberar energía cuando se quema, y luego cambiar o transformar su estructura química. Supone la liberación de una energía de su forma potencial a una forma utilizable. Entre los combustibles sólidos se incluyen el carbón, la madera y la turba. Entre los combustibles fluidos, se encuentran los líquidos como el gasóleo, el queroseno o la gasolina (o nafta) y los gaseosos, como el gas natural o los gases licuados de petróleo (GLP), representados por el propano y el butano.

2.10.6. Gasolina

La gasolina es una mezcla de hidrocarburos, derivada del petróleo, que se utiliza como combustible en motores de combustión interna con encendido a chispa.

2.10.7. Accesorios y repuestos para venta

Los accesorios y repuestos destinados a la venta son aquellos utilizados, además de complementar cualquier trabajo de mano de obra, para ser vendidos al público en general.

CAPÍTULO III

FASE DE INICIO

3.1. Descripción de la empresa

3.1.1. Ubicación geográfica de la empresa

La sede principal y única de HIDROCONS, C. A. está ubicada en la ciudad de Puerto La Cruz – Estado Anzoátegui, Venezuela, específicamente en el municipio Juan Antonio Sotillo, Sector La Caraqueña – Parroquia Pozuelos, en el taller N°. 21.

3.1.2. Naturaleza de la empresa

HIDROCONS, C. A. es una empresa principalmente de servicios, con personalidad Jurídica domiciliada en Puerto La Cruz – Estado Anzoátegui, inscrita ante el Registro de Primera Instancia en lo Civil, Mercantil del Trabajo y del Tránsito de la Circunscripción Judicial del Estado Monagas y el Estado Delta Amacuro.

3.1.3. Objetivos de la empresa

El objetivo principal de la empresa HIDROCONS, C. A. está relacionado con ofrecer servicios de construcción de obras civiles en general, instalación, reconstrucción, mantenimiento y reparación de equipos de estaciones de servicios dispensadoras de combustible para vehículos automotores, electricidad, plomería y pintura, así como también la compra, venta, importación y exportación de todo tipo de material de construcción.

3.1.4. Organigrama de la empresa

Actualmente la estructura organizativa de la empresa HIDROCONS, C. A. está definida como se observa en la figura 3.1 mostrada a continuación. Como se puede inferir, los empleados que podrían ser usuarios del sistema en cuestión vendrían a ser el Director Gerente, Administrador, Contador y Secretaria.

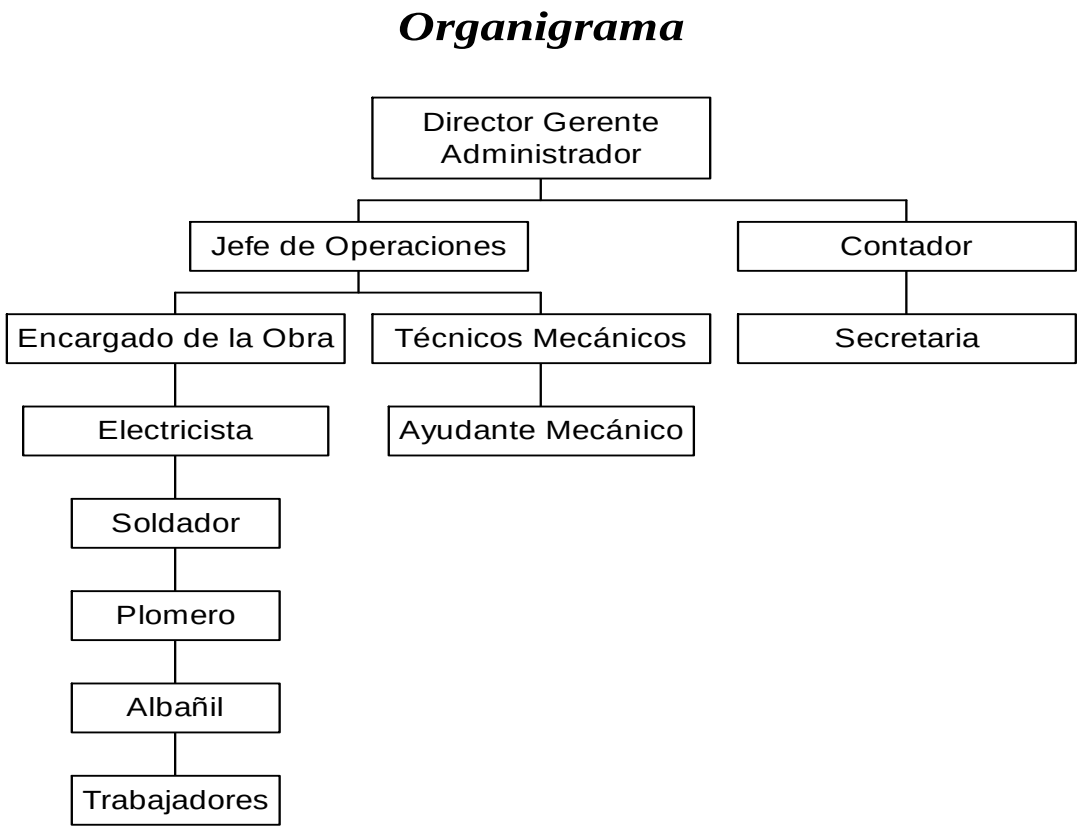


Figura 3.1. Organigrama de HIDROCONS, C. A.
Fuente: HIDROCONS, C. A., 2007

3.1.5. Misión de la empresa

La misión de la empresa HIDROCONS, C. A. es, en líneas generales, proveer, fomentar e impartir servicios de mantenimiento y remodelación de establecimientos

dispensadores de combustibles para vehículos automotores, ofreciendo productos innovadores y con tecnología de punta, atención personalizada y excelente servicio, con un personal altamente capacitado, calificado e identificado con la organización, y tecnología de punta que permita la total y garantizada satisfacción de los clientes.

3.1.6. Visión de la empresa

La visión de la Empresa HIDROCONS, C. A. es, en resumen, ser la empresa contratista por excelencia en el ramo de mantenimiento de Estaciones de Servicio suplidoras de combustible, a la vez de ofrecer servicios y productos destinados para tal fin, siendo líderes en el mercado de esta rama, y consolidándose como empresa responsable, dinámica, eficiente y altamente competitiva en dicho ramo económico.

3.2. Introducción

La fase de inicio de todo proyecto de desarrollo de software tiene sus orígenes en el Proceso Unificado Racional de desarrollo de software (RUP). RUP es la metodología que rige el marco de trabajo para el desarrollo de este proyecto y la misma se dá a través de cuatro fases de desarrollo, a saber: 1.- *Fase de inicio*, fase a la cual pertenece esta sección del proyecto, 2.- *Fase de elaboración*, 3.- *Fase de construcción* y 4.- *Fase de transición*. La mayoría de los proyectos requieren de una etapa inicial breve en la que se estudian los siguientes tipos de preguntas:

- ¿Cuál es la visión y el análisis del negocio para este proyecto?
- ¿Cuánto ha de costar aproximadamente el proyecto?
- ¿Es viable?

- ¿Qué recursos se necesitarán?
- ¿Deberíamos abordarlo o no seguir?
- ¿Qué procesos seguir y qué herramientas usar?

En la fase de inicio se identifican los requerimientos y/o necesidades del programa a elaborar, tomando en cuenta las características más importantes de los procesos involucrados en el manejo del mismo. Estos requerimientos se obtienen a través de observaciones, entrevistas, recolección de los datos, análisis de la información existente, (como entradas de datos, el flujo de información, control y salidas requeridas) lo que permite delimitar el alcance del proyecto, para así modelar el sistema y obtener una arquitectura base. Además se establece la forma en que los usuarios llevan a cabo los procesos, permitiendo determinar los requisitos adicionales para el funcionamiento del sistema e identificar los riesgos del mismo lo que conlleva a una vista del diseño completo. Para definir la visión y obtener una estimación del orden de magnitud del proyecto a desarrollar es necesario llevar a cabo alguna exploración de los requisitos. Sin embargo, el objetivo de esta etapa de inicio no es definir todos los requisitos existentes, o generar una estimación creíble o plan de proyecto. Aún a riesgo de simplificar demasiado, la idea es hacer la investigación justa para formar una opinión racional y justificable del propósito global y la viabilidad del sistema en cuestión. [15]

3.3. Planificación de la fase de inicio

Entender lo que se desea alcanzar en esta fase ayudará a aplicar el enfoque RUP a los proyectos con más eficacia. Se seleccionan y realizan solamente las actividades que contribuyan a alcanzar los objetivos de un proyecto en particular. La meta más importante de esta fase es alcanzar consenso de los objetivos del ciclo de vida del

proyecto entre todos los inversionistas y afectados por el desarrollo del proyecto. [9] En otras palabras, la idea de esta fase es realmente entender el alcance del proyecto y los objetivos, obtener suficiente información para confirmar qué se debería proceder con el proyecto o tal vez concluir qué no se debería hacer en el desarrollo del mismo. Esta fase tiene cuatro objetivos básicos, que son:

- **Entender qué se va a construir:** determinar la visión, el alcance del sistema y los límites de este, es decir, qué está dentro del sistema y qué está afuera. Establecer el alcance del proyecto de software y las condiciones que lo limitan, incluyendo una visión operacional, criterio de aceptación y qué está previsto hacer en el producto y qué no. La visión debería estar completa y estable al final de esta fase, pero se puede continuar refinando a través del proyecto. Es importante que la visión sea pública, compartida y constantemente revisada de modo que nadie pueda decir que no la entendía o no la conocía. Esta descripción requiere dos actividades básicas: [9]

  - **Identificar los usuarios típicos del sistema (actores):** esta es una actividad que se realiza para encontrar los actores que interactúan con el sistema. Es bastante útil para establecer las permisologías y privilegios de cada usuario, la relación entre actores y una visión clara sobre el alcance de cada actor o entidad que interactúe con el sistema. Los actores son los usuarios que interactúa con el sistema que se esta diseñando, e inclusive puede ser otro sistema o subsistema.
  - **Identificar las interacciones típicas con el sistema o casos de uso:** cada forma en que los actores usan el sistema, o sea, la permisología y privilegios de los mismos, se representa con un caso de uso. Es una secuencia de un conjunto de acciones que realiza un sistema con respecto a un actor particular interesado en este. Los casos de uso ayudan a descubrir requerimientos, capturar las necesidades de los usuarios,

formular planes de prueba del sistema y controlar las iteraciones e integración del mismo. El modelo de caso de uso permite formar el sistema de manera como lo entendería el usuario, se usa en el levantamiento de la información, debido a que el usuario puede tener una mejor visión del sistema sin entrar en profundidad de programación.

- **Comprender los costos, cronogramas y riesgos asociados con el proyecto:** entender lo que se va a construir es clave, pero determinar cómo construirlo y cuánto cuesta es crucial. Para determinar si se debería continuar con un proyecto se necesita comprender ¿cuánto costará el proyecto? La mayoría de estos costos están relacionados a ¿qué recursos se necesitarán?, ¿cuánto tiempo tomará terminar el proyecto?, etc. Combinando todo este conocimiento con el entendimiento de la funcionalidad requerida y el valor de la misma para los usuarios, se puede construir un caso de negocios para el proyecto. Durante la fase de elaboración se enfrentará la gran mayoría de los riesgos relacionados con la tecnología y la arquitectura que se identificaron durante la fase de inicio. [9]

- **Decidir qué proceso seguir y qué herramientas usar:** es importante que el equipo de trabajo comparta un conjunto de vista común de cómo será construido el software, es decir, cuál será el proceso de software a seguir. Se debería racionalizar el proceso, para minimizar la carga de trabajo innecesario y asegurar que el proceso de software trate las necesidades específicas del proyecto. La idea es plantear un proceso de software y una herramienta de desarrollo y trabajar en ella en la primera iteración. [9]

- **Identificar por lo menos una arquitectura candidata:** puesto que la meta de la fase de inicio es determinar si tiene sentido de continuar con el proyecto, es necesario cerciorarse de que haya por lo menos una arquitectura candidata

que permitirá construir el sistema con una mínima cantidad de riesgos a un costo razonable. [9]

Estos cuatro puntos anteriormente descritos constituyen prácticamente lo que ser refiere a la fase de inicio de un proceso de desarrollo de software, y por lo que se vé, se refiere a un análisis exclusivo para determinar lo viable y factible de la aplicación en cuestión. A continuación se lleva a cabo el análisis de esta fase de inicio según los puntos y objetivo especificados anteriormente.

3.4. Análisis

3.4.1. Requisitos esenciales

Los requisitos son declaraciones de los servicios que debe proporcionar el sistema, son un conjunto de ideas donde se percibe un bosquejo general del software que se quiere desarrollar. Los mismos se originan según las necesidades de cada usuario y aparecen durante la vida de un sistema. Especifican la manera en que este debe reaccionar a determinadas entradas, como debe comportarse en situaciones particulares y declarar explícitamente lo que no debe hacer el sistema. Estos deben ser precisos, pues deben definir exactamente que es lo que se va a desarrollar. Algunos de los requisitos candidatos que el sistema podría suponer son los que a continuación se mencionan:

- Tener un registro actualizado de las compañías y de sus representantes clientes de HIDROCONS, C. A, así como de los presupuestos, facturas y solicitudes de repuestos emitidos.

- Permitir que los clientes puedan ver los presupuestos de algún trabajo a realizarse en su Estación de Servicio o infraestructura en la comodidad de su casa u oficina, para aprobarlos o rechazarlos en el menor tiempo posible.

- Facilitarles a las compañías una vista de sus facturas emitidas, tanto de facturas canceladas como la que se encuentra por cancelar, para que estén al tanto de sus estados de cuenta.

- Darles a conocer a los clientes información acerca de los repuestos de dispensadores de combustible y bombas sumergibles, así como de los servicios que realiza.

- Permitir que los clientes registrados en el sistema realicen solicitudes de repuestos e inclusive envío de comentarios y sugerencias vía On-Line.

- Todo esto enmarcado en una aplicación Web cuya interfaz sea amigable, clara y sencilla, para que los usuarios puedan interactuar con la misma sin complicaciones mayores.

3.4.2. Requisitos adicionales

Los requisitos adicionales son fundamentalmente requisitos no funcionales que no pueden asociarse a ningún caso de uso. A continuación se muestra la lista de requisitos adicionales obtenida en el estudio del sistema propuesto:

- La empresa debe tener un servidor que procese las solicitudes de los usuarios y maneje la base de datos, y así cubrir con las necesidades de funcionamiento.

- Es necesario contar con un sistema operativo para el equipo servidor tal como Microsoft Windows XP o superior, o en su defecto, Linux en cualquiera de sus versiones. Igualmente el software de navegación o programa visualizador de Internet usado para interactuar con la aplicación puede ser Microsoft Internet Explorer 6.0 o superior, e inclusive, Netscape Navegator o Mozilla.

- Se deben establecer criterios de autorización para el acceso a la información, es decir, sólo personas autorizadas tendrán acceso a la información debidamente clasificada para mantener la privacidad de la data.

- El sistema propuesto debe ser práctico, consistente y flexible para acelerar el proceso de aprendizaje de uso.

- Se necesita que el sistema esté disponible a los usuarios cuando éstos lo consideren necesario. Generalmente el horario del mismo puede coincidir con los horarios de oficina de la empresa HIDROCONS, proveedora del servicio.

3.4.3. Contexto del sistema

El contexto muestra la visión general que se obtiene de una situación en particular. Para la descripción de este contexto fue necesario realizar un estudio para percibir las actividades relacionadas con los procesos administrativos, técnicos y de inventario de la empresa HIDROCONS, C. A. El estudio está basado en la realización de entrevistas al personal encargado de las áreas operativas y administrativas, observaciones para visualizar el desarrollo de los procesos que se realizan en tales áreas y el estudio de la terminología de las mismas. Esto permitió conocer los procesos involucrados en el sistema, al igual que las necesidades o requerimientos de los usuarios, la forma de ejecución y operaciones de los mismos. La modalidad de trabajo de la empresa es la siguiente: el cliente contacta

eventualmente a la empresa para un trabajo de cualquier índole o compra de repuesto. En caso de ser una solicitud de servicio técnico, bien sea correctivo o preventivo, a efectuarse en las infraestructuras de la empresa que solicita el servicio, el personal se dirige hacia la misma, hace los estudios necesarios, regresa a la sede de HIDROCONS y emite un presupuesto para hacerlo llegar al cliente, quien decide si se aprueba o no el mismo. Al ser aprobado, el personal se traslada nuevamente al local, satisface la demanda y emite la facturación respectiva. Si lo que se solicita es repuestos, pues el cliente contacta a la contratista y hace conocer su solicitud, luego se le informa de los precios y estos deciden si se procede con la compra o no.

El sistema propone llevar un control automatizado de los procedimientos administrativos de la empresa HIDROCONS, C. A., en lo referente a clientes y empresas a las cuales se les presta servicio, información de repuestos y servicios a ofrecer, registro, divulgación y aprobación de presupuestos y de solicitudes de cotizaciones de repuestos, además de la facturación propia para cada empresa cliente y sus correspondientes estados de cuenta.

3.4.4. Riesgos del sistema

El modo que se planifica el desarrollo de un nuevo sistema esta influenciado en gran medida por los riesgos que se perciben del mismo. Por lo tanto uno de los pasos propios de la fase de inicio es identificar los posibles riesgos y solventar aquellos que podrían hacer que el sistema fallara. Los riesgos identifican problemas potenciales de diseño, implementación, de interfaz, verificación y de mantenimiento, además de amenazas contra la calidad y la planificación temporal del software que hay que producir. Ésta actividad se inicia en la primera etapa de un proyecto de software y se solventa a lo largo de todo su ciclo de vida (hasta la aceptación del producto del proyecto). La identificación de los riesgos del sistema, nos permite especificar y determinar los elementos de riesgos potenciales del software en desarrollo con el fin

de evitarlos cuando sea posible y controlarlos cuando sea necesario. [15] A continuación, en la tabla 3.1, se muestra una lista conteniendo los riesgos del sistema que se han identificados hasta los momentos y su categoría correspondiente:

Tabla 3.1: Identificación de los Riesgos del sistema

Ubicación	Responsable	Problema o Riesgo	Contingencia / Solución
Interfaz	Desarrollador del Software	Interfaz confusa e incomprensible para el usuario.	1.- Desarrollar una interfaz amigable y fácil de entender para el usuario. 2.- Capacitar al personal con respecto al buen funcionamiento del sistema. 3.- Elaborar un módulo de ayuda adaptado a los requerimientos de los diversos tipos de usuarios.
Software	Desarrollador del Software	Los algoritmos diseñados no gestionan de forma correcta los distintos procesos presentes en el sistema.	Verificar exhaustivamente el comportamiento de cada uno de los procesos de forma gradual, hasta lograr el funcionamiento eficiente y eficaz del sistema.
		Uso de herramientas inadecuadas para el desarrollo del sistema.	Estudiar detalladamente las herramientas que servirán de apoyo para el desarrollo de los procesos.
		La configuración del sistema es ineficiente.	Diseñar un módulo de configuración robusto que conlleve a un sistema apto según los requisitos exigidos por el servicio.

Tabla 31: Identificación de los Riesgos del sistema (Continuación)

Ubicación	Responsable	Problema o Riesgo	Contingencia / Solución
Plataforma	Desarrollador del Software	Incompatibilidad de la integración de la aplicación con la plataforma.	Comprobar si existe la compatibilidad entre el sistema con la plataforma de hardware y la de software.
Servidor	HIDROCONS, C. A.	Ausencia de conexión y de servicios de Internet.	Comprobar si existe servicio de Internet, en caso contrario, instalarlo y reponerlo.
Base de Datos	Desarrollador del Software	Acceso fallido a la base de datos.	Verificar que la conexión a la base de datos se efectúe de manera correcta y exitosa.
		Sentencias SQL algo erróneas.	Comprobar que las sentencias SQL utilizadas, realicen las acciones requeridas para las cuales fueron diseñadas.
		Pérdida formato de información.	Crear un respaldo adecuado de los datos de modo que no haya inconsistencia entre ellos y los originales.
Hardware	HIDROCONS, C. A.	No contar con el hardware necesario (rendimiento, función, capacidad de la memoria, entre otros) para realizar la implementación del software en cuestión.	La empresa puede estar en capacidad de adquirir algún hardware necesario para el buen funcionamiento del sistema, en la medida de sus posibilidades.
		La arquitectura del sistema no es lo suficientemente apta y robusta, es decir, no puede ocuparse de muchas demandas de clientes simultáneas sin sacrificar su buen funcionamiento.	Evaluar exhaustivamente las arquitecturas candidatas y determinar la más apropiada para la posible demanda del sistema.

Tabla 31: Identificación de los Riesgos del sistema (Continuación)

Ubicación	Responsable	Problema o Riesgo	Contingencia / Solución
Usuarios Finales	Usuarios Finales	Aparición de nuevos requerimientos y riesgos a lo largo del desarrollo de la aplicación.	Presentación y muestra constante y periódica de prototipos del sistema a los usuarios finales, a fin de que alguna observación que tengan que hacer sea remediada a tiempo.
		Insatisfacción en el momento de finalizar y presentar a los usuarios el sistema en cuestión.	
		El usuario no obtiene toda la accesibilidad a ciertas zonas deseadas y permitidas para ella medida que ofrece el software.	

3.4.5. Actores del sistema

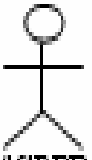
Los actores del sistema son los sujetos o entidades las cuales interactúan con el mismo, es decir, hacen uso de dicha aplicación. Constituyen un conjunto de personas, sub-sistemas o sistemas adicionales que se relacionan con este. Tienen la propiedad de ser externos a la aplicación en cuestión. Se debe tener en cuenta que un usuario puede acceder al sistema como un actor en particular o inclusive, como diversos actores, según sea el tipo de usuario con el cual esté registrado y de acuerdo a los privilegios y permisologías que se le atribuyan. La representación gráfica de un actor en los diferentes diagramas a los cuales se le hace alusión se denotará mediante el dibujo de un monigote o muñeco.

Seguidamente se muestra la tabla 3.2, el cual incluye un resumen de cuáles actores intermedian con dicha aplicación.

Tabla 32. Descripción de los Actores que interactúan con el sistema

Nombre del Actor	Representa	Actividades que realiza
 Empleado HIDROCONS	Este actor del sistema está registrado, como su nombre lo indica, como empleado certificado de HIDROCONS, C. A., en donde intervienen también los directivos de dicha empresa.	Los mismos pueden inscribir al sistema compañías nuevas y nuevos clientes (representantes autorizados de las compañías), aprobar, modificar o rechazar las solicitudes de inscripción de compañías y clientes, consultar los datos de las compañías y clientes inscritos hasta el momento, visualizar las visitas que realicen todos los usuarios, además de los comentarios y sugerencias que emitan los clientes y usuarios externos, realizar, modificar y consultar presupuestos, facturas y solicitudes de repuestos, registrar nuevos productos y servicios, así como la consulta de los mismos.
 Cliente Prospecto	Constituye a todos los representantes legales de empresas que desean registrarse en el sistema, no importando si dicha empresa a la cual representan está o no registrada.	Estos actores pueden hacer lo que cualquier usuario externo, visitar las diversas páginas comunicacionales e informativas de la aplicación, sin embargo ellos pueden llenar solicitudes de registro de la empresa que representan y de ellos mismos en caso de que la dicha empresa ya esté previamente registrada.
 Cliente HIDROCONS	Este actor en cuestión representa a cada uno de los distintos clientes registrados de manera formal en el sistema, autorizados por las diversas compañías inscritas igualmente en la aplicación.	Los clientes están en la capacidad de consultar y modificar datos tanto de la empresa que representan como de su persona, consultar, aprobar o rechazar algún presupuesto en espera, consultar sus estados de cuenta en cuanto a facturación se refiere, informarse acerca de los repuestos que la empresa HIDROCONS, C. A. tiene en stock y de los servicios y actividades que puesta actualmente, y hacer solicitudes de repuestos de acuerdo a la

Tabla 32. Descripción de los Actores que interactúan con el sistema (Continuación)

Nombre del Actor	Representa	Actividades que realiza
 SGBDD HIDROCONS	Sistema Gestionador de Base de Datos	Ejecuta las transacciones relacionadas con el manejo de data (inserción, modificación, eliminación y consulta), en forma transparente, es decir, su participación está implícita en cada una de las tareas solicitadas por los otros actores del sistema, sin que estos se percaten nidencuenta de ello.

3.4.6. Interacciones de los actores con el sistema (Casos de uso)

En el diagrama de casos de uso se describen las funcionalidades y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas, especificando la relación existente entre los actores y dichos casos. El sistema en estudio posee casos de usos que son “genéricos” en cierta manera, debido a su repetición constante, y la realización de muchos de ellos se lleva a cabo de la misma forma, razones por las cuales, además de cuestiones de espacio, se decidió reducir el número de casos de uso que se analizarán, ya que sería muy repetitivo colocarlos todos. Se puede observar en la figura 3.2, la cual representa el diagrama de casos de uso general, que están presentes los diferentes actores que interactúan en el sistema, además de las relaciones existentes entre ellos, y cuáles casos de uso dicho actor puede desempeñar. Así, por ejemplo, entre el actor Empleado de HIDROCONS y Usuario existe la relación “de Herencia”, es decir, este último (señalado por la punta de la flecha que indica la generalización) le cede sus atributos al primero. Además, se puede observar que todos los actores, exceptuando el Cliente Prospecto pueden desempeñar todos los casos de uso mencionados dentro de los óvalos que se observan. Esto se describe mediante una línea sólida que une al actor en cuestión con su caso de uso atribuido. Igualmente ocurre en los diagramas de casos de uso posteriores, los cuales se derivan de este diagrama general.

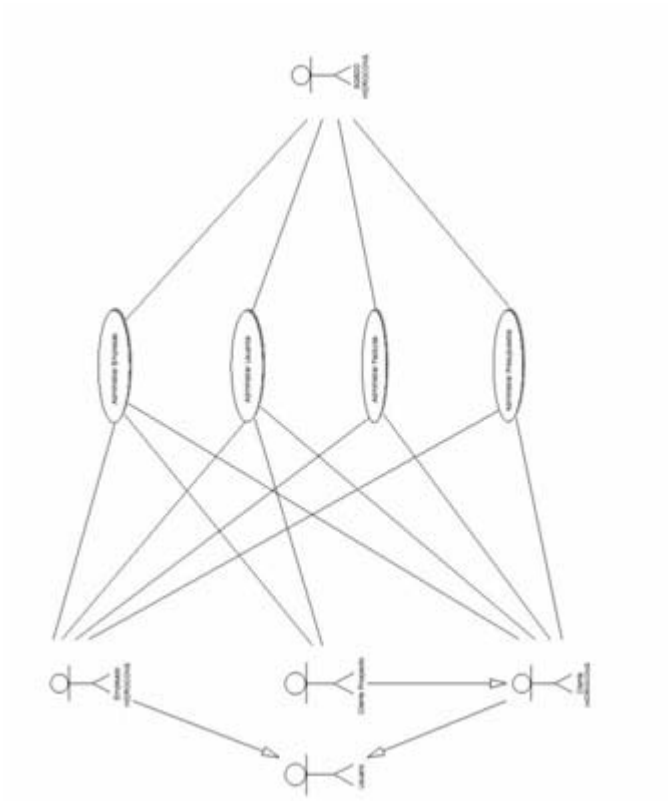


Figura 3.2. Diagrama General de Casos de Uso del Sistema
Fuente: Propia, 2009

- **Diagrama de Casos de Uso "Administrar Empresas":** en el diagrama 3.3 se puede detallar que tanto el actor Cliente HIDROCONS como el actor Empleado HIDROCONS heredan las características del actor Usuario, como lo indica las flechas que provienen de estos actores y que desembocan en este último actor. Igualmente ocurre entre el actor Cliente Prospecto, quien hereda las propiedades del actor Cliente HIDROCONS. Se puede observar que el actor SGBDD HIDROCONS tiene la facultad de realizar todos los casos de uso presentes, ya que este es el que gestiona todo lo referente al acceso a los datos del sistema, al igual que el actor Empleado. Sin embargo el actor Cliente de HIDROCONS, e inclusive el actor Cliente Prospecto no tienen tantos privilegios. Por ejemplo, el Cliente Prospecto sólo está en capacidad de "Registrar Empresa", mientras que el Cliente de HIDROCONS no puede

aprobar la solicitud de inscripción de una empresa aplicando el caso de uso "Aprobar Afiliación Empresa", acción que efectivamente sí puede hacer el Empleado.

- **Diagrama de Casos de Uso "Administrar Usuarios":** se puede observar en la figura 3.4 que, al igual que la anterior figura, ocurren las mismas relaciones existentes entre los diferentes actores presentes. De igual modo se detallan las conexiones existentes entre dichos actores y los respectivos casos de usos que puede realizar. Sin embargo, aparecen nuevos elementos o estereotipos, específicamente nuevas relaciones entre casos de uso, tal como ocurre con el caso de uso "Registrar Usuario" y "Determinar Código", cuyo estereotipo existente se denomina "includes", es decir, el primer caso de uso o caso base incluye al segundo caso. Estos casos de uso incluidos nunca trabajan solos, ya que contienen sentencias en común que se agrupan en un solo caso para luego funcionar como parte de varios casos de uso que lo incluyen, simplificando el diagrama en cuestión. Se representa mediante una flecha abierta punteada que sale desde el caso de uso base y apunta al caso de uso que desea incluir.

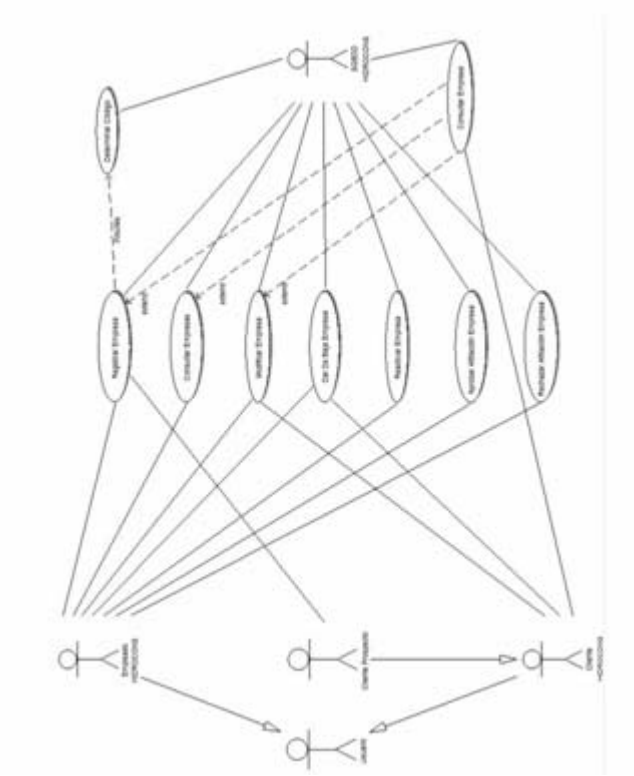


Figura 3.3. Diagrama de Casos de Uso "Administrar Empresas"
Fuente: Propia, 2009

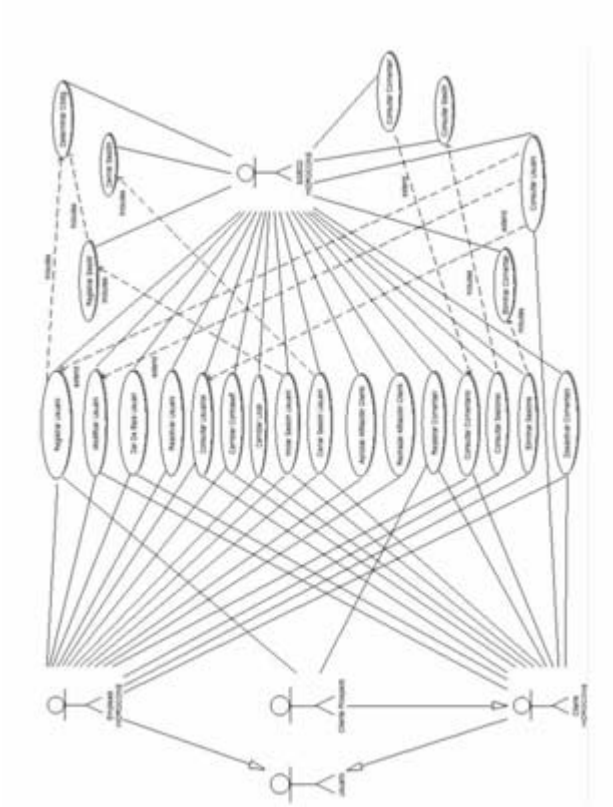


Figura 3.4. Diagrama de Casos de Uso "Administrar Usuarios"
Fuente: Propia, 2009

- **Diagrama de Casos de Uso "Administrar Presupuestos":** al igual que las figuras anteriores, se puede observar en la figura 3.5 que todos los actores se encuentran relacionados de igual manera, y que los mismos presentan ciertos casos de uso asignados que se observan mediante su respectivo enlace, a excepción del actor Cliente Prospecto, el cual no tiene nada que ver con la administración de presupuestos, ya que debido a que como no se encuentra registrado en el sistema, ni se encuentra asociado a alguna empresa registrada en la aplicación, no se le puede adjudicar presupuesto alguno. Por otro lado, aparecen nuevas relaciones, denominadas estereotipos, entre casos de uso existentes, tal como ocurre con el caso de uso "Consultar Presupuestos"

(plural), el cual está relacionado con el caso de uso “Consultar Presupuesto” (singular) mediante el estereotipo “extend”. Esta nueva relación, denotada por una flecha abierta punteada señalando al caso de uso base, indica que el comportamiento de dicho caso base (Consultar Usuarios en este caso) es extendido o ampliado por el segundo caso mencionado. Se emplea este estereotipo, a diferencia del “includes”, ya que ambos casos pueden trabajar por separado (independientemente el uno del otro) en caso de ser necesario.

- **Diagrama de Casos de Uso “Administrar Facturas”:** en la figura 3.6 sucede un comportamiento parecido a la anterior figura, en cuanto a las relaciones existentes entre los actores presentes, y entre ellos mismos y los casos de uso adjudicados. Igualmente se puede observar que el actor Cliente Prospecto no tiene asociado ningún caso de uso en este diagrama, debido a que, como ocurre con la administración de presupuestos, no se le tiene asignado ninguna factura, ya que dicho cliente prospecto aún no es considerado un cliente definitivo y estable del sistema en cuestión, ni perteneciente a empresa registrada en la aplicación. También se observa en este diagrama la existencia de relaciones tanto de tipo “extend” como de tipo “includes”, explicadas anteriormente, entre diversos casos de uso.

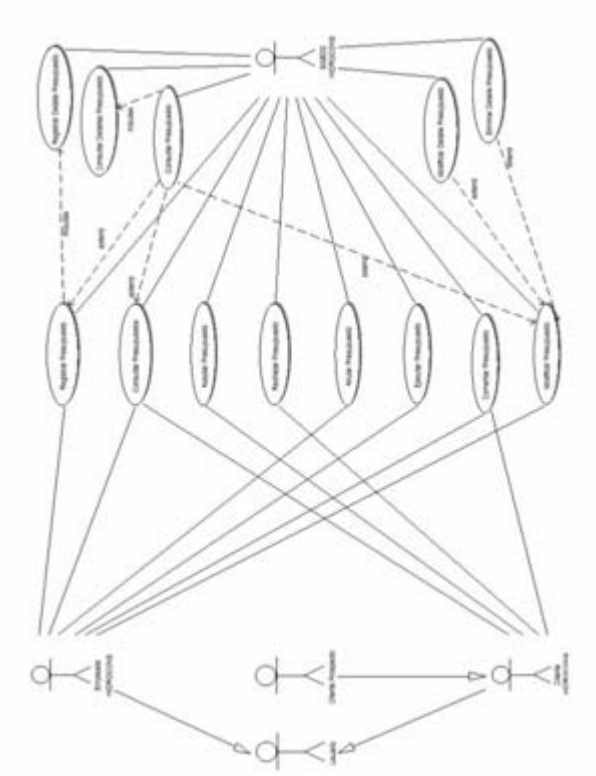


Figura 3.5. Diagrama de Casos de Uso "Administrar Presupuestos"
Fuente: Propia, 2009

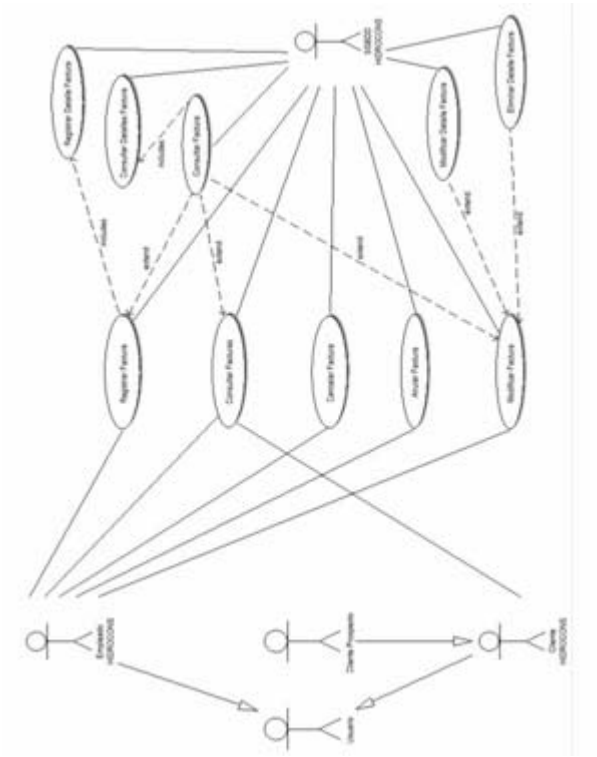


Figura 3.6. Diagrama de Casos de Uso "Administrar Facturas"
Fuente: Propia, 2009

Como complemento de los diagramas anteriores, se detallará en las siguientes tablas la realización o flujo de procesos de ciertos casos de usos, para observar los pasos que efectúan tanto el actor ejecutor de dicho caso como el sistema en cuestión, y así comprender el funcionamiento del mismo. Lo que se denomina "Curso Típico" es la secuencia de pasos que se lleva a cabo sin ocurrir excepción o inconveniente alguno, es decir, el procedimiento normal, en cambio que los "Cursos Excepcionales" se refieren al comportamiento en caso de que dichos inconvenientes aparezcan.

Tabla 33: Flujo de Procesos del Caso de Uso "RegistrarEmpresa"

Nombre Caso de Uso	REGISTRAR EMPRESA	N°. C. de U.	01
Actores:	Empleado HIDROCON y Cliente Prospecto		
Descripción:	Permite ingresar los datos de alguna empresa que desee registrarse en la aplicación.		
Casos de Uso Relacionados:	➢ DETERMINAR CODIGO ➢ CONSULTAR EMPRESA		
Entradas:	Datos de la nueva empresa (Nombre, RIF, NIT, Estado del País, Ciudad, Dirección Fiscal, Teléfonos, Correo Electrónico).		
Salidas:	Registro actualizado de la empresa en la Base de Datos del sistema.		

Curso Típico	
Acción del Actor	Respuesta del Sistema
1.- El actor introduce todos los datos necesarios propios de la empresa en el formulario apropiado que muestra el sistema.	
2.- El actor realiza la solicitud y confirmación respectivas, posicionando el botón apropiado para registrar los datos.	
	3.- El sistema verifica que la empresa no esté afiliada, tomando como referencia el RIF, o en su defecto, el NIT de dicha empresa.
	4.- El sistema muestra un mensaje de transacción exitosa.
	5.- El sistema actualiza la Base de Datos en donde se registrarán los valores ingresados.

Tabla 33: Flujo de Procesos del Caso de Uso "RegistrarEmpresa" (Continuación)

Cursos Excepcionales	
Curso Excepcional #1: La empresa ya está registrada en la BDD	
Pre-Condición:	En el paso 1 del Curso Típico el cliente ingresa RIF y/o NIT de la empresa.
Acción del Actor	Respuesta del Sistema
	1.- En la consulta de la empresa se consigue una ocurrencia de registro de la misma.
	2.- El sistema muestra un mensaje, indicando que la empresa ya se

Tabla 34: Flujo de Procesos del Caso de Uso "ConsultarEmpresa"

Nombre Caso de Uso	CONSULTAR EMPRESA	N°. C. de U.	02
Actores	Empleado HIDRICON y Cliente HIDRICON		
Descripción	Permite visualizar y verificar los datos de alguna empresa ya registrada previamente en la aplicación.		
Casos de Uso Relacionados:	➢ CONSULTAR EMPRESA ➢ REGISTRAR EMPRESA ➢ MODIFICAR EMPRESA		
Entradas:	Algunos datos de empresa considerados por el sistema como criterios de búsqueda (Código, RIF, Nombre de la Empresa a consultar, Estado del país, Fecha de Registro).		
Salidas:	Datos de la empresa en cuestión, registrados en la BDD.		

Curso Típico	
Acción del Actor	Respuesta del Sistema
1.- El actor introduce el dato o los datos de la empresa que se usarán como criterio de búsqueda.	
2.- El actor realiza la solicitud y confirmación respectivas, presionando el botón apropiado para consultar los datos.	
	3.- El sistema procede con la búsqueda, consultando a la BDD, según sean los criterios introducidos por el usuario.
	4.- El sistema muestra un formulario con los datos extraídos de la BDD referentes a la empresa consultada.

Tabla 34: Flujo de Procesos del Caso de Uso "ConsultarEmpresa" (Continuación)

Cursos Excepcionales	
Curso Excepcional #1: La empresa no se encuentra registrada en la BDD	
Pre-Condición:	En el paso 1 del Curso Típico el empleado introduce algún dato de la empresa considerado como criterio de búsqueda (Código, RIF, Nombre de la Empresa, Estado del país, Fecha de Registro).
Acción del Actor	Respuesta del Sistema
	1.- Al consultar la empresa no se consigna alguna ocurrencia de registro de la misma.
	2.- El sistema muestra un mensaje, indicando que la empresa no se encuentra registrada.
3.- El caso de uso continúa en el paso 1 del Curso Típico.	

Tabla 35: Flujo de Procesos del Caso de Uso "Modificar Empresa"

Nombre Caso de Uso	MODIFICAR EMPRESA	N°. C. de U.	03
Actores:	Empleado HIDROCON y Cliente HIDROCON		
Descripción:	Permite modificar los datos de alguna empresa ya registrada previamente en la aplicación.		
Casos de Uso Relacionados:	➢ CONSULTAR EMPRESA.		
Entradas:	Datos a modificar de la empresa (Nombre de la empresa, RIF, NIT, Estado del país, Ciudad, Dirección Fiscal, Teléfonos, Correo Electrónico).		
Salidas:	➢ Registro actualizado de la empresa en la Base de Datos del sistema, después de ser modificada.		

Curso Típico	
Acción del Actor	Respuesta del Sistema
1.- El actor introduce todos los datos a modificar necesarios de la empresa en el formulario apropiado que muestra el sistema.	
2.- El actor realiza la solicitud y confirmación respectivas, pulsando el botón apropiado para modificar los datos.	
	3.- El sistema verifica que el RIF o NIT ingresados por pantalla no estén adjudicados a otra empresa distinta a la que se está modificando.

Tabla 35: Flujo de Procesos del Caso de Uso "Modificar Empresa" (Continuación)

Curso Típico	
Acción del Actor	Respuesta del Sistema
	4.- El sistema muestra un mensaje de transacción exitosa.
	5.- El sistema actualiza la Base de Datos en donde se registraron los valores ingresados propios de la modificación realizada.

Cursos Excepcionales		
Curso Excepcional #1: La empresa se le asigna un RIF o NIT de otra empresa		
Pre-Condición:	En el paso 1 del Curso Típico el cliente ingresa RIF y/o NIT de la empresa.	
Acción del Actor		Respuesta del Sistema
		1.- El sistema determina que el RIF o NIT ingresados por pantalla pertenecen

Tabla 36: Flujo de Procesos del Caso de Uso "Dar De Baja Empresa"

Nombre Caso de Uso	DAR DE BAJA EMPRESA	N°. C. de U.	04
Actores:	Empleado HIDROCON y Cliente HIDROCON		
Descripción:	Permite dar de baja a alguna empresa ya registrada previamente en la aplicación. Cabe destacar que esta eliminación no es física sino lógica, es decir, una vez registrada la empresa como constancia que la misma estuvo presente en algún momento en la base de datos, pero con un indicador que la coloca como INACTIVA o CESANTE.		
Casos de Uso Relacionados:			
Entradas:	No requiere de dato de entrada alguno, ya que este es un proceso que consiste solo de posicionar un botón determinado designado para tal fin.		
Salidas:	Registro actualizado de la empresa en la Base de Datos, en donde la misma, así como sus clientes asociados quedan en situación de inactividad.		

Tabla 36: Flujo de Procesos del Caso de Uso "Dar De Baja Empresa" (Continuación)

Curso Típico	
Acción del Actor	Respuesta del Sistema
1.- El actor realiza la solicitud para dar de baja a la empresa en cuestión, posicionando el botón apropiado.	
	2.- El sistema muestra un mensaje de transacción exitosa.
	3.- El sistema actualiza la Base de Datos en donde se ve inhabilitada la empresa seleccionada.
Cursos Excepcionales	
No existen cursos atípicos o excepcionales para este Caso de Uso	

Tabla 37: Flujo de Procesos del Caso de Uso "Registrar Presupuesto"

Nombre Caso de Uso	REGISTRAR PRESUPUESTO	N°. C. de U.	05
Actores:	Empleado HIDROCON'S		
Descripción	Permite ingresar los datos de un presupuesto o cotización de servicio que la empresa HIDROCON'S, C. A. le ofrezca y le emita alguna empresa cliente en particular.		
Casos de Uso Relacionados:	➢ CONSULTAR PRESUPUESTO ➢ REGISTRAR DETALLE PRESUPUESTO		
Entradas:	➢ Datos del encabezado del presupuesto (Referencia, fecha, empresa a la cual se dirige, termino de obra). ➢ Lista de ítems o renglones propios del cuerpo del presupuesto en cuestión (Descripción del ítem, medida o presentación, cantidad, precio unitario).		
Salidas:	➢ Se obtiene un registro actualizado de la BDD, incrementándose un nuevo presupuesto cuya situación es considerada "EN ESPERA".		
Curso Típico			
Acción del Actor		Respuesta del Sistema	
1.- El actor introduce todos los datos del encabezado del presupuesto, es decir, los datos de identificación del mismo, en el formulario que muestra el sistema en una ventana modal por pantalla.			
2.- El actor realiza la confirmación de ingreso de datos del encabezado, posicionando el botón apropiado para registrar los datos.			

Tabla 37: Flujo de Procesos del Caso de Uso "Registrar Presupuesto" (Continuación)

Curso Típico	
Acción del Actor	**Respuesta del Sistema**
	3.- El sistema verifica que el número de referencia del presupuesto no esté registrado, es decir, que no exista un presupuesto adicional con igual código.
4.- El actor introduce todos los datos necesarios para detallar mobiliario o partida del presupuesto, es decir, un detalle de trabajo particular que en conjunto con otros ítems, forman el cuerpo del mismo.	
5.- El actor realiza la confirmación de registro del nuevo ítem, presionando el botón apropiado para registrar los datos.	
	6.- El sistema muestra un mensaje de ingreso exitoso de datos.
7.- El actor repite los pasos 4 y 5 tantas veces como ítems existan y sean necesarios ingresar.	
8.- El actor confirma finalmente el registro del presupuesto, presionando el botón apropiado para ingresar los datos.	
	9.- El sistema muestra mensaje de transacción exitosa.
	10.- El sistema actualiza la Base de Datos del sistema en donde se registran los valores ingresados.
Cursos Excepcionales	
Curso Excepcional #1: El presupuesto se encuentra registrado en la BDD	
Pre-Condición:	En el paso 1 del Curso Típico el empleado introduce el número de referencia del presupuesto.
Acción del Actor	**Respuesta del Sistema**
	1.- El sistema consigna una ocurrencia de presupuesto con igual número de referencia que el ingresado por pantalla.
	2.- El sistema muestra un mensaje, indicando que dicho presupuesto ya se encuentra registrado.
3.- El caso de uso continua en el paso 1 del Curso Típico.	

Tabla 3.8: Flujo de Procesos del Caso de Uso "Aceptar Presupuesto"

Nombre Caso de Uso	**ACEPTAR PRESUPUESTO**	N°. C. de U.	**06**
Actores	Cliente HIDROCONS		
Descripción	Permite aprobar un presupuesto específico por parte del cliente de alguna empresa a la cual va dirigido dicho documento, lo que daría a entender que se está de acuerdo para comenzar el trabajo detallado y especificado por el presupuesto en cuestión.		
Casos de Uso Relacionados:			
Entradas:	No requiere de dato de entrada alguno, ya que es un proceso que consiste sólo de presionar un botón determinado.		
Salidas:	Situación modificada de dicho presupuesto en la BDD.		
Curso Típico			
Acción del Actor	Respuesta del Sistema		
1.- El actor realiza la solicitud y confirmación respectivas para aceptar y aprobar el presupuesto que se observa en pantalla, presionando el botón apropiado.			
	2.- El sistema muestra un mensaje de transacción exitosa.		
	3.- El sistema actualiza la Base de Datos donde se encuentra el presupuesto en cuestión, pero ya aprobado.		
Cursos Excepcionales			
No existen cursos atípicos o excepcionales para este Caso de Uso			

3.4.7. Modelo del dominio

Un modelo de dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés. También se le denomina modelo conceptual, modelo de objetos del dominio, modelo de objetos de análisis o modelo de clases. El proceso unificado de desarrollo de software RUP define un modelo de

dominio como uno de los artefactos que podrían crearse en la disciplina del modelado del negocio. El diagrama de clase del dominio se elabora para establecer las relaciones entre las clases del dominio, bien sea de dependencia, herencia, asociación, agregación y composiciones que existen entre ellas. Este diagrama permite visualizar el comportamiento general del sistema en función de los objetos del dominio. [15] Una clase es el marco de trabajo para un conjunto de objetos. En un diagrama de clases, según UML, el nombre de cada una de estas clases debe constar de una sola palabra simple, que comience con letra mayúscula, y si el nombre está compuesto por dos o más palabras, las mismas deben estar unidas comenzando cada una igualmente en mayúscula. Lo importante es que los nombres estén contenidos dentro de una sola palabra, sin espacios vacíos que los separen.

Utilizando la notación de UML, un modelo del dominio se representa mediante un diagrama determinado en la que se pueden observar:

- Objetos del dominio o clases conceptuales.
- Relaciones entre estas clases conceptúales.
- Cardinalidad o multiplicidad entre dichas clases conceptuales.

El modelo de dominio es un tipo de diagrama estático, lo más representativo posible al sistema a proponer, que describe la estructura de un sistema mostrando sus clases, las relaciones y la cardinalidad entre ellas. Aquí están presentes todas las entidades que intervienen en el sistema tales como Usuario, EmpleadoHIDROCONS, ClienteHIDROCONS, Cliente Prospecto, Empresa, Factura, Detalle_Factura, Presupuesto, Detalle_Presupuesto, Sesión, Cartero, entre otras, enmarcadas dentro de un paquete o contenedor especial, el cual fue denominado “modelo”. Una clase determinada puede encontrarse enlazada a otra distinta mediante relaciones. Ello

ocurre entre la clase Usuario y la clase Empleado por ejemplo, donde existe una relación denominada “de Herencia”, debido a que la clase principal o superclase, la cual es la señalada por la punta de la flecha (Usuario), le transfiere sus propiedades a la subclase, que en este caso viene dada por la clase Empleado, indicada por el otro extremo de la flecha. Esta relación es llamada también “es un tipo de”, por lo que se lee de manera que “un Empleado es un tipo de Usuario”. Mientras tanto, entre la clase Empresa y las clases Factura o Presupuesto existe una relación conocida como relación “de Dependencia”, denotada mediante una línea o flecha abierta y punteada. Dicha relación es sinónimo de que una clase determinada “utiliza” a otra clase específica para que se realice o lleve a cabo una tarea en particular, en otras palabras, que un elemento usa otro para que se realice una proceso específico, por lo que un cambio en el elemento que se está usando, en este caso la Empresa (clase que es apuntada por la flecha) afecta al elemento que lo usa (Factura o Presupuesto). Lo inverso no necesariamente es cierto. De allí que un cambio en la interfaz de Empresa (por ejemplo, cambio del nombre de la empresa) afectará a las otras dos clases, ya que para que se genere una factura o presupuesto por ejemplo, estas requieren hacer uso de este dato en cuestión.

La relación “de Asociación” es una relación que conecta dos clases, denotando la conexión estructural entre ellas. La notación UML para una asociación es una línea sólida que conecta las dos clases que participan en dicha conexión jugando un rol específico. Dicha relación se pueden observar, por ejemplo, tanto entre las clases Empresa y Usuario, así como entre las clases Usuario y la clase Cartero. En ambas relaciones la “asociación” viene expresada mediante una línea sólida, acompañada de un nombre que indica la relación y de las cantidades de objetos que pueden ser conectados entre ellos, hecho conocido como *cardinalidad* o *multiplicidad*. Así por ejemplo, las asociaciones descritas se leerían “una Empresa tiene inscrito ningún o muchos Usuarios” y “un Usuario recibe ninguno o muchos Carteros”. De igual modo, la relación “de Agregación”, también presente en este tipo de diagramas UML,

es una extensión de la relación de asociación, es decir, que se encuentra presente exactamente donde se encuentre cualquier relación de asociación, trabajando en conjunto con esta misma. Esta nueva relación se conoce como una relación "todo-parte", lo que quiere decir que una clase en específico "es parte de" otra clase, o inclusive en sentido contrario, también es sinónimo de relación "tiene una", de manera que una clase "tiene una" clase diferente, o sea, la clase en un extremo de la relación (el todo) contendrá las clases presentes en el otro extremo (las partes). De este modo las relaciones explicadas en el párrafo anterior, las cuales presentan asociaciones específicas se pueden leer, en función de la agregación, del siguiente modo: "ninguno o muchos Usuarios son parte de una Empresa" o en su defecto "una Empresa tiene ningún o varios Usuarios", y "un Usuario tiene ningún o muchos Carteros" o "ningún o muchos Carteros son parte de un Usuario".

Existen dos formas de agregación, la simple y la compuesta. En la agregación simple el tiempo de vida del todo y el de las partes no están enlazados, es decir, si el todo deja de existir las partes siguen existiendo. Esto es lo que ocurre entre las clases Usuario y Cartero, ya que al desaparecer del sistema un usuario, el Cartero puede seguir enviando posteriormente mensajes al mismo usuario eliminado. Esta relación de agregación se muestra usando un diamante sin rellenar ubicado cerca de la clase que forma el todo en la relación, lo que en este caso sería cerca de la clase Usuario. Sin embargo, la relación de agregación compuesta es todo lo contrario, el tiempo de vida del todo y de las partes sí se encuentran enlazados, lo que significa que si el todo deja de existir, simultáneamente ocurre lo mismo con sus partes. Ello se puede observar en la relación existente entre la Empresa y el Usuario, debido a que si desaparece la Empresa, ya no tiene sentido que el Usuario o Cliente de dicha empresa siga existiendo en el sistema, por lo que igualmente desaparecería. Esta relación se muestra como un diamante negro o relleno que se coloca, al igual que la agregación simple, cerca de la clase que representa el todo en la relación, es decir, cerca de la clase Empresa. De esta forma se quiere dar a entender que todas las clases que

intervienen en el sistema se asocian a otras clases mediante relaciones claras y definidas, como se puede observar en la figura 3.7, la cual muestra un diagrama de clases o modelo de dominio, donde se destacan todas las entidades o clases, junto a sus respectivas asociaciones o relaciones:

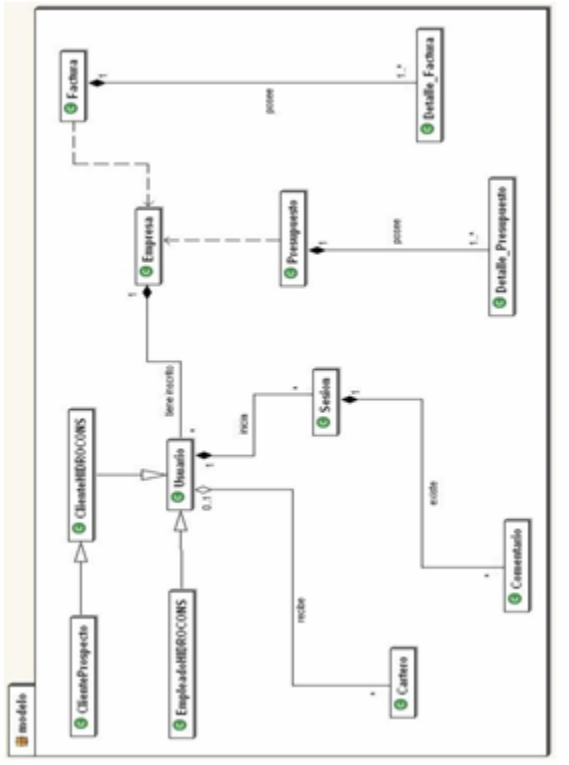


Figura 3.7. Modelo de Dominio o Diagrama de Clases del sistema
Fuente: Propia, 2009

El diagrama de clases del dominio descrito anteriormente establece las relaciones entre las clases del dominio que intervienen en el sistema propuesto. En la siguiente tabla, la 3.9, se describen cada una de las clases presente en dicho diagrama.

Tabla 3.9: Descripción de Clases del Modelo de Dominio

Clases del Dominio	Descripción
Empresa	Representa a cualquier compañía o empresa (en su mayoría Estaciones de Servicio dispensadoras de combustible) a la cual HIDROCONS, C. A. está en capacidad de ofrecer y prestar sus servicios.
Usuario	Esta clase es la superclase de las clases que representan los actores que normalmente harán uso del sistema en cuestión (Empleado, Cliente y Cliente Prospecto).
Empleado HIDROCONS	Esta clase representa a los empleados y directivos pertenecientes a la empresa HIDROCONS, C. A., es decir, todo aquel que trabaja en dicha compañía.
Cliente HIDROCONS	Corresponde al otro grupo de usuarios del sistema que trabajan o son representantes legales de las empresas inscritas a las cuales se le venden repuestos o se les hace u ofrece algún trabajo.
Cliente Prospecto	Constituye un representante autorizado de alguna empresa y potencial cliente a ser registrado en el sistema.
Presupuesto	Esta clase encierra los datos del encabezado de los documentos denominados “Presupuestos”, como su número de referencia, la fecha de emisión, el código del cliente al cual va dirigido y la descripción de la obra a ejecutar.
Detalle_Presupuesto	Esta clase posee los datos de cada uno de los ítems o partidas propios de un presupuesto, en donde se describe la información del trabajo a efectuarse y se

	cotizan sus cantidades y precios.
Factura	Se refiere a los datos del encabezado del documento denominado "Factura", como el lugar y fecha de emisión de dicho documento, código del cliente al cual va dirigido, el impuesto asociado y la forma de pago del mismo.
Detalle_Factura	Esta clase encierra los datos del cada uno de los ítems o partidas pertenecientes a una factura, en donde se describe la información del trabajo efectuado y sus cantidades y precios.

Tabla 3.9: Descripción de Clases del Modelo de Dominio (Continuación)

Clases del Dominio	Descripción
Cartero	Esta clase se encarga de enviar correo electrónico automáticamente a los usuarios clientes registrados en el sistema, en los momentos que se considere necesario.
Sesión	Dicha clase registra la información de la fecha de ingreso, la hora de entrada y salida de algún usuario registrado al sistema, así como de las acciones realizadas durante su estadía para efectos de auditoría.
Comentario	Dicha clase tiene que ver con todos los comentarios y mensajes que algún usuario, esté registrado como Cliente de HIDROCONS o no, envían al sistema para que sea leído por cualquier Empleado de esta empresa.
Objeto Común	Esta es una clase auxiliar que contiene funciones útiles comunes para varios casos de uso. No es necesario mostrarla en el diagrama, ni instanciar esta clase para hacer uso de sus funciones ya que las mismas son de tipo "estático", es decir, para usarlas sólo se menciona esta clase en el caso de uso específico y se escoge la función a utilizar.
SGBDD (Sistema Gestionador de	Representa la base de datos en la cual se registran, eliminan, modifican y consultan todos los registros de las operaciones inherentes en el sistema.

Base de Datos)	

3.4.8. Arquitectura candidata del sistema

Para la fase inicial, el flujo de trabajo de diseño es muy breve porque aún no se poseen todos los detalles del complejo sistema que se construirá. Lo que se obtiene de este flujo de trabajo es un bosquejo del modelo de diseño y una arquitectura inicial estable.

3.4.8.1. Arquitectura de software

La computación distribuida puede definirse como una red de computadores que son alojamientos (hosts) para componentes que se comunican y coordinan unos a otros por medio de pase de mensajes. Dichos componentes pueden ser componentes de hardware o software (programas). Se refiere a un ambiente en el cual los programas, los datos que ellos usan y los cómputos actuales se extienden a través de una red. En la computación distribuida los programas realizan llamadas a otros espacios de direcciones generalmente presentes en otras máquinas, mientras que en la computación local dichos componentes comparten un espacio de direcciones comunes para comunicarse; la velocidad de comunicación local es mucho mayor comparada con la distribuida; y cuando ocurre una falla parcial en una porción de la computación distribuida, las otras partes pueden no enterarse y continuar la ejecución como si nada hubiese ocurrido, a diferencia de la computación local, en la que por causa de dicha falla todos los componentes se pueden colgar, o en su defecto actúa un detector encargado de transmitir la información de la falla a aquellos componentes que no estén afectados por la misma. [8] Para el escenario planteado en este sistema, se puede proponer una arquitectura de computación distribuida que divida el trabajo del sistema en cuatro capas, como se puede observar en la figura 3.8:

- **Capa de interfaz de usuario**: en esta capa estarán todos los componentes relacionados con el diseño, captura de datos, validación de entradas e inicio y cierre de sesión, es decir, con todo lo que tenga que ver con las acciones llevadas a cabo por los usuarios a emplear el sistema.

- **Capa de controlador de interfaz**: en esta capa se encuentran todos los componentes que toman decisiones sobre el flujo de eventos, dependiendo de las solicitudes de los usuarios del sistema (llámese solicitud a cualquier tarea que se le pide al sistema ejecutar de manera automática o no, desde consultas hasta actualizaciones). Aquí están los componentes que determinan cuáles otros componentes atenderán una solicitud específica.

- **Capa de lógica del negocio**: se encuentran inmersas todas las entidades (clases) que fueron definidas para implementar las funcionalidades y comportamientos (casos de uso) del sistema, según los requerimientos de la aplicación, el alcance del sistema y las políticas de la empresa.

- **Capa de acceso a datos**: aquí se encuentran los componentes dedicados a establecer la conexión con las bases de datos asociadas al sistema. Estos componentes son responsables de establecer los privilegios de los diferentes tipos de usuarios que intentan acceder a la información presente en las Bases de Datos respectivas.

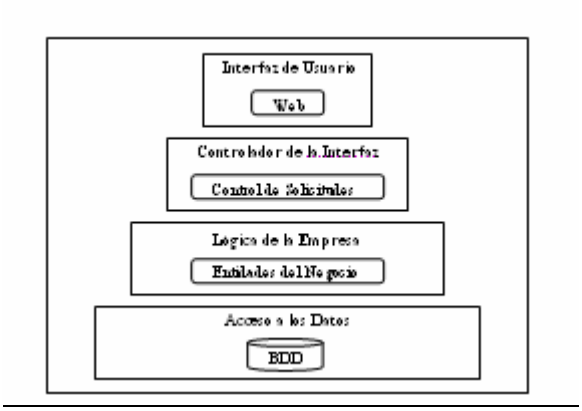


Figura 3.8. Modelo de Arquitectura de software Candidata
Fuente: Guía de Estudiante del Curso de Java Empresarial Libro Nº 1 de IBM. Código del Curso: CY760. Versión 4.0., 2006

- **Tecnologías de Computación Distribuida [8]:** en el mercado hay diversas tecnologías para ayudar a los programadores a construir sistemas distribuidos, todas ellas propias de diversas plataformas que pueden o no ser compatibles, entre las cuales se nombran las siguientes que son las más reconocidas:
  - RPC.
  - Java RMI (Java Remote Method Invocation).
  - DCOM (Distributed Component Object Model).
  - CORBA (Common Object Request Broker Architecture).

Todas estas tecnologías difieren unas de otras de alguna forma, y cualquiera de ellas puede usarse para construir sistemas distribuidos eficientes y robustos tomando en consideración factores como el lenguaje de programación soportado por dicha tecnología y el sistema operativo en el que puedan ser usadas. Para este proyecto en desarrollo se empleará la tecnología Java RMI, la cual es una poderosa tecnología de redes de Java proporcionada por Sun Microsystems, cuyo principal objetivo es

asociar la robustez de la programación Java con las ventajas de la computación distribuida. Es una arquitectura que se caracteriza por que un objeto en un proceso que reside en una computadora determinada puede invocar al método de un objeto en otro proceso que en la mayoría de los casos reside en una Máquina Virtual Java (JVM – Java Virtual Machine) diferente, tanto en la misma máquina física como en una máquina remota. A los objetos cuyos métodos pueden ser invocados por objetos de otros procesos, se denominan objetos remotos. En Java RMI, un objeto remoto Java existe en un servidor, y los métodos de este objeto remoto pueden ser invocados por medio de la invocación remota, por cualquier otro objeto Java en un computador cliente. Los procesos que alojan objetos remotos son referidos como procesos servidores y los procesos que alojan objetos que invocan métodos de objetos remotos se llaman procesos clientes.

3.4.8.2. Arquitectura de red y hardware

Inicialmente, y tomando en cuenta las posibilidades y necesidades de la empresa, se toma en cuenta sólo un componente esencial: el servidor Web. Preferiblemente se considera este sólo componente debido a que se optó por la configuración de arquitectura de un solo servidor (ya que no se trabaja con redes especiales ni Intranets) donde se encuentra inmerso el conjunto de componentes básicos de una aplicación Web, tales como el motor de ejecución de Scripts y el servidor de base de datos, todos instalados en una misma máquina. Los clientes y empleados de HIDROCONS, C. A. acceden al sistema vía Internet, donde pueden hacer sus peticiones HTTP al servidor Web. El servidor puede ser una PC, una estación de trabajo, o hasta un Mainframe, con un progreso en el rendimiento, pero también de costo. En general, el rendimiento de una arquitectura de un solo servidor esta limitada por el hecho de que el motor del Script y el sistema de manejo de base de datos son ambos aplicaciones de un alto uso de la memoria y del procesador, y así, crean conflicto en el uso de los recursos de la máquina, lo cual puede producir un

cuello de botella. No se requiere un servidor que esté concebido para paralelismos masivos, ya que la mayoría de las solicitudes son sólo de consulta. La disponibilidad no está garantizada en caso de fallo, ya que en la configuración más simple, con un solo procesador y un proceso por cada componente de software, cada elemento de software y hardware es un único punto de ruptura: si falla, el sistema completo se cuelga. Esta es una desventaja de este tipo de arquitectura, como se explicó antes.

La tolerancia a fallos puede ser mejorada al precio de incrementar el costo y la complejidad, agregando recursos de hardware redundantes, por ejemplo múltiples procesadores y varios discos y mejoras en paralelo. Sin embargo, por ahora, la empresa HIDROCONS, C. A. no esta interesada en este tipo de mejoras. En cuanto a seguridad, este es el aspecto más débil de la arquitectura de un solo servidor, porque los atacantes que traspasen el Firewall o inclusive, el mismo servidor Web pueden tomar control del host y ganar acceso directo a la base de datos, violando la protección de los mismos. Con respecto al costo, la arquitectura de un solo servidor es de bajo precio, mientras un paralelismo masivo no sea requerido. Agregar más procesadores, memoria RAM, y discos más rápidos al servidor, o cambiar a una clase de hardware más sofisticada, incrementa el costo, y a veces puede superar el gasto en comparación a una configuración alternativa.

La solución de un solo servidor, implementada en un PC de bajo costo o estación de trabajo, es viable a pequeña escala, para aplicaciones Web no críticas, como es el caso de la aplicación a desarrollar, donde la simplicidad y la inversión limitada en el área de tecnologías de la información son las metas prominentes, y la seguridad de datos no es tan esencial. Muchos proveedores de alojamiento Web ofrecen configuraciones similares, alquilando a sus clientes una sola máquina o incluso una porción de una máquina que aloja muchas aplicaciones de diferentes clientes. Sin embargo, si es exigido un alto rendimiento, alta disponibilidad y acceso seguro del Internet, se necesita una arquitectura más articulada.

3.5. Evaluación de la fase de inicio

Al final de esta fase se encuentra el primer y principal hito, meta u objetivo del proyecto, llamado “Hito de Objetivos del Sistema”. En este punto se examinan, como su nombre lo indica, los objetivos del ciclo de vida del proyecto, y de esa manera se evalúa básicamente la viabilidad del proyecto. Fue un período breve en cuanto a tiempo de realización, donde se hizo una sola iteración para lograr los objetivos de la misma. En esta iteración se trabajaron y definieron los flujos de requisitos, análisis de viabilidad y muy brevemente, el diseño. Al revisar este hito aplicado a este proyecto en desarrollo se observaron ciertos criterios de evaluación, arrojándose los siguientes resultados:

- Efectivamente hubo consenso entre los directivos de la empresa HIDROCONS, C. A. y el desarrollador de software en la definición del alcance, estimación inicial de costo y cronograma del proyecto.

- Dicho acuerdo permitió la captura exitosa de un conjunto significativo y correcto de requerimientos necesarios para el desarrollo del futuro sistema de información de la empresa, mediante el entendimiento común y exhaustivo de estos, además de los requisitos que permitieron obtener una arquitectura candidata del mismo.

- Se identificaron la mayoría de los riesgos iniciales, consiguiéndose estrategias de mitigación de riesgos para cada uno.

- Se logró analizar el contexto del sistema, lo cual permitió obtener un diseño potencial del modelo de dominio, que constituye un punto clave para iniciar la captura de requisitos y procedimientos de las actividades administrativas de la

empresa HIDROCONS, C. A., y de esta forma llevarlas al plano computacional, a nivel de sistema de información. Se definieron las clases conceptuales que representan el negocio, y se describieron para comprender este modelo conceptual.

- Se definieron los actores principales del sistema, se relacionaron con los respectivos casos de uso y la función que cumple cada uno en el sistema.

- Posteriormente se obtuvo un esbozo del sistema usando las clases hasta ahora definidas y tablas de realizaciones de casos de uso, que guiaron al siguiente paso: definir la arquitectura del sistema.

CAPÍTULO IV

FASE DE ELABORACIÓN

4.1. Introducción

La elaboración es la serie inicial de iteraciones durante las cuales el equipo de desarrollo de software lleva a cabo un análisis mas profundo de RUP. Se define el núcleo central de la arquitectura, se aclara la mayoría de los requisitos, y se abordan los problemas de alto riesgo. Por tanto, el trabajo inicial podría incluir la implementación de los escenarios que se consideran importantes, pero que no son especialmente arriesgados desde el punto de vista técnico. [15] En la fase de inicio se obtienen muchos detalles esenciales como son la visión del sistema, la arquitectura candidata, los usuarios típicos, etc. Pero estos sólo son bocetos, los cuales serán refinados y algunos de ellos validados en esta nueva fase. La meta de la fase de elaboración es definir y establecer la base de la arquitectura del sistema, brindando así un soporte estable para la mayor parte del esfuerzo de diseño e implementación en la fase de construcción. Esta meta general se traduce en cuatro objetivos importantes, donde cada uno trata un área imprescindible de riesgo. Se manejan riesgos asociados con los requerimientos y con la arquitectura. También se manejan los riesgos asociados con los costos, cronogramas y finalmente se necesita manejar los problemas relacionados al proceso y al ambiente de desarrollo. Manejar estos riesgos asegura un mínimo de problemas cuando se pase a la fase de construcción. [12]

Los diagramas de secuencia UML y el diseño de la base de datos son puntos que se toman en cuenta en esta fase del desarrollo del proyecto. Los primeros explican como es la interacción entre un conjunto de objetos específicos para llevar a cabo cierto caso de uso, mientras que el segundo modelo muestra la distribución de

los datos de manera que la redundancia de los mismos sea mínima. Considerando que uno de los requisitos para este sistema es que sea una aplicación Web, se seleccionó WebML, una herramienta novedosa que ofrece modelos para representar sistemas basados en Web, y que integrando dicha herramienta con el proceso unificado RUP, se obtendrá un buen soporte en la modelación e implementación del área Web del proyecto. Estos modelos serán de gran ayuda al momento de describir de manera sencilla el funcionamiento de las páginas Web del sistema, y los mismos colaboran significativamente en el proceso de codificación de las páginas e interfaces que se realizarán de forma definitiva en la fase siguiente a la presente.

4.2. Planificación de la fase de elaboración

Hasta ahora se ha obtenido un esbozo claro de la arquitectura que se desea implementar. Se estima culminar esta fase con una o dos iteraciones. El sistema en desarrollo es extenso y complejo, principalmente por el amplio conjunto de operaciones que se manejan, la cantidad masiva de datos que se procesan y el conjunto de interfaces a diseñar. Se continuarán las actividades de recopilación de los requisitos que queden pendientes mediante el modelo o diagrama de casos de uso, aunque en la fase de inicio se obtuvo un modelo parcialmente completo de este. WebML proporciona los “SiteViews” que permiten representar la organización lógica y física de las páginas de la aplicación. La siguiente actividad del flujo de trabajo de análisis será describir la realización de casos de uso nuevos y realizar el análisis de las interfaces. Es importante también realizar el diseño lógico de las páginas mediante el modelo de hipertexto de WebML. En esta fase se realizará el diseño de la base de datos del sistema mediante el modelo relacional. Para el final de esta fase se ha de obtener la arquitectura que sustentará al sistema y un diseño conceptual de las páginas bastante sólido para proceder a la codificación de las mismas. Los cuatro objetivos de esta fase, de los cuales se hicieron mención previamente son:

- **Obtener un entendimiento más detallado de los requerimientos:** durante la fase de elaboración es necesario tener un buen entendimiento de la gran mayoría de los requerimientos, ya que los mismos fueron descritos breve y superficialmente durante la fase de inicio. Esto permite crear un plan más detallado, así como también ganar un entendimiento amplio de los requerimientos más críticos y relevantes, para validar y garantizar que la arquitectura propuesta y seleccionada cubra con todas las bases, algo que únicamente puede alcanzarse construyendo una implementación parcial de estos requerimientos en cuestión. [9]

- **Diseñar, implementar y validar la arquitectura base:** la funcionalidad del sistema no estará completa, pero la mayoría de las interfaces entre los bloques de construcción son implementadas durante la fase de elaboración, para poder compilar y probar la arquitectura. A esto se le denomina "arquitectura ejecutable" hasta el punto que se puede conducir alguna carga inicial de datos y ejecutar ciertas pruebas sobre la arquitectura. [9]

- **Mitigar los riesgos significativos, producir un cronograma más exacto y estimar el costo:** durante esta fase se manejan los riesgos significativos y primordiales. La mayoría serán manejados como el resultado detallado de los requerimientos, el diseño, la implementación y prueba de la arquitectura. Al final de esta fase, se tendrá información más exacta que permitirá actualizar el cronograma, el plan de proyecto y la estimación de costos. [9]

4.3. Requisitos

4.3.1. Modelos de casos de uso

Entre los materiales o requisitos generales y básicos que se emplean en esta nueva fase de RUP para proseguir con este estudio se encuentran todos o la gran mayoría de los diagramas de casos de uso detallados en la fase previa a la presente, acompañado obviamente de las descripciones y realizaciones de dichos casos. Esto debido a que dichos casos de uso van a ser empleados para desarrollar nuevos artefactos y diagramas referentes a otras áreas del desarrollo del sistema, tales como los Diagramas de Secuencia, o los Modelos de Navegación y de Hipertexto de WebML, además de ciertas pantallas o interfaces que permiten la realización de los mismos, e inclusive para etapas y fases posteriores a esta. Para ello es necesario obtener una nueva y última versión mejorada del modelo o diagrama de casos de uso que hasta ahora se posee, de manera que si aparecen nuevos casos de uso en esta etapa, los mismos se han de anexar a los casos de uso conseguidos hasta ahora en la fase de inicio de RUP.

Se debe recordar que existe un esquema gráfico del modelo de casos de uso que es general y único, del cual se derivan, por limitaciones de tamaño, los demás casos de uso de una manera más detallada y específica. Para esta sección de requisitos de la actual fase de desarrollo de RUP se han de considerar los casos de uso conseguidos previamente, pero obteniéndose un esquema general modificado, el cual se muestra en la figura 4.1, en una versión mejorada, añadiéndole además los nuevos casos de uso conseguidos en esta fase. Durante el nuevo estudio de requerimientos y de necesidades del sistema propios de esta fase de elaboración se determinaron dos (2) nuevos casos de uso generales: Administración de Repuestos (Véase Figura 4.2) y Administración de Servicios (Véase Figura 4.3), sin olvidar todos los nuevos casos de uso particulares y propios que se derivan de estos dos nuevos amplios casos.

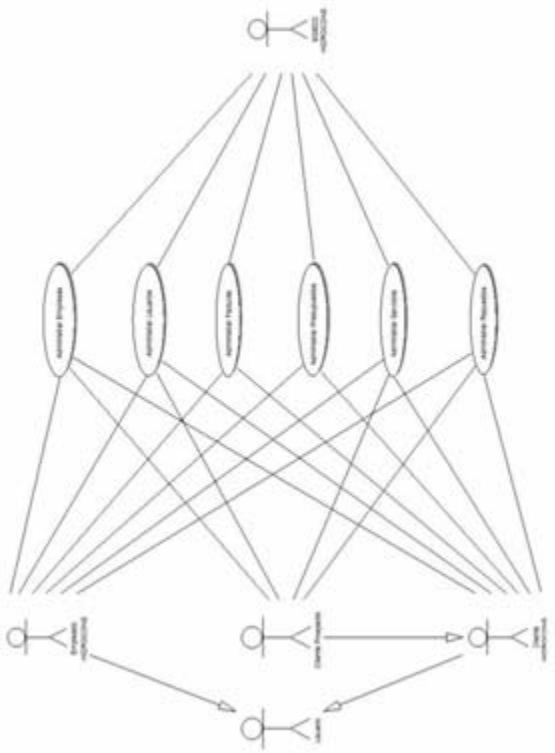


Figura 4.1. Diagrama General de Casos de Uso Mejorado del Sistema
Fuente: Propia, 2009

- **Diagrama de Casos de Uso "Administrar Repuestos":** en la figura 4.2, al igual que las diagramas de casos de uso obtenidos en la fase anterior, ocurren las mismas relaciones de generalización o herencia existentes entre los diferentes actores presentes, mientras se observa además que el actor SGBDD HIDROCONS realiza todos los casos, el Empleado HIDROCONS realiza la mayoría de ellos, el Cliente HIDROCONS, un poco menos que los del Empleado HIDROCONS y el Cliente Prospecto, solamente un caso de uso. También se observa en este diagrama la existencia de relaciones tanto de tipo "extend" como de tipo "includes" entre diversos casos de uso. El "includes" se destaca entre los casos de uso "Registrar Solicitud Repuestos" y "Registrar Repuesto Solicitado", y se usa este estereotipo ya que para registrar una Solicitud de Repuestos es imperativo registrar obviamente uno o más

repuestos solicitados, sin embargo el caso inverso no es factible, y mucho menos un caso aislado entre ambos casos de uso es imposible. En otras palabras, el caso de uso “Registrar Repuesto Solicitado” no trabaja por sí solo.

- **Diagrama de Casos de Uso “Administrar Servicios”**: este diagrama observado en la figura 4.3, es un poco más sencillo que el anterior, sin embargo las relaciones de herencia entre los diversos actores se conservan entre sí. El actor SGBDD HIDROCONS es el que posee el monopolio del control de todos los casos de uso, mientras que el Cliente de HIDROCONS y el Cliente Prospecto solo tienen facultad de llevar a cabo una consulta como caso de uso asociado, y el Empleado HIDROCONS, un número considerable de ellos. Al igual que en todos los diagramas de casos de uso particulares descritos hasta ahora, se consiguen relaciones “includes” y “extend” entre algunos casos de uso clave. Por ejemplo, la relación “extend” se puede notar entre los casos de uso “Consultar Servicios” (Plural) y “Consultar Servicio” (Singular). Se hace la aclaratoria que entre dichos casos impera este estereotipo ya que ambos casos pueden trabajar y funcionar por separado.

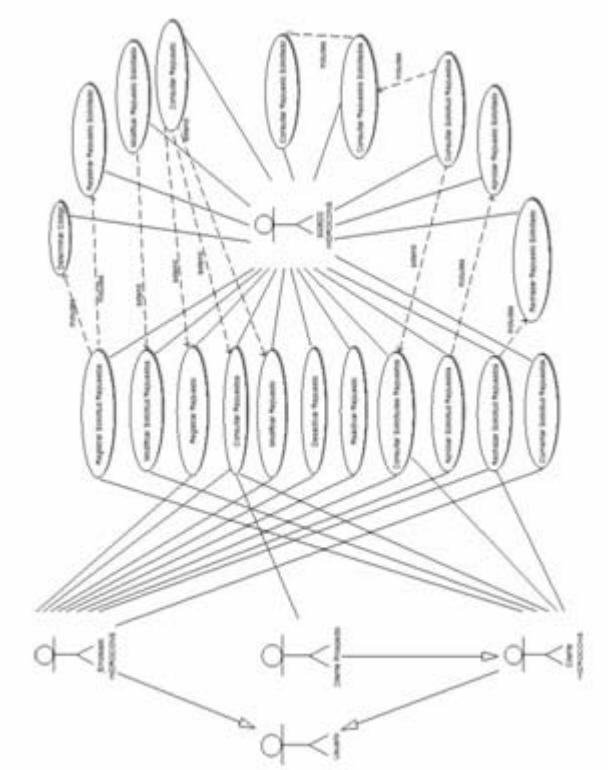


Figura 4.2. Diagrama de Casos de Uso "Administrar Repuestos"
Fuente: Propia, 2009

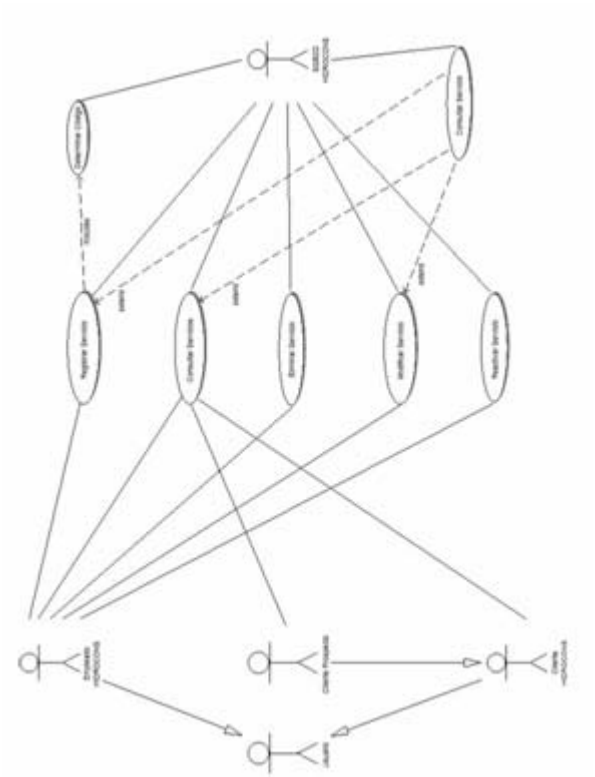


Figura 4.3. Diagrama de Casos de Uso "Administrar Servicios"
Fuente: Propia; 2009

4.3.2. Actores, requisitos y arquitectura del sistema

Con respecto a los actores, se mantienen los mismos encontrados en la fase de inicio, al igual que los mismos requisitos funcionales y adicionales determinados. Y en cuanto a la arquitectura, tanto de software como de hardware, se mantiene la que se escogió previamente, pero se detallarán ciertas especificaciones adicionales de la misma con la finalidad de mejorar la ya expuesta, y explicarla de mejor manera, para así dar a entender definitivamente en qué consistirá dicha arquitectura en cuestión.

4.3.3. Modelo de dominio o diagrama de clases

Otro de los artefactos indispensables para la continuación de este proyecto en la fase actual es el modelo o diagrama de dominio, obtenido en la etapa previa o fase de

inicio. Al igual que en los modelos de casos de uso obtenidos, también este diagrama presenta nuevos elementos, constituidos por las clases que tienen que ver con los repuestos y los servicios que la empresa ofrece a su clientela, las cuales se describen en la tabla 4.1 que se muestra a continuación:

Tabla 4.1: Descripción de Nuevas Clases del Modelo de Dominio Mejorado

Clases del Dominio	Descripción
Servicio	Representa a todos los servicios y actividades clasificadas, que puede ser inclusive una obra ejecutada y facturada, que HIDROCONS, C. A. pone a la disposición de sus clientes.
Repuesto	Clase correspondiente a los repuestos o equipos que pone en venta HIDROCONS o que incluye como complemento de la mano de obra en sus presupuestos o facturas.
Solicitud_Repuestos	Representa las solicitudes de repuestos, conocidas también como cotizaciones, que los clientes registrados hacen a la empresa HIDROCONS, C. A.
Repuesto_Solicitado	Constituye cada uno de los repuestos que solicitó el cliente en una cotización o solicitud de repuestos determinada.

En comparación con el modelo de dominio previo, aparecen nuevas clases, acompañadas de sus respectivas relaciones, acompañados por cuestión de espacio, de sus atributos esenciales, es decir, las propiedades principales (códigos y claves de conexión), como por ejemplo Código de Usuario, Código de Empresa, Código de Presupuesto, etc., además del nombre del correspondiente constructor de cada clase, con lo cual se considera completo el modelo de dominio definitivo. Las nuevas relaciones “de Asociación”, que no estaban en el diagrama de clases de la fase de inicio, se pueden observar, por ejemplo entre la clase Repuesto_Solicitado y la clase base Solicitud_Repuestos, así como entre las clases Detalle_Presupuesto o

Detalle_Factura y la clase base Repuesto. De este modo, dichas asociaciones se leerían “una Solicitud_Repuestos posee uno o muchos Repuesto_Solicitado”, mientras que si se considera la otra relación antes mencionada, se obtendría que “un Repuesto está en ningún o en muchos Detalle_Presupuesto o Detalle_Factura”.

Toda relación “de Asociación” tiene incluida una relación “de Agregación”, por lo que en la nueva relación presente entre las clases Repuesto_Solicitado y Solicitud_Repuestos impera una relación “de Agregación” de tipo compuesto, denotada por el diamante relleno colocado cerca de la clase que representa el “todo”, que sería Solicitud_Repuestos, lo que se refiere a que si es eliminado una solicitud de repuestos automáticamente todos sus repuestos solicitados dejan de existir. Mientras que la agregación entre las clases Repuesto y Detalle_Presupuesto o Detalle_Factura es simple, indicada por un diamante vacío colocado cerca de la clase base Repuesto. De este modo las relaciones indicadas se pueden leer “uno o muchos Repuesto_Solicitado son parte de una Solicitud_Repuestos”, o en sentido inverso, “una Solicitud_Repuestos tiene uno o varios Repuesto_Solicitado”. Igual ocurre entre las otras clases mencionadas, donde la relación de agregación se leería “un Repuesto es parte de ningún o de muchos Detalle_Presupuesto o Detalle_Factura”, o lo que es igual, “un Detalle_Presupuesto o Detalle_Factura tiene ningún o varios Repuestos”. En la figura 4.4 siguiente se muestra el diagrama de dominio definitivo:

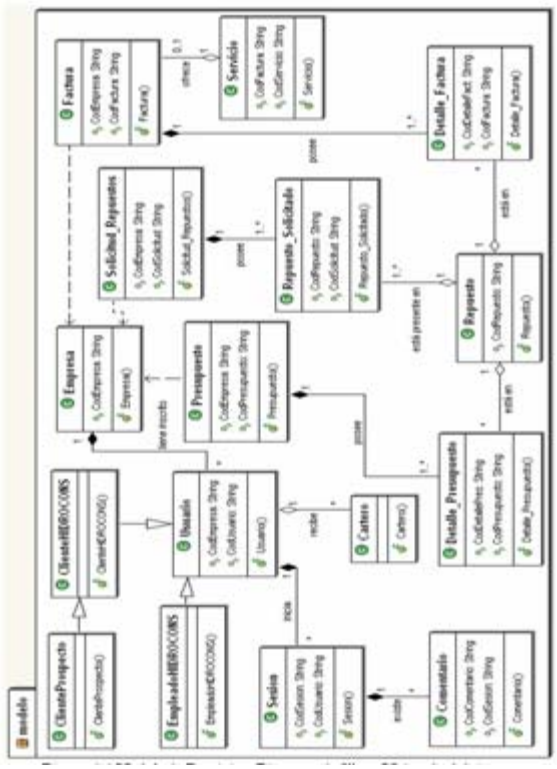


Figura 4.4. Modelo de Dominio o Diagrama de Clases Mejorado del sistema
Fuente: Propia, 2009

4.4. Análisis

En el flujo de trabajo de análisis se tomará como datos de entrada tanto los diagramas de casos de uso, como la información presente en el diagrama de modelo de dominio o de clases mejorado obtenidos hasta el momento en el anterior flujo de trabajo de requisitos. Se desea obtener una vista más dinámica del sistema y sus elementos, para así tener una base de conocimiento y análisis lo suficientemente estable y fuerte para realizar un diseño óptimo y efectivo. [15] Haciendo uso de

dichos casos de uso se desarrollan otros diagramas vitales para la continuidad de este proyecto, como son los diagramas de secuencia que muestran el comportamiento y tiempo de vida de los objetos que interactúan en estos casos, y una introducción de algunos modelos del lenguaje WebML, para reforzar los diagramas UML de secuencia mencionados anteriormente, además de orientarlos a la navegabilidad de las futuras páginas Web que se han de fabricar.

4.4.1. Diagramas de secuencia

Un diagrama de secuencia es una descripción de una colección de objetos que interactúan para implementar un cierto comportamiento dentro de un contexto. Describe una sociedad de objetos cooperantes unidos para realizar un cierto propósito. Los objetos trabajan juntos para realizar tareas comunicándose el uno con el otro, por lo tanto los diagramas de secuencia modelan a nivel de objetos más que a nivel de clases. Dicho diagrama utiliza dos elementos fundamentales:

- **Línea de vida del objeto**: la notación de la línea de vida de un objeto es una combinación de un objeto con una línea de tiempo, la cual es una línea discontinua que sale de la base del objeto mismo. El largo de dicha línea representa el tiempo que el objeto interactúa en el escenario.

- **Mensajes**: es la definición para la comunicación entre objetos. Estos mensajes pueden invocar una operación, causar una creación o destrucción de un objeto, etc. Los mensajes se representan con una flecha de línea punteada, pero según el tipo de mensaje el tipo de flecha puede cambiar. Mensajes asíncronos (que no ocurren simultáneamente, sino uno después del otro) son los que imperan en estos diagramas, y los mismos están representados por una flecha con punta completa; a diferencia de los síncronos (ocurren al mismo tiempo), que se denotan con una flecha con media punta. La cola de la flecha

representa al remitente del mensaje, mientras que la punta representa el receptor del mismo. [6]

A continuación se muestran algunos diagramas de secuencia referentes a ciertos casos de uso que intervienen en esta aplicación:

- **Diagrama de Secuencia del Caso de Uso "Iniciar Sesión Usuario"**: como se puede observar en la figura 4.5, la cual describe la realización de todo este procedimiento, este caso de uso lo puede llevar a cabo cualquier tipo de actor del sistema, es decir, cualquier Empleado o Cliente de HIDROCONS registrado en la aplicación. Para realizar este caso de uso el usuario introduce en los campos respectivos su nombre de usuario o login y su contraseña o password. Al presionar el botón de ingreso que desencadenará los procesos de autentificación para este actor, el sistema crea una instancia del objeto USUARIO y le es traspasado dichos datos (login y clave) acompañados del mensaje "Iniciar Sesión Usuario". Este objeto USUARIO le envía al objeto SGBDD, igualmente creado con anterioridad por el sistema, el mensaje de "Consultar Datos", acompañado como es debido de los datos ingresados por el actor en uso de la aplicación. Esto con la finalidad de determinar si es un usuario registrado en el sistema el que está intentando ingresar al mismo, mas sin embargo no basta con que su nombre de usuario y contraseña estén presentes en la base de datos, porque en dicha consulta de datos también se verifica si el mismo está en Situación "ACTIVO", o sea, que no está dado de baja, y que no esté conectado al sistema en ese mismo instante, es decir, no pretenda ingresar nuevamente a la aplicación estando previamente ya conectado al sistema en cuestión.

En caso de cumplir con todos estos requisitos se le devuelve al objeto USUARIO la respuesta de que efectivamente es un usuario aprobado, junto con los

demás datos presentes en la base de datos propios de dicho usuario (Cédula del usuario, nombres, apellidos, cargo, teléfonos, correo electrónico, etc.). Posteriormente el sistema crea un objeto SESIÓN y el objeto USUARIO le envía un mensaje de “Iniciar Sesión” junto con el código del usuario extraído de la consulta realizada a comienzos de este caso de uso; este último objeto a su vez le envía al objeto SGBDD previamente creado la orden de “Registrar Datos”, acompañado de los datos de sesión a ingresar (Código de la nueva sesión, código del usuario que ingresa, fecha y hora de ingreso del mismo, estos últimos datos tomados directamente del sistema operativo obviamente).

Finalizado este ingreso exitoso de datos, el objeto SGBDD le envía al objeto SESIÓN la respuesta de que fueron ingresados los datos de la nueva sesión, para luego ser destruido por el sistema. Este objeto SESIÓN a su vez también devuelve al objeto USUARIO, antes de ser destruido por el sistema, el mensaje de que la “Sesión fue iniciada”, junto con el código de la sesión asignado por el sistema. Esto para que cuando se necesite consultar los datos de la sesión en procedimientos posteriores, por ejemplo, al “Cerrar la Sesión del Usuario”, se conozca cual código de sesión está asignado al usuario. Finalmente el objeto SESIÓN envía un mensaje de “Sesión Iniciada y Usuario Conectado” al actor que actualmente usa el sistema, quien lo puede constatar al acceder a su página principal, bien sea un Empleado o un Cliente de HIDROCONS. Finalmente el sistema destruye al objeto USUARIO.

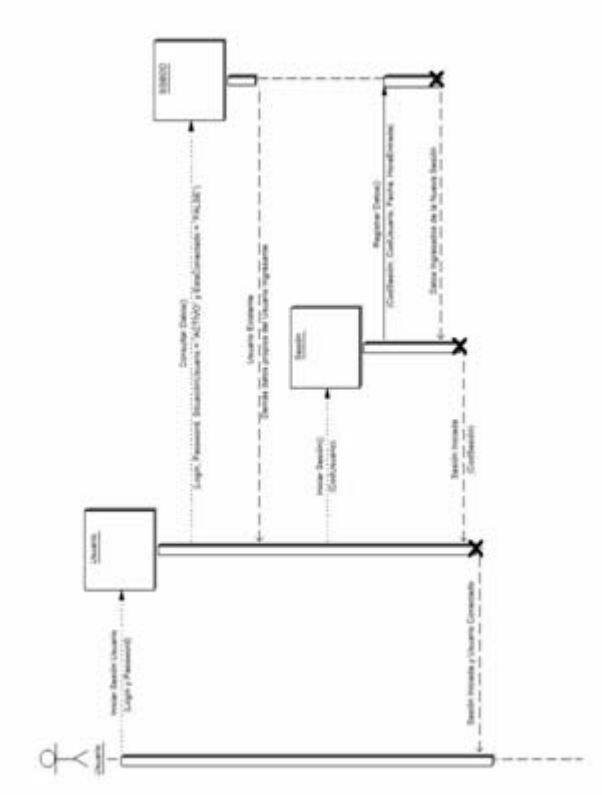


Figura 4.5. Diagrama de Secuencia del Caso de Uso "Iniciar Sesión Usuario"
Fuente: Propia, 2009

- **Diagrama de Secuencia del Caso de Uso "Registrar Empresa"**: considérese que de todos los usuarios que pueden realizar este caso de uso sea el Empleado HIDROCONS el que lo lleve a cabo. Como se observa en la figura 4.6, en el momento en que dicho usuario origina la solicitud, el sistema ordena la creación de un objeto de tipo USUARIO, el cual le envía el mensaje "Registrar Empresa" con todos los datos de la nueva empresa al objeto EMPRESA igualmente creado por el sistema, pero posteriormente. Automáticamente el mismo objeto EMPRESA se autoenvía el mensaje

> “Consultar Empresa” acompañado sólo del RIF o del NIT de la empresa a ingresar. Luego de ello, el sistema crea un objeto SGBDD y el objeto EMPRESA le envía el mensaje “Consultar Datos” con el RIF o NIT mencionados anteriormente. Dicho objeto averigua si existe registrada en la base de datos alguna empresa con dicho NIT, o en su defecto, con el RIF ingresado. Si no existe tal empresa el objeto SGBDD devuelve al objeto EMPRESA el mensaje “Empresa No Existente”, por lo cual el objeto EMPRESA envía el mensaje “Registrar Datos” al objeto SGBDD acompañado de todos los datos de la nueva empresa. El objeto SGBDD, luego de registrar efectivamente la empresa envía un mensaje al objeto EMPRESA de “Empresa Registrada Exitosamente”.

Inmediatamente después, el objeto EMPRESA le envía al objeto CARTERO, creado por el sistema, la orden de enviar un correo a la empresa ya aprobada, dando la bienvenida al sistema, para luego ser destruido. Posteriormente se crea el objeto SESIÓN y el objeto EMPRESA le ordena “Modificar Sesión” con el código de la sesión designado al instante de iniciarla, para ser modificada por el SGBDD mediante el mensaje “Modificar Datos”. El objeto SGBDD devuelve el mensaje de éxito al objeto SESIÓN, quien a su vez se lo envía al objeto EMPRESA, quien igualmente se lo envía al objeto USUARIO, todo ello antes de que todos estos objetos sean destruidos por el sistema. Finalmente dicho mensaje es recibido por el Empleado.

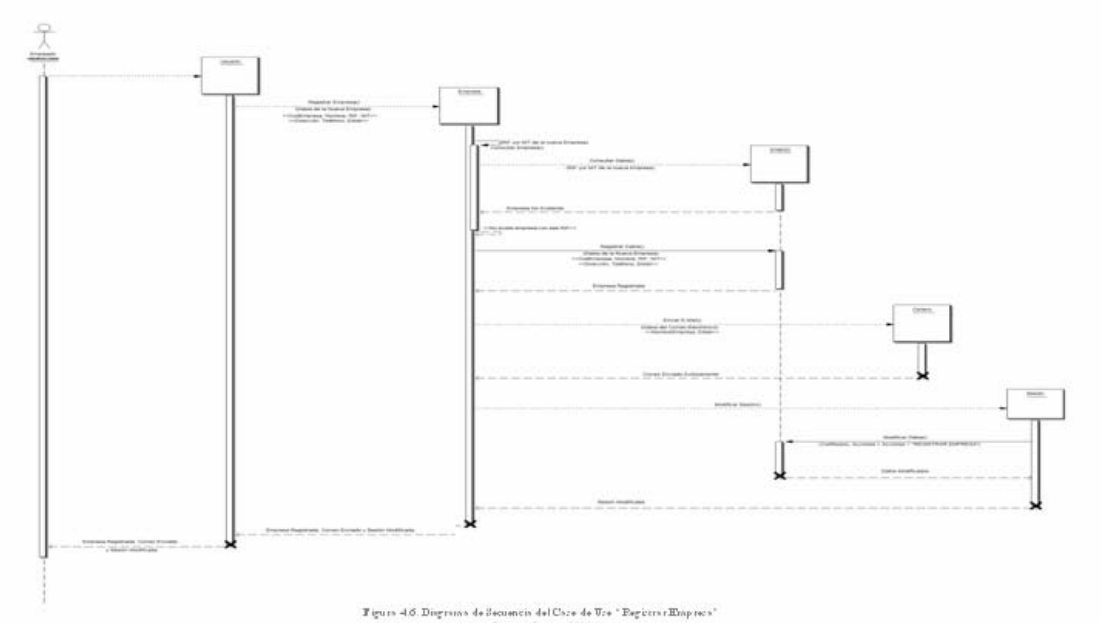


Figura 4.6 Diagrama de Secuencia del Caso de Uso "RegistrarEmpresa"
Fuente: Propia, 2009

- **Diagrama de Secuencia del Caso de Uso "Consultar Empresa"**: como se puede observar en la figura 4.7 vista a continuación, considérese que de todos los usuarios que pueden realizar este caso de uso, sea un Cliente HIDROCONS el que lleve a cabo el mismo. En el momento que dicho cliente genera la solicitud, el sistema crea una instancia del objeto USUARIO, el cual posee todos los datos completos registrados en la BDD del Cliente en cuestión. Posterior a todo ello, el sistema crea un objeto de tipo EMPRESA y el objeto USUARIO creado previamente le envía a éste nuevo objeto el mensaje "Consultar Empresa" acompañado del Código de Usuario del propio Cliente.

Luego el sistema crea un objeto SGBDD (el cual se encarga de gestionar todo lo referente a transacciones que tienen que ver con la Base de Datos del sistema) quien recibe de parte del objeto EMPRESA el mensaje "Consultar Datos", para que este realice la consulta en la BDD con el Código de Usuario que es facilitado

igualmente por parte del objeto USUARIO. Después, el objeto SGBDD le devuelve como respuesta al objeto EMPRESA los datos de la empresa a la cual pertenece el Cliente que hace la solicitud (Código de la empresa, nombre, RIF y NIT de la misma, dirección completa y teléfonos registrados para su ubicación, así como su correo electrónico), para luego ser destruido por el sistema. Este objeto EMPRESA a su vez le devuelve toda esta información recibida del objeto SGBDD al objeto USUARIO antes de ser igualmente destruido. Finalmente el objeto USUARIO muestra este conjunto de datos obtenidos al Cliente en su función de actor del sistema, para ser posteriormente destruido.

Se puede notar en la figura una especie de etiqueta, colocada entre el actor y el objeto USUARIO, la cual constituye una ventaja que posee este tipo de diagramas, ya que representa un comentario que puede ser colocado en el mismo con la finalidad de especificar o de explicar de mejor manera el diagrama mismo.

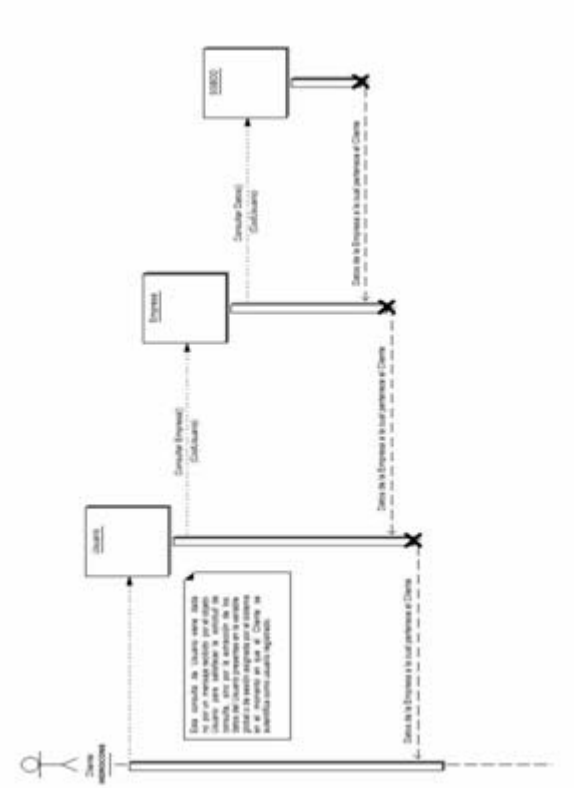


Figura 4.7. Diagrama de Secuencia del Caso de Uso "Consultar Empresa"
Fuente: Propia, 2009

- **Diagrama de Secuencia del Caso de Uso "Modificar Repuesto"**: este caso de uso es realizado exclusivamente por un Empleado HIDROCONS, ya que este actor es el que está habilitado únicamente para llevar a cabo este proceso. En la figura 4.8 se puede observar que este actor, al momento de hacer la solicitud de modificación, el sistema crea una instancia del objeto USUARIO, el cual posee todos los datos completos registrados en la BDD del usuario o Empleado de HIDROCONS en cuestión. Posteriormente a ello, el sistema crea un objeto REPUESTO. Luego, el objeto USUARIO envía el mensaje "Modificar Repuesto" al objeto REPUESTO recién creado, acompañado de los nuevos datos a modificar del repuesto (nombre, presentación, precio, tipo, sub-tipo, foto, situación, etc). El objeto REPUESTO se autoenvía la orden "Consultar Repuesto" junto con el código y nombre del repuesto, que a su vez

lo releva al objeto SGBDD igualmente creado por el sistema, mediante el mensaje “Consultar Datos”. Esto con la finalidad de determinar si existe en la base de datos un repuesto exactamente con el mismo nombre recién ingresado que tenga un código diferente al del repuesto actual, y así evitar que se repitan y dupliquen nombres de repuestos.

De no existir un repuesto con el mismo nombre recién ingresado, el objeto REPUESTO recibe dicha respuesta y procede a enviar los nuevos datos que modificarán el estado de dicho repuesto mediante el mensaje “Modificar Datos” al objeto SGBDD. Luego de modificar los datos en cuestión, el SGBDD envía un mensaje de éxito al objeto REPUESTO. Luego el sistema crea un objeto de tipo SESIÓN y el objeto REPUESTO a su vez envía a este objeto SESIÓN recién creado, la orden de “Modificar Sesión” junto con el código de sesión para anexar esta acción. Éste último se lo reenvía al objeto SGBDD y este hace lo propio, devolviendo un mensaje de éxito. El objeto SESIÓN lo recibe y antes de ser destruido lo envía a REPUESTO, y éste luego lo envía a USUARIO, antes de ser eliminado igualmente por el sistema. El Empleado recibe al final el mensaje de “Repuesto Modificado”.

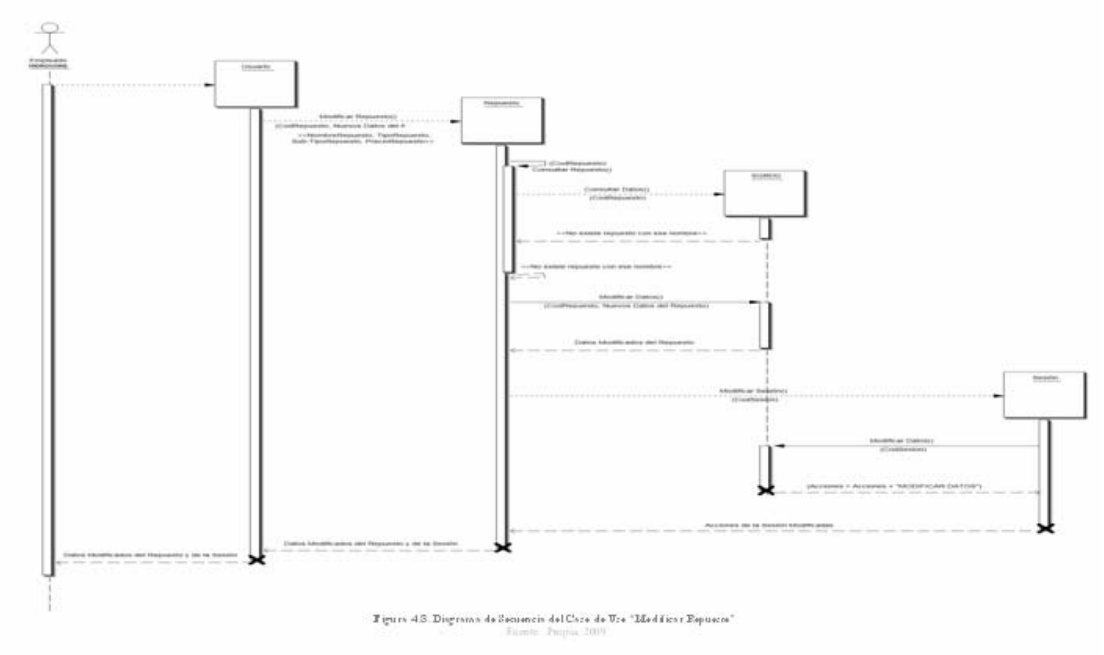


Figura 4.8. Diagrama de Secuencia del Caso de Uso "Modificar Repuesto"
Fuente: Propia, 2009

- **Diagrama de Secuencia del Caso de Uso "Dar de Baja Usuario"**: de todos los usuarios que pueden realizar este proceso, se puede considerar que sea el mismo Cliente HIDROCONS el que lleve a cabo la realización de este caso. Según se observa en la figura 4.9 de la página siguiente, en el momento que dicho cliente lleva a cabo la solicitud, el sistema crea un objeto de tipo USUARIO, el cual posee todos los datos completos registrados en la base de datos del Cliente en cuestión. Posteriormente a ello, dicho objeto USUARIO se autoenvía el mensaje "Dar De Baja Usuario", junto con el código de usuario respectivo, por lo cual el sistema crea un objeto SGBDD y el objeto USUARIO le indica la orden de "Desactivar Datos", acompañado de dicho código de usuario que se da de baja. El SGBDD desactiva al cliente, devolviéndole a USUARIO el mensaje respectivo. Luego este objeto envía al objeto CARTERO creado por el sistema la orden de enviar un correo informando al cliente eliminado de su baja, para luego ser destruido por el sistema. Posteriormente el objeto USUARIO se autoenvía la orden "Cerrar Sesión Usuario", para luego enviar el mensaje "Modificar Datos" al objeto

SGBDD, que haciendo uso del Código de Usuario que acompaña dicho mensaje, modifica la situación del usuario a No Conectado, regresando al objeto USUARIO el mensaje de “Usuario Desconectado”.

Luego se crea un objeto SESIÓN y el objeto USUARIO le indica la orden de “Cerrar Sesión” acompañado del código de sesión asignado al momento de iniciar su visita en el sistema”. El objeto SESIÓN a su vez envía el mensaje “Modificar Datos” al SGBDD, junto con el código de la sesión en cuestión y la hora actual tomada del sistema, con lo que actualiza dicha sesión, y antes de ser destruido por el sistema, devuelve al objeto SESIÓN el mensaje de éxito, el cual igualmente, antes de ser destruido por el sistema, devuelve un mensaje de “Sesión Cerrada” al objeto USUARIO. El objeto USUARIO le devuelve al Cliente el mensaje de éxito, antes de que el sistema lo destruya finalmente. El Cliente queda fuera del sistema e inactivo.

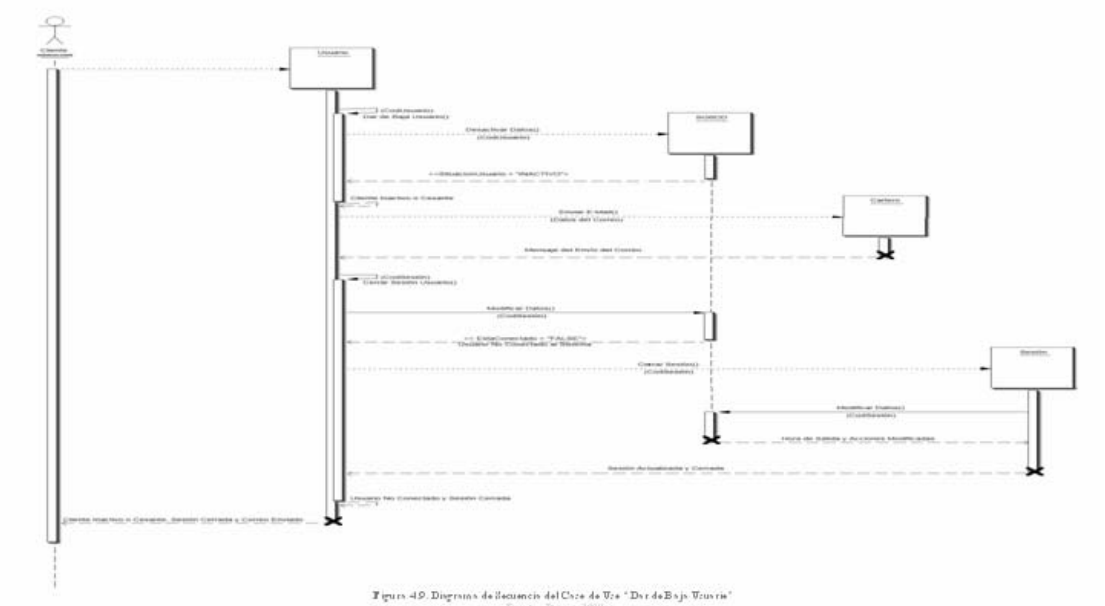


Figura 49. Diagrama de Secuencia del Caso de Uso “Dar de Baja Usuario”
Fuente: Propia, 2009

- **Diagrama de Secuencia del Caso de Uso "Registrar Presupuesto"**: este caso de uso es llevado a cabo por un Empleado HIDROCONS. Según la figura 4.10, este actor hace la solicitud de registrar el presupuesto y el sistema crea un objeto USUARIO, que le envía a un objeto PRESUPUESTO igualmente creado el mensaje de "Registrar Presupuesto", enviándole además los datos completos del encabezado del presupuesto en cuestión. Posterior a ello el objeto PRESUPUESTO se autoenvía el mensaje "Consultar Presupuesto" junto con el código del mismo. Luego el sistema crea un objeto SGBDD y este objeto PRESUPUESTO le envía el mensaje "Consultar Datos" acompañado de dicho código. Esto con la finalidad de determinar que dicha referencia no se encuentra registrada en la base de datos. De no existir este presupuesto, se da pie para que el objeto PRESUPUESTO le envíe el mensaje "Registrar Datos" al objeto SGBDD junto con todos los datos de la cabecera del documento facilitados por el Empleado. Posteriormente a ello el sistema procede a registrar la lista de detalles del presupuesto que siguen al encabezado del mismo. Para ello se crea un objeto DETALLE_PRESUPUESTO y el objeto PRESUPUESTO le envía la lista completa de todos los detalles o ítems del presupuesto mediante la orden "Registrar Detalles de Presupuesto" (Plural) y, por cada uno de los ítems o detalles que conforman el cuerpo del presupuesto, el mismo objeto DETALLE_PRESUPUESTO se autoenvía cada uno de los datos de dichos ítems mediante el mensaje "Registrar Detalle de Presupuesto" (Singular). A su vez este objeto DETALLE_PRESUPUESTO le envía el mensaje "Registrar Datos" a SGBDD acompañado por dichos datos del detalle.

Llegado al final de la lista de ítems del presupuesto, el objeto PRESUPUESTO envía al objeto CARTERO ya creado por el sistema la orden de enviar un correo al E-Mail de la empresa señalándole que tienen un presupuesto ya preparado en el sistema esperando que se le apruebe o rechace. Luego de ello se procede a registrar esta

transacción realizada por el usuario. Para ello el sistema crea un objeto SESIÓN, quien recibe de parte del objeto PRESUPUESTO la orden “Modificar Sesión” acompañado del código de sesión asignado al usuario al momento de entrar en el sistema. Con este código el objeto SESIÓN envía el mensaje “Modificar Datos” al objeto SGBDD para que actualice el registro de dicha sesión, anexándole un comentario de este proceso llevado a cabo. Una vez hecho esto el objeto SGBDD es destruido por el sistema, el objeto SESIÓN le envía un mensaje de éxito al objeto PRESUPUESTO, el cual se lo devuelve también al objeto USUARIO y este le indica lo mismo al Empleado, para que finalmente los objetos SESIÓN, PRESUPUESTO y USUARIO también sean destruidos por el sistema.

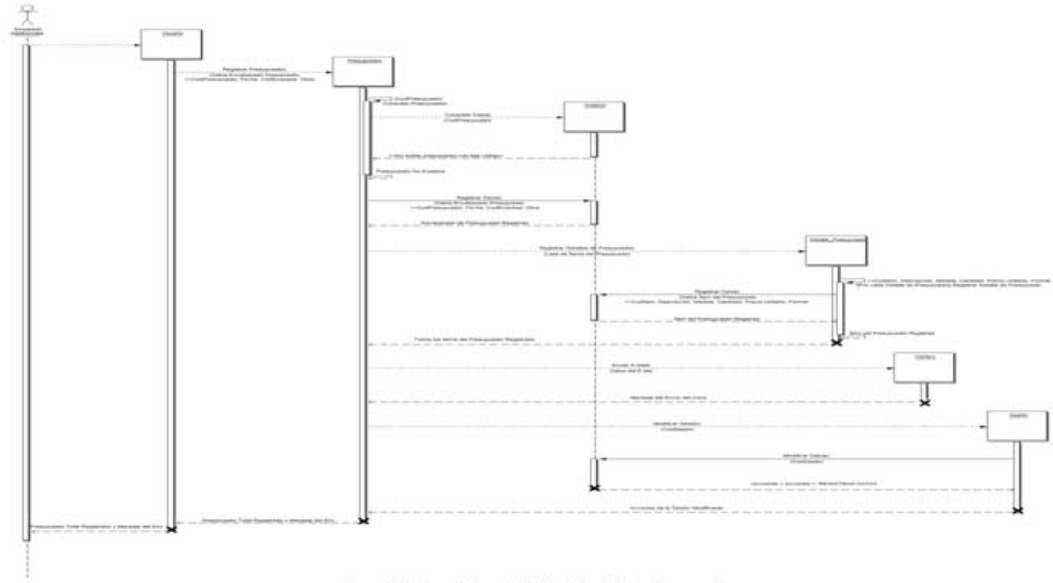


Figura 4.10 Diagrama de Secuencia del Caso de Uso “Egresos e Ingresos”
Fuente: Propia, 2009

- **Diagrama de Secuencia del Caso de Uso “Consultar Facturas”**: considérese que de todos los actores que pueden realizar este caso de uso, sea el Empleado HIDROCONS el que lo lleve a cabo. Véase la figura 4.11, donde el Empleado hace la solicitud de consulta y el sistema crea un objeto FACTURA, al cual se le envía el mensaje “Consultar Facturas” acompañado de los datos que sirven de criterio de búsqueda (todas las facturas, número de

factura, código de la empresa a la cual se facturó, situación de las facturas y/o fecha de facturación). El objeto FACTURA se envía a sí mismo un mensaje de "Consultar Factura", igualmente junto con los criterios de búsqueda, que, a diferencia del mensaje previamente recibido, este consulta una sola factura a la vez. El sistema crea luego un objeto SGBDD al cual le es enviado por parte del objeto FACTURA el mensaje "Consultar Datos" acompañado por dichos criterios de búsqueda. El objeto SGBDD le devuelve al objeto FACTURA los datos del encabezado de la factura en cuestión (Número de Factura, Lugar de Emisión, fecha de facturación, código de la empresa a la cual va dirigida, IVA y forma de pago). Posteriormente el sistema crea un objeto EMPRESA y el objeto FACTURA le envía el mensaje "Consultar Empresa" junto con el código de la empresa obtenido en la consulta previamente descrita. A su vez el objeto EMPRESA le envía el mensaje "Consultar Datos" al objeto SGBDD junto con dicho código de empresa. Esto con la finalidad de que el objeto SGBDD le devuelva al objeto EMPRESA los demás datos del encabezado de la factura correspondientes a la empresa a la cual fue emitida dicha factura (Nombre de la empresa, RIF, teléfono y dirección completa), y este objeto EMPRESA a su vez, antes de ser destruido por el sistema, se lo devuelva al objeto FACTURA.

Ahora se procede a extraer los datos de los ítems o renglones que conforman el cuerpo de la factura. El sistema crea un objeto denominado DETALLE_FACTURA y el objeto FACTURA le envía el mensaje "Consultar Detalles de Factura" junto con el código de la factura. Este objeto a su vez se autoenvía el mensaje "Consultar Detalle de Factura", es decir, un solo ítem a la vez, acompañado del código de factura recibido. Posteriormente el objeto DETALLE_FACTURA le envía dicho código al objeto SGBDD mediante el mensaje "Consultar Datos", quien le devuelve a su vez los datos del ítem esperado (cantidad, descripción, presentación o medida, formato y precio unitario). Este mensaje igualmente es reenviado al mismo objeto

DETALLE_FACTURA como respuesta al mensaje de consulta de ítem individual. Sin embargo, si aún quedan varios ítems por extraer de la factura consultada, el mensaje individual “Consultar Detalle de Factura” se repite tantas veces como ítems existan en la factura en cuestión. Al recuperar hasta el último renglón del cuerpo de la factura en consulta, esta lista de ítems a su vez se devuelve por parte del objeto DETALLE_FACTURA, quien es destruido por el sistema finalmente, al objeto FACTURA, con lo que se completó la información total de una factura para responder a la solicitud del usuario. Sin embargo, si existen más facturas que cumplan con los criterios de búsqueda introducidos al principio de este proceso, el mensaje “Consultar Factura” se reenvía nuevamente, repitiéndose este proceso hasta extraer la información de la última factura. Finalmente el objeto FACTURA devuelve al usuario la “Lista de Facturas obtenidas” y es destruido por el sistema.

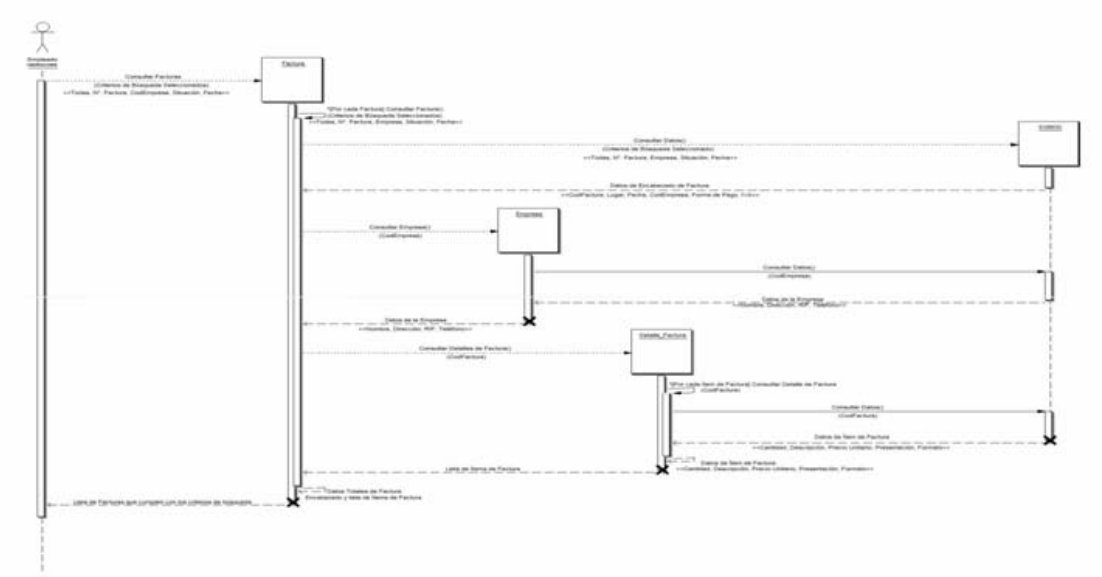


Figura 411 Diagrama de Secuencia del Caso de Uso “ConsultarFacturas”
Fuente: Propia, 2009

- **Diagrama de Secuencia del Caso de Uso "Consultar Solicitudes de Repuestos"**: considérese que sea el Empleado HIDROCONS el que lo lleve a cabo este proceso. Según la figura 4.12, la cual es la que especifica este caso de uso, dicho Empleado hace la solicitud de consulta y el sistema crea un objeto SOLICITUD_REPUESTOS, al cual se le envía el mensaje "Consultar Solicitudes de Repuestos" donde el criterio de búsqueda va a ser todas las solicitudes que se encuentren en situación de EN ESPERA. El objeto SOLICITUD_REPUESTOS se envía a sí mismo un mensaje de "Consultar Solicitud de Repuestos" que, a diferencia del mensaje previamente recibido, este consulta una sola solicitud a la vez. El sistema crea luego un objeto SGBDD al cual le es enviado por parte del objeto SOLICITUD_REPUESTOS el mensaje "Consultar Datos". El objeto SGBDD le devuelve al objeto SOLICITUD_REPUESTOS los datos del encabezado de la solicitud en cuestión (Código de la solicitud en espera, código de la empresa que hizo la solicitud, fecha y situación de dicha solicitud). Posteriormente el sistema crea un objeto EMPRESA y el objeto SOLICITUD_REPUESTOS le envía el mensaje "Consultar Empresa" junto con el código de la empresa obtenido en la consulta previamente descrita. A su vez el objeto EMPRESA le envía el mensaje "Consultar Datos" al objeto SGBDD junto con dicho código de empresa. Esto con la finalidad de que el objeto SGBDD le devuelva al objeto EMPRESA sólo el nombre de la empresa que realizó la solicitud de repuestos, y éste a su vez, antes de ser destruido por el sistema, se lo devuelva al objeto SOLICITUD_REPUESTOS.

Ahora se procede a extraer los datos de los ítems o renglones que conforman el cuerpo de la solicitud. El sistema crea un objeto denominado REPUESTO_SOLICITADO y el objeto SOLICITUD_REPUESTOS le envía el mensaje "Consultar Repuestos Solicitados" junto con el código de la solicitud. Este objeto a su vez se envía el mensaje "Consultar Repuesto Solicitado", es decir, un solo

repuesto a la vez, acompañado del código de solicitud recibido. Posteriormente el objeto REPUESTO_SOLICITADO le envía dicho código al objeto SGBDD mediante el mensaje “Consultar Datos”, objeto que le devuelve a su vez los datos del repuesto esperado (Código del repuesto solicitado, cantidad solicitada del repuesto, el precio por el cual se hizo la solicitud y la situación o estado del mismo). Sin embargo, estos datos del repuesto no están completos, es decir, no son todos los que se necesitan. Para ello el sistema crea un objeto REPUESTO al cual le es enviado por parte del objeto REPUESTO_SOLICITADO el mensaje “Consultar Repuesto” acompañado del Código del repuesto solicitado. Este objeto REPUESTO a su vez envía al objeto SGBDD el mensaje “Consultar Datos” con el código del repuesto anterior. El SGBDD le devuelve al objeto REPUESTO los datos que faltan del repuesto en cuestión (Nombre y presentación o medida de dicho repuesto) para ser descrito más detalladamente en el cuerpo de la solicitud de repuestos. Igualmente el objeto REPUESTO, antes de ser destruido, le devuelve esa misma información al objeto REPUESTO_SOLICITADO, quien a su vez se lo reenvía a sí mismo junto con los demás datos extraídos anteriormente como respuesta al mensaje individual de consulta de repuesto solicitado.

Si aún quedan varios repuestos por extraer de la solicitud consultada, el mensaje individual “Consultar Repuesto Solicitado” se repite tantas veces como repuestos solicitados existan cuyo código de solicitud sea el que actualmente se está consultando. Esta lista de repuestos a su vez se devuelve por parte del objeto REPUESTO_SOLICITADO, antes de ser destruido por el sistema, al objeto SOLICITUD_REPUESTOS, con lo que se completó la información total de una sola Solicitud de Repuestos. Sin embargo, si existen más solicitudes cuya situación se encuentre EN ESPERA, todo este proceso se produce nuevamente hasta extraer la información de la última solicitud. Finalmente el objeto SOLICITUD_REPUESTOS devuelve al Empleado la “Lista de Solicitudes que se encuentran EN ESPERA” y es destruido por el sistema.

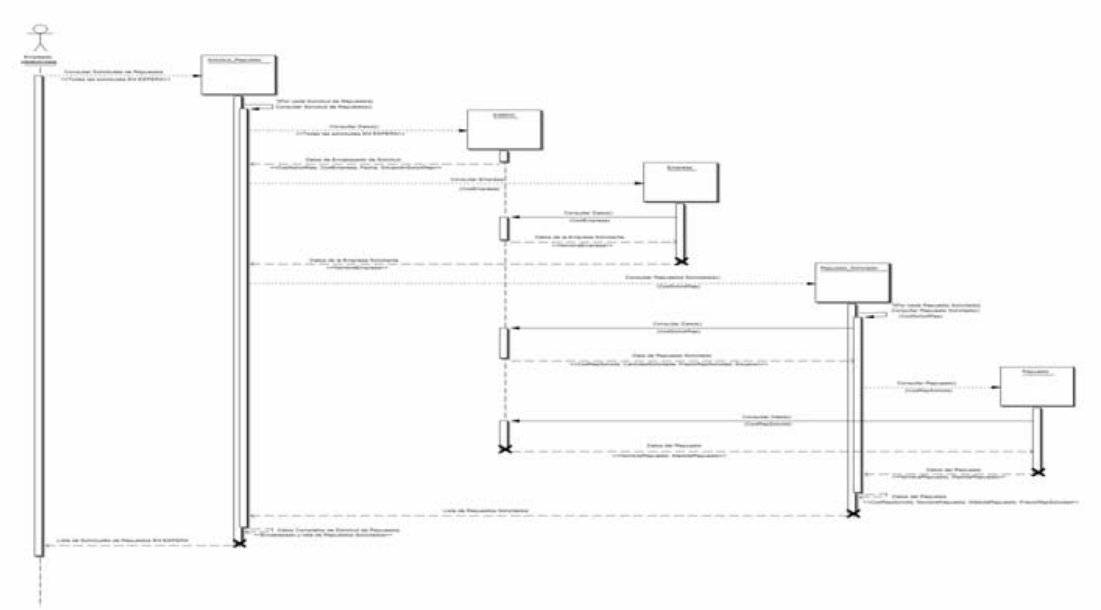


Figura 412.Diagrama de Secuencia del Caso "Consultar Solicitudes de Espacios"
Fuente: Propia, 2009

4.4.2. Diagramas WEBML

Los diagramas WEBML más resaltantes que encajan en esta fase de elaboración son el diagrama o Modelo Estructural y el diagrama o Modelo de Hipertexto. El primero muestra los objetos publicados en el sitio Web y cómo ellos están relacionados con respecto a los objetos que describe. El segundo, como ya se mencionó anteriormente es una fusión del modelo de Navegación y del modelo de Composición. En el modelo de Navegación se observan la navegabilidad y accesibilidad entre las diversas páginas, mientras que en el de Composición se observan las unidades de contenido, las páginas y los enlaces.

4.4.2.1. Modelo estructural WEBML

El modelo estructural propio de este proyecto se observa en la figura 4.13. El mismo muestra los objetos publicados en el sitio Web y cómo ellos están relacionados. Se observan las características comunes de todo objeto, tales como **Entidad**: el nombre o tipo de objeto (Usuario, Factura, Repuesto, etc). **Atributos**: propiedades escalares de una entidad, que por cuestiones de espacio se usarán atributos claves tales como códigos. **Relaciones**: conexiones entre entidades, como las existentes entre Usuario y Sesión, donde un (1:1) Usuario puede iniciar ninguna o muchas Sesiones (0:N), o como la que existe entre los objetos Factura y Detalle_Factura, ya que una (1:1) Factura posee de uno a muchos Detalles (1:N). **Jerarquía ES_UN**: clasifica y agrupa a las entidades, tal como se observa en los objetos actores, ya que un Empleado o Cliente constituyen tipos de Usuario, o un Cliente Prospecto representa un tipo de Cliente. El modelo estructural en cierta forma tiene un cierto parecido con el Diagrama de Clases propio de UML, pero para efectos de diseño se tratan ambos diagramas por separado, no como un mismo artefacto, ya que uno es para la construcción de clases y este otro para la construcción de páginas Web.

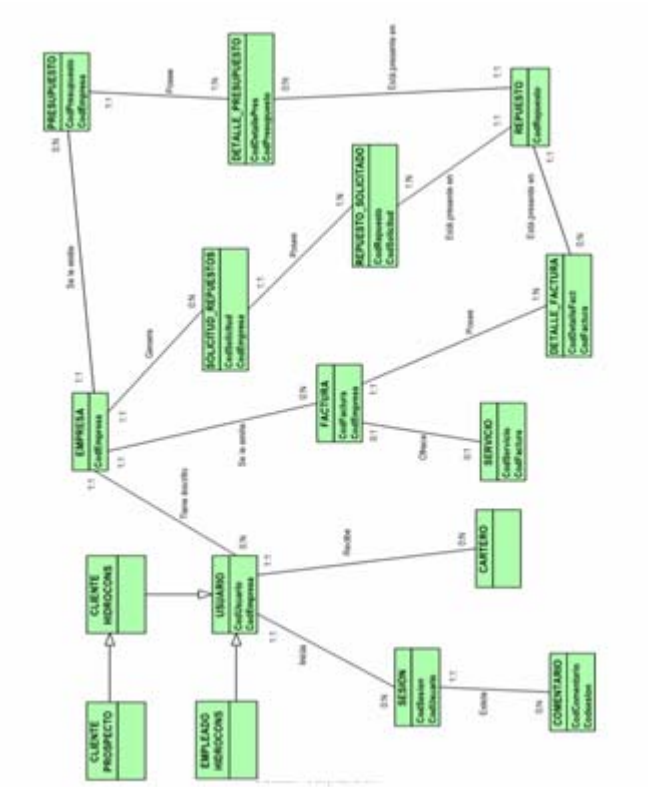


Figura 4.13. Modelo Estructural WebML del sistema
Fuente: Propia, 2009

4.4.2.2. Siteviews WEBML

Un SiteView es un sub-conjunto de páginas en la que el usuario puede experimentar como un sitio Web completo. Representa un número de páginas y/o áreas que forman una vista coherente del sitio Web. A continuación, se realiza una descripción de los SiteViews, mediante tablas explicativas, tanto público como privados, propios de cada tipo de usuario, además de sus respectivos mapas de SiteView, y que conforman la estructura lógica y organizacional de la aplicación Web.

Tabla 4.2. SiteView Público "Principal Externo"

Nombre del SiteView	**PRINCIPAL EXTERNO**	
Descripción	Es la primera vista que se le muestra a cualquier usuario que accede al dominio o dirección URL del sistema SIADCON. Como su nombre lo indica, es la vista o interfaz externa del sistema. Se observan ciertos servicios de tipo informativos, tanto estáticos, que no consultan a ninguna base de datos (Home, información de la empresa, razones para afiliarse a SIADCON, pasos para afiliarse, etc.) así como dinámicos, extraídos desde la base de datos e ingresados a la misma (Consulta de repuestos, consulta de los servicios que presta la empresa, registro de nueva empresa y afiliación de nuevo cliente, recuperación de claves olvidadas, envío de comentarios, etc.).	
Grupo de Usuarios	Usuarios en general (Cualquier usuario que acceda a la página principal de SIADCON), esté o no registrado en el mismo.	
Casos de Uso Asociados	➢ Administrar Empresas ➢ Administrar Servicios	➢ Administrar Usuarios ➢ Administrar Repuestos

Tabla 4.3. Mapa del SiteView Público "Principal Externo"

Nombre del área	**Descripción del área**
Home	Regresa al usuario a la página principal externa del sistema SIADCON.
Nuestra Empresa	Páginas que contienen suficiente y resumida información concerniente a la historia, logros, experiencias y alcances de la empresa HIDROCONS, C. A.

Tabla 4.3. Mapa del SiteView Público "Principal Externo" (Continuación)

Nombre del Área	**Descripción del área**
Repuestos	Muestra los diferentes repuestos y accesorios disponibles a la venta al público en general.

Servicios	Se observan un conjunto de actividades y servicios detallados que explican las obras que la empresa está en capacidad de ofrecer a los clientes y llevar a cabo.
HIDROCONS On Line	Muestra las diversas maneras de cómo comunicarse con los empleados de HIDROCONS, C. A. (correo electrónico, mensajería, ubicación, teléfonos), incluyendo la ventaja de afiliarse al sistema SIADCON y los pasos a seguir para realizar dicha afiliación.
Contáctenos	Es una extensión del área mencionada previamente (HIDROCONS On Line), en donde exclusivamente se explican las posibles maneras de comunicarse con la empresa HIDROCONS, C. A. (ubicación geográfica de la sede, números telefónicos, correo electrónico, mensajería directa, etc.).
¿Nuevo Usuario?	Es una vista donde cualquier usuario inscrito en cualquier empresa cliente de HIDROCONS, C. A. puede solicitar la afiliación de dicha empresa como de su persona.
¿Olvidó su Contraseña?	Forma útil y segura para recuperar los datos de autentificación de cualquier usuario, como lo son su nombre de usuario y su contraseña.
SIADCON	Representa el acceso a las vistas privadas del sistema, sólo para usuarios registrados de dicho sistema.

Tabla 4.4. SiteView Privado "Principal para Empleados de HIDROCONS"

Nombre del SiteView	**PRINCIPAL PARA EMPLEADOS DE HIDROCONS, C. A.**	
Descripción	Constituye la primera vista que se le muestra al empleado cuando este se autentifica e ingresa como usuario válido al sistema. En esta se muestran las opciones y privilegios a las que puede acceder y que les son permitidas a dicho empleado en cuanto a administración y gestión de empresas, de usuarios, de presupuestos y facturas, así como de servicios y actividades que presta la empresa y repuestos que vende.	
Grupo de Usuarios	Empleados de HIDROCONS, C. A. que estén inscritos en el sistema SIADCON	
Casos de Uso Asociados	➢ Administrar Empresas ➢ Administrar	➢ Administrar Usuarios

	Presupuestos ➢ Administrar Servicios	➢ Administrar Facturas ➢ Administrar Repuestos

Tabla 4.5. Mapa del SiteView Privado "Principal para Empleados de HIDROCONS"

Nombre del Área	Descripción del área
Empresas	Se ingresan compañías al sistema, se consultan, modifican y dan de baja a las ya registradas, y se ven las solicitudes de afiliación.
Usuarios	Permite anexar un representante a una empresa registrada, anexar otro empleado, cambiar sus claves personales, consultar los clientes, consultar sus comentarios y visitas o sesiones.
Presupuestos	Permite manejar los datos que corresponden a los presupuestos elaborados, tanto al registrar uno nuevo, como consultar, modificar (eliminar o anexar ítems del presupuesto, cambiar datos del encabezado del mismo) y anular y dar por ejecutados los ya ingresados.
Facturas	Permite manejar los datos correspondientes a las facturas elaboradas, tanto al registrar una nueva factura, como consultar, modificar (eliminar o anexar ítems de la factura, cambiar datos del encabezado de la misma), dar por canceladas o pagadas y anular alguna ya ingresada.
Servicios	Permite registrar, consultar, modificar y eliminar datos relacionados con los servicios y actividades que presta u ofrece la empresa HIDROCONS, C. A. a las empresas clientes.
Repuestos	Permite gestionar los procesos relacionados con lo que tiene que ver con repuestos que HIDROCONS, C. A. tiene a la venta, como registrar, consultar, modificar y eliminar dichos repuestos.
Ayuda	Módulo de ayuda al servicio del empleado, en caso de que este tenga alguna duda acerca del uso de algún proceso del sistema.

Tabla 4.6. SiteView Privado "Principal para Clientes de HIDROCONS"

Nombre del SiteView	**PRINCIPAL PARA CLIENTES DE HIDROCONS, C. A.**	
Descripción	Es la primera vista que se le muestra al cliente cuando este ingresa como usuario válido al sistema. En esta vista se muestran las opciones a las que puede acceder y que les son permitidas a dicho cliente en cuanto a administración de empresa, de clientes, de presupuestos y facturas, así como de servicios que presta la empresa y repuestos que esta misma pone en venta.	
Grupo de Usuarios	Clientes que sean Representantes Autorizados de las Empresas Afiliadas al sistema SIADCON	
Casos de Uso Asociados	➢ Administrar Empresas ➢ Administrar Presupuestos ➢ Administrar Servicios	➢ Administrar Usuarios ➢ Administrar Facturas ➢ Administrar Repuestos

Tabla 4.7. Mapa del SiteView Privado "Principal para Clientes de HIDROCONS"

Nombre del Área	**Descripción del área**
Empresas	El cliente tiene la facultad de consultar y modificar datos de su empresa, así como dar de baja a la empresa a la cual pertenece.
Usuarios	Permite al cliente consultar sus datos registrados en el sistema, para modificarlos o darse de baja.
Presupuestos	Permite consultar los presupuestos elaborados para la compañía a la cual pertenece.
Facturas	Permite consultar las facturas elaboradas para la compañía a la cual representa.
Servicios	Le da la potestad al cliente de consultar los servicios o actividades que presta y ofrece la empresa HIDROCONS, C. A. a sus empresas inscritas en general.
Repuestos	Permite gestionar todos los procesos relacionados con lo que tiene que ver con la consulta de los repuestos que HIDROCONS, C. A. tiene a la venta, así como la potestad de efectuar solicitudes de repuestos destinados a facturación futura.
Ayuda	Módulo de ayuda al servicio del cliente, en caso de que este tenga alguna duda acerca del uso de algún proceso del sistema.

4.5. Diseño

El flujo de trabajo de diseño toma como entrada el análisis previamente realizado de los requisitos. En el diseño se realizarán actividades dedicadas y cuidadosas que darán como resultado los componentes que se implementarán en la siguiente fase. Entre ellos tenemos el diseño de prototipos de interfaces, diseño de la base de datos, y el diseño de la arquitectura de software y de hardware.

Un prototipo de interfaz representa un ejemplar original o primer molde en que se fabrica la interfaz visual definitiva, por ello se denomina prototipo (una representación limitada del diseño de un producto que permite a las partes responsables de su creación experimentar, probarlo en situaciones reales y explorar su uso). En lo que respecta a la base de datos, se comenzará con el diseño de las tablas que intervienen en el sistema, con la descripción de todos y cada uno de sus campos, para luego obtener sus respectivos Scripts generadores. En cuanto a la arquitectura del software se especifica el sistema manejador de bases de datos que se utilizará para este proyecto, la arquitectura J2EE como especificaciones estadarizadas para desarrollar aplicaciones empresariales en ambiente Web, y los modelos de hipertexto de WebML para observar el comportamiento visual en lo que a la navegabilidad de la aplicación se refiere. Mientras que en lo que corresponde a la arquitectura del hardware, en esta sección se detallan las especificaciones indicadas en la fase de inicio en cuanto a este punto o tema.

4.5.1. Prototipos de interfaces

El prototipo de una interfaz es como un boceto de lo que verdaderamente serán las futuras interfaces de usuario definitivas del sistema. Es recomendable realizar

estos esquemas una vez se hayan descritos todos los casos de uso del sistema, en otras palabras, que se hayan definido la mayoría de los requerimientos, los usuarios que interactuarán con el sistema y la forma en que interactuarán con el mismo, respectivamente.

- **Prototipo de la Interfaz de "Inicio de Sesión"**: se considera al caso de uso Iniciar Sesión Usuario, que se invoca cada vez que un usuario registrado, desea empezar a interactuar con el sistema. Es una interfaz que presenta un espacio donde se puede colocar el logotipo que identifica a la empresa, además de un pie de página informativo. No sólo incluye los cuadros de texto receptores de información de los usuarios registrados, como son el nombre de usuario y su contraseña, sino que también posee un menú desplegable horizontal que puede ser llenado en un futuro con diversas opciones y páginas informativas que pueden ser vista por cualquier visitante, sea usuario registrado o no, de manera que pueda tener una idea general de lo que es la compañía HIDROCONS, C. A. Igualmente posee un espacio más amplio situado al lado derecho del área de autentificación donde se puede colocar fotos, texto, formularios, en fin, cualquier dato que concuerde con la opción seleccionada en el menú desplegable. Véase la figura 4.14 de esta interfaz.

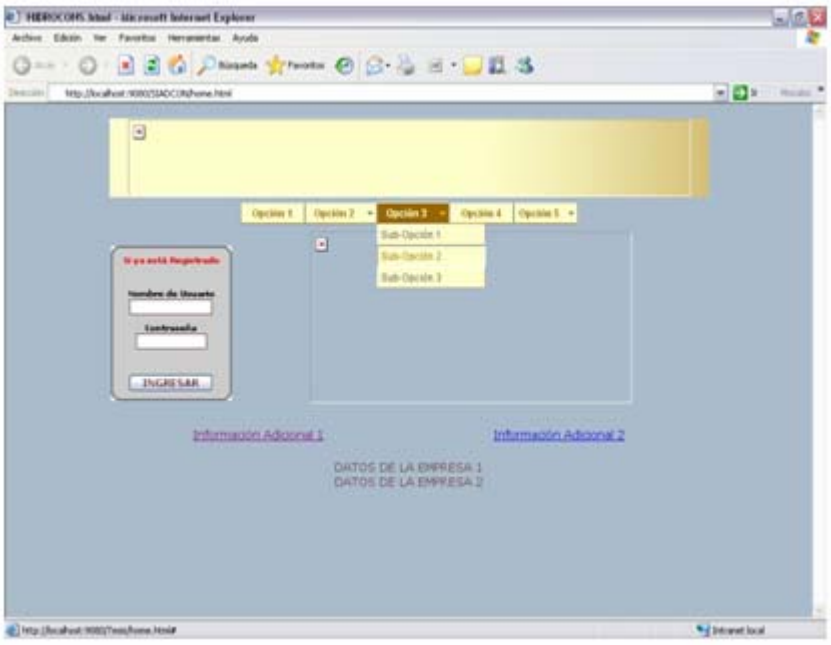


Figura 4.14. Prototipo de interfaz "Principal Externa" o de "Inicio de Sesión"
Fuente: SIADCON, 2009

- **Prototipo de la "Interfaz Principal"**: esta interfaz posee, al igual que la interfaz externa, un marco superior que representa el encabezado, con un logotipo de la empresa HIDROCONS, C. A. acompañado del logo del sistema SIADCON, el izquierdo muestra un menú desplegable vertical que le permite al usuario navegar por las diferentes opciones, bien sea Empleado o Cliente de HIDROCONS, C. A., y el derecho, donde se observa la información según la opción seleccionada en el menú desplegable vertical. Véase figura 4.15.

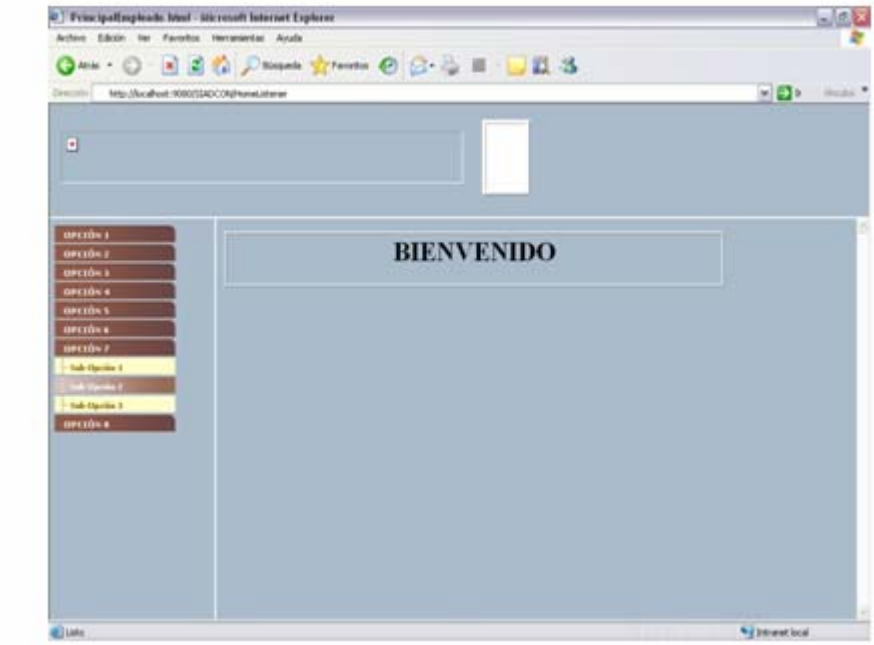


Figura 4.15. Prototipo de interfaz "Principal para Usuarios del sistema"
Fuente: SIADCON, 2009

4.5.2. Diseño de la base de datos

El almacenamiento de la información ingresada a través de la aplicación en desarrollo implica el uso de una base de datos. Para ello se implementó el modelo relacional, mediante asociaciones de entidades e interrelaciones, el cual sintetiza eficazmente la implementación de la base de datos, ya que elimina la redundancia de datos.

- **Identificación de Tablas:** durante el proceso de definición de la estructura de la Base de Datos se consiguieron 12 tablas que conformarán el sistema de datos del software, las cuales se detallan en las siguientes tablas descriptivas. En las mismas los campos clave de las tablas, definidos de acuerdo a las relaciones conseguidas en el diseño mismo, son señalados mediante un fondo

de celda amarillo, un formato de letra en **negrita** y un dibujo de una llave  que hace suponer que representa una "clave principal".

Tabla 4.8. Tabla EMPRESAS y sus respectivos campos

Nombre Tabla	**EMPRESAS**	
Descripción Tabla	Esta tabla encierra todos los datos propios de las empresas clientes de HIDROCONS, C. A. afiliadas a este sistema.	
Datos de los Campos		
Nombre Campo	**Tipo Dato**	**Descripción**
CodEmpresa	Texto(6)	Código que le asigna el sistema a la empresa.
NombreEmpresa	Texto(70)	Nombre fiscal de la empresa.
RIFEmpresa	Texto(15)	Registro de Información Fiscal de la empresa.
NITEmpresa	Texto(15)	Número de Información Tributaria de la empresa. (No imprescindible)
FechaRegEmpresa	Texto(10)	Fecha en que se registró la empresa en el sistema.
DireccionEmpresa	Texto(200)	Dirección fiscal de la empresa.
CiudadEmpresa	Texto(50)	Ciudad principal donde se encuentra ubicada.
EstadoEmpresa	Texto(15)	Estado donde se encuentra ubicada
CodTelefEmpresa1	Texto(5)	Código del teléfono principal de la empresa. Preferiblemente y en la medida de lo posible que sea fijo.
TelefonEmpresa1	Texto(8)	Teléfono principal para poder ubicar a la empresa. Preferiblemente fijo.
CodTelefEmpresa2	Texto(5)	Código del teléfono secundario de la empresa. (No imprescindible)
TelefonEmpresa2	Texto(8)	Teléfono fijo o móvil secundario de la empresa. (No imprescindible)
CorreoEmp	Texto(50)	Correo electrónico propio de la empresa al

resa		cual poder enviar información en algún momento.
SituacionEmpresa	Texto(10)	Indica si la empresa es clienta activa del sistema. Toma tres valores: EN ESPERA, ACTIVA e INACTIVA.

Tabla 4.9. Tabla USUARIOS y sus respectivos campos

Nombre Tabla	**USUARIOS**	
Descripción Tabla	Esta tabla encierra todos los datos de los representantes de las empresas clientes de HIDROCONS, C. A., y que se encuentran registradas en este sistema.	
Datos de los Campos		
Nombre Campo	**Tipo Dato**	**Descripción**
CodUsuario	Texto(6)	Código que le asigna el sistema al usuario al momento de registrarse en el mismo.
CodEmpresa	Texto(6)	Código que le asigna el sistema a la empresa, y que permite asociarla con el usuario en cuestión.
NombresUsuario	Texto(60)	Nombres completos del usuario.
ApellidosUsuario	Texto(60)	Apellidos completos del usuario.
CedulaUsuario	Texto(15)	Nacionalidad y cédula de identidad del usuario.
FechaRegUsuario	Texto(10)	Fecha en la cual se registró el usuario en el sistema.
DireccionUsuario	Texto(200)	Dirección lo más precisa posible del empleado, incluyendo ciudad y estado. Para el cliente no aplica este campo.
CodTelefUsuario1	Texto(5)	Código del teléfono principal para poder contactar al usuario que se está afiliando.
TelefonUsuario1	Texto(8)	Teléfono principal que permite contactar al usuario que se está afiliando.
CodTelefUs	Texto(5)	Código de algún teléfono secundario del

uario2		usuario. (No imprescindible).
TelefUsuario2	Texto(8)	Teléfono secundario del usuario. (No imprescindible).
CargoUsuario	Texto(70)	Cargo que desempeña el empleado o persona solicitante en la empresa.
CorreoUsuario	Texto(50)	Correo electrónico propio del usuario, por medio del cual poder enviarle alguna información.
UsernameUsuario	Texto(20)	Nombre de usuario (login) asignado o escogido por el usuario.
ClaveUsuario	Texto(20)	Contraseña o clave (password) asignada o escogida por el usuario.
TipoUsuario	Texto(12)	Indica qué tipo de usuario es el que está inscrito en el sistema. Toma tres valores: DESCONOCIDO, EMPLEADO y CLIENTE.

Tabla 4.9. Tabla USUARIOS y sus respectivos campos (Continuación)

Datos de los Campos		
Nombre Campo	**Tipo Dato**	**Descripción**
SituacionUsuario	Texto(10)	Indica en cuál estado o situación se encuentra el cliente en el sistema. Toma tres valores: ACTIVO, INACTIVO y EN ESPERA.
EstaConectado	Texto(5)	Expresa si el usuario en cuestión está o no actualmente conectado al sistema SIADCON.

Tabla 4.10. Tabla SESIONES y sus respectivos campos

Nombre Tabla	**SESIONES**	
Descripción Tabla	Esta tabla posee todos los registros de los ingresos y visitas llevadas a cabo por todos los diversos usuarios del sistema.	
Datos de los Campos		
Nombre Campo	**Tipo Dato**	**Descripción**
 **CodSesion**	Texto(6)	Código que le asigna el sistema a la visita, sesión o

			ingreso del usuario en el sistema.
	CodUsuario	Texto(6)	Código que le asigna el sistema al usuario.
FechaSesion		Texto(10)	Fecha en la cual el usuario ingresó autentificándose al sistema.
HoraEntrada		Texto(12)	Hora en que el usuario ingresó al sistema.
HoraSalida		Texto(12)	Hora en que el usuario abandonó el sistema.
AccionesSesion		Texto(1500)	Expresa las acciones que realizó el usuario durante el tiempo que estuvo conectado al sistema con la finalidad de hacer un seguimiento a sus actos. Sólo se registran acciones importantes como registro, modificación y eliminación de datos, mas no de consulta.

Tabla 4.11. Tabla COMENTARIOS y sus respectivos campos

Nombre Tabla	**COMENTARIOS**
Descripción Tabla	Esta tabla presenta los datos de los comentarios y sugerencias que envían los usuarios, tanto externos al sistema (que no se encuentran registrados en el mismo) como los Clientes de HIDROCONS inscritos formalmente.

Tabla 4.11. Tabla COMENTARIOS y sus respectivos campos (Continuación)

Datos de los Campos		
Nombre Campo	**Tipo Dato**	**Descripción**
CodComentario	Texto(6)	Código que le asigna el sistema al comentario.
CodSesi	Texto(6)	Código que le asigna el sistema a la visita,

on	sesión o ingreso del usuario en el sistema.
Fecha Comentario	Texto(10)	Campo que representa la fecha en que el usuario envió el comentario.
Hora Comentario	Texto(12)	Indica la hora exacta en la que se envió el comentario o sugerencia en cuestión.
RemiteComentario	Texto(6)	Representa el código de usuario del actor del sistema que envió el comentario.
CorreoComentario	Texto(50)	Correo de la persona que registró su comentario.
AsuntoComentario	Texto(100)	Idea principal del comentario registrado.
Comentario	Texto(1500)	Comentarios y sugerencias.
SituacComentario	Texto(6)	Indica si la los comentario están recién registrados o no. Toma dos valores: NUEVO y LEÍDO.

Tabla 4.12. Tabla SERVICIOS y sus respectivos campos

Nombre Tabla	**SERVICIOS**	
Descripción Tabla	Esta tabla incluye los datos de las diversas actividades y servicios que presta la empresa HIDROCONS, C. A.	
Datos de los Campos		
Nombre Campo	**Tipo Dato**	**Descripción**
 **CodServicio**	Texto(6)	Código que le asigna el sistema al servicio.
 **CodFactura**	Texto(6)	Código de la factura de la cual proviene el servicio que se describe como ofrecimiento al público. En caso de que no exista factura asociada a esta obra, este campo tomará el valor "000000" previamente ingresado al sistema.
NombreServicio	Texto(500)	Nombre completo del

		servicio. Representa un resumen o título del mismo.
DescripcnServicio	Texto(1200)	Descripción del servicio o actividad que se presta. Es un poco más detallada, específica y extensa que el campo anterior.

Tabla 4.12. Tabla SERVICIOS y sus respectivos campos (Continuación)

Datos de los Campos		
Nombre Campo	**Tipo Dato**	**Descripción**
SituacionServicio	Texto(10)	Indica si la actividad está actualmente en servicio, es decir, si la empresa la está ofreciendo al público. Toma dos valores: ACTIVO e INACTIVO.
FotoServicio1	Texto(140)	Dirección o URL del archivo contentivo de la primera foto seleccionada para detallar gráficamente el servicio en cuestión.
FotoServicio2	Texto(140)	Dirección o URL del archivo contentivo de la segunda foto seleccionada para detallar gráficamente el servicio en cuestión.
FotoServicio3	Texto(140)	Dirección o URL del archivo contentivo de la tercera foto seleccionada para detallar gráficamente el servicio en cuestión.
FotoServicio4	Texto(140)	Dirección o URL del archivo contentivo de la cuarta foto seleccionada para detallar gráficamente el servicio en cuestión.
FotoServicio5	Texto(140)	Dirección o URL del

		archivo contentivo de la quinta foto seleccionada para detallar gráficamente el servicio en cuestión.
ComentarioFoto1	Texto(300)	Expresa una explicación o comentario detallado de la primera foto seleccionada para este servicio.
ComentarioFoto2	Texto(300)	Expresa una explicación o comentario detallado de la segunda foto seleccionada para este servicio.
ComentarioFoto3	Texto(300)	Expresa una explicación o comentario detallado de la tercera foto seleccionada para este servicio.
ComentarioFoto4	Texto(300)	Expresa una explicación o comentario detallado de la cuarta foto seleccionada para este servicio.
ComentarioFoto5	Texto(300)	Expresa una explicación o comentario detallado de la quinta foto seleccionada para este servicio.

Tabla 4.13. Tabla REPUESTOS y sus respectivos campos

Nombre Tabla	**REPUESTOS**
Descripción Tabla	Esta tabla incluye los datos de los diversos repuestos que pone a la venta la empresa HIDROCONS, C. A.

Tabla 4.13. Tabla REPUESTOS y sus respectivos campos (Continuación)

Datos de los Campos			
Nombre Campo		**Tipo Dato**	**Descripción**
	**CodRepuesto**	Texto(6)	Código que le es asignado al repuesto por parte del sistema.

TipoRepuesto	Texto(25)	Indica de cuál maquinaria es el repuesto, si está a la venta al público o si es un artículo consumible no disponible a la venta sino de uso complementario para los servicios o actividades que presta la empresa. Sus valores son: ACCESORIOS, BOMBAS SUMERGIBLES, NINGUNO, DISPENSADORES, y OTROS.
SubTipoRepuesto	Texto(25)	Toma diversos valores según el tipo de repuesto seleccionado. Si es un repuesto de tipo: • ACCESORIOS: son los únicos que están a la venta al público, y pueden ser vistos por cualquier usuario, tanto de tipo Empleado, así como Cliente registrado o externo. Los valores que toma este campo son Eléctrico, Electrónico e Hidráulico. • BOMBAS SUMERGIBLES: toma valores de las marcas de las bombas sumergibles más usadas en la empresa, como FE PETRO y RED JACKET. • DISPENSADORES: toma valores de las

		marcas de los dispensadores para los cuales están diseñados estos repuestos, tales como WAYNE, GILBARCO H-111 y LEGACY. • OTROS: se refiere los repuestos que no entran en ninguno de los tipos de repuestos descritos anteriormente. Sus opciones son CONSUMIBLE, ELECTRICIDAD, PLOMERÍA, SISTEMA CONTRA INCENDIO y OTROS.
NombreRepuesto	Texto(70)	Nombre completo del repuesto a ingresar, incluyendo la Marca, el Tipo, el Modelo, el Serial del mismo, etc.

Tabla 4.13. Tabla REPUESTOS y sus respectivos campos (Continuación)

Datos de los Campos		
Nombre Campo	**Tipo Dato**	**Descripción**
MedidaRepuesto	Texto(5)	Indica la presentación o medida del repuesto. Algunos valores que toma son: UN, PG, JUEGO, ML, M2, M3, LTS, GL.
PrecioRepuesto	Flotante	Precio actualizado del repuesto.

FotoRepuesto	Texto(200)	Dirección o URL del archivo contentivo de la foto del repuesto.
SituacionRepuesto	Texto (10)	Indica si el repuesto está actualmente en venta. Toma dos valores: ACTIVO e INACTIVO.

Tabla 4.14. Tabla PRESUPUESTOS y sus respectivos campos

Nombre Tabla		**PRESUPUESTOS**	
Descripción Tabla		Esta tabla presenta los datos de identificación de los presupuestos elaborados por trabajos y obras a efectuarse.	
Datos de los Campos			
Nombre Campo		**Tipo Dato**	**Descripción**
	**CodPresupuesto**	Texto(6)	Referencia o número del presupuesto registrado.
	**CodEmpresa**	Texto(6)	Código que le asigna el sistema a la empresa.
FechaPresupuesto		Texto(10)	Fecha de elaboración del presupuesto.
SituacPresupuesto		Texto(10)	Indica situación del presupuesto en cuestión. Toma los valores siguientes: EN ESPERA, APROBADO, ANULADO, RECHAZADO y EJECUTADO.
ObraPresupuesto		Texto(500)	Describe un resumen de la obra del trabajo a efectuar.
RazonPresupuesto		Texto(3000)	Comentarios sobre el presupuesto en cuestión.

Tabla 4.15. Tabla DETALLES_PRESUPUESTO y sus respectivos campos

Nombre Tabla	**DETALLES_PRESUPUESTO**
Descripción Tabla	Esta tabla incluye los registros de las diversas partidas, renglones o ítems que conforman los presupuestos elaborados por la empresa HIDROCONS, C. A.

Tabla 4.15. Tabla DETALLES_PRESUPUESTO y sus campos (Continuación)

Datos de los Campos			
Nombre Campo		**Tipo Dato**	**Descripción**
	**CodDetallePres**	Texto(6)	Código asignado al ítem del presupuesto.
	**CodPresupuesto**	Texto(6)	Referencia o número del presupuesto registrado al que pertenece este ítem o detalle.
ItemPresupuesto		Texto(6)	Indica el tipo de ítem del presupuesto (ningun: cualquier ítem de comentario, que no posee cantidad ni precio; activd: ítem que describe una trabajo de mano de obra, o código del repuesto: el código del repuesto seleccionado).
MedidaDetallePres		Texto(5)	Unidad de medida del ítem del presupuesto. Toma valores tales como: UN, PG, JUEGO, ML,

		M2, M3, LTS, GL.
DescripcDetallPres	Texto(1500)	Descripción propia y completa del ítem del presupuesto.
CantidDetallePres	Flotante	Cantidad de unidades del ítem en cuestión. Si la medida del ítem es "PG", este campo toma el valor de uno (1).
PrecioDetallePres	Flotante	Precio unitario asignado al ítem o renglón del presupuesto.
FormatoDetallePres	Texto(10)	Indica la presentación y formato del ítem. Sus valores son: NORMAL, NEGRITA y CURSIVA.

Tabla 4.16. Tabla FACTURAS y sus respectivos campos

Nombre Tabla		**FACTURAS**	
Descripción Tabla		Esta tabla presenta los datos de identificación de las facturas elaboradas por trabajos realizados y ventas de repuestos.	
Datos de los Campos			
Nombre Campo		**Tipo Dato**	**Descripción**
	**CodFactura**	Texto(6)	Número de la factura ingresada.
	**CodEmpresa**	Texto(6)	Código de identificación que le asigna el sistema a la empresa al momento de

			registrarse.
LugarFactura		Texto(35)	Lugar de emisión de la factura.
FechaFactura		Texto(10)	Fecha de elaboración de la factura.

Tabla 4.16. Tabla FACTURAS y sus respectivos campos (Continuación)

Datos de los Campos		
Nombre Campo	**Tipo Dato**	**Descripción**
SituacionFactura	Texto(12)	Indica la situación de la factura. Toma tres valores: ANULADA, POR CANCELAR y CANCELADA.
IVAFactura	Flotante	Porcentaje del Impuesto al Valor Agregado.
FormaPagoFactura	Texto(10)	Indica forma de pago de la factura. Toma cinco valores: EFECTIVO, CHEQUE, DÉBITO, CRÉDITO y OTRO.
OtraFormaPago	Texto(20)	Si el campo forma de pago está presente la opción "OTRO", he aquí los valores respectivos: CRÉDITO A 30 DÍAS, A 60 DÍAS, A 90 DÍAS.

Tabla 4.17. Tabla DETALLES_FACTURA y sus respectivos campos

Nombre Tabla	**DETALLES_FACTURA**	
Descripción Tabla	Esta tabla incluye los registros de las partidas o ítems que conforman las facturas elaboradas por HIDROCONS, C. A.	
Datos de los Campos		
Nombre Campo	**Tipo Dato**	**Descripción**
CodDetalleFact	Texto(6)	Código asignado por el sistema al ítem de la factura.
CodFactura	Texto(6)	Número de la factura ingresada.

ItemFactura	Texto(6)	Indica el tipo de ítem de factura (ninguno, actividad o repuesto).
MedidaDetalleFact	Texto(5)	Unidad de medida del ítem de la factura. Toma valores tales como: UN, PG, JUEGO, ML, M2, M3, LTS, GL.
DescripcDetallFact	Texto(1500)	Descripción propia del ítem de la factura.
CantidDetalleFact	Flotante	Cantidad de unidades del ítem. Si la medida del ítem es "PG", este toma el valor de uno (1).
PrecioDetalleFact	Flotante	Precio unitario asignado al ítem de la factura.
FormatoDetalleFact	Texto(10)	Indica la presentación y formato del ítem. Sus valores son: NORMAL, NEGRITA y CURSIVA.

Tabla 4.18. Tabla SOLICITUDES_REPUESTOS y sus respectivos campos

Nombre Tabla		**SOLICITUDES_REPUESTOS**	
Descripción Tabla		Esta tabla encierra toda la información de las solicitudes de repuestos realizadas por los clientes a la empresa HIDROCONS, C. A.	
Datos de los Campos			
Nombre Campo		**Tipo Dato**	**Descripción**
	**CodSolicitud**	Texto(6)	Código que le asigna el sistema a la solicitud.
	**CodEmpresa**	Texto(6)	Código que le asigna el sistema a la empresa.
FechaSolicitud		Texto(10)	Fecha en que la solicitud fue realizada.
SituacionSolicitud		Texto(12)	Indica la situación de la solicitud. Toma tres valores: EN ESPERA, APROBADA y RECHAZADA.
ComentarSolicitud		Texto(3000)	Comentarios sobre la solicitud en cuestión.

Tabla 4.19. Tabla REPUESTOS_SOLICITADOS y sus respectivos campos

Nombre Tabla		REPUESTOS_SOLICITADOS	
Descripción Tabla		Esta tabla detalla todos y cada uno de los repuestos solicitados observados en una solicitud de repuestos respectiva.	
Datos de los Campos			
Nombre Campo		Tipo Dato	Descripción
	**CodRepuesto**	Texto(6)	Código del repuesto que se está solicitando. Corresponde al mismo código presente en la tabla REPUESTOS.
	**CodSolicitud**	Texto(6)	Código que le asigna el sistema a la solicitud.
CantidRepSolicitad		Flotante	Cantidad a solicitar del repuesto en cuestión.
PrecioRepSolicitad		Flotante	Precio del repuesto a solicitar. Esto con la finalidad de que se conserve el precio al momento de hacer la solicitud en caso de que se actualice el precio en la tabla REPUESTOS.
SituacRepSolicitad		Texto(12)	Indica la situación del repuesto solicitado. Toma tres valores: EN ESPERA, APROBADO y RECHAZADO.

- **Scripts de la Base de Datos:** La herramienta modeladora de Base de Datos usada para crear las sentencias SQL que darán origen a las tablas previamente mencionadas y descritas se denomina **Azzuri Clay**, la cual corre como un Plug-in dentro de la aplicación conocida como ECLIPSE, otro entorno de desarrollo integrado de código abierto multiplataforma en lenguaje Java diseñado por IBM.

A continuación se describe los scripts que generan las tablas detalladas anteriormente, para ser implantadas en el SMBDR IBM DB2 Versión 9 a utilizar,

donde cada uno de ellos emplea la sentencia “CREATE TABLE” seguido del nombre de la tabla a crear, acompañado por sus respectivos campos en orden de aparición y colocación dentro de la tabla, así como el tipo de dato que impera en ese campo, bien sea cadena de caracteres corta (CHAR), cadena de caracteres larga (VARCHAR), incluyendo la longitud de cada uno de ellos, y punto flotante (FLOAT) entre otros tipos de datos. También se indica cuales campos representan tanto clave primaria (PRIMARY KEY) como claves foráneas (FOREIGN KEY) en la tabla.

```
CREATE TABLE H.EMPRESAS (
      CodEmpresa CHAR(6) NOT NULL
    , NombreEmpresa CHAR(70)
    , RIFEmpresa CHAR(15)
    , NITEmpresa CHAR(15)
    , FechaRegEmpresa CHAR(10)
    , DireccionEmpresa CHAR(200)
    , CiudadEmpresa CHAR(50)
    , EstadoEmpresa CHAR(15)
    , CodTelefEmpresa1 CHAR(5)
    , TelefonEmpresa1 CHAR(8)
    , CodTelefEmpresa2 CHAR(5)
    , TelefonEmpresa2 CHAR(8)
    , CorreoEmpresa CHAR(50)
    , SituacionEmpresa CHAR(10)
    , PRIMARY KEY (CodEmpresa)
);
CREATE TABLE H.USUARIOS (
      CodUsuario CHAR(6) NOT NULL
    , CodEmpresa CHAR(6) NOT NULL
    , NombresUsuario CHAR(60)
    , ApellidosUsuario CHAR(60)
    , CedulaUsuario CHAR(15)
    , FechaRegUsuario CHAR(10)
    , DireccionUsuario CHAR(200)
    , CodTelefUsuario1 CHAR(5)
    , TelefonUsuario1 CHAR(8)
    , CodTelefUsuario2 CHAR(5)
    , TelefonUsuario2 CHAR(8)
    , CargoUsuario CHAR(70)
```

```
    , CorreoUsuario CHAR(50)
    , UsernameUsuario CHAR(20)
    , ClaveUsuario CHAR(20)
    , TipoUsuario CHAR(12)
    , SituacionUsuario CHAR(10)
    , EstaConectado CHAR(5)
    , PRIMARY KEY (CodUsuario)
    , FOREIGN KEY (CodEmpresa)  REFERENCES H.EMPRESAS (CodEmpresa));


CREATE TABLE H.REPUESTOS (
      CodRepuesto CHAR(6) NOT NULL
    , TipoRepuesto CHAR(25)
    , SubTipoRepuesto CHAR(25)
    , NombreRepuesto CHAR(70)
    , MedidaRepuesto CHAR(5)
    , PrecioRepuesto FLOAT
    , FotoRepuesto CHAR(200)
    , SituacionRepuesto CHAR(10)
    , PRIMARY KEY (CodRepuesto));


CREATE TABLE H.SOLICITUDES_REPUESTOS (
      CodSolicitud CHAR(6) NOT NULL
    , CodEmpresa CHAR(6) NOT NULL
    , FechaSolicitud CHAR(10)
    , SituacionSolicitud CHAR(12)
    , ComentarSolicitud VARCHAR(3000)
    , PRIMARY KEY (CodSolicitud)
    , FOREIGN KEY (CodEmpresa)
            REFERENCES H.EMPRESAS (CodEmpresa));
CREATE TABLE H.REPUESTOS_SOLICITADOS (
      CodRepuesto CHAR(6) NOT NULL
    , CodSolicitud CHAR(6) NOT NULL
    , CantidRepSolicitad FLOAT
    , PrecioRepSolicitad FLOAT
    , SituacRepSolicitad CHAR(12)
    , PRIMARY KEY (CodRepuesto, CodSolicitud)
    , FOREIGN KEY (CodSolicitud)
            REFERENCES H.SOLICITUDES_REPUESTOS (CodSolicitud)
    , FOREIGN KEY (CodRepuesto)
            REFERENCES H.REPUESTOS (CodRepuesto)
);
```

```
CREATE TABLE H.SESIONES (
      CodSesion CHAR(6) NOT NULL
    , CodUsuario CHAR(6) NOT NULL
    , FechaSesion CHAR(10)
    , HoraEntrada CHAR(12)
    , HoraSalida CHAR(12)
    , AccionesSesion VARCHAR(1500)
    , PRIMARY KEY (CodSesion)
    , FOREIGN KEY (CodUsuario)
            REFERENCES H.USUARIOS (CodUsuario)
);



CREATE TABLE H.COMENTARIOS (
      CodComentario CHAR(6) NOT NULL
    , CodSesion CHAR(6) NOT NULL
    , FechaComentario CHAR(10)
    , HoraComentario CHAR(12)
    , RemiteComentario CHAR(6)
    , CorreoComentario CHAR(50)
    , AsuntoComentario CHAR(100)
    , Comentario VARCHAR(1500)
    , SituacComentario CHAR(6)
    , PRIMARY KEY (CodComentario)
    , FOREIGN KEY (CodSesion)
            REFERENCES H.SESIONES (CodSesion)
);
CREATE TABLE H.FACTURAS (
      CodFactura CHAR(6) NOT NULL
    , CodEmpresa CHAR(6) NOT NULL
    , LugarFactura CHAR(35)
    , FechaFactura CHAR(10)
    , SituacionFactura CHAR(12)
    , IVAFactura FLOAT
    , FormaPagoFactura CHAR(10)
    , OtraFormaPago CHAR(20)
    , PRIMARY KEY (CodFactura)
    , FOREIGN KEY (CodEmpresa) REFERENCES H.EMPRESAS (CodEmpresa));

CREATE TABLE H.DETALLES_FACTURA (
```

```
    CodDetalleFact CHAR(6) NOT NULL
  , CodFactura CHAR(6) NOT NULL
  , ItemFactura CHAR(6)
  , MedidaDetalleFact CHAR(5)
  , DescripcDetallFact VARCHAR(1500)
  , CantidDetalleFact FLOAT
  , PrecioDetalleFact FLOAT
  , FormatoDetalleFact CHAR(10)
  , PRIMARY KEY (CodDetalleFact, CodFactura)
  , FOREIGN KEY (CodFactura) REFERENCES H.FACTURAS (CodFactura));

CREATE TABLE H.SERVICIOS (
    CodServicio CHAR(6) NOT NULL
  , CodFactura CHAR(6) NOT NULL
  , NombreServicio VARCHAR(500)
  , DescripcnServicio VARCHAR(1200)
  , SituacionServicio CHAR(10)
  , FotoServicio1 VARCHAR(140)
  , FotoServicio2 VARCHAR(140)
  , FotoServicio3 VARCHAR(140)
  , FotoServicio4 VARCHAR(140)
  , FotoServicio5 VARCHAR(140)
  , ComentarioFoto1 VARCHAR(300)
  , ComentarioFoto2 VARCHAR(300)
  , ComentarioFoto3 VARCHAR(300)
  , ComentarioFoto4 VARCHAR(300)
  , ComentarioFoto5 VARCHAR(300)
  , PRIMARY KEY (CodServicio, CodFactura)
  , FOREIGN KEY (CodFactura) REFERENCES H.FACTURAS (CodFactura));
CREATE TABLE H.PRESUPUESTOS (
    CodPresupuesto CHAR(6) NOT NULL
  , CodEmpresa CHAR(6) NOT NULL
  , FechaPresupuesto CHAR(10)
  , SituacPresupuesto CHAR(10)
  , ObraPresupuesto VARCHAR(500)
  , RazonPresupuesto VARCHAR(3000)
  , PRIMARY KEY (CodPresupuesto)
  , FOREIGN KEY (CodEmpresa)
        REFERENCES H.EMPRESAS (CodEmpresa)
);

CREATE TABLE H.DETALLES_PRESUPUESTO (
```

```
    CodDetallePres CHAR(6) NOT NULL
  , CodPresupuesto CHAR(6) NOT NULL
  , ItemPresupuesto CHAR(6)
  , MedidaDetallePres CHAR(5)
  , DescripcDetallPres VARCHAR(1500)
  , CantidDetallePres FLOAT
  , PrecioDetallePres FLOAT
  , FormatoDetallePres CHAR(10)
  , PRIMARY KEY (CodDetallePres, CodPresupuesto)
  , FOREIGN KEY (CodPresupuesto)
        REFERENCES H.PRESUPUESTOS (CodPresupuesto)
);
```

De igual forma se deben de anexar los siguientes Scripts al momento de elaborar la base de datos, ya que los mismos permiten el buen desenvolvimiento del sistema en diversos casos excepcionales, debido a que controlan lo referente a comentarios y sugerencias remitidos por empresas y/o clientes que no se encuentran registrados en el sistema, es decir, mensajes provenientes de la interfaz externa del sistema, así como los datos de la empresa HIDROCONS, C. A. que igualmente debe estar registrada en la aplicación en función de los empleados pertenecen a la misma, que a la final constituyen usuarios del sistema, además de los servicios y obras que ofrece la empresa o que ésta haya realizado, pero que no se encuentran por diversas razones asociadas a factura alguna. Estos Scripts adicionales en cuestión son los que a seguir se describen:

```
INSERT INTO H.EMPRESAS VALUES ('ningun', 'EMPRESA NO REGISTRADA', '', '', '0000/00/00', 'PARA EMPRESAS QUE NO ESTÁN REGISTRADAS EN EL SISTEMA', '', '', '0000', '0000000', '', '', '', 'ACTIVA');

INSERT INTO H.EMPRESAS VALUES ('000000', 'HIDROCONS, C. A.', 'J-08018455-6', '0023725177', '2008/12/14', 'CALLE CUMANÁ N°. 21, SECTOR LA CARAQUEÑA', 'PUERTO LA CRUZ', 'ANZOÁTEGUI', '0281', '2636606', '', '', 'hidrocons@cantv.net', 'ACTIVA');

INSERT INTO H.USUARIOS VALUES ('000000', 'ningun', 'USUARIO', 'EXTERNO', 'X00000000', '0000/00/00', 'IDENTIFICA A LOS USUARIOS
```

EXTERNOS DEL SISTEMA, LOS QUE NO ESTÁN INSCRITOS EN EL MISMO', '0000', '0000000', '', '', 'NINGUNO', 'usuario_externo@desconocido.com', '', '', 'DESCONOCIDO', 'ACTIVO', 'FALSE');

INSERT INTO H.FACTURAS VALUES ('000000', '000000', 'FACTURA PARA SERVICIOS SIN FACTURA', '0000/00/00', 'POR CANCELAR', 0.0, '-', '-');

INSERT INTO H.SESIONES VALUES ('000000', '000000', '0000/00/00', 'AM 00:00:00', 'PM 00:00:00', 'REGISTRO PARA ASOCIAR COMENTARIOS PROVENIENTES DE USUARIOS EXTERNOS');

En la figura 4.16 se muestra el modelo relacional de la Base de Datos propio de esta aplicación donde se observan las diferentes tablas que interactúan en el sistema de base de datos escogido, cada una con sus respectivos nombres de tabla y con todos los campos denotados en las tablas anteriormente explicadas, y de las cuales se derivaron los diferentes Scripts que se describieron en líneas previas. Se puede observar que los campos que representan claves principales o foráneas se encuentran indicadas con un tipo de letra un poco más oscura. De igual modo se pueden detallar las diversas relaciones de multiplicidad existente entre determinadas tablas, enlazadas por medio de sus claves, como por ejemplo la que existe entre la tabla EMPRESA y la tabla USUARIOS, donde dicha relación expresa que una (1) Empresa tiene muchos (∞) Usuarios inscritos. Todo esto con la finalidad de diseñar una base de datos lo más compacta posible y con la menor redundancia de datos.

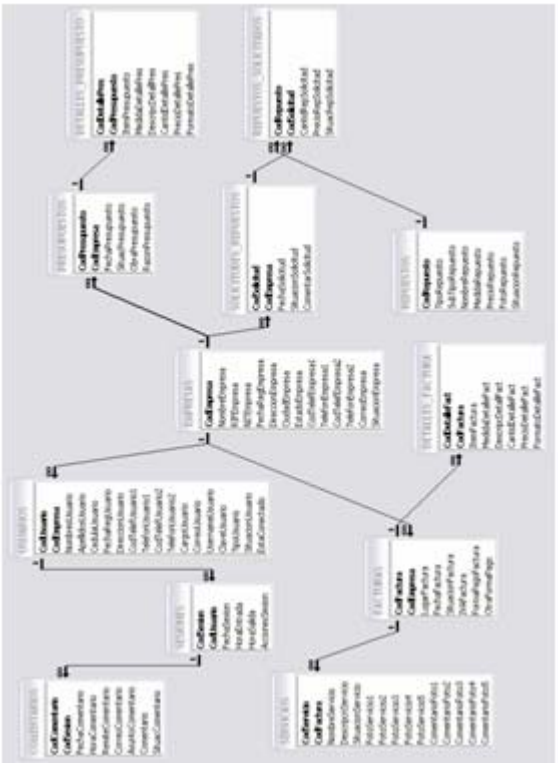


Figura 4.16. Modelo Relacional de la Base de Datos
Fuente: Propia, 2009

4.5.3. Arquitectura de software

La arquitectura candidata seleccionada en la fase de inicio se conserva en la etapa presente, sin embargo en esta parte del proyecto se ha de detallar y especificar un poco más la misma, obteniéndose una versión mejorada. En esta sección se explica entre otras cosas, las razones por las cuales se seleccionó para este proyecto un determinado sistema manejador de base de datos relacionales, se explica el estándar J2EE como arquitectura distribuida para desarrollar aplicaciones empresariales multinivel apta para ser empleada en este sistema en desarrollo, y finalmente se ahonda en el diseño de modelos de Hipertexto propios de WebML, clasificados en Modelos de Navegación y Modelos de Composición.

4.5.3.1. Sistema de base de datos

Con lo referente al diseño y administración de la base de datos a utilizar, en el mercado se encuentran disponibles diferentes tipos de Sistemas Manejadores de Bases de Datos Relacionales (RDBMS), de los cuales los más populares son Oracle, Microsoft SQL Server, MySQL, PostgreSQL y el que se va a emplear para este proyecto que es un sistema administrador de bases de datos conocido como IBM DB2 Versión 9, el cual tiene como características y propiedades que, además de ser un manejador de base de datos propietario, es un sistema que responde rápidamente a la demanda de transacciones en los periodos de mayor carga de trabajo, se amplía para acoger cantidades crecientes de información, posee las características de manejar pedidos de procesamiento de datos de forma exacta y eficiente, escalabilidad a medida que el negocio crece, y permite soporte para multiprocesamiento simétrico y procesadores masivamente paralelos, es decir, puede configurarse para correr en múltiples servidores y procesadores (maneja lo que se conoce como multihilos).

4.5.3.2. Visión de la arquitectura J2EE

El estándar conocido como J2EE (Java 2 Platform Enterprise Edition – Edición 2 en Java para Plataformas Empresariales) es un conjunto unificado de especificaciones proporcionadas por Sun Microsystems muy útiles para desarrollar aplicaciones empresariales distribuidas a través de un modelo de aplicaciones basado en componentes. Las mismas comprenden una colección de librerías que soportan del desarrollo de aplicaciones empresariales. Esta arquitectura se puede dividir en las siguientes categorías:

- ***Capa Cliente:*** se refiere a la interfaz de los diferentes tipos de usuarios que interactúan con la aplicación. El usuario puede ser un Cliente de HIDROCONS que accede a la aplicación a través de un Firewall, o un

Empleado de HIDROCONS que accede desde dentro del Firewall corporativo. Para efectos del presente proyecto, sólo estarán permitido el acceso a usuarios Web, tanto fuera como dentro del Firewall corporativo.

- ***Capa intermedia:*** proporciona servicios intermedios a cualquier tipo de usuario Web. Consta de los siguientes componentes:

  - **Lógica de Presentación**: se refiere a la formación de contenido dinámico producto de las solicitudes y respuestas de los diferentes usuarios en un instante determinado.

  - **Lógica del Negocio**: maneja las reglas, políticas y procesos del negocio al cual es destinado esta aplicación en cuestión. Permite que el sistema automatizado refleje y asemeje el comportamiento de los procesos administrativos de la empresa HIDROCONS, C. A. Esta capa posee todas las clases descritas previamente y que interactúan en el software.
  - **Servicios de Infraestructura**: proporciona funcionalidades adicionales requeridas por los componentes de la aplicación, como mensajería y transacciones.

- ***Capa de Datos EIS (Enterprise Information System – Sistema de Información Empresarial):*** proporciona los datos de la aplicación. Aquí se incluyen las bases de datos.

El diagrama 4.17, el cual explica las diversas capas comentadas y descritas anteriormente, es el siguiente:

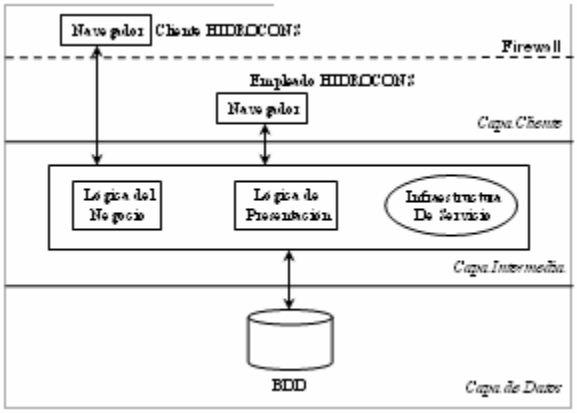


Figura 4.17. Nueva versión de la arquitectura de software candidata
Fuente: Guía de Estudiante del Curso de Java Empresarial Libro Nº. 1" de IBM. Código del Curso: CY760. Versión 4.0., 2006

4.5.3.3. Modelos de navegación WEBML

Un modelo de navegación expresa cómo las páginas, áreas y SiteViews son enlazadas para formar los hipertextos. Usando las tablas de SiteViews descritas anteriormente, se procede a realizar los diversos modelos de navegación que regirán el desplazamiento y/o movilidad que, de acuerdo a los privilegios y atribuciones designados según su tipo, el usuario puede realizar. Se observa que dichos diagramas presentan un aspecto de árbol invertido donde en la raíz se encuentra el usuario habilitado para recorrerlos. Se observan ciertos nodos en forma de carpeta de las cuales se despliegan un número definido de páginas Web, en las cuales el usuario en cuestión está en capacidad de llevar a cabo ciertos casos de uso que les son adjudicados. A continuación los modelos de navegación para este proyecto son:

- **Modelo de Navegación Público Externo para Usuarios en General**: muestra las páginas Web, grupos de páginas u opciones a las que puede acceder cualquier tipo de usuario, tanto externo como registrado en el sistema. El último caso que se observa en este diagrama es exclusivo sólo para usuarios autorizados y registrados en el sistema. Véase la figura 4.18.

- **Modelo de Navegación Privado para Empleados**: este diagrama muestra páginas a las cuales puede tener acceso cualquier Empleado de HIDROCONS activo y registrado en la aplicación. Del nodo denominado "Ayuda" se derivan muchas páginas Web alusivas a dicha sección, pero por cuestiones de espacio se resumen en unas pocas. Para ello véase la figura 4.19.

- **Modelo de Navegación Privado para Clientes**: los Clientes de HIDROCONS activos y autentificados pueden desplazarse mediante los diversos caminos descritos en este diagrama. Véase figura 4.20.

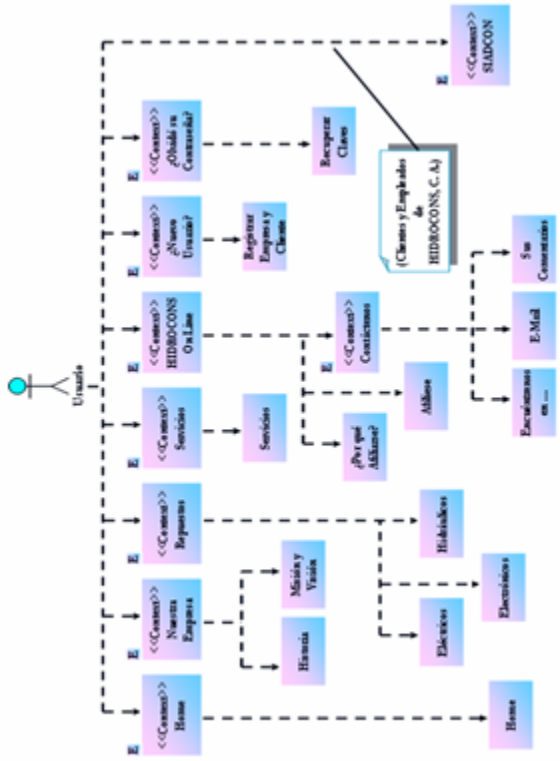


Figura 4.18. Modelo de Navegación "Principal Externo"
Fuente: Propia, 2009

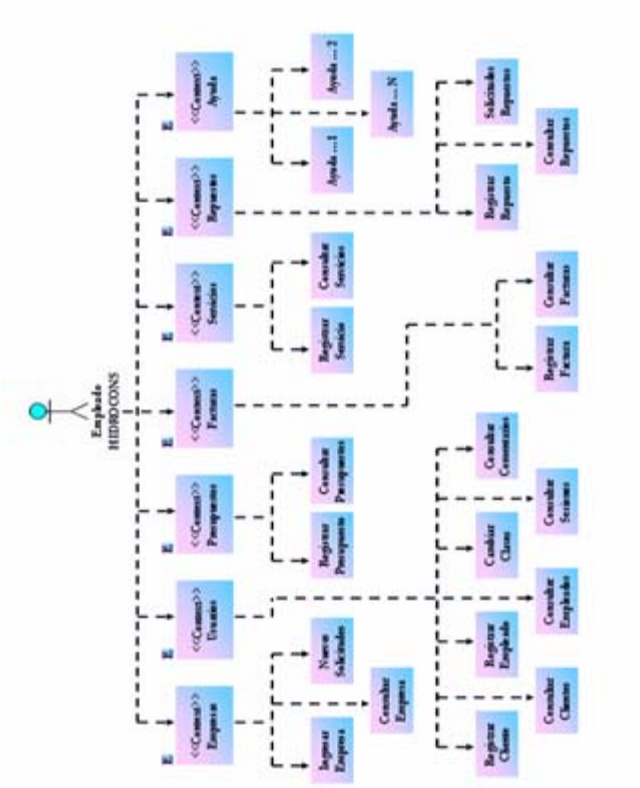


Figura 4.19. Modelo de Navegación "Principal para Empleados"
Fuente: Propia, 2009

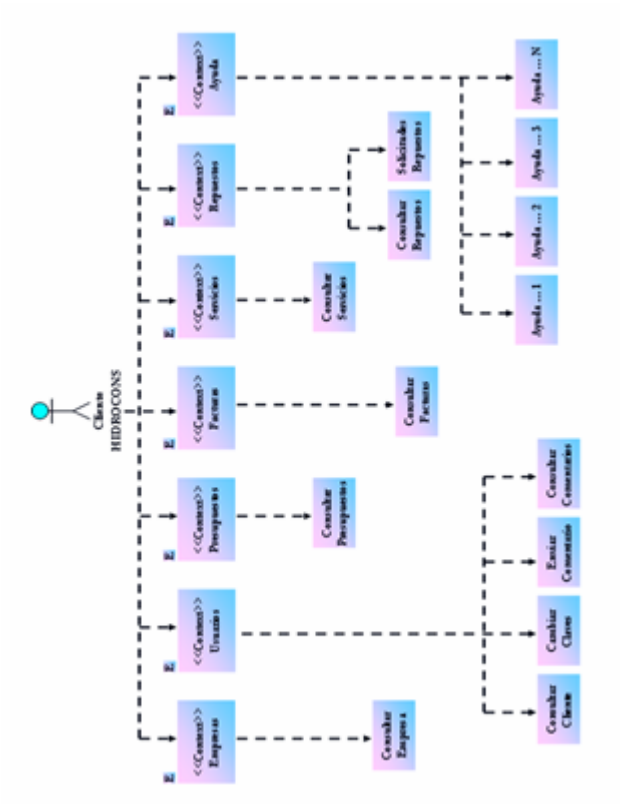


Figura 4.20. Modelo de Navegación "Principal para Clientes"
Fuente: Propia, 2009

4.5.3.4. Modelos de composición o de gestión de contenido WEBML

El modelo de navegación comentado anteriormente no es suficiente para expresar el manejo del contenido de las páginas de la aplicación, por lo que se utiliza otra extensión del modelo de hipertexto llamado modelo de Gestión de Contenido o modelo de Composición. Este modelo comprende el uso de operaciones, que sirven para expresar como se manejan los datos en el sistema, ya sea mediante el uso de bases de datos, variables globales, envío de E-Mails, acceso a sistemas, transacciones, salidas de sesiones, transporte de datos, etc. El modelo de gestión de contenido o de composición nos otorga una visión más completa del funcionamiento intrínseco de la aplicación Web. El modelo de composición en WebML es el encargado de mostrar al

usuario el contenido conceptual de las páginas que formarán parte de la aplicación Web a desarrollar, y lo hacen utilizando los SiteViews que conforman el sistema, las áreas por la que están formadas los SiteViews, las páginas que contiene cada área y por supuesto, el contenido de cada página utilizando los elementos que nos proporciona WebML.

El diseño de hipertextos es una de las actividades que consume más tiempo en esta fase de desarrollo de RUP. El mismo define cómo ha de ser el comportamiento del sistema en función de páginas Web, es decir, cómo será la navegación entre páginas interrelacionadas pertenecientes a un caso de uso específico del sitio Web de la aplicación. Durante esta actividad se diseñarán el conjunto de páginas que integrarán los SiteViews requeridos. En este tipo de diagramas se observan ciertas unidades denominadas unidades de contenido, que indican los procesos a los que son sometidos los datos: unidades de entrada de datos, unidades de consulta de datos, unidades de índice, unidades de creación, modificación y eliminación de datos, transacciones, entre otros.

A continuación se muestran algunos modelos de composición de ciertos casos de uso emblemáticos del software en construcción:

- **Diagrama de Composición o de Gestión de Contenido del Caso de Uso "Iniciar Sesión Usuario"**: este diagrama se detalla en la figura 4.21, y en dicho diagrama se puede observar el proceso que realizaría cualquier usuario, sea Empleado o Cliente de HIDROCONS para poder ingresar a la aplicación. El caso de uso a nivel de páginas Web comienza con la interfaz exterior del sistema, donde se observa, entre otras cosas, un formulario con dos campos para que el usuario coloque su nombre de usuario y su contraseña. Al hacer la solicitud de ingreso a la aplicación el sistema consulta si existe un usuario registrado cuyo login y password sean los indicados, además que su situación

esté Activo y no esté conectado. Esto es llevado a cabo utilizando una unidad de consulta sencilla o simple encargada de buscar dichos datos en la tabla USUARIOS. En caso de que no se cumpla alguna de estas condiciones se muestra un mensaje de rechazo y se vuelve a mostrar la interfaz externa. De lo contrario, si cumple todos los requisitos y es un usuario registrado y autentificado, automáticamente se le abrirán las puertas e ingresará al sistema para iniciar una sesión.

Posteriormente se uso de una unidad de ingreso o registro de datos, la cual se encarga de ingresar en la tabla SESIONES un registro que identifique la visita de este usuario (Código de sesión, código de usuario, fecha y hora de entrada, etc.). El sistema lo llevará a su respectivo SiteView Principal, dependiendo si es un Empleado o un Cliente, generándose automáticamente una variable global, conocida como variable de sesión, que posee todos los datos propios del usuario que acaba de ingresar, aunado al código de la sesión establecido por la aplicación, para ser empleados en casos de uso posteriores. Si ocurriese un inconveniente durante esta última acción, el usuario será llevado a la página exterior nuevamente.

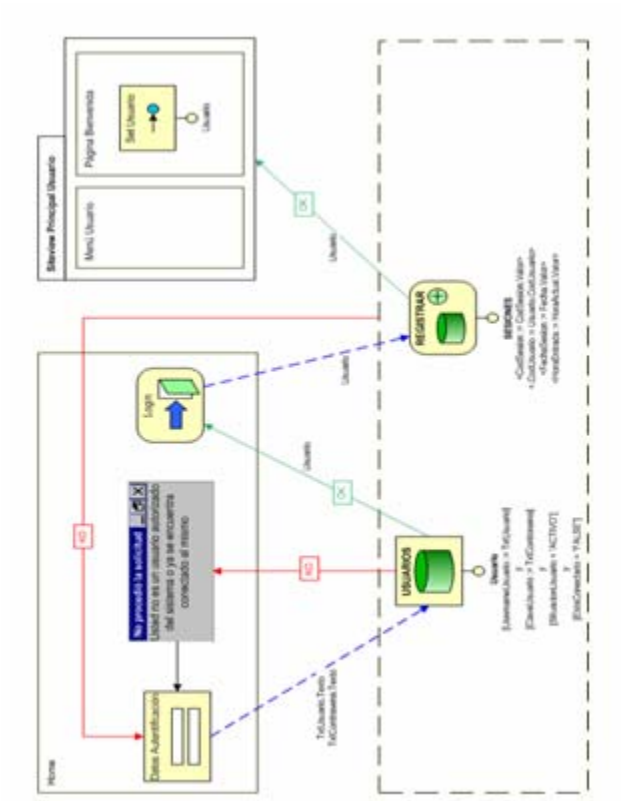


Figura 4.21. Diagrama de Composición del Caso de Uso "Iniciar Sesión Usuario"
Fuente: Propia, 2009

- **Diagrama de Composición o de Gestión de Contenido del Caso de Uso "Registrar Empresa"**: se puede detallar este modelo en la figura 4.22 presente en la siguiente página, la cual especifica y detalla la realización desde principio a fin de este caso de uso a nivel de páginas y tecnología Web. Considérese en este caso que un Empleado HIDROCONS sea el que lleve a cabo este caso de uso. El proceso comienza observándose la interfaz de Registrar Empresa donde el empleado llena los campos del formulario y solicita el registro o ingreso de los mismos, acción la cual lleva al sistema a consultar si existe una empresa que posea el Número de RIF o número de NIT ingresado por el empleado. Esta búsqueda se realiza empleando una unidad de consulta simple o sencilla la cual revisa en la tabla EMPRESAS alguna

ocurrencia del RIF o en su defecto, del NIT ingresado en el formulario en cuestión. Si existe alguna empresa que concuerde con esos datos se muestra un mensaje indicando la existencia de la misma. De lo contrario, una unidad de registro o ingreso de datos se encarga de registrar los datos de la nueva empresa ingresados por el empleado a la base de datos.

Luego, de manera automática, a la variable global o de sesión, la cual posee información extraída de la base de datos referente a este usuario que hace uso de esta aplicación, se le extrae el código de sesión asignado al usuario al momento de autentificarse. Esto para que, empleando una unidad de modificación, se modificase el registro presente en la tabla SESIONES cuyo código de sesión concuerde con el obtenido recientemente, anexando el comentario que denote esta acción recién hecha. Posteriormente la entidad Cartero le envía un correo electrónico a la empresa recién registrada al sistema, dándole la bienvenida al sistema SIADCON. Finalmente el sistema lleva al empleado a la misma interfaz de Registro de Empresa, con los campos del formulario vacíos en espera de introducir nuevos datos de otra nueva empresa. En caso de que ocurra algún inconveniente durante todo este proceso la aplicación mostrará un mensaje al Empleado indicando que hubo una excepción.

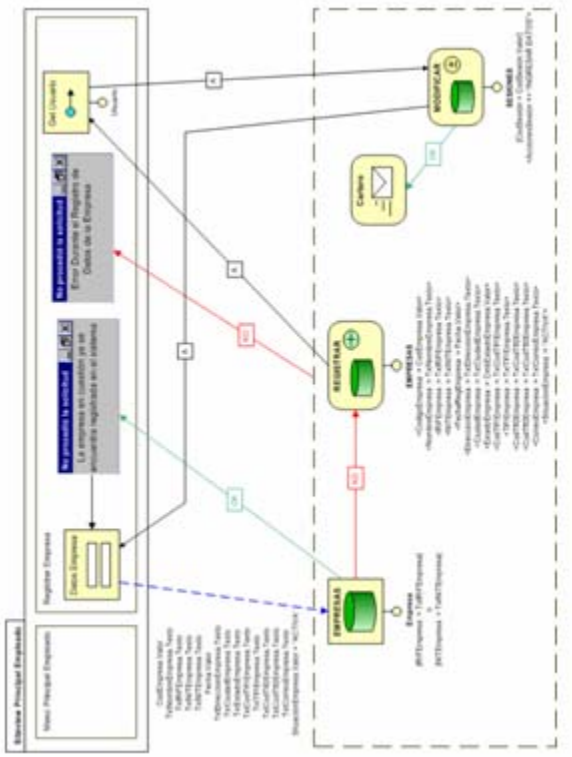


Figura 4.22. Diagrama de Composición del Caso de Uso "Registrar Empresa"
Fuente: Propia, 2009

- **Diagrama de Composición o de Gestión de Contenido del Caso de Uso "Consultar Empresa"**: considérese en este caso que de todos los usuarios que pueden llevar a cabo este caso de uso, el que lo realice sea un Cliente HIDROCONS, quien obviamente desee consultar los datos de la empresa a la cual representa. Como se puede observar en la figura 4.23, la cual explica detalladamente el funcionamiento de este proceso de consulta, en el instante en que dicho cliente hace la solicitud de consulta de la empresa a la cual pertenece, el sistema toma de la variable global o de sesión, variable que contiene información referente a dicho actor y que es generada al momento de iniciar su visita a la aplicación, el valor del código de usuario que le fue asignado a dicho usuario por la aplicación al momento de solicitar la afiliación al sistema. Haciendo uso de este código de usuario, el sistema hace

la consulta en la base de datos de los datos de la empresa que le pertenezca el usuario cuyo código de usuario concuerde con el valor proveniente de la variable de sesión. Esta acción es llevada a cabo utilizando una unidad de consulta sencilla o simple, la cual se encarga de buscar en la tabla EMPRESAS una ocurrencia de alguna empresa que esté asociada con el usuario que actualmente lleva a cabo la solicitud de consulta, es decir, alguna empresa asociada con el código de usuario del Cliente en cuestión.

Todo actor del sistema que actúe como cliente de HIDROCONS está asociado obligatoriamente con alguna empresa en particular, de allí que no es posible el caso de que no exista la empresa a la cual pertenece el cliente. Posteriormente, después de conseguida la empresa en cuestión, el sistema devuelve todos los datos de dicha empresa extraídos de la base de datos y los coloca en los campos respectivos de un formulario adaptado para ello de la nueva interfaz de consulta de empresa, los cuales son finalmente visualizados por el Cliente solicitante. En caso de que ocurriese algún error o inconveniente durante la consulta de estos datos, se mostrará al usuario el mensaje respectivo, indicando tal anomalía.

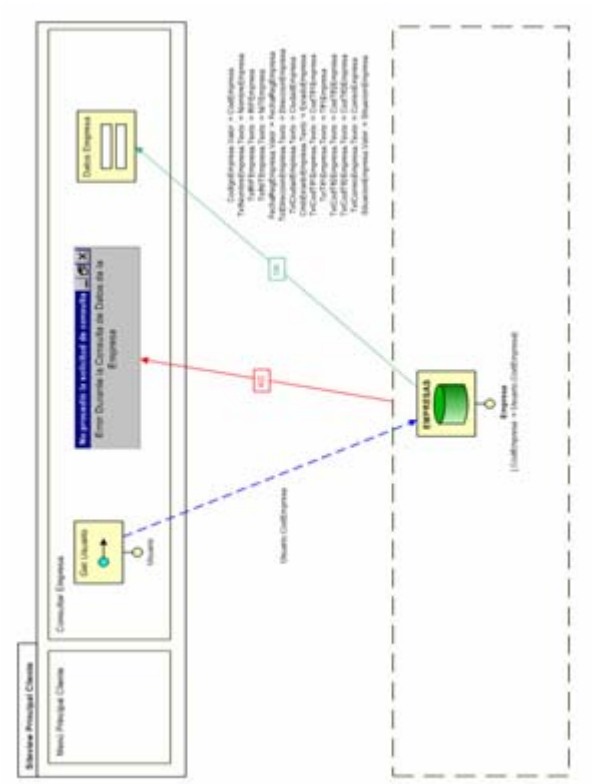


Figura 4.23. Diagrama de Composición del Caso de Uso "Consultar Empresa"
Fuente: Propia, 2009

- **Diagrama de Composición o de Gestión de Contenido del Caso de Uso "Modificar Repuesto"**: este caso de uso es realizado exclusivamente por algún Empleado HIDROCONS, ya que es el tipo de actor autorizado para tal finalidad. Obsérvese detalladamente la figura 4.24, la cual corresponde a la realización completa a nivel de tecnología Web de este modelo de composición. Considérese que la página donde se encuentran todos los campos de datos del repuesto a modificar se está observando actualmente en la interfaz del sistema. El Empleado hace las modificaciones de rigor correspondientes a dicho repuesto y procede a solicitar y confirmar la modificación de los datos. Una unidad de consulta simple o sencilla verifica si existe en la base de datos, específicamente en la tabla REPUESTOS, un

repuesto con código distinto al del repuesto recién modificado pero con igual nombre de repuesto. De ser así se observa el mensaje de duplicación de repuestos (el repuesto ya existe). Esto con la finalidad de evitar que se use nuevamente el nombre de un equipo ya utilizado previamente y que genere confusiones a los usuarios al momento de hacer uso de ellos. De lo contrario una unidad de modificación de datos recibe los datos substitutivos, y dicha unidad se encarga de realizar el cambio de la información en la tabla REPUESTOS de la base de datos, para luego devolverlos a la interfaz del sistema y ser colocados nuevamente en sus campos respectivos.

Posteriormente y de manera automática, a la variable global o de sesión, asignada al momento de autentificarse como usuario registrado del sistema, se le extrae el código de sesión que el sistema designó al usuario en dicho instante. Esto con la finalidad de anexarle a la tabla SESIONES, específicamente en el registro que concuerde con dicho código de sesión, y haciendo uso de otra unidad de modificación, el comentario de que se modificó un registro de la tabla REPUESTOS. En caso de ocurrir un error o excepción durante la modificación se muestra un mensaje indicando dicho inconveniente.

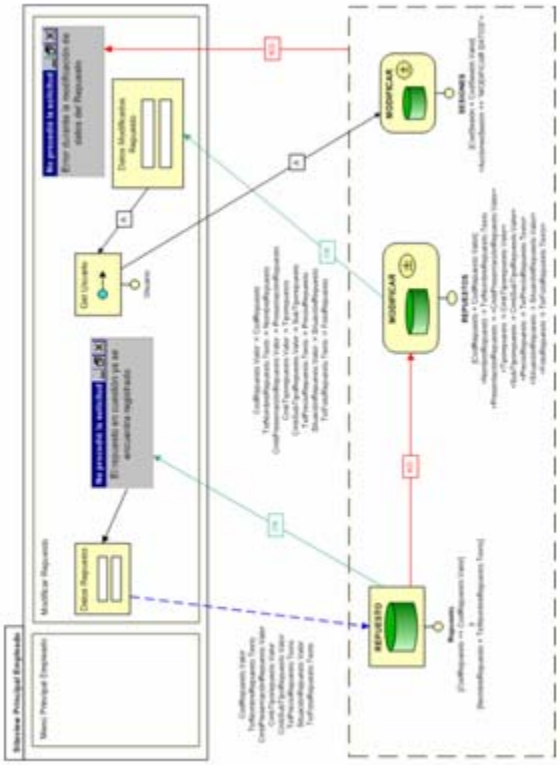


Figura 4.24. Diagrama de Composición del Caso de Uso "Modificar Repuesto"
Fuente: Propia, 2009

- **Diagrama de Composición o de Gestión de Contenido del Caso de Uso "Dar de Baja Usuario"**: este modelo se puede observar en el diagrama o figura 4.25. Considérese que de todos los usuarios que pueden realizar este caso de uso, sea el mismo Cliente HIDROCONS el que desee darse de baja del sistema, y que sea él mismo el que lleve a cabo este procedimiento. En la interfaz del sistema se muestra el formulario que contiene todos los datos del cliente en cuestión, y las distintas opciones que se pueden procesar con los datos del mismo. Entre todas estas opciones está precisamente la de dar de baja a dicho usuario.

El cliente procede con dicha solicitud presionando el botón que así lo da a entender y después de confirmar que realmente desea ser desactivado, mediante la

aparición de un mensaje emergente de confirmación, la aplicación operará de la manera que a continuación se describe: El sistema toma de la variable global de sesión, asignada al momento de autentificarse como usuario registrado del sistema, es decir, en el instante de iniciar la sesión e ingresar a la aplicación, el código de usuario asignado al momento de registrar su solicitud de afiliación. Haciendo uso del valor de esta variable, una unidad de modificación de datos se encarga de modificar algunos campos que tienen que ver con la tabla USUARIOS, es decir, los campos que denotan si el cliente está conectado y la situación del mismo, colocándolo con el valor de No Conectado y modificando su situación a Inactivo respectivamente. Se emplea una unidad de modificación y no de eliminación de datos porque este proceso de dar de baja es una desactivación, es decir, una eliminación virtual o lógica, en donde el cliente aún sigue registrado en el sistema pero en condiciones distintas. Esto es diseñado así para tener la posibilidad de que este usuario pueda ser reactivado en una próxima oportunidad en caso de que él así lo desee. Seguidamente el sistema procede a enviar un E-Mail a la dirección de correo electrónico extraída de igual modo de la variable global asignada por el sistema al comenzar la sesión, comunicándole al usuario en cuestión que efectivamente ha sido desactivado de la aplicación.

Posteriormente, la aplicación hace uso del código de sesión presente nuevamente en la variable de sesión global asignad al instante de que el cliente haya iniciado su actual sesión en el sistema. Con esta variable se procede a modificar algunos de los campos de la tabla SESIONES, empleando para ello una nueva unidad de modificación de datos que asigna la hora actual (al momento de darse de baja) extraída del sistema operativo, al campo que tiene que ver con la hora de salida de esta sesión en cuestión, e igualmente ocurre con el campo denominado AccionesUsuario de la misma tabla SESIONES, que almacena todas las acciones importantes que efectuó dicho usuario durante su visita al sistema, incluyendo la eliminación lógica del mismo efectuada recientemente. Finalmente el sistema cierra la sesión del usuario, quien además sale del SiteView principal al cual pertenece y es

llevado a la interfaz externa de la aplicación. Durante todo este proceso, si ocurre algún inconveniente a nivel de modificación, se mostrará un mensaje de aviso.

- **Diagrama de Composición o de Gestión de Contenido del Caso de Uso "Registrar Presupuesto"**: se puede distinguir este modelo de composición precisamente en la figura 4.26. El Empleado de HIDROCONS es el que se encarga de llevar a cabo este caso de uso, ya que es el actor habilitado para tal fin. Se observa una ventana modal con campos que recibirán los datos propios del encabezado del presupuesto. Luego de ingresar estos datos en cuestión y solicitar la continuación y confirmación de este proceso mediante un cuadro de diálogo apropiado, se obliga al sistema a verificar si existe algún presupuesto con número de referencia o código igual al recién ingresado. Esto se realiza utilizando una unidad de consulta simple o sencilla, que realiza dicha búsqueda en la tabla PRESUPUESTOS. De ser cierta la existencia en la base de datos de un presupuesto con igual número de referencia, se muestra un mensaje indicando dicha situación (que se está duplicando un presupuesto existente) y se procede a rectificar este dato al observarse nuevamente la ventana modal con los datos ingresados. De lo contrario, se observa otra interfaz conformada por otra ventana modal con un formulario en el cual se ingresa un ítem, partida o renglón del presupuesto en cuestión. Ingresado éste en el formulario se procede a registrarlo presionando el botón adecuado y confirmando dicha solicitud. Esta acción se repite tantas veces como ítems o renglones deban ingresarse al presupuesto.

Al no existir más ítems o partidas que ingresar, el usuario solicita la continuación del proceso de registro del presupuesto e inmediatamente se originan dos unidades de creación y registro de datos, una primera unidad para registrar los datos del encabezado del presupuesto recibidos en la primera ventana modal, y la otra posteriormente para registrar la lista de todos los ítems propios del mismo, recibidos

en la segunda ventana modal mencionada anteriormente. En este último caso esta unidad se emplea tantas veces como ítems existan, es decir, tantos ítems que se hayan ingresado en la ventana modal correspondiente, como lo indica la flecha que sale y regresa a esta misma unidad de creación.

Posteriormente y de manera automática, a la variable global o de sesión, asignada al momento de autentificarse como usuario registrado del sistema, y que contiene toda la información del usuario mismo que representa al actor que hace uso de la aplicación, se le extrae el valor de la variable de código de sesión, de manera que pueda actualizar el registro en la tabla SESIONES que posea dicho código, adicionándole al campo AccionesUsuario de la misma tabla un comentario del proceso de registro de datos realizado en este caso. Luego la entidad Cartero le envía un correo a la dirección electrónica de la empresa para la cual fue elaborado el presupuesto, indicando que ya lo tienen a la disposición en el sistema para la próxima vez que ingresen al mismo. Finalmente el sistema muestra nuevamente la misma ventana modal observada al inicio para recibir datos de otro nuevo presupuesto posiblemente a registrar. En caso de que ocurra algún inconveniente durante todo este proceso, se mostrará al Empleado el mensaje respectivo.

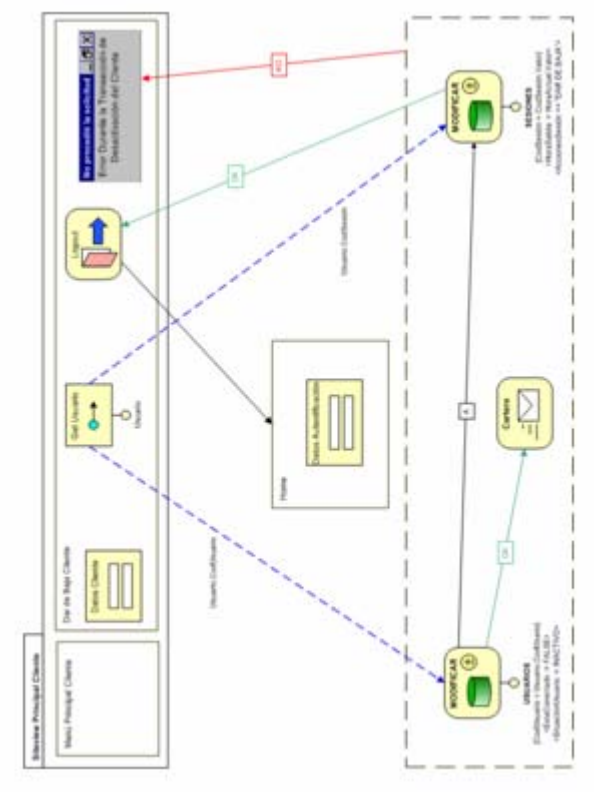


Figura 4.25. Diagrama de Composición del Caso de Uso "Dar de Baja Usuario"
Fuente: Propia, 2009

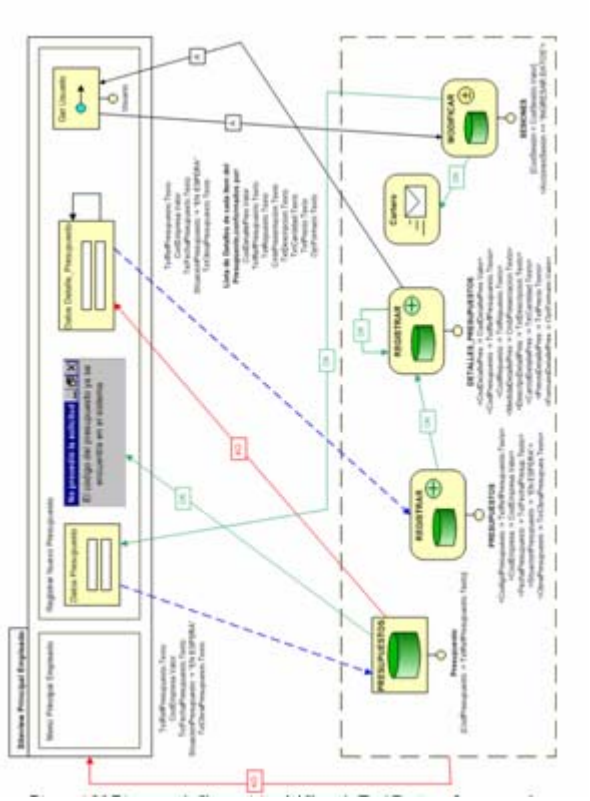


Figura 4.26. Diagrama de Composición del Caso de Uso "Registrar Presupuesto"
Fuente: Propia, 2009

- **Diagrama de Composición o de Gestión de Contenido del Caso de Uso "Consultar Facturas"**: considérese que sea un Empleado de HIDROCONS el que lleve a cabo este caso de uso. Según se puede observar en la figura 4.27 correspondiente a este diagrama, el cual explica gráficamente todo este caso de uso, el proceso comienza con una ventana modal que contiene un formulario con campos que reciben los criterios de búsqueda que ha de ingresar el usuario que hace la consulta. En el mismo el Empleado selecciona cómo va a ser su búsqueda de facturas (todas las facturas hasta ahora registradas, facturas de una empresa en particular, facturas elaboradas en un período de tiempo estipulado, facturas canceladas o pendientes por cobrar, y casos combinados). Al proceder con la consulta respectiva el sistema determina las facturas que cumplen con los criterios de búsqueda introducidos, haciendo uso de una unidad de consulta

múltiple que busca específicamente en la tabla FACTURAS de la base de datos. Para cada una de las facturas perteneciente a la lista de facturas conseguidas por el sistema, el mismo hace la consulta, mediante otra unidad de consulta múltiple, de cuál empresa es propietaria de cada factura, comparando el código de la empresa perteneciente a cada factura en sí con el código de empresa de la tabla EMPRESAS.

Ocurre algo parecido con la tabla DETALLES_FACTURAS, para determinar todos los ítems pertenecientes a cada factura consultada, utilizando obviamente otra unidad de consulta múltiple, es decir, para cada factura conseguida en esta consulta se determinan los detalles e ítems de cada una, comparando el código de factura de cada una de ellas con el código de factura presente en la tabla DETALLES_FACTURAS. Todas estas consultas provenientes de todas estas tablas llevan todos estos datos, obviamente ordenados, a una unidad de navegación, cuya función es distribuir los resultados obtenidos de la consulta pero de 10 en 10 (las 10 facturas previas y siguientes), para luego ser observados mediante una lista jerárquica. En caso de ocurrir un error durante la consulta, es emitido un mensaje referente a ello.

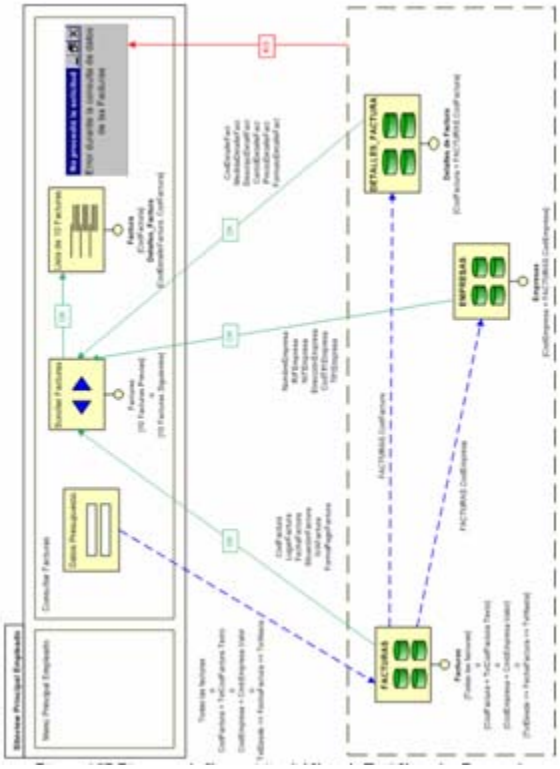


Figura 4.27. Diagrama de Composición del Caso de Uso "Consultar Facturas"
Fuente: Propia, 2009

- **Diagrama de Composición o de Gestión de Contenido del Caso de Uso "Consultar Solicitudes de Repuestos"**: considérese que de todos los usuarios registrados del sistema, sea un Empleado de HIDROCONS el que lleve a cabo este caso de uso. En la figura 4.28 se puede constatar todo el desarrollo de este caso de uso a nivel de páginas y aplicación Web. Al proceder con la consulta respectiva el sistema determina todas las solicitudes de repuestos cuya situación esté En Espera, haciendo uso de una unidad de consulta de datos múltiple que busca específicamente en la tabla SOLICITUDES_REPUESTOS de la base de datos. Para todas y cada una de estas solicitudes conseguidas en esta búsqueda, dicho sistema hace una nueva consulta, mediante otra unidad de consulta de datos múltiple, de cuál empresa es propietaria o fue la que generó dicha solicitud de repuestos, comparando el código de la empresa presente en

cada cotización o solicitud de repuestos encontrada con los códigos de empresa presentes en la tabla EMPRESAS.

Ocurre algo parecido con la tabla REPUESTOS_SOLICITADOS, para determinar todos los ítems o repuestos pertenecientes en cada solicitud consultada, utilizando obviamente otra unidad de consulta múltiple. Del mismo modo, para cada consulta de repuesto solicitado extraído de dicha tabla, se consulta en la tabla REPUESTOS para determinar los datos que faltan de los mismos, como lo son el nombre del repuesto y su presentación o medida, utilizando otra vez una nueva unidad de consulta múltiple, la cual sólo utiliza los códigos del repuesto extraídos de la tabla REPUESTOS_SOLICITADOS en la consulta previa. Todas estas consultas provenientes de todas estas tablas llevan todos estos datos, obviamente ordenados por código de solicitud y por fecha, y en formato de solicitud de repuestos de HIDROCONS, a una unidad de navegación, la cual mostrará, haciendo uso de una lista jerárquica, cada una de estas solicitudes pendientes por aprobarse. En caso de que ocurra algún inconveniente durante todo este proceso, se mostrará al Empleado el mensaje respectivo.

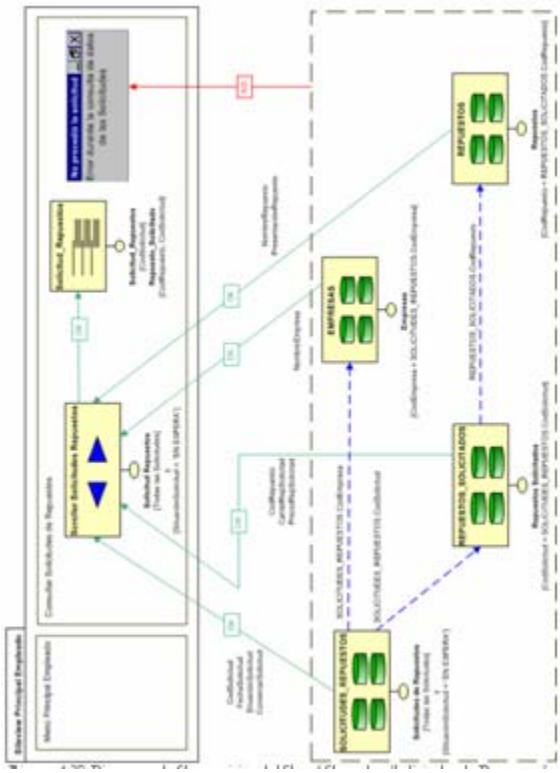


Figura 4.28. Diagrama de Composición del Caso "Consultar Solicitudes de Repuestos"
Fuente: Propia, 2009

4.5.4. Arquitectura de red y hardware

Basándose en la arquitectura candidata expuesta en la fase anterior, se mantiene el hecho de considerar la arquitectura de un solo servidor y varios clientes, conectados por medio de Internet. El sistema SIADCON (nombre del software o producto final) incluye una capa de aplicación basada en el uso de páginas Web HTML y Scripts de servidor. Dichas páginas son desplegadas mediante el subsistema navegador Web que en este caso puede ser representado por Internet Explorer, Firefox, Netscape, Mozilla, etc. Para procesar las peticiones del cliente, el navegador utiliza el subsistema servidor Web que consiste del servidor WebSphere el cual genera las páginas Web de tendencia dinámica consultando los datos en el SRMBDD IBM DB2. 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, además del protocolo HTTP que representa el protocolo de comunicación entre el servidor y los clientes, por medio de la vía de Internet. En la figura 4.29 siguiente se muestra un bosquejo del despliegue de la arquitectura de red:

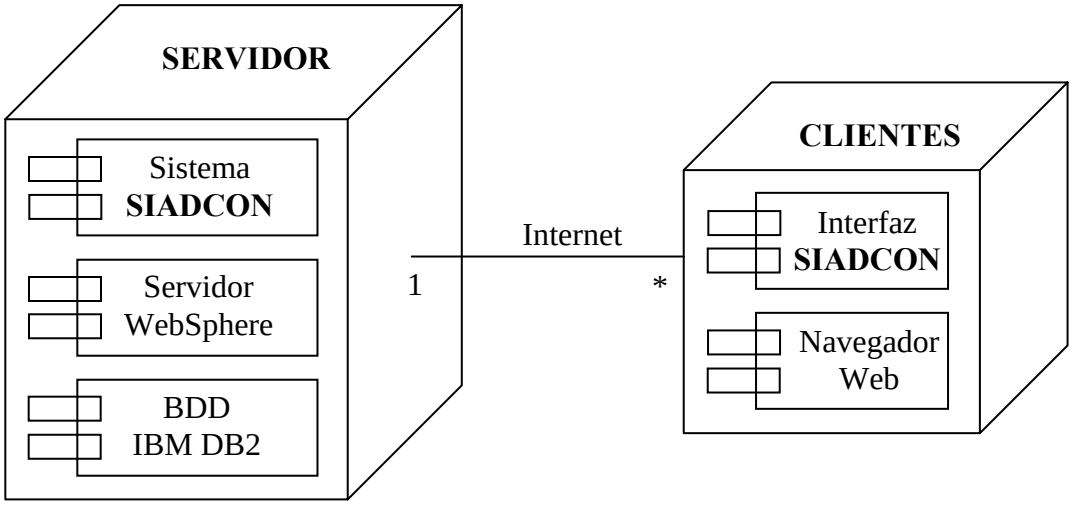


Figura 4.29. Diagrama de despliegue del sistema
Fuente: Guía de Estudiante del Curso de Java Empresarial Libro Nº. 1" de IBM. Código del Curso: CY760. Versión 4.0., 2006

4.6. Evaluación de la fase de elaboración

Al final de esta fase se encuentra el "Hito de Arquitectura del Sistema". En este punto se examina en detalle el alcance y objetivos del sistema, la selección de la arquitectura y la resolución de la mayoría de los riesgos. Se requirió de una sola iteración para completar los objetivos planteados al inicio de la fase. Al revisar este hito aplicado al proyecto que actualmente se desarrolla se constataron ciertos criterios de evaluación, arrojándose los siguientes resultados:

- Se reafirma la viabilidad y estabilidad de la visión y los requerimientos del proyecto, así como de la arquitectura del mismo.

- El diseño de las bases de datos propias para el almacenamiento de la información relacionada con los procesos administrativos a ser manejados por el sistema en cuestión fue llevado a cabo de manera exitosa, al igual que el diseño de la Arquitectura del Software, en función de las restricciones computacionales existentes.

- Los planes de iteración para la construcción son lo suficientemente detallados y fiables para permitir iniciar el trabajo de desarrollar la aplicación como tal.

- Todos las personas involucradas en el proyecto (inversionistas, directivos de HIDROCONS, C. A., diseñador de software, empleados, etc.) están de acuerdo con la visión actual del mismo.

- Son aceptables los gastos actuales de recursos versus los gastos planeados con anterioridad.
- Se logró hacer un diseño efectivo de las páginas Web que conformarán el sistema mediante el uso del modelo Estructural y los modelos de Navegación de WebML.

- En definitiva se considera que ya se han obtenido todos los componentes necesarios para construir el sistema en la siguiente fase.

CAPÍTULO V

FASE DE CONSTRUCCIÓN

5.1. Introducción

La fase de construcción se enfoca de forma detallada en etapas de diseño, implementación y pruebas hasta lograr un sistema completo, con una alta calidad a un costo efectivo. La meta de esta fase es resolver los requerimientos restantes y completar el desarrollo del sistema sobre la arquitectura base. A esta altura del desarrollo muchos casos de uso posiblemente no hayan sido implementados del todo, sólo parcialmente o lo suficiente para probar algunas hipótesis y mitigar riesgos. A medida que se implementen más y más casos de uso, se refinarán los requerimientos para asegurar que la funcionalidad correcta será entregada, buscando un balance entre calidad, alcance, tiempo y refinación de las características. Por eso esta fase es típicamente la que más tiempo consume. [9]

5.2. Planificación de la fase de construcción

Esta fase tiene como objetivos los siguientes: [9]

- Minimizar los costos de desarrollo por medio de la optimización de recursos, evitando el trabajo innecesario.
- Desarrollar iterativa e incrementalmente un producto completo que esté listo para la transición a la comunidad de usuarios.

En líneas generales, el principal objetivo de esta fase es obtener una solución de software operativa, satisfaciendo todos los requerimientos planteados en las fases anteriores, ofreciendo una alta calidad funcional y practicidad. Esta versión del software que se obtendrá se cataloga como "versión beta", una versión lista para ser entregada a los usuarios finales de la aplicación. Para lograr los objetivos planteados en esta fase, se realiza la codificación de las páginas Web que se diseñaron en el modelo de hipertexto de WebML. La implementación de estas páginas implica también la construcción visual de las mismas; deben tener un aspecto elegante y un esquema de navegabilidad práctico, siguiendo con el prototipo de interfaz planteado en la fase de elaboración. No sólo la codificación de las diferentes interfaces se refleja en esta fase de desarrollo sino también lo referente a interacciones entre las diversas entidades u objetos del sistema, que igualmente deben de ser desarrollados para funcionar como un todo. Los diagramas de secuencia descritos en la fase anterior y que detallan los casos de uso que intervienen en este sistema ofrecen una vista general de lo que representan las interacciones entre los diferentes objetos y entidades que actúan en el sistema a construir. De allí que son estos diagramas, entre otros adicionales, como los diagramas WebML, los que constituyen piezas fundamentales a requerir para esta nueva fase de desarrollo del proyecto en cuestión, en lo que a desarrollo de interfaces, codificación y pruebas o evaluaciones de módulos se refiere.

5.3. Diseño

Para la fase de construcción del desarrollo del proyecto se salta toda etapa de análisis para pasar directamente a una etapa de diseño, ya que se procede a la construcción de la aplicación en sí. Por otra parte, en esta etapa de la construcción ya se habla de manera más formal de las herramientas a emplear para el desarrollo y construcción de la aplicación como tal.

5.3.1. Herramientas de desarrollo empleadas

Para el diseño del software se hizo uso de las siguientes herramientas. Cada una contribuye al buen desempeño del proyecto en cuestión, y son piezas fundamentales para el correcto funcionamiento del mismo. Entre ellas se encuentran:

- **Sistema Manejador de Base de Datos Relacionales IBM DB2 Versión 9**: el RDBMS DB2 Universal Database es un sistema administrador de base de datos que responde rápidamente a la demanda de transacciones en los períodos de mayor carga de trabajo, se amplía para acoger cantidades crecientes de información distribuida en distintas bases de datos y crece al ritmo de su infraestructura informativa desde uno a varios procesadores.

- **Software de aplicaciones de computación distribuida IBM RATIONAL 6.0**: constituye un software del tipo middleware para sistemas distribuidos. El middleware apunta a ocultar la heterogeneidad de las redes subyacentes, hardware, sistemas operativos y lenguajes de programación, ya que uno de los principales desafíos que enfrentan los diseñadores de sistemas distribuidos es la diversidad de estos aspectos. Con esta herramienta los programadores pueden concentrarse en escribir programas que trabajen a un nivel abstracto para comunicarse en un ambiente distribuido.

- **Servidor WebSphere 5.0:** dispositivo de software original de IBM que trabaja bajo plataforma Java 2 EE de Sun MicroSystem y que ofrece servicios de aplicación Web a las computadoras clientes. Gestiona la mayor parte de las funciones de lógica de negocio y de acceso a los datos del sistema.

5.4. Implementación

La etapa de implementación de la fase de construcción de RUP consta en este caso de la elaboración de las interfaces del programa, así como de la codificación que manipulará los datos que ingrese, consulte y modifique el usuario. Para la codificación de las páginas se usará el lenguaje de programación **Java** a través del ambiente de desarrollo que ofrece la herramienta **Racional 6.0**. La integración de todos estos elementos conformará la arquitectura de trabajo estable que se diseñó en la fase anterior.

5.4.1. Interfaces gráficas de usuario

Partiendo de los prototipos de interfaces diseñadas en la fase de elaboración, se desarrolla el resto de las interfaces que, al mejorarlas, constituyen las definitivas a utilizar por sus respectivos usuarios. A continuación se muestran algunas de las interfaces finales que formarán parte definitiva de la aplicación en desarrollo, clasificadas según el tipo de usuario:

- **Interfaz de "Principal Externa" o "Inicio de Sesión":** es la pantalla inicial del sistema donde los usuarios se identifican para ser autenticados e iniciar su sesión. Es una interfaz que no sólo incluye un logotipo que identifica a la empresa HIDROCONS, C. A., si no que además incluye los cuadros de texto receptores de información propia de los usuarios registrados, como son el nombre de usuario y su contraseña. Aunado a ello también se observa un menú desplegable horizontal que al mostrar sus opciones y visitar sus páginas, cualquier visitante, sea usuario registrado o no, puede tener una idea general de lo que es la compañía (su historia, su misión y visión, repuestos que ofrece, servicios que presta, cómo contactar y comunicarse con algún empleado, etc). Véase la figura 5.1 para tal finalidad.

Figura 5.1. Interfaz "Principal Externa" o de "Inicio de Sesión"
Fuente: SIADCON, 2009

- **Interfaz "Principal para Empleados de HIDROCONS":** esta página muestra un mensaje de bienvenida al sistema para el Empleado de HIDROCONS que fue autentificado exitosamente, además de la hora y fecha de inicio de sesión. Constituye una página que a sus vez enmarca tres páginas más: la página horizontal superior presenta el logo de HIDROCONS, C. A. acompañado del logo del sistema SIADCON, la página izquierda inferior posee un menú desplegable por medio del cual el usuario puede navegar y trasladarse hacia las diferentes secciones de la aplicación en cuestión que le son permitidas, y finalmente la página inferior derecha que constituye la página de información o de contenido, en donde se muestran las interfaces solicitadas según la opción seleccionada por el usuario en el menú desplegable anterior. A continuación, en la figura 5.2 se puede observar la vista gráfica de la interfaz principal para los empleados que ingresan al sistema:

Figura 5.2. Interfaz "Principal para Empleados de HIDROCONS"
Fuente: SIADCON, 2009

- **Interfaz "Registrar Empresa" para Empleados de HIDROCONS:** haciendo uso de esta página el Empleado de HIDROCONS, quien es el usuario capacitado para llevar a cabo este caso de uso, puede ingresar los datos necesarios de una empresa en particular, tales como Nombre de la compañía, RIF y NIT de la misma, Estado y Ciudad en que reside, Dirección Fiscal de la empresa, un correo electrónico y hasta dos teléfonos fijos o móviles para poder contactarla en caso de ser necesario. La mayoría de los datos antes mencionados son de obligatorio ingreso, por lo que es imperativo que sean escritos, de lo contrario la solicitud no procederá. Los otros datos, tales como el NIT de la empresa, e inclusive un segundo número telefónico pueden ser omitidos. En caso de que se omita algún dato necesario el sistema mostrará un mensaje al presionar el botón adecuado, indicando la falta del mismo. De igual modo, al ser exitoso el registro de la empresa solicitante, la

aplicación emitirá un mensaje anunciando el registro definitivo de la misma al sistema. Véase la interfaz en la figura 5.3.

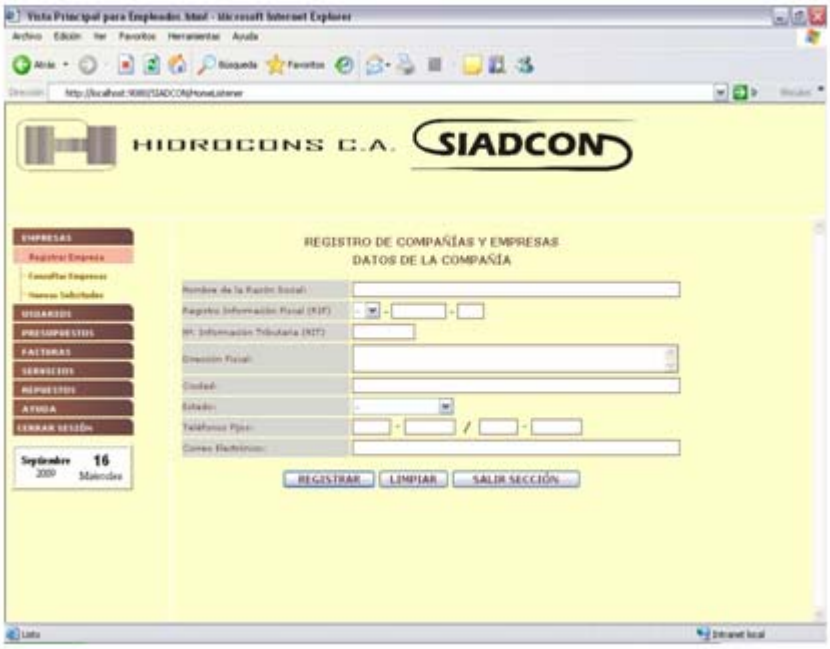


Figura 5.3. Interfaz "Registrar Empresa" para Empleados de HIDROCONS
Fuente: SIADCON, 2009

- **Interfaz "Consultar Empresa" para Clientes de HIDROCONS:** al seleccionar esta opción se muestra en el área de información o de contenido un formulario con los datos propios de la empresa a la cual pertenece el Cliente de HIDROCONS que hace la consulta, como se puede observar en la figura 5.4. Los datos de dicha razón social incluye campos como Nombre de la empresa, RIF y NIT de la misma, Estado y Ciudad en que reside, Dirección Fiscal de la compañía, un correo electrónico, etc. En esta misma interfaz se ofrece la posibilidad de modificar dichos datos directamente, e inclusive hasta dar de baja la empresa misma, presionando obviamente el botón apropiado.

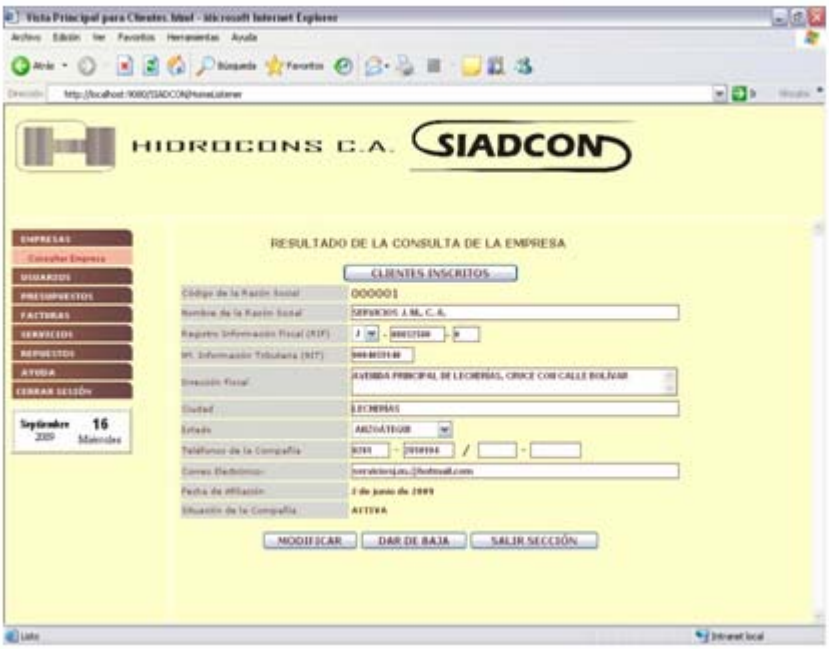


Figura 5.4. Interfaz "Consultar Empresa" para Clientes de HIDROCONS
Fuente: SIADCON, 2009

- **Interfaz "Registrar Presupuesto":** el ingreso de un presupuesto sólo lo lleva a cabo un Empleado de HIDROCONS y se realiza en tres partes: una primera en donde por medio de una ventana modal se ingresan los datos del encabezado del presupuesto en cuestión; una segunda donde se registran, mediante otra nueva ventana modal cada una de las partidas, renglones o ítems que conforman el cuerpo de dicho presupuesto; y una tercera parte, que es la que se observa en la figura 5.5, donde se puede visualizar el presupuesto final, para detallar cómo será visto por la empresa cliente al cual se emite, y donde se puede eliminar o modificar cualquiera de estos ítems a conveniencia del transcriptor. En esta última parte el mismo será registrado en la BDD presionando el botón adecuado, de lo contrario se puede cerrar dicha ventana.

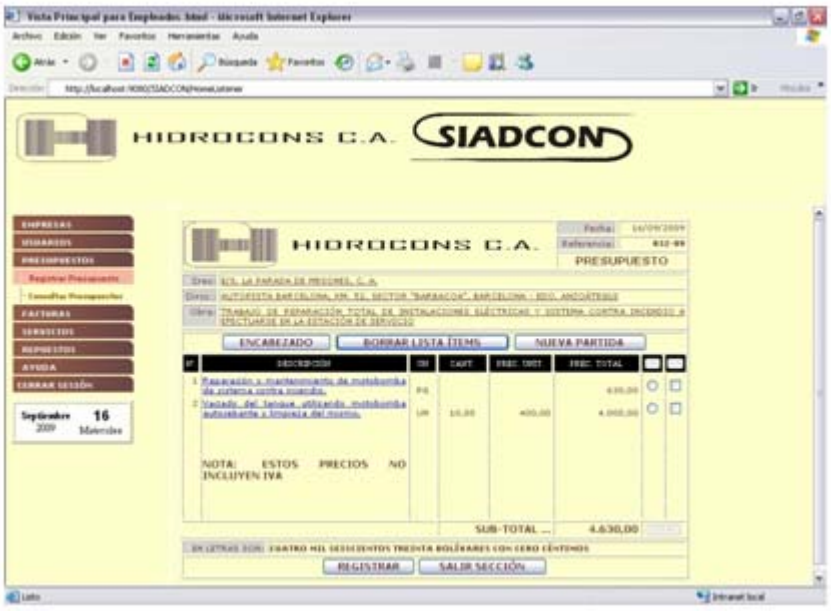


Figura 5.5. Interfaz "Registrar Presupuesto"
Fuente: SIADCON, 2009

- **Interfaz "Consultar Presupuestos" para Clientes de HIDROCONS:** en esta opción aparece una ventana modal que recibe los criterios de búsqueda por los cuales se van a consultar los presupuesto, que pueden consistir en: todos los presupuestos de la empresa del Cliente de HIDROCONS, uno en particular según su número de referencia, presupuestos cuya fecha de elaboración esté dentro de un rango específico, etc. Si no existe algún presupuesto registrado con alguno de los criterios de búsqueda anteriores se muestra un mensaje explicando tal situación, de lo contrario, como se observa en la figura 5.6, se muestra una lista de todos los presupuestos que cumplen con los requisitos de búsqueda indicados, con sus respectivas informaciones generales. En caso de que la consulta arroje más de diez (10) presupuestos, se pueden usar los enlaces en forma de flecha para observarlos en su totalidad. Para poder visualizar cada uno de los presupuestos mostrados se presiona en los enlaces conformados por los números de referencia de los presupuestos, con lo que se abre una nueva ventana con el presupuesto completo solicitado.

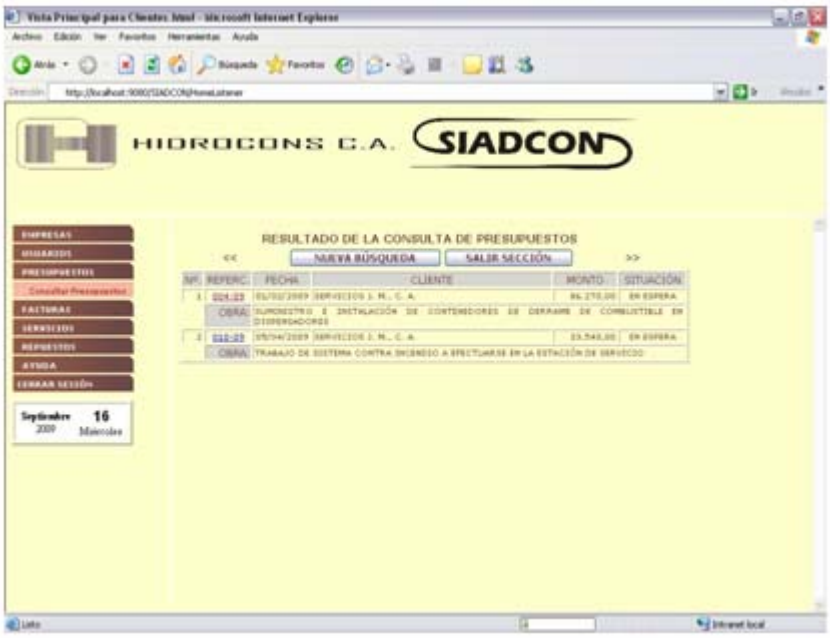


Figura 5.6. Interfaz "Consultar Presupuestos" para Clientes de HIDROCONS
Fuente: SIADCON, 2009

5.4.2. Código fuente

Las páginas Web que han sido mostradas se deben implementar en código fuente. Para la codificación de las páginas se usará el lenguaje de programación Java a través del ambiente de desarrollo que ofrece el **Software de computación distribuida IBM RATIONAL 6.0**. Las operaciones de base datos serán gestionadas por el **Sistema Manejador de Base de Datos Relacionales IBM DB2 Versión 9**. La integración de todos estos elementos formará la arquitectura de trabajo estable que se diseñó en la fase de Elaboración de RUP.

En la construcción de páginas Web estáticas se emplea el lenguaje HTML puro, este es el lenguaje de marcado predominante para la construcción de este tipo de páginas Web. Entiéndase como página Web estática la que no tiene que acceder a ninguna información presente en alguna base de datos ni se construye mediante la

intervención de algún servidor Web, sino que su información siempre ha de ser constante. A diferencia de las páginas Web dinámicas, cuyo contenido se genera a partir de lo que un usuario introduce en un formulario, o alguna consulta que este realice en específico, el contenido de dicha página no está incluido en un archivo HTML como en el caso de las páginas estáticas, sino que otro lenguaje, como por ejemplo Java, genera automáticamente dicho código HTML. Para la codificación de las páginas dinámicas se ha de emplear lo que se conoce como Servlet. Un Servlet es un componente del lado del servidor escrito en Java que reside y se ejecuta en dicho servidor y que extiende la funcionalidad del mismo para entregar al cliente contenido dinámico. Se llama Servlet porque se usa para servir de recursos al cliente en el lado del servidor.

Para lograr una codificación efectiva e incrementar la calidad de producción de páginas se elaboraron una serie de componentes que representan librerías reutilizables en toda la aplicación. Se crearon librerías **Javascript** que suministran las funciones de validación, características gráficas y controles prácticos durante la ejecución de la aplicación en el explorador Web, del lado del cliente. Para los aspectos de formato y de presentación de los contenidos, tales como tipo de fuente y tamaño de letras, justificación del texto, colores y fondos, etc. se crean librerías de **hojas de estilo** (*style sheets*), las cuales pueden ir a veces en forma de archivo externo o incrustados a los archivos de texto que modifican, permitiendo que se separe la codificación del diseño y procesamiento de datos de una página Web del código que le da presentación a la página en cuestión. En el sistema resultante se codificaron alrededor de 90 páginas de servidor. A continuación se observa una muestra del código fuente de las páginas dinámicas que gestionan el ingreso al sistema de un Servicio que ofrece HIDROCONS, C. A., incluyendo los códigos de formulario en HTML, validaciones de campos usando JavaScript, hojas de estilos para la presentación de la página y el formulario en cuestión, además de codificación Java

para el procesamiento de datos y las interacciones entre todos los objetos codificados que intervienen en este caso de uso.

Código fuente del Caso de Uso "Registrar Servicio"

JavaScript.js

```
/*********************************************************************
*Esta función cuenta el número de caracteres que ingresan en un cuadro de texto*
*determinado y acepta una cantidad de caracteres hasta un máximo permitido,*
*indicado por una variable recibida como argumento. Si se ingresa uno o varios*
*caracteres más de lo permitido, los mismos desaparecen. Recibe de argumentos*
*el texto completo (objeto) y la cantidad permitida de caracteres (cantidad). *
*********************************************************************/
function contarCaracteres(objeto, cantidad)
   {var q = objeto.value;
   a = q.length;
   if (a == cantidad-1)
      valor_listo = q;
   if (a > cantidad-1)
      objeto.value = valor_listo;
   }
/*********************************************************************
*Se pregunta al usuario si efectivamente desea salir de alguna sección determinada*
de la aplición. No recibe argumento alguno, mientras que devuelve una valiable*
booleana cuyo valor depende de la opción del usuario al confirmar la orden de salida*
*********************************************************************/
function permisoSalirSeccion()
   {var resp = false;
      if(confirm('¿Desea salir de esta sección?\nPerderá todos los datos ingresados en
      esta vista'))
         resp = true;
      return resp;
   }

/*********************************************************************
*Esta función permite que al seleccionar una foto de un servicio, automáticamente*
*se visualize la imagen y se escriba el URL de dicha foto en un cuadro de texto*
*determinado. Recibe como argumentos el documento o página Web completo, el*
*valor del URL de la foto y el lugar o posición de la foto seleccionada *
*********************************************************************/
```

```
function origenFotosServ(doc, valor, puesto)
   {var imagen = doc.getElementsByName("ImgRepuesto");
   var direccion = doc.getElementsByName("TxtFileFoto");
   var comentario = doc.getElementsByName("txtComentario");

   imagen[puesto].src = valor.toString();
   direccion[puesto].value = valor;
   comentario[puesto].focus();
   }

/**********************************************************************
*Limpia todo rastro de alguna foto de servicio seleccionada previamente, tanto en la*
*imagen como en el cuadro de texto del URL. Recibe como argumentos el*
*documento Web y la posición de la imagen a limpiar o borrar *
**********************************************************************/
function borrarFotoServ(doc, lugar)
   {var imagen = doc.getElementsByName("ImgRepuesto");
   var direccion = doc.getElementsByName("TxtFileFoto");
   var comentario = doc.getElementsByName("txtComentario");

   imagen[lugar].src = 'imagenes/Servicios/ImagenNoDisponible.jpg';
   direccion[lugar].value = "";
   comentario[lugar].value = "";
   }
/**********************************************************************
*Verifica que todos los campos que reciben los datos respectivos a un servicio*
*específico (enunciado del servicio, descripción del mismo, sus fotos, etc.) estén*
*completa y adecuadamente llenos. Recibe como argumentos el documento y el*
*formulario Web, mientras que devuelve un mensaje en caso de faltar algún dato *
**********************************************************************/
function validarServicioServ(doc, formul)
   {var direccion = doc.getElementsByName("TxtFileFoto");
   var comentario = doc.getElementsByName("txtComentario");
   var resp = false, entro = true, existe = false;

   for(i = 0; i < direccion.length; i++)
      {if(direccion[i].value != "")
         if(comentario[i].value == "")
            {entro = false;
            break;
            }
         else
            existe = true;
```

```
    }

  if(formul.cmbPresupuestos.selectedIndex != 0)
    if(formul.txtEnunciado.value.length != 0 &&
    !/^\s+$/.test(formul.txtEnunciado.value))
      if(formul.txtDescripcion.value.length != 0 &&
      !/^\s+$/.test(formul.txtDescripcion.value))
        if(entro)
          if(existe)
            {if(confirm('¿Los datos ingresados son los correctos?'))
              resp = true;
            }
          else
            {alert('Ingrese al menos una foto que refleje el servicio en cuestión');
            direccion[0].focus();
            }
        else
          {alert('Ingrese el comentario de la foto N°. ' + (i + 1));
          comentario[i].focus();
          }
      else
        {alert('Ingrese una descripción completa y detallada del servicio');
        formul.txtDescripcion.focus();
        }
    else
      {alert('Ingrese un encabezado o enunciado del servicio');
      formul.txtEnunciado.focus();
      }
  else
    {alert('Seleccione un código de presupuesto');
    formul.cmbPresupuestos.focus();
    }
  return resp;
  }
```

TodosEstilos.css

```
/**********************************************************************
*Este estilo es asignado a ciertos  botones de formulario Web,  de modo que el título*
*de  los mismos  sean de  ocho (8) pixeles,  esté alineado  verticalmente en la  mitad,*
*posea un color de código específico y el tipo de letra sea uno de los indicados      *
**********************************************************************/
.boton8
  {FONT-SIZE: 8px; VERTICAL-ALIGN: middle;
```

```
   COLOR:  #663333;
   FONT-FAMILY: Verdana, Arial, Helvetica, sans-serif
   }

/*********************************************************************
*Este estilo es asignado a ciertos  botones de formulario Web,  de modo que el título*
*de  los mismos sean de doce (12) pixeles,  esté alineado verticalmente en  la  mitad,*
*posea un color de código específico y el tipo de letra sea uno de los mencionados    *
*********************************************************************/
.boton12
   {FONT-WEIGHT: bold;   FONT-SIZE: 12px;
   VERTICAL-ALIGN: middle; COLOR: #663333;
   FONT-FAMILY: Verdana, Arial, Helvetica, sans-serif
   }

/*********************************************************************
*Estilo que se  le asigna a determinadas celdas de algunas tablas Web.   Las letras de*
*dicha celda son gruesas y de 10 pixeles, la alineación de la misma es centrada        *
*********************************************************************/
.celdaGrizCent10b
   {FONT-SIZE: 10px; FONT-FAMILY: Verdana, Arial;
   TEXT-ALIGN: center; BACKGROUND: #D7D7D2;
   COLOR: #663333; FONT-WEIGHT: bold
   }
/*********************************************************************
*Estilo que se le asigna  a determinadas  celdas de algunas tablas Web.  Las letras de*
*dicha  celda son de doce (12) pixeles,  la alineación  de  la  misma  es  centrada,  un*
*color de fondo indicado, así como tipo y color de letras según las opciones vistas    *
*********************************************************************/
.celdaGrizCent12
   {FONT-SIZE: 12px; FONT-FAMILY: Verdana, Arial;
   TEXT-ALIGN: center; BACKGROUND: #D7D7D2;
   COLOR: #663333
   }

/*********************************************************************
*Estilo  que se le asigna  a determinadas celdas de algunas tablas Web.  Las letras de*
*dicha celda son gruesas y de 10 pixeles, la alineación de la misma es a la izquierda *
*********************************************************************/
.celdaGrizIzq10
   {FONT-SIZE: 10px; FONT-FAMILY: Verdana, Arial;
   TEXT-ALIGN: left; BACKGROUND: #D7D7D2;
   COLOR: #663333
```

```
   }

/*********************************************************************
*Estilo que  se le asigna a ciertos cuadros de texto.  Especifica los colores de borde,*
*la alineación de su contenido y características de las letras que ingresan al cuadro  *
*********************************************************************/
.textboxIzq11
   {BORDER-RIGHT: #663333 1px solid; BORDER-TOP: #663333 1px solid;
   FONT-WEIGHT: bold; FONT-SIZE: 11px;
   BORDER-LEFT: #663333 1px solid;  COLOR: #663333;
   BORDER-BOTTOM: #663333 1px solid; TEXT-DECORATION: none;
   FONT-FAMILY: Times-New-Roman, Arial, Helvetica, sans-serif;
   TEXT-ALIGN: left
   }

/*********************************************************************
*Estilo que  se le asigna a ciertos cuadros de texto.  Especifica los colores de borde,*
*la alineación de su contenido y características de las letras que ingresan al cuadro  *
*********************************************************************/
.textoCent14b
   {TEXT-ALIGN: center; FONT-SIZE: 14px;
   FONT-WEIGHT: bold; FONT-FAMILY: Verdana, Arial;
   COLOR: #663333
   }
/*********************************************************************
*Estilo que  se le asigna a ciertos cuadros de texto.  Especifica los colores de borde,*
*la alineación de su contenido y características de las letras que ingresan al cuadro  *
*********************************************************************/
.textoIzq12
   {FONT-SIZE: 12px; FONT-FAMILY: Verdana, Arial;
   TEXT-ALIGN: left; COLOR: #663333
   }

/*********************************************************************
*Estilo que  se le asigna a ciertos cuadros de texto.  Especifica los colores de borde,*
*la alineación  de su contenido y características  de las letras que ingresan al cuadro,*
*como letra gruesa, de un color y tipo específico y con un tamaño de 12 pixeles       *
*********************************************************************/
.textoIzq12b
   {FONT-SIZE: 12px; FONT-FAMILY: Verdana, Arial;
   TEXT-ALIGN: left; FONT-WEIGHT: bold;
   COLOR: #663333
   }
```

SGBDD.java

```
package modelo;

/*****************************************************************
*Las  siguientes son  las librerías  que se  importan o  incluyen para que  esta clase o*
*función se lleve a cabo con total normalidad y sin inconvenientes                      *
*****************************************************************/
import javax.naming.*;

import java.io.Serializable;
import java.sql.*;
import javax.sql.DataSource;
import javax.swing.JOptionPane;

/*****************************************************************
*Esta función representa la clase SGBDD, la cual es pública, implementa  la interfaz*
*"Serializable" y presenta atributos privados para otras clase.  Dicha clase se refiere *
*a ciertos objetos que sirven para la conexión con la Base de Datos                     *
*****************************************************************/
public class SGBDD implements Serializable
   {private DataSource ds = null;
   private ResultSet rs = null;
   private Connection con = null;
/*****************************************************************
*Esta función representa el constructor de la clase SGBDD.  La sección "try" evalúa*
*si alguna de sus sentencias genera algún error y "catch", qué hacer en dicho error  *
*****************************************************************/
   public SGBDD()
      {super();
      try
         {InitialContext ctx = new InitialContext();
         ds = (DataSource) ctx.lookup("java:comp/env/jdbc/BaseDeDatos");
         }
      catch (NamingException e)
         {JOptionPane.showMessageDialog(null, "ERROR EN  LA  CONEXIÓN A
         LA BASE DE DATOS", "ERROR", JOptionPane.WARNING_MESSAGE);
         }
      }

/*****************************************************************
*Función pública que  no recibe  argumento alguno,  mas devuelve un objeto de tipo*
*RestultSes, es decir, el resultado de una consulta a la base de datos del sistema      *
```

```
**********************************************************************/
   public ResultSet getRs()
      {return rs;
      }

/**********************************************************************
*Función pública re recibe una cadena de caracteres con una sentencia  SQL (varsql)*
*la cual permite el registro de algunos datos en la BDD, y un mensaje (mensaje) que*
*señala el éxito de tal acción.  También se vé secciones “try” y “catch”          *
**********************************************************************/
   public void RegistrarDatos(String varsql, String mensaje)
      {try
         {con = ds.getConnection("db2admin", "db2admin");
         Statement stm = con.createStatement();
         boolean flag = stm.execute(varsql);

         if(!mensaje.equalsIgnoreCase(""))
            JOptionPane.showMessageDialog(null, mensaje, "REGISTRO DE
            DATOS", JOptionPane.WARNING_MESSAGE);
         }
      catch (SQLException e)
         {JOptionPane.showMessageDialog(null, "ERROR EN EL REGISTRO DE
         DATOS", "ERROR", JOptionPane.WARNING_MESSAGE);
         }
      finally
         {if(con != null)
            {try
               {con.close();
               }
            catch (SQLException e1)
               {JOptionPane.showMessageDialog(null, "NO SE CERRÓ
               CORRECTAMENTE LA BASE DE DATOS", "ERROR",
               JOptionPane.WARNING_MESSAGE);
               }
            }
         }

/**********************************************************************
*Esta función  pública recibe  una sentencia SQL  que consulta ciertos datos propios*
*la base de datos.   La sección “try” presenta setencias que permiten la conexión con*
*la BDD, y si alguna de éstas genera cierta excepción, el “catch” indica que hacer    *
**********************************************************************/
   public void ConsultarDatos(String varsql)
```

```
    {boolean flag;

    try
       {con = ds.getConnection("db2admin", "db2admin");
       Statement stm = con.createStatement();
       flag = stm.execute(varsql);
       if(flag)
          rs = stm.getResultSet();
       }
    catch (SQLException e)
       {JOptionPane.showMessageDialog(null, "ERROR EN LA CONSULTA DE
       DATOS", "ERROR", JOptionPane.WARNING_MESSAGE);
       }
    }

/**********************************************************************
*Esta función pública recibe una sentencia SQL que consulta en la base de datos*
*todos los registros presentes en cualquier tabla, de manera que con la sentencia*
*"while" presente en esta función se cuente la cantidad de registros devueltos y así *
*determinar el código del próximo registro a ingresar                *
**********************************************************************/
   public int DeterminarCodigo(String sqlvar)
      {boolean flag;
      int cantidades = 0;
/**********************************************************************
*La sección "try" presenta setencias que permiten la conexión con la BDD, con la*
*finalidad de hacer una consulta determinada, y si alguna de estas sentencias genera*
*cierta excepción, el "catch" indica que hacer en dicho caso          *
**********************************************************************/
      try
         {con = ds.getConnection("db2admin", "db2admin");
         Statement stm = con.createStatement();
         flag = stm.execute(sqlvar);

         if(flag)
            {rs = stm.getResultSet();
            while(rs.next())
               {cantidades++;
               }
            }
         }
      catch (SQLException e)
         {JOptionPane.showMessageDialog(null, "ERROR AL CONSULTAR LOS
```

```
        DATOS", "ERROR", JOptionPane.WARNING_MESSAGE);
        }
      return cantidades;
      }

/*********************************************************************
*Esta  función pública  presente en la clase  SGBDD no recibe argumentos alguno ni*
*devulve algún valor.  La misma  tiene la función de cerrar la conexión con la BDD*
*Si  ocurre un error  en la zona del "try",  muestra el mensaje de la zona del "catch"*
*********************************************************************/
   public void CerrarBDD()
      {if(con != null)
         {try
            {con.close();
            System.out.println("SE CERRO LA BASE DE DATOS");
            }
         catch (SQLException e1)
            {JOptionPane.showMessageDialog(null, "NO SE CERRÓ
            CORRECTAMENTE LA BASE DE DATOS", "ERROR",
            JOptionPane.WARNING_MESSAGE);
            }
         }
      }
   }
```

ObjetoComun.java

```
package modelo;
/*********************************************************************
*Librería que es importada por esta clase o archivo para su normal funcionamiento  *
*********************************************************************/
import java.text.DecimalFormat;

/*********************************************************************
*Clase  pública que  contiene ciertas  funciones Java  comunes para  otras clases,  de*
*manera que puedan ser solicitadas en cualquier instante que se necesite          *
*********************************************************************/
public class ObjetoComun
   {public ObjetoComun()
      {super();
      }

/*********************************************************************
*Esta función altera el orden de colocación de los caracteres de un dato tipo "fecha",*
*de manera que en la interfaz de usuario se muestre ordenada, y en la BDD, al revés *
```

```
**********************************************************************/
   public static String cambioFechas(String fecha)
      {String nuevaFecha[] = null;

      if(fecha.length() > 0)
         {nuevaFecha = fecha.split("/");
         nuevaFecha[2] += "/" + nuevaFecha[1];
         nuevaFecha[2] += "/" + nuevaFecha[0];
         return nuevaFecha[2];
         }
      else
         return fecha;
      }

/**********************************************************************
*Esta función determina  la longitud del  argumento recibido (valor),  y de acuerdo a*
*dicha longitud, se le anexa ceros a dicha variable para completar seis dígitos.  Al  *
*final devuelve la misma variable recibida (valor) pero modificada                    *
**********************************************************************/
   public static String codigos(String valor)
      {switch(valor.length())
         {case 1:  valor = "00000" + valor; break;
         case 2:  valor = "0000" + valor; break;
         case 3:  valor = "000" + valor; break;
         case 4:  valor = "00" + valor; break;
         case 5:  valor = "0" + valor;
         }
      return valor;
      }
   }
```

Usuario.java

```
package modelo;
/**********************************************************************
*Librerías que son importadas por esta clase o archivo para su normal desempeño     *
**********************************************************************/
import java.io.Serializable;
import java.sql.SQLException;
import java.util.Date;
import java.util.Random;

/**********************************************************************
```

```
*Esta es una clase pública que implementa  la interfaz “Serializable”,  que comienza*
*con el conjunto de variables protegidas,  ya que de esta se derivan otras clases,  son*
*heredadas por las clases que extienden de esta, de allí su carácter protegido          *
**********************************************************************/
public class Usuario implements Serializable
   {protected String CodUsuario;
   protected String CodEmpresa;
   protected String DireccionUsuario;
   protected String CedulaUsuario;
   protected String NombresUsuario;
   protected String ApellidosUsuario;
   protected String FechaRegistroUsuario;
   protected String CodTlfUsuario1;
   protected String TelefonoUsuario1;
   protected String CodTlfUsuario2;
   protected String TelefonoUsuario2;
   protected String LoginUsuario;
   protected String ClaveUsuario;
   protected String CargoUsuario;
   protected String CorreoUsuario;
   protected String TipoUsuario;
   protected String SituacionUsuario;
   protected String CodVisita;
   protected boolean EstaConectado;
/**********************************************************************
*Esta función pública representa el constructor  de la clase Usuario,  no recibe algún*
*argumento,  tampoco  devuelve resultado alguno,  y se inicializan los valores de los*
*diversos atributos mencionados en la sección anterior,  con un valor acoorde con su*
*tipo de variable asignado                                                          *
**********************************************************************/
   public Usuario()
      {super();
      CodUsuario = "";
      CodEmpresa = "";
      DireccionUsuario = "";
      CedulaUsuario = "";
      NombresUsuario = "";
      ApellidosUsuario = "";
      FechaRegistroUsuario = "";
      CodTlfUsuario1 = "";
      TelefonoUsuario1 = "";
      CodTlfUsuario2 = "";
      TelefonoUsuario2 = "";
```

```
        LoginUsuario = "";
        ClaveUsuario = "";
        CargoUsuario = "";
        CorreoUsuario = "";
        TipoUsuario = "";
        SituacionUsuario = "ACTIVO";
        CodVisita = "";
        EstaConectado = false;
        }

/*************************************************************************
*Estas dos  funciones siguientes  se encargan  exclusivamente de  recibir y enviar  el*
*valor  de la  variable codVisita.  Se conocen en Java como funciones getter y setter*
*respectivamente,  las cuales  permiten el  encapsulamiento de  los datos como*
*propiedad característica de la programación orientada a objetos                *
*************************************************************************/
    public String getCodVisita()
        {return CodVisita;
        }

    public void setCodVisita(String codVisita)
        {CodVisita = codVisita;
        }
    }
```

Sesion.java

```
package modelo;
/*************************************************************************
*Librerías que son importadas por esta clase o archivo para su normal desempeño     *
*************************************************************************/
import java.io.Serializable;
import java.sql.SQLException;
import java.text.DateFormat;
import java.util.ArrayList;
import java.util.Date;

/*************************************************************************
*Esta clase pública es la precursora del objeto Sesión,  implementa la interfaz*
*"Serializable",  y presenta  algunas variables privadas de tipo caracter propias de el*
*objeto Sesión en cuestión                                                      *
*************************************************************************/
public class Sesion implements Serializable
    {private String CodSesion;
    private String CodUsuario;
```

```
   private String FechaSesion;
   private String HoraEntradaSesion;
   private String HoraSalidaSesion;
   private String AccionesSesion;

/**************************************************************************
*Esta  función es  el constructor  de la  clase Sesion.  Es pública  para que pueda ser*
*accedida por cualquier otra clase desde otro lugar  del entorno Web.  Se observa la*
*inicialización de todas las variables mencionadas en la sección anterior,  además de*
*nombrarse otras nuevas variables                                                    *
**************************************************************************/
   public Sesion()
      {super();
      DateFormat fecha =
      DateFormat.getDateInstance(java.text.DateFormat.DEFAULT);
      DateFormat hora =
      DateFormat.getTimeInstance(java.text.DateFormat.DEFAULT);
      CodSesion = "";
      CodUsuario = "000000";
      FechaSesion = ObjetoComun.cambioFechas(fecha.format(new Date()));
      HoraEntradaSesion = hora.format(new Date());
      HoraSalidaSesion = HoraEntradaSesion;
      AccionesSesion = "";
      }
/**************************************************************************
*Estas son las funciones getter y setter de las variables privadas de la clase Sesion    *
**************************************************************************/
   public String getCodSesion()
      {return CodSesion;
      }

   public void setCodSesion(String codSesion)
      {CodSesion = codSesion;
      }

   public String getCodUsuario()
      {return CodUsuario;
      }

   public void setCodUsuario(String codUsuario)
      {CodUsuario = codUsuario;
      }
```

```
   public String getFechaSesion()
      {return FechaSesion;
      }

/**************************************************************************
*En  esta función se  hace un  llamado de  forma estática  a la función  presente en la*
*clase ObjetoComun, es decir, sin instanciar el objeto llamado "ObjetoComun"        *
**************************************************************************/
   public void setFechaSesion(String fechaSesion)
      {FechaSesion = ObjetoComun.cambioFechas(fechaSesion);
      }

   public String getHoraEntradaSesion()
      {return HoraEntradaSesion;
      }

   public void setHoraEntradaSesion(String horaEntradaSesion)
      {HoraEntradaSesion = horaEntradaSesion;
      }

   public String getHoraSalidaSesion()
      {return HoraSalidaSesion;
      }
   public void setHoraSalidaSesion(String horaSalidaSesion)
      {HoraSalidaSesion = horaSalidaSesion;
      }

   public String getAccionesSesion()
      {return AccionesSesion;
      }

   public void setAccionesSesion(String accionesSesion)
      {AccionesSesion = accionesSesion;
      }

/**************************************************************************
*Esta función  recibe como  argumento una  variable que  posee el  código de sesión*
*a modificarse, mientras que no devuelve resultado alguno (void).  La misma busca*
*dicha sesión en la tabla SESIONES y la modifica posteriormente según se necesita *
**************************************************************************/
   public void ModificarSesion(String sesion)
```

```
    {SGBDD cdbdd = new SGBDD();
    Sesion sesi = new Sesion();
    String varsql = "SELECT * FROM H.SESIONES WHERE CodSesion = '" +
    sesion + "';";
    sesi = sesi.ConsultarSesion(varsql);
    if(!sesi.AccionesSesion.equalsIgnoreCase(""))
       if(!sesi.AccionesSesion.endsWith(this.AccionesSesion))
          this.AccionesSesion = sesi.AccionesSesion + ", " + this.AccionesSesion;
    varsql = "UPDATE H.SESIONES SET AccionesSesion = '" +
    this.AccionesSesion + "' WHERE CodSesion = '" + sesion + "';";
    cdbdd.ModificarDatos(varsql, "", "");
    }
  }
```

Servicio.java

```
package modelo;

/**************************************************************************
*Las siguientes representan las librerías que son importadas por la clase Servicio*
*para su normal funcionamiento                                             *
**************************************************************************/
import java.io.Serializable;
import java.sql.SQLException;
import java.util.ArrayList;
/**************************************************************************
*Clase pública precursora del objeto Servicio, implementa la interfaz ”Serializable”,*
*y presenta algunas variables privadas de tipo carácter propias del objeto Servicio  *
**************************************************************************/
public class Servicio implements Serializable
  {private String CodServicio;
  private String CodFactura;
  private String EnunciadoServicio;
  private String DescripcionServicio;
  private String SituacionServicio;
  private String[] FotosServicios = {"", "", "", "", ""};
  private String[] ComentarioFoto = {"", "", "", "", ""};

/**************************************************************************
*Constructor de la clase Servicio, donde se inicializan las variables de la misma      *
**************************************************************************/
  public Servicio()
    {super();
```

```
        CodServicio = "";
        CodFactura = "";
        EnunciadoServicio = "";
        DescripcionServicio = "";
        SituacionServicio = "ACTIVO";
        }

/**********************************************************************
*Estas son las funciones getter y setter de las variables privadas de la clase Servicio *
**********************************************************************/
    public String getCodServicio()
        {return CodServicio;
        }

    public void setCodServicio(String codServicio)
        {CodServicio = codServicio;
        }

    public String getCodFactura()
        {return CodFactura;
        }

    public void setCodFactura(String codFactura)
        {CodFactura = codFactura;
        }
    public String getDescripcionServicio()
        {return DescripcionServicio;
        }


    public void setDescripcionServicio(String descripcionServicio)
        {DescripcionServicio = descripcionServicio.trim().toUpperCase();
        }

    public String getEnunciadoServicio()
        {return EnunciadoServicio;
        }


    public void setEnunciadoServicio(String enunciadoServicio)
        {EnunciadoServicio = enunciadoServicio.trim().toUpperCase();
        }
```

```
   public String getSituacionServicio()
      {return SituacionServicio;
      }


   public void setSituacionServicio(String situacionServicio)
      {SituacionServicio = situacionServicio;
      }

   public String[] getComentarioFoto()
      {return ComentarioFoto;
      }


   public void setComentarioFoto(String[] comentarioFoto)
      {ComentarioFoto = comentarioFoto;
      }

   public String[] getFotosServicios()
      {return FotosServicios;
      }

   public void setFotosServicios(String[] fotosServicios)
      {FotosServicios = fotosServicios;
      }
/**********************************************************************
*Esta función se encarga de asignar un código adecuado a un nuevo servicio o*
*actividad. No recibe argumento alguno y devuelve un valor entero. Le envía una*
*sentencia SQL a la función "DeterminarCodigo" presente en la clase SGBDD      *
**********************************************************************/
   public int DeterminarCodigo()
      {String varsql = "SELECT * FROM H.SERVICIOS;";
      SGBDD cdbdd = new SGBDD();
      int codigo;

      codigo = cdbdd.DeterminarCodigo(varsql);
      cdbdd.CerrarBDD();
      return ++codigo;
      }

/**********************************************************************
*Esta función se encarga de ingresar los datos de un servicio o actividad en la BDD.*
*Recibe una variable de tipo cadena de caracteres y no devuelve variable alguna.*
```

```
*Posee una sentencia SQL que registra en la tabla SERVICIOS los datos del mismo *
**********************************************************************/
   public void RegistrarServicio(String mensaje)
      {SGBDD cdbdd = new SGBDD();
      String varsql = "INSERT INTO H.SERVICIOS VALUES ('" + this.CodServicio
      + "', '" + this.CodFactura + "', '" + this.EnunciadoServicio + "', '" +
      this.DescripcionServicio + "', '" + this.SituacionServicio + "', '" +
      this.FotosServicios[0] + "', '" + this.FotosServicios[1] + "', '" +
      this.FotosServicios[2] + "', '" + this.FotosServicios[3] + "', '" +
      this.FotosServicios[4] + "', '" + this.ComentarioFoto[0] + "', '" +
      this.ComentarioFoto[1] + "', '" + this.ComentarioFoto[2] + "', '" +
      this.ComentarioFoto[3] + "', '" + this.ComentarioFoto[4] + "');";
      cdbdd.RegistrarDatos(varsql, mensaje);
      }

/**********************************************************************
*Esta función se encarga de consultar datos propios de la  tabla SERVICIOS usando*
*una sentencia SQL recibida como argumento.  Devuelve un objeto de tipo Servicio.*
*Presenta ciertas nuevas varibles, y secciones “try” y “catch” para resolver errores   *
**********************************************************************/
   public Servicio ConsultarServicio(String varsql)
      {SGBDD cdbdd = new SGBDD();
      Servicio servicio = null;
      String direcciones[] = new String[5], comentarios[] = new String[5];
      int i;
      cdbdd.ConsultarDatos(varsql);
      try
         {if(cdbdd.getRs().next())
            {servicio = new Servicio();
            servicio.setCodServicio(cdbdd.getRs().getString(1).trim());
            servicio.setCodFactura(cdbdd.getRs().getString(2).trim());
            servicio.setEnunciadoServicio(cdbdd.getRs().getString(3).trim());
            servicio.setDescripcionServicio(cdbdd.getRs().getString(4).trim());
            servicio.setSituacionServicio(cdbdd.getRs().getString(5).trim());
            for(i = 6; i < 11; i++)
               {direcciones[i-6] = cdbdd.getRs().getString(i).trim();
               comentarios[i-6] = cdbdd.getRs().getString(i+5).trim();
               }
            servicio.setFotosServicios(direcciones);
            servicio.setComentarioFoto(comentarios);
            }
         }
      catch (SQLException e)
```

```
        {e.printStackTrace();
        }
      cdbdd.CerrarBDD();
      return servicio;
      }
  }
```

Factura.java

```
package modelo;

/*************************************************************************
*Librerías que son importadas por esta clase o archivo para su normal desempeño     *
*************************************************************************/
import java.io.Serializable;
import java.sql.SQLException;
import java.text.DateFormat;
import java.util.ArrayList;
import java.util.Date;

/*************************************************************************
*Clase pública precursora del objeto Factura,  implementa  la interfaz ”Serializable”,*
*y presenta  algunas variables privadas de tipo carácter propias de el objeto Factura  *
*************************************************************************/
public class Factura implements Serializable
   {private String CodFactura;
   private String OrigenFactura;
   private String ProcedenciaFactura;
   private String LugarFactura;
   private String FechaFactura;
   private String EdSFactura;
   private String SituacionFactura;
   private String FormaPagoFactura[];
   private String PagoOtraFormaFactura;
   private double IVAFactura;

/*************************************************************************
*Este es  el constructor  de la clase Factura,  es de  caracter público,  y se  observa la*
*inicialización de todos las variables mencionadas en la sección anterior,  además de*
*nombrarse otras nuevas variables                                                      *
*************************************************************************/
   public Factura()
      {super();
      DateFormat fecha = DateFormat.getDateInstance(DateFormat.DEFAULT);
```

```
        CodFactura = "";
        OrigenFactura = "nueva";
        ProcedenciaFactura = "000000";
        LugarFactura = "Puerto La Cruz";
        FechaFactura = fecha.format(new Date());
        EdSFactura = "-";
        SituacionFactura = "POR CANCELAR";
        PagoOtraFormaFactura = "-";
        IVAFactura = 0.0;
        }

/**************************************************************************
*Estas son las funciones getter y setter de las variables privadas de la clase Factura   *
**************************************************************************/
    public String getEdSFactura()
        {return EdSFactura;
        }

    public void setEdSFactura(String edsFactura)
        {EdSFactura = edsFactura;
        }

    public String getProcedenciaFactura()
        {return ProcedenciaFactura;
        }
    public void setProcedenciaFactura(String procedenciaFactura)
        {ProcedenciaFactura = procedenciaFactura;
        }

    public String getLugarFactura()
        {return LugarFactura;
        }

    public void setLugarFactura(String lugarFactura)
        {LugarFactura = lugarFactura.trim().toUpperCase();
        }

    public String getCodFactura()
        {return CodFactura;
        }

    public void setCodFactura(String codFactura)
        {CodFactura = codFactura;
```

```
    }

public String getFechaFactura()
    {return FechaFactura;
    }

public void setFechaFactura(String fechaFactura)
    {FechaFactura = ObjetoComun.cambioFechas(fechaFactura);
    }

public double getIVAFactura()
    {return IVAFactura;
    }

public void setIVAFactura(double factura)
    {IVAFactura = factura;
    }

public String getSituacionFactura()
    {return SituacionFactura;
    }

public void setSituacionFactura(String situacionFactura)
    {SituacionFactura = situacionFactura;
    }
public String getOrigenFactura()
    {return OrigenFactura;
    }
public void setOrigenFactura(String origenFactura)
    {OrigenFactura = origenFactura;
    }

public String getFormaPagoFactura(int i)
    {return FormaPagoFactura[i];
    }

public String[] getFormaPagoFactura()
    {return FormaPagoFactura;
    }

public void setFormaPagoFactura(String[] formaPagoFactura)
    {FormaPagoFactura = formaPagoFactura;
    }
```

```
   public String getPagoOtraFormaFactura()
      {return PagoOtraFormaFactura;
      }

   public void setPagoOtraFormaFactura(String pagoOtraFormaFactura)
      {PagoOtraFormaFactura = pagoOtraFormaFactura;
      }

/*********************************************************************
*Esta función consulta los datos propios de la tabla FACTURAS mediante una*
*sentencia SQL recibida como argumento.  Devuelve un objeto de tipo Factura  *
*********************************************************************/
   public Factura ConsultarFactura(String varsql)
      {SGBDD cdbdd = new SGBDD();
      String valor;
      Factura factura = null;

      cdbdd.ConsultarDatos(varsql);
      try
         {if(cdbdd.getRs().next())
            {factura = new Factura();
            factura.setCodFactura(cdbdd.getRs().getString(1).trim());
            factura.setEdSFactura(cdbdd.getRs().getString(2).trim());
            factura.setLugarFactura(cdbdd.getRs().getString(3).trim());
            factura.setFechaFactura(cdbdd.getRs().getString(4).trim());
            factura.setSituacionFactura(cdbdd.getRs().getString(5).trim());
            factura.setIVAFactura(cdbdd.getRs().getFloat(6));
            factura.setFormaPagoFactura(cdbdd.getRs().getString(7).trim().split(","));
            factura.setPagoOtraFormaFactura(cdbdd.getRs().getString(8).trim());
            }
         }
      catch (SQLException e)
         {e.printStackTrace();
         }

      cdbdd.CerrarBDD();
      return factura;
      }

/*********************************************************************
*Esta función se encarga de consultar datos propios de la  tabla FACTURAS usando*
*una  sentencia SQL  recibida como  argumento.  Devuelve  un objeto de tipo*
```

```
* ArrayList, es decir, una lista de objetos, que en este caso son de tipo Factura         *
**********************************************************************/
   public ArrayList ConsultarFacturas(String varsql)
      {SGBDD cdbdd = new SGBDD();
      ArrayList FacturaList = new ArrayList();
      Factura factura = null;

      cdbdd.ConsultarDatos(varsql);
      try
         {while(cdbdd.getRs().next())
            {factura = new Factura();
            factura.setCodFactura(cdbdd.getRs().getString(1).trim());
            varsql = "SELECT * FROM H.FACTURAS WHERE CodFactura = '" +
            factura.getCodFactura() + "';";
            factura = factura.ConsultarFactura(varsql);
            FacturaList.add(factura);
            }
         }
      catch (SQLException e)
         {e.printStackTrace();
         }
      cdbdd.CerrarBDD();
      return FacturaList;
      }
   }
```

Servicios.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<HTML>
   <HEAD>
      <META http-equiv = "Content-Type" content = "text/html;
      charset=ISO-8859-1">
      <META NAME = "GENERATOR" CONTENT = "IBM Software Development
      Platform">
      <META http-equiv = "Content-Style-Type" content = "text/css">
      <LINK HREF = 'CSS/TodosEstilos.css' TYPE = 'text/css' REL = 'stylesheet'>
      <TITLE>EMPRESAS.html</TITLE>
   </HEAD>
   <BODY>
<!-- Aquí se muestra el código HTML que genera una página Web sencilla y de
contenido estático, es decir, que no cambia con el tiempo ni según el usuario que la
solicita o consulta, y que muestra una imagen informativa que hace alusión a los
Servicios y Actividades que ofrece la empresa HIDROCONS, C. A. a su clientela.
```

```
Posee etiquetas propias de este lenguaje para formar tablas, imágenes, así como
enlaces hacia otros archivos para la buena presentación de la página en cuestión -->
      <TABLE BORDER = "1">
         <TR>
            <TD WIDTH = "623">
               <IMG BORDER = '0' SRC = imagenes/Repuestos/SERVICIOS.jpg'
               WIDTH = '615' HEIGHT = '165'>
            </TD>
         </TR>
      </TABLE>
   </BODY>
</HTML>
```

NewServicioEmpl.java

```
package vista;

/**********************************************************************
*Las siguientes representan las librerías que son importadas por este servlet*
*(un servlet es un generador dinámico Java de páginas y respuestas a solicitudes*
*Web) para su normal funcionamiento                                   *
**********************************************************************/
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.Iterator;
import javax.servlet.RequestDispatcher;
import javax.servlet.Servlet;
import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import modelo.Factura;
import modelo.ObjetoComun;
import modelo.Servicio;

/**********************************************************************
*Esta es una clase muy diferente a las que hasta ahora se han diseñado, ya que la*
*misma implementa y hereda interfaces que tienen quer ver con el concepto de*
*“Servlet”, además de trabajar con protocolo HTTP                     *
```

```
**********************************************************************/
public class NewServicioEmpl extends HttpServlet implements Servlet
   {public NewServicioEmpl()
      {super();
      }

/**********************************************************************
*Esta función  propia de un servlet,  se encarga de procesar solicitudes  Web del tipo*
*GET,  es decir,  no tienen ningún contenido de cuerpo,  de manera que dichos datos*
*se  pasan al  servidor  como cadena de  consulta en el URL.   Tiene dos argumentos*
*(req) que representa la solicitud, y (resp) la respuesta.  Dicha función se encarga de*
*generar de forma  dinámica, es decir,  cambiante,  de acuerdo a ciertas condiciones,*
*la página de respuesta a alguna solicitud elaborada por algún usuario autentificado  *
**********************************************************************/
   protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws
   ServletException, IOException
      {HttpSession session = req.getSession(false);
      ServletContext context = this.getServletContext();
      RequestDispatcher rd = null;
      PrintWriter out = resp.getWriter();
      Servicio servicio = new Servicio();
      ArrayList listaPresupuestos;
      Iterator iter = null;
      Factura factura = new Factura();
      Usuario usuario = (Usuario) session.getAttribute("UsuarioSes");
      Sesion sesion = new Sesion();
      String varsql, comando = req.getParameter("comando"), direcciones[],
      comentarios[];
      String direcciones1[] = {"", "", "", "", ""}, comentarios1[] = {"", "", "", "", ""};
      int i, numero = 0, a, posicion = 267, p = 0;

      numero = servicio.DeterminarCodigo();
      if(comando != null)
         if(comando.equalsIgnoreCase("REGISTRAR"))
            {servicio.setCodServicio(ObjetoComun.codigos(String.valueOf(numero)));
            servicio.setCodFactura(req.getParameter("cmbPresupuestos"));
            servicio.setEnunciadoServicio(req.getParameter("txtEnunciado"));
            servicio.setDescripcionServicio(req.getParameter("txtDescripcion"));
            direcciones = req.getParameterValues("TxtFileFoto");
            comentarios = req.getParameterValues("txtComentario");

            for(i = 0; i < direcciones.length; i++)
               {if(!direcciones[i].equalsIgnoreCase(""))
```

```
            {direcciones1[p] = direcciones[i];
            comentarios1[p++] = comentarios[i];
            }
         }

      for(i = 0; i < direcciones1.length; i++)
         {if(!direcciones1[i].equalsIgnoreCase(""))
            {a = direcciones1[i].lastIndexOf("imagenes");
            direcciones1[i] = direcciones1[i].substring(a);
            }
         else
            direcciones1[i] = "imagenes/Servicios/ImagenNoDisponible.jpg";
         }

      servicio.setFotosServicios(direcciones1);
      servicio.setComentarioFoto(comentarios1);
      servicio.RegistrarServicio("SERVICIO REGISTRADO CON ÉXITO");
      numero++;
      sesion.setAccionesSesion("INGRESAR SERVICIO");
      sesion.ModificarSesion(usuario.getCodVisita());
      servicio = new Servicio();
      }
   else
      {rd = context.getRequestDispatcher("SERVICIOS.html");
      rd.forward(req, resp);
      }
out.println("<!DOCTYPE HTML PUBLIC '-//W3C//DTD HTML 4.01
Transitional//EN'>"
+"<HTML><HEAD><META http-equiv='Content-Type' content='text/html;
charset=ISO-8859-1'>"
+"<META NAME = 'GENERATOR' CONTENT = 'IBM Software
Development Platform'><TITLE>NuevoCliente.html</TITLE>"
+"<SCRIPT LANGUAGE = 'JavaScript' SRC =
'JavaScripts/JavaScript.js'></SCRIPT>"
+"<LINK HREF = 'CSS/TodosEstilos.css' TYPE = 'text/css'
REL = 'stylesheet'></HEAD>"
+"<BODY BGCOLOR = '#FFFFCC'><FORM NAME = 'formul' METHOD =
'POST' ACTION = 'NewServicioEmpl' TARGET = 'principal'>"
+"<TABLE BORDER = '0'><TR><TD COLSPAN = '2' WIDTH = '623'
CLASS = 'textoCent14b'>DATOS DEL SERVICIO QUE PRESTA LA
EMPRESA</TD></TR>"
+"<TR><TD HEIGHT = '10' COLSPAN = '2'></TD></TR></TABLE>"
+"<TABLE BORDER = '0'><TR><TD WIDTH = '90'
```

```
CLASS = 'celdaGrizIzq10'>C&oacute;digo</TD><TD WIDTH = '130'
CLASS = 'textoIzq12b'>" + ObjetoComun.codigos(String.valueOf(numero)) +
"</TD>"
+"<TD WIDTH = '75' CLASS = 'celdaGrizIzq10'>Factura</TD><TD
WIDTH = '310'><SELECT NAME = 'cmbPresupuestos' STYLE = 'WIDTH:76;'
CLASS = 'textboxIzq11' OnChange = 'formul.txtEnunciado.focus();'><OPTION
VALUE = '-'>-</OPTION><OPTION VALUE =
'000000'>Ninguna</OPTION>");

varsql = "SELECT * FROM H.FACTURAS ORDER BY CodFactura;";
listaPresupuestos = factura.ConsultarFacturas(varsql);
iter = listaPresupuestos.iterator();

while(iter.hasNext())
   {factura = (Factura) iter.next();
   varsql = "SELECT * FROM H.SERVICIOS WHERE CodFactura = '" +
   factura.getCodFactura() + "' AND CodFactura <> '000000';";
   servicio = new Servicio();
   servicio = servicio.ConsultarServicio(varsql);

   if(servicio == null)
      out.println("<OPTION VALUE = '" + factura.getCodFactura() + "'>" +
      factura.getCodFactura() + "</OPTION>");
   }

out.println("</SELECT></TD></TR>"
+"<TR><TD CLASS = 'celdaGrizIzq10'>Enunciado</TD><TD
COLSPAN = '4'><TEXTAREA ROWS = '2' COLS = '82'
NAME = 'txtEnunciado'
CLASS = 'textoIzq12' OnChange = 'contarCaracteres(this, 500);'
OnKeyUp = 'contarCaracteres(this, 500);'></TEXTAREA></TD></TR>"
+"<TR><TD CLASS = 'celdaGrizIzq10'>Descripci&oacute;n</TD><TD
COLSPAN = '4'><TEXTAREA ROWS = '5' COLS = '82'
NAME = 'txtDescripcion' CLASS = 'textoIzq12'
OnChange = 'contarCaracteres(this, 1200);'
OnKeyUp = 'contarCaracteres(this, 1200);'></TEXTAREA></TD></TR>
</TABLE>"
+"<P><TABLE BORDER = '0'><TR><TD COLSPAN = '2'
CLASS = 'celdaGrizCent12'>FOTOS</TD></TR>");

for(i = 0; i < 5; i++)
   {out.println("<TR><TD WIDTH = '218' ROWSPAN = '4'><IMG
   BORDER = '0' SRC = 'imagenes/Servicios/ImagenNoDisponible.jpg'
```

```
        WIDTH = '217' HEIGHT = '160' NAME = 'ImgRepuesto'></TD>"
        +"<TD WIDTH = '399' HEIGHT = '20'
        CLASS = 'celdaGrizCent10b'>Direcci&oacute;n URL</TD></TR>"
        +"<TR><TD HEIGHT = '64' ALIGN = 'CENTER'>
        <DIV STYLE = 'position:absolute;
        left:255px; top:" + posicion + "px; visibility: visible'><INPUT
        TYPE = 'TEXT' NAME = 'TxtFileFoto' SIZE = '50' READONLY
        CLASS = 'textboxIzq11'></DIV>"
        +"<INPUT TYPE = 'file' NAME = 'FileFoto' SIZE = '50'
        CLASS = 'textboxIzq11'
        OnChange = (document,this.value," + i + ");>"
        +"<BR><INPUT TYPE = 'BUTTON' NAME = 'comando'
        VALUE = 'BORRAR URL Y COMENTARIO' CLASS = 'boton8'
        OnClick = borrarFotoServ(document," + i + ");></TD></TR>"
        +"<TR><TD height = '20'
        CLASS = 'celdaGrizCent10b'>Comentario</TD></TR>"
        +"<TR><TD HEIGHT = '56' ALIGN = 'CENTER'><TEXTAREA
        ROWS = '3' COLS = '60' NAME = 'txtComentario' CLASS = 'textoIzq12'
        OnChange = 'contarCaracteres(this, 300);'
        OnKeyUp = 'contarCaracteres(this, 300);'></TEXTAREA></TD></TR>");
        posicion += 168;
        }

    out.println("<TABLE BORDER = '0'><TR><TD HEIGHT = '10'
    WIDTH = '623'></TD></TR>"
    +"<TR><TD ALIGN = 'CENTER'><INPUT TYPE = 'SUBMIT'
    NAME = 'comando'
    VALUE = 'REGISTRAR' CLASS = 'boton12'
    OnClick = 'return validarServicioServ(document, formul);'> "
    +"<INPUT TYPE = 'RESET' VALUE = 'LIMPIAR' CLASS = 'boton12'> "
    +"<INPUT TYPE = 'SUBMIT' CLASS = 'boton12' NAME = 'comando'
    VALUE = 'SALIR SECCIÓN'
    OnClick = 'return permisoSalirSeccion();'></TD></TR>"
    +"</TABLE></FORM></BODY></HTML>");
    }

protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException
    {this.doGet(req, resp);
    }
}
```

5.5. Pruebas

Las pruebas constituyen una de las etapas más importantes del ciclo de vida de desarrollo de software. Un producto de software que se desarrolla se debe entregar al cliente libre de defectos y de errores. La prueba es el proceso de ejecutar un programa con la intención de descubrir defectos en el mismo. De allí que el objetivo de esta etapa es detectar en la aplicación cualquier posible funcionamiento erróneo antes de entregarlo a los usuarios definitivos.

- **Pruebas de Unidad:** las pruebas de unidad evalúan los componentes implementados como unidades individuales. Ayudan a que el módulo probado se haga independiente, es decir, sin tomar en cuenta el resto del sistema. Conociendo la función específica para la cual se diseñará el producto, se aplican pruebas que demuestren que cada función es plenamente operacional, mientras se buscan los errores en dichas funciones. Para efectuar las pruebas de unidad se utiliza la técnica de pruebas de Caja Negra. La prueba de Caja Negra, también conocida como prueba Funcional, es la que se aplica a la interfaz del software. Una prueba de este tipo examina algún aspecto funcional de un sistema que tiene poca relación con la estructura lógica del software. La prueba de Caja Negra se concentra en la funcionalidad global del software y considera la selección de los datos de prueba y la interpretación de los resultados de dicha prueba realizada basándose en las características funcionales del software. [10]

- **Partición Equivalente:** la partición equivalente es un método de prueba de Caja Negra que divide el dominio de entrada de un programa en clases de datos a partir de las cuales pueden derivarse casos de prueba. El objetivo principal es definir tan solo un caso de prueba en cada clase de equivalencia en lugar de múltiples casos de prueba que pertenecen a la misma clase de

equivalencia. La partición equivalente se esfuerza por definir un caso de prueba que descubra ciertas clases de errores, reduciendo así el número total de casos de prueba que deben desarrollarse. [10]

Una clase de equivalencia representa un conjunto de estados válidos y no válidos para las condiciones de entrada. Por lo general, una condición de entrada es un valor numérico específico, un rango de valores, un conjunto de valores relacionados o una condición booleana. Las clases de equivalencia se definen de acuerdo con ciertas directrices que ayuden a determinar todos los posibles casos de clases de equivalencia que imperan en un sistema a ser examinado.

5.5.1. Determinación de clases de equivalencia

Al aplicar estos conceptos para la derivación de clases de equivalencia, se desarrollarán y ejecutarán los casos de prueba para cada objeto de los datos del dominio de entrada. A continuación, en la tabla 5.1 de esta sección se detallan todas las posibles clases de equivalencia que intervienen en los casos de prueba del sistema en general.

Tabla 5.1. Clases de Equivalencia generales presentes en el sistema

N°. Clase	Nombre de la Clase	Descripción	Ejemplo a Utilizar
1	Fecha	Esta clase de equivalencia recibe primero dos números que representan el día seguido del carácter especial "/", luego dos números que representan el mes, igualmente seguido de otro "/", y finalmente cuatro números que indican el año.	10/07/2009
2	Numérico	Recibe de entrada sólo números enteros positivos. Tal es el caso	12345678

		de las cédulas, números de facturas, códigos de registros, NIT's y RIF's de empresas.	
3	Flotante	Recibe números seguidos de un punto decimal, y de uno, dos o ningún número adicional que constituya los decimales. Aquí se incluyen los precios y las cantidades de repuestos y servicios.	1234.56
4	Alfanumérico	Recibe caracteres alfabéticos en mayúscula o en minúscula, además de numéricos y caracteres especiales. Este tipo de datos son los que recibe los nombres de las empresas, de los clientes autentificados, nombre de los repuestos en venta, nombre de los servicios que presta la empresa.	?LU"IS_car'los/3
5	E-Mail	Procesa tipos de datos formados por una cadena de caracteres alfanuméricos, seguida por el carácter arroba "@", luego otra cadena de caracteres seguida de un punto para finalmente terminar con una cadena de hasta tres caracteres más.	Ab-2@kmn3.xyz
6	Valor vacío	Se refiere a que ningún dato se ingresa en el campo, también es conocido como una cadena de caracteres vacía. Existen algunos campos que sí aceptan este valor (dejando sus campos libres) mientras que otros no.	""

5.5.2. Casos de prueba de caja negra

A continuación se muestran ciertos casos de prueba de Caja Negra en que se resumen el uso de las clases de equivalencia anteriormente descritas. La idea de estos

casos de prueba es ingresar en los campos de texto requeridos un valor representativo de cada clase de equivalencia y determinar si es un argumento válido o no para el campo en cuestión, tanto en lo que se refiere a tipo de datos que representa como en longitud de dicho conjunto de caracteres. Así, en cada campo se ingresarán cada una de las seis clases de equivalencia encontradas anteriormente, y si dicho valor representativo de dicha clase de equivalencia es aceptable teóricamente, y a la vez es aceptado por el sistema, en función de sus requerimientos tanto de tipo de dato (cadena de caracteres, numérico, flotante, fecha, formato de E-Mail, cadena vacía, etc.) como de longitud (por ejemplo, un campo cédula sólo acepta 8 caracteres, ni un caracter más, ni un caracter menos) pues se cotejará como "Válido" en dicho campo, de lo contrario será declarado "No Válido". Véase tabla 5.2.

Tabla 5.2. Caso de Prueba de Caja Negra "Iniciar Sesión Usuario"

Nombre del Campo	Caso de Prueba	Válido	No Válido	Clase de Equiv. Cubierto
Nombre de Usuario o Login	10/07/2009	✓		1
	12345678	✓		2
	1234,56	✓		3
	?LU"IS_car'los/3	✓		4
	Ab-2@kmn3.xyz	✓		5
	""		✓	6
Contraseña o Password	10/07/2009	✓		1
	12345678	✓		2
	1234,56	✓		3
	?LU"IS_car'los/3	✓		4
	Ab-2@kmn3.xyz	✓		5
	""		✓	6

La tabla 5.3 siguiente muestra las pruebas del caso de uso "Registrar Empresa".

Tabla 5.3. Caso de Prueba de Caja Negra "Registrar Empresa"

Nombre del Campo	Caso de Prueba	Válido	No Válido	Clase de Equiv. Cubierto
Nombre de la Empresa	10/07/2009	✓		1
	12345678	✓		2
	1234,56	✓		3
	?LU"IS_car'los/3	✓		4
	Ab-2@kmn3.xyz	✓		5
	""		✓	6
RIF de la Empresa	10/07/2009		✓	1
	12345678	✓		2
	1234,56		✓	3
	?LU"IS_car'los/3		✓	4
	Ab-2@kmn3.xyz		✓	5
	""		✓	6
NIT de la Empresa	10/07/2009		✓	1
	12345678		✓	2
	1234,56		✓	3
	?LU"IS_car'los/3		✓	4
	Ab-2@kmn3.xyz		✓	5
	""	✓		6
Dirección Fiscal donde reside la Empresa	10/07/2009	✓		1
	12345678	✓		2
	1234,56	✓		3
	?LU"IS_car'los/3	✓		4
	Ab-2@kmn3.xyz	✓		5
	""		✓	6
Ciudad de ubicación de la Empresa	10/07/2009	✓		1
	12345678	✓		2
	1234,56	✓		3
	?LU"IS_car'los/3	✓		4
	Ab-2@kmn3.xyz	✓		5
	""		✓	6

Tabla 5.3. Caso de Prueba de Caja Negra "Registrar Empresa" (Continuación)

Nombre del Campo	Caso de Prueba	Válido	No Válido	Clase de Equiv. Cubierto
Correo Electrónico de la Empresa	10/07/2009		✓	1
	12345678		✓	2
	1234,56		✓	3
	¿LU"IS_car'los/3		✓	4
	Ab-2@kmn3.xyz	✓		5
	""		✓	6

Imagínese que algún usuario, como un Empleado de HIDROCONS, modifica los datos propios o de otro usuario. He aquí en la tabla 5.4 las pruebas respectivas:

Tabla 5.4. Caso de Prueba de Caja Negra "Modificar Usuario"

Nombre del Campo	Caso de Prueba	Válido	No Válido	Clase de Equiv. Cubierto
Nombres del Usuario	10/07/2009	✓		1
	12345678	✓		2
	1234,56	✓		3
	¿LU"IS_car'los/3	✓		4
	Ab-2@kmn3.xyz	✓		5
	""		✓	6
Apellidos del Usuario	10/07/2009	✓		1
	12345678	✓		2
	1234,56	✓		3
	¿LU"IS_car'los/3	✓		4
	Ab-2@kmn3.xyz	✓		5
	""		✓	6
Cédula de Identidad del Usuario	10/07/2009		✓	1
	12345678	✓		2
	1234,56		✓	3
	¿LU"IS_car'los/3		✓	4
	Ab-2@kmn3.xyz		✓	5
	""		✓	6

Tabla 5.4. Caso de Prueba de Caja Negra "Modificar Usuario" (Continuación)

Nombre del Campo	Caso de Prueba	Válido	No Válido	Clase de Equiv. Cubierto
Dirección del Usuario	10/07/2009	✓		1
	12345678	✓		2
	1234,56	✓		3
	¿LU"IS_car'los/3	✓		4
	Ab-2@kmn3.xyz	✓		5
	""		✓	6
2do Teléfono del Usuario	10/07/2009		✓	1
	12345678	✓		2
	1234,56		✓	3
	¿LU"IS_car'los/3		✓	4
	Ab-2@kmn3.xyz		✓	5
	""		✓	6
E-Mail del Usuario	10/07/2009		✓	1
	12345678		✓	2
	1234,56		✓	3
	¿LU"IS_car'los/3		✓	4
	Ab-2@kmn3.xyz	✓		5
	""		✓	6

Si el Empleado de HIDROCONS ingresa alguna factura en particular acreditada a las cuentas por cobrar, la siguiente (tabla 5.5) muestra las pruebas de rigor:

Tabla 5.5. Caso de Prueba de Caja Negra "Registrar Factura"

Nombre del Campo	Caso de Prueba	Válido	No Válido	Clase de Equiv. Cubierto
Número de la Factura	10/07/2009		✓	1
	12345678		✓	2
	1234,56		✓	3
	¿LU"IS_car'los/3		✓	4
	Ab-2@kmn3.xyz		✓	5

	“”		✓	6

Tabla 5.5. Caso de Prueba de Caja Negra “Registrar Factura” (Continuación)

Nombre del Campo	Caso de Prueba	Válido	No Válido	Clase de Equiv. Cubierto
Lugar de emisión	10/07/2009	✓		1
	12345678	✓		2
	1234,56	✓		3
	?LU”IS_car’los/3	✓		4
	Ab-2@kmn3.xyz	✓		5
	“”		✓	6
Impuesto al Valor Agregado (IVA)	10/07/2009		✓	1
	12345678		✓	2
	1234,56		✓	3
	?LU”IS_car’los/3		✓	4
	Ab-2@kmn3.xyz		✓	5
	“”		✓	6
Descripción del ítem	10/07/2009	✓		1
	12345678	✓		2
	1234,56	✓		3
	?LU”IS_car’los/3	✓		4
	Ab-2@kmn3.xyz	✓		5
	“”		✓	6
Cantidad del ítem	10/07/2009		✓	1
	12345678	✓		2
	1234,56	✓		3
	?LU”IS_car’los/3		✓	4
	Ab-2@kmn3.xyz		✓	5
	“”		✓	6
Precio unitario del ítem	10/07/2009		✓	1
	12345678	✓		2
	1234,56	✓		3
	?LU”IS_car’los/3		✓	4
	Ab-2@kmn3.xyz		✓	5
	“”		✓	6

5.5.3. Casos de prueba de integración

Conociendo el funcionamiento interno del producto, se aplican pruebas para asegurarse que "todas las piezas encajan", en otras palabras, están integradas correctamente. Se prueban las rutas lógicas del software y la colaboración entre los componentes, al proporcionar casos de prueba que ejerciten conjuntos específicos de condiciones, bucles o ambos. Esta técnica de evaluación de software es denominada pruebas de Caja Blanca. Las pruebas se aplicaron a varios componentes de software desarrollados, comprobando así la exitosa integración de la aplicación. Además, se probaron los enlaces de las páginas, que no hubiesen enlaces rotos o vacíos, ni enlaces que direccionaran hacia alguna página indebida. Se comienza con el caso de uso que realiza un usuario al tratar de autentificarse, como se muestra en la tabla 5.6 para lograr la meta de ingresar como usuario válido:

Tabla 5.6. Caso de Prueba de Integración "Iniciar Sesión Usuario"

N°. Caso de Prueba	**01**	**Caso de Prueba**	**Iniciar Sesión Usuario**
Entrada	➢ **Nombre de usuario**: LuisGuevara ➢ **Contraseña**: 13167548.		
Resultado	El usuario es autentificado y de inmediato ingresa al sistema SIADCON.		
Condiciones	➢ El usuario debe estar registrado previamente en la BDD. ➢ El usuario debe estar en situación o estado ACTIVO. ➢ No debe haber iniciado sesión alguna.		
Procedimiento	1. Escribir el nombre de usuario y la contraseña en los cuadros de texto correspondientes. 2. Presione el botón "INGRESAR". 3. Se ingresa finalmente al sistema.		
Situación del Caso de	✓ **EXITOSO**		

Prueba

Luego de iniciar satisfactoriamente la sesión, se procede a registrar los datos de una empresa en el sistema, como se observa en la tabla 5.7. Dicho procedimiento lo puede realizar un Cliente Prospecto o representante de una potencial empresa cliente:

Tabla 5.7. Casos de Prueba de Integración "Registrar Empresa" y "Registrar Usuario"

N°. Caso de Prueba	**02**	**Casos de Prueba**	**Registrar Empresa y Registrar Usuario**
Entrada	➢ **Nombre de la Compañía**: E/S. RÍO CARIBE, C. A. ➢ **RIF de la Razón Social**: J-03136496-1 ➢ **NIT de la Razón Social**: 0044187353 ➢ **Estado**: Sucre ➢ **Ciudad**: Carúpano ➢ **Dirección Fiscal**: Carretera Carúpano - Río Caribe, Con Calle Principal De Puerto Santo, Sector "El Río". ➢ **Primer Teléfono**: (0294) 6640033 ➢ **Segundo Teléfono**: "" ➢ **Correo Electrónico**: es-rio-caribe@cantv.net ➢ **Nombres del Cliente**: Pedro José ➢ **Apellidos del Cliente**: Jiménez Zamora ➢ **Cédula de Identidad**: V25468697 ➢ **Primer Teléfono del Cliente**: (0414) 6856165 ➢ **Segundo Teléfono del Cliente**: "" ➢ **Cargo del Usuario**: Gerente General ➢ **E-Mail del Cliente:** pe-zamora@gmail.es		
Resultado	Los datos de la compañía nueva y del nuevo usuario son guardados en el sistema.		
Condiciones	➢ No existe un registro de empresa en la Base de Datos con un RIF o NIT igual al ingresado por pantalla. ➢ No existe un registro de usuario en la Base de Datos con una cédula igual a la ingresada por pantalla.		
Procedimiento	1. Seleccionar del menú desplegable izquierdo la opción de Registrar Nueva Empresa.		

	2. Escribir en los respectivos campos todos los datos de entrada de la empresa descritos anteriormente. 3. Escribir todos los datos de entrada del representante legal descritos anteriormente en sus respectivos campos. 4. Presionar el botón adecuado para registrar los datos. 5. Confirmar el mensaje de solicitud de procesamiento que el sistema presenta. 6. La compañía y el usuario ya se registraron en el sistema.
Situación del Caso de Prueba	✓ **EXITOSO**

Seguidamente de agregar los datos de la nueva empresa y registrarlos, y de ser aprobada su solicitud de ingreso al sistema, se procede a realizar una modificación de los datos del cliente ingresados previamente para comprobar la integración entre estos módulos. En este caso los procedimientos los llevará a cabo un Empleado de HIDROCONS registrado del sistema.

Tabla 5.8. Caso de Prueba de Integración "Modificar Usuario"

N°. Caso de Prueba	**03**	**Caso de Prueba**	**Modificar Usuario**
Entrada	➢ **Cédula de identidad del cliente como criterio usado de búsqueda**: V25468697 ➢ Modificar los siguientes datos: o Nombres anteriores del Usuario: Pedro José **Nuevos Nombres del Usuario**: Pedro Alejandro o Apellidos anteriores del Usuario: Jiménez Zamora **Nuevos Apellidos del Usuario**: Jiménez Zamora o Cédula de Identidad anterior: V25468697 **Nueva cédula de Identidad**: V8591268 o 1er Teléfono anterior del Usuario: (0414) 6856165		

	Nuevo 1er teléfono del Usuario: (0416) 0369540 o Segundo Teléfono anterior del Usuario: (“”) “” **Nuevo 2do teléfono del Usuario**: (0294) 6632596 o Cargo del Usuario: Gerente General **Nuevo Cargo del Usuario**: Gerente General o E-Mail anterior del Usuario**:** pe-zamora@gmail.es **Nuevo E-Mail del Cliente**: p-a-j-z@hotmail.com o **Razón Social a la cual pertenece el usuario:** E/S. RÍO CARIBE, C. A. **Nueva Razón Social propia del usuario:** E/S. MNICENTRO KOPA, C. A.
Resultado	Se actualizan los datos modificados del usuario en cuestión en la base de datos.
Condiciones	➢ El Empleado de HIDROCONS que realiza este caso de uso debe estar registrado en el sistema como usuario. ➢ Dicho usuario debe estar previamente autentificado, es decir, haber cumplido requisitos para entrar al sistema. ➢ Existe un registro del usuario a modificar en la base de datos con el o los datos considerados como criterios de búsqueda. ➢ El usuario que lleva a cabo este caso de uso posee la permisología y privilegios adecuados para realizar este tipo de modificaciones.

Tabla 5.8. Caso de Prueba de Integración “Modificar Usuario” (Continuación)

N°. Caso de Prueba	**03**	**Caso de Prueba**	**Modificar Usuario**
Procedimiento	1. Seleccionar del menú la opción de Consultar Clientes. 2. Escoger algún criterio de búsqueda en la ventana modal emergente (todos los clientes, clientes de una determinada compañía, o según la situación del mismo).		

	3. Buscar el cliente a modificar con los botones de recorrido de registros. 4. Una vez conseguido el cliente en cuestión, cambiar los valores de los campos necesarios por los nuevos datos indicados anteriormente. 5. Presionar el botón adecuado para proceder con la modificación los datos del cliente. 6. Confirmar el mensaje de solicitud de procesamiento que el sistema presenta. 7. El cliente ya se encuentra modificado en el sistema.
Situación del Caso de Prueba	✓ **EXITOSO**

Otro de los casos de prueba que puede realizar un usuario del tipo Empleado de HIDROCONS con la empresa ingresada anteriormente puede ser el que consiste en redactar una nueva factura a dicha empresa, acreditándosela a sus cuentas por cobrar.

Tabla 5.9. Caso de Prueba de Integración "Registrar Factura"

N°. Caso de Prueba	**04**	**Caso de Prueba**	**Registrar Factura**
Entrada	➢ **Número de la factura**: 001598 ➢ **Lugar de emisión de la factura**: Puerto La Cruz ➢ **Fecha de elaboración de la factura**: 20/07/2009 ➢ **Impuesto al Valor Agregado (IVA)**: 12% ➢ **Cliente al que va dirigida**: E/S. RÍO CARIBE, C. A. ➢ **Forma de pago**: Otra (Crédito a 90 días) ➢ **Tipo de servicio del ítem N°. 1**: Actividad ➢ **Descripción del ítem N°. 1**: Suministro e instalación de picos de ¾"Ø para dispensador Marca WAYNE. ➢ **Cantidad del ítem N°. 1**: 3 ➢ **Presentación o unidad de medida del ítem N°. 1**: UN		

	➢ **Precio unitario del ítem N°. 1**: 200.00

Tabla 5.9. Caso de Prueba de Integración "Registrar Factura" (Continuación)

N°. Caso de Prueba	**04**	**Caso de Prueba**	**Registrar Factura**
Entrada	➢ **Tipo de servicio del ítem N°. 2**: Actividad ➢ **Descripción del ítem N°. 2**: Reparación e instalación de picos de 1"Ø en dispensador Marca LEGACY. ➢ **Cantidad del ítem N°. 2**: 8 ➢ **Presentación o unidad de medida del ítem N°. 2**: UN ➢ **Precio unitario del ítem N°. 2**: 250.00		
Resultado	Se registra en la base de datos la nueva factura, con sus respectivos ítems asociados.		
Condiciones	➢ El usuario debe estar previamente autentificado. ➢ El número de factura no debe estar presente en la BDD. ➢ La empresa a la cual va dirigida la factura debe estar registrada previamente en el sistema.		
Procedimiento	1. Seleccionar la opción Registrar Factura en el menú. 2. Ingresar en los campos respectivos de la nueva ventana modal el número de factura, el lugar de emisión, la fecha de elaboración y el impuesto (IVA). 3. Seleccionar del cuadro combinado la empresa a la cual se le emitirá la factura en cuestión. 4. Seleccionar las distintas formas de pago de factura. 5. Presionar el botón apropiado para continuar. 6. Presionar el botón para mostrar la ventana modal con el formulario de ingresar nuevos ítems o partidas. 7. En el nuevo formulario, seleccionar el tipo de servicio asociado al ítem, e ingresar la cantidad, la unidad de medida, la descripción y el precio unitario del mismo. 8. Presionar el botón adecuado para aceptar y confirmar el procesamiento del ítem. 9. Repetir los pasos 6, 7 y 8 tantas veces como se		

	necesite. 10. Presionar el botón para ingresar finalmente la factura. 11. Confirmar el mensaje de solicitud de procesamiento que el sistema presenta. 12. La factura ha sido registrada exitosamente.
Situación del Caso de Prueba	✓ **EXITOSO**

Si dicha factura ingresada es cancelada posteriormente, es decir, es pagada por el cliente al cual fue emitida dicha factura, el Empleado de HIDROCONS igualmente puede realizar este caso de prueba como a continuación se detalla:

Tabla 5.10. Caso de Prueba de Integración "Cancelar Factura"

N°. Caso de Prueba	**05**	**Caso de Prueba**	**Cancelar Factura**
Entrada	➢ **Número de la factura como criterio de búsqueda**: 001598		
Resultado	La factura que cumple con el requisito de búsqueda es mostrada en la pantalla, y posteriormente cancelada.		
Condiciones	➢ El usuario debe estar registrado en la Base De Datos. ➢ El usuario debe estar previamente autentificado. ➢ El número de factura debe estar registrado en la BDD. ➢ La situación o estado de la factura en cuestión debe indicar que está "POR CANCELAR".		
Procedimiento	1. Seleccionar la opción Consultar Facturas en el menú desplegable izquierdo. 2. Aparece una ventana modal donde se ingresará en el campo adecuado como criterio de búsqueda el número de la factura a consultar. 3. Presionar el botón de consulta. 4. Aparece una sola factura que satisface el criterio de búsqueda indicado, y algunos datos de información propios de ella para corroborar la pertenencia y procedencia de la misma, tales como fecha, cliente al cual va dirigida, situación y monto		

	sin IVA de la misma. 5. Presionar en el enlace del código o número de factura que aparece en la pantalla. 6. En la nueva ventana presionar el botón que cancelará dicha factura, retirándola del estado de cuentas por cobrar del cliente en cuestión y colocando su estado como “CANCELADA”. 7. Confirmar el mensaje de solicitud de procesamiento que el sistema presenta. 8. La factura ha sido cancelada exitosamente.
Situación del Caso de Prueba	✓ **EXITOSO**

El representante legal de la empresa ingresada también le corresponde unos ciertos casos de prueba, pero su rol es como “Cliente HIDROCONS”. Aquí se muestran algunos de estos casos de prueba que el cliente puede efectuar, en donde se le atribuye ciertos privilegios como realizar solicitudes o peticiones de repuestos, así como aprobar y/o rechazar presupuestos propuestos.

Tabla 5.11. Caso de Prueba de Integración “Registrar Solicitud de Repuestos”

N°. Caso de Prueba	**06**	**Caso de Prueba**	**Registrar Solicitud de Repuestos**
Entrada	➢ **Selección de uno o varios repuestos con los checkboxes colocados al lado de la información de cada repuesto**: o **Repuesto 1**: Pico OPW 11AP de ¾"Ø x ½"Ø. o **Repuesto 2**: Swivel OPW de ¾"Ø. o **Repuesto 3**: Breakaway OPW de ¾"Ø. o **Repuesto 4**: Capacitador de arranque de 25 µf. o **Repuesto 5**: Tarjeta controladora T-20076 para dispensador LEGACY Modelo JHA. ➢ **Cantidad de unidades de los repuestos seleccionados**: o **Cantidad Repuesto 1**: 20 UN		

Resultado	o **Cantidad Repuesto 2**: 20 UN o **Cantidad Repuesto 3**: 5 UN o **Cantidad Repuesto 4**: 10 UN o **Cantidad Repuesto 5**: 15 UN La solicitud de la lista de repuestos se registra en la Base de Datos del sistema
Condiciones	➢ El Cliente de HIDROCONS debe estar registrado en el sistema como usuario activo. ➢ El Cliente debe estar previamente autentificado.
Procedimiento	1. Seleccionar la opción Consultar Repuestos en el menú desplegable del lado izquierdo de la vista principal del cliente autentificado. 2. Seleccionar qué tipo de accesorio va a solicitar (Eléctrico, Electrónico o Hidráulico). 3. Escojer el repuesto de su preferencia propio del tipo de repuesto que está actualmente escogido. 4. Indicar la cantidad requerida de dicho repuesto. 5. Repetir los pasos 2, 3 y 4 tantas veces como sea necesario. 6. Presionar el botón indicado para visualizar la cotización definitiva y totalizada sin impuesto (IVA). 7. Confirmar el mensaje de solicitud de procesamiento. 8. La solicitud de repuestos ha sido generada exitosamente.
Situación del Caso de Prueba	✓ **EXITOSO**

Supóngase que la empresa cliente en cuestión, así como tuvo una factura acreditada a su cuenta, también ha de tener un presupuesto en situación de espera de algún trabajo que se le ha de realizar por parte de la compañía HIDROCONS, C. A. Pues igualmente el usuario Cliente HIDROCONS o representante legal de dicha empresa puede proceder a realizar la aprobación o rechazo del presupuesto en cuestión. A continuación se muestra el procedimiento que realizaría el cliente para llevar a cabo el caso de prueba de integración de aprobación de un presupuesto.

Tabla 5.12. Caso de Prueba de Integración "Aceptar Presupuesto"

N°. Caso de Prueba	**07**	**Caso de Prueba**	**Aceptar Presupuesto**
Entrada	➢ **Referencia del presupuesto como criterio de búsqueda**: 001-09		
Resultado	El presupuesto que cumple con los requisitos de búsqueda indicados es mostrado en la pantalla, y posteriormente es aprobado para su próxima ejecución.		
Condiciones	➢ El usuario debe estar registrado previamente en la BDD. ➢ El usuario debe estar en situación o estado ACTIVO. ➢ El usuario no debe haber iniciado sesión alguna.		
Procedimiento	1. Seleccionar la opción Consultar Presupuestos en el menú desplegable. 2. Aparece una ventana modal donde se ingresará en el campo adecuado como criterio de búsqueda el número de referencia del presupuesto a consultar. 3. Presionar el botón de consulta. 4. Aparece un solo presupuesto que es el que satisface el criterio de búsqueda, y algunos datos de información adicionales para corroborar la pertenencia y procedencia del mismo, tales como fecha del presupuesto, cliente al cual va dirigido, situación y monto del mismo, entre otros. 5. Presionar en el enlace del código o referencia del presupuesto que aparece en la pantalla. 6. En la ventana emergente se procede a presionar el botón que aceptará dicho presupuesto, colocando su situación como “APROBADO”. 7. Confirmar el mensaje de solicitud de procesamiento que el sistema presenta. 8. El presupuesto ha sido aprobado exitosamente.		
Situación del Caso de Prueba	✓ **EXITOSO**		

5.6. Evaluación de la fase de construcción

La fase de construcción termina con el “Hito de la Capacidad Operacional”, hito usado para determinar cuándo el producto está listo para ser publicado en una prueba beta. Se puede decir que esta fase se ha ejecutado con éxito. Se requirió una iteración donde se abarcaron los flujos de trabajo de implementación y pruebas. Durante la implementación se realizó la codificación efectiva de las páginas Web y los distintos componentes que conforman el software. Al revisar este hito aplicado al proyecto que actualmente se desarrolla, se revisaron ciertos criterios de evaluación, arrojándose los siguientes resultados:

- Se diseñaron las interfaces del programa que permiten la interacción e intercambio de información entre el sistema y el usuario de manera cómoda y amigable.

- Tanto la codificación de los módulos que conforman el proyecto, como la integración de cada uno de ellos se realizaron de manera exitosa, en combinación con la respectiva base de datos.

- Las diversas pruebas necesarias y llevadas a cabo para la certificación de la confiabilidad y efectividad de la aplicación permitió la detección y corrección oportuna de las fallas existentes.

- Las herramientas que ofrece el lenguaje de modelado WebML fueron de mucha utilidad, ya que facilitan la codificación mediante estándares de programación Web.

- En esta fase, se le aplicaron pruebas específicas al software, las cuales permitieron mejorar la calidad del mismo, ya que con la corrección de los errores presentados se garantiza la futura estabilidad operativa de la aplicación.

- La ejecución de los procesos de implementación y pruebas han dado como resultado un producto estable y maduro como para ser entregado a la comunidad de usuarios para ser probado y evaluado.

- Se certifica que todas las personas involucradas en el proyecto están listas para la transición a la comunidad de usuarios.

- Ahora que se obtuvo una versión funcional del producto, se sigue a la fase que determinará si la transición de dicha versión es hacia otro ciclo de desarrollo, es decir, hacia otra iteración, o hacia los usuarios finales del mismo.

CAPÍTULO VI

FASE DE TRANSICIÓN

6.1. Introducción

La Fase de Transición del Proceso Unificado de desarrollo de software abarca las últimas etapas de la actividad genérica de construcción. Esta fase se enfoca en asegurar que el software esté listo para los usuarios finales. En este punto la mayor parte de la estructura debería estar terminada y la retroalimentación de los usuarios debería enfocarse principalmente en la entonación, configuración, instalación y resultados de reutilización. Es posible que la fase de transición se extienda algunas iteraciones, incluyendo las pruebas del producto dentro de la preparación para su publicación y el hacer los ajustes menores basados en la retroalimentación de los usuarios. La fase de transición en el enfoque de RUP difiere del desarrollo tradicional, principalmente porque se entra en esta fase con una versión del sistema estable, integrada y probada. [9]

En la fase anterior se obtuvo un software preliminar que es capaz de cumplir a cabalidad con los requerimientos establecidos por el departamento de administración de HIDROCONS, C. A., minimizando los riesgos y maximizando la productividad y efectividad de los usuarios. La fase de transición cubre el periodo durante el cual el producto se convierte en versión beta. En la versión beta un número reducido de usuarios con experiencia prueba el producto e informa de defectos y deficiencias. Los desarrolladores corrigen los problemas e incorporarán alguna de las mejores sugeridas en una versión general dirigida a la totalidad de la

comunidad de usuarios. La fase de transición conlleva actividades como la fabricación, formación del cliente, el proporcionar una línea de ayuda y asistencia, y la corrección de los defectos en dos categorías: los que tienen suficiente impacto en la operación para justificar una versión incrementada y los que pueden corregirse en la siguiente versión nominal. El principal objetivo que se ha planteado para esta fase es garantizar la satisfacción de todos los usuarios finales del sistema. También se provee el soporte básico necesario a través de un manual de usuario. [15]

El software se entrega a los usuarios para realizar pruebas beta, y la retroalimentación del usuario reporta tanto defectos como cambios necesarios. Además, el equipo de software crea la información de soporte necesaria (por ejemplo, manuales de usuario, guías de resolución de problemas, procedimientos de instalación) para el lanzamiento.

6.2. Planificación de la fase de transición [9]

Al final de la fase de transición los objetivos del ciclo de vida deben haber sido alcanzados y el proyecto debería estar en posición de ser cerrado. El fin del actual ciclo de vida puede coincidir con el inicio de otro ciclo de vida, sobre el mismo producto, dirigiéndose a la próxima generación o versión del producto. Los principales objetivos de esta fase están constituidos por:

- Una prueba beta, para validar el nuevo sistema contra las expectativas del usuario y la operación en paralelo con los sistemas heredados que dicho sistema reemplazará.

- Entrenar a los usuarios y a las personas encargadas de mantener el sistema, para alcanzar la adopción exitosa del sistema.

- En caso de que el producto sea destinado a la venta, la distribución del mismo debe de llevarse a cabo, lo que implica empaquetamiento y producción comercial del software, fuerza de ventas, publicidad y otras actividades que ayudan al lanzamiento exitoso del producto (ingeniería de publicación).

- Alcanzar un consenso con todas las personas involucradas en el proyecto, en que la base para la publicación está completa y es consistente con el criterio de evaluación de la visión.

Se estima que la duración de esta fase sea más corta que la anterior, ya que la realización de actividades en cualquier flujo de trabajo (requisitos, análisis, diseño, implementación o pruebas) depende del resultado que arroje las pruebas sobre la versión beta del software. La corrección de todos los posibles errores y modificaciones que aparezcan sugeridas por los usuarios es lo que determinará la calidad funcional del software, debido a que son los usuarios finales los que reportarán las fallas o modificaciones aceptables en pro del mejoramiento del producto final que utilizarán.

6.3. Implementación

En la fase de transición del RUP se procede directamente a la implementación del proyecto en cuestión, de allí que no se lleve a cabo ninguna etapa de diseño, y mucho menos de análisis. En esta fase el número de iteraciones puede variar de algo muy directo a algo extremadamente complejo, dependiendo del producto. Por ejemplo, una nueva versión beta puede ser muy simple, requiriendo simplemente la corrección de algunos errores menores, que se pueden realizar en una iteración. Sin embargo se necesitan más iteraciones si se van a agregar características adicionales al producto o se va a integrar con otros sistemas. Las actividades ejecutadas en las iteraciones de transición dependen de las metas a alcanzar.

6.3.1. Preparación de la versión beta

Para la preparación del lanzamiento de la versión beta del software, se ha planeado instalar la estructura de hardware necesaria para que funcione el software en un área de trabajo determinado y seleccionar un grupo de usuarios para que utilicen la aplicación. Los usuarios seleccionados fueron uno de cada tipo de actor que interacciona con el software, es decir, un directivo de HIDROCONS, C. A., y un cliente o representante de una estación dispensadora de combustible. A cada uno de ellos se le realizó un curso inductivo con el objeto de cubrir el entrenamiento del personal para el correcto uso de la aplicación, que incluye requisitos mínimos para la instalación y uso del software, niveles de permisología por usuario, es decir, las restricciones o limitaciones de acceso de cada usuario, y un manual de usuario.

6.3.2. Instalación de la versión beta

La instalación del software se realizó en una computadora de la oficina de HIDROCONS, C. A. para los usuarios Empleados HIDROCONS o directivos. Se instalaron todas las herramientas requeridas para el correcto funcionamiento del software bajo la plataforma Microsoft Windows XP, así como el manejador de base de datos DB2 Versión 9 con todas las actualizaciones respectivas. Por otro lado, en la computadora de la oficina de la estación dispensadora de combustible no fue necesario instalar aplicación alguna, ya que lo que se necesita para ese caso ya se encontraba implantado, como lo es el sistema operativo Microsoft Windows XP y el navegador Web Explorer Versión 6.

Los computadores utilizados para la instalación de la versión beta cumplen con las características mínimas que se necesitan para un desempeño óptimo del software. Tales requerimientos mínimos se mencionan a continuación:

- Procesador PENTIUM IV 3.0 GHz.
- Memoria RAM de 512 MB o más.
- Disco duro 80 GB o más.
- Teclado, monitor y mouse.

Con la finalidad de explicar el funcionamiento del programa SIADCON, se llevó a cabo una especie de curso al respecto. En el siguiente cuadro (véase tabla 6.1) se muestra cómo fue el horario de adiestramiento a los tipos de usuarios del sistema, lo cual constituye un curso intensivo que se puede llevar a cabo en menos de un día:

Tabla 6.1. Distribución de horas durante adiestramiento de usuarios de SIADCON

Tipo de Usuario / Tema de adiestramiento	Empleado HIDROCONS	Cliente HIDROCONS
Interfaz Externa de SIADCON	15 minutos	15 minutos
Administración de Empresas	30 minutos	15 minutos
Administración de Usuarios	30 minutos	15 minutos
Administración de Presupuestos	45 minutos	30 minutos
Administración de Facturas	30 minutos	15 minutos
Administración de Servicios	30 minutos	10 minutos
Administración de Repuestos	30 minutos	20 minutos
Sección de Ayuda	10 minutos	10 minutos

Horas Totales …	**220 minutos**	**130 minutos**

Durante el transcurso de las pruebas hechas por los usuarios, se les impartieron instrucciones para que fueran accediendo a las diferentes partes que conforman el software, para así poder diagnosticar las posibles fallas que se puedan presentar con respecto a la funcionalidad del mismo.

6.4. Evaluación de la fase de transición

El hito que marca el final de la fase de transición es el hito de la versión del producto, donde se decide si los objetivos del proyecto fueron alcanzados o si es necesario comenzar otro ciclo de desarrollo. Llegado al final de esta fase, se encontraron los siguientes resultados:

- El desarrollo de este proyecto se ha completado con éxito; el hecho de haber culminado esta fase así lo confirma.

- Se comprobó que el sistema implantado ha cumplido con los requerimientos y necesidades del departamento de administración de la empresa solicitante, así como también se han satisfecho los objetivos planteados al inicio del proyecto.

- El manual de usuario del sistema SIADCON es lo suficientemente comprensible e ilustrativo al usar.

- No fue necesario corregir o modificar significativamente algún componente de software.

- La arquitectura implementada es robusta y estable.

CONCLUSIONES

1. Se efectuó un análisis minucioso de los procedimientos y normas asociadas al departamento de administración de la empresa HIDROCONS, C. A., lo que permitió la captura y definición de manera exitosa y completa de las necesidades actuales del sistema, de un conjunto significativo y correcto de requerimientos necesarios para el desarrollo del sistema de información de la empresa, mediante el entendimiento común y exhaustivo de estos, además de los requisitos que permitieron la obtención de una arquitectura candidata lo más apropiada del mismo.

2. Al llevarse a cabo, mediante técnicas del modelado relacional, un diseño efectivo de la base de datos propia para el almacenamiento de la información relacionada con los procesos administrativos a ser manejados por el sistema, aunado a fases de normalización, se logró de manera efectiva y poco redundante gestionar todas las transacciones que ocurren entre la aplicación y dicho manejador de datos.

3. De igual manera dicho análisis minucioso permitió conseguir un diseño definitivo de la Arquitectura del Software del sistema, en función de las restricciones computacionales existentes.

4. Utilizar el lenguaje de modelado UML, acompañado también del WebML para diseñar los módulos facilitó la construcción de la aplicación en lo que a

navegabilidad y funcionamiento del sistema se refiere, así como a visualización e integración de los módulos.

5. Las interfaces diseñadas para la aplicación permiten la interacción e intercambio dinámico de información entre el sistema y el usuario de una manera sencilla y amigable.

6. Tanto el proceso de codificación de módulos por separado como el de integración de cada uno de los módulos que conforman el proyecto se realizaron sin inconvenientes considerables y de manera exitosa, en combinación con la respectiva base de datos, obteniéndose una aplicación robusta, confiable y segura.

7. Mediante la realización de diversas pruebas necesarias y llevadas a cabo para la certificación de la confiabilidad y efectividad de la aplicación, se permitió la detección y corrección de las fallas existentes, garantizando un producto de buena calidad.

8. Se elaboró un manual de usuario que brinda soporte al empleo del sistema para todo empleado o cliente que haga uso del mismo.

9. Los módulos de “Administración de Repuestos” y “Administración de Servicios” permiten dar a conocer a los clientes los repuestos que HIDROCONS, C. A. tiene para ofrecer al público en general, así como los servicios, trabajos y actividades que esta empresa les presta en materia de mantenimiento de estaciones de servicio y construcción civil en general.

10. Mediante el módulo “Administración de Presupuestos” los clientes pueden estar al tanto y conocer al respecto de todos los trabajos a realizarse y de todos

los ya realizados o ejecutados, inherentes a su empresa, para realizar transacciones vía Web con relación a dichos documentos.

RECOMENDACIONES

1. Anexar al sistema un módulo de mantenimiento, manejado por otro tipo de usuario de sistema que permita llevar a cabo rutinas de mantenimiento periódicamente a la base de datos, realizando respaldos y optimizaciones a las tablas que la conforman, para que se mantengan actualizados los índices de búsqueda y el buen rendimiento de la misma.

2. Crear un módulo que envíe mensualmente un estado de cuenta a los usuarios de las factura por cancelar mediante correo electrónico, de esta forma estarán al tanto de sus deudas y tendrán la oportunidad de estar al día con dichas cuentas.

3. Diseñar un módulo que permita al empleado clasificar los diferentes ítems o renglones de los presupuestos y facturas de manera que pertenezcan a un grupo específico de actividad.

4. Modificar convenientemente la base de datos del sistema, de manera que se informe la fecha de la última conexión, visita o sesión de cualquier usuario que haga uso de la aplicación.

BIBLIOGRAFÍA

[1] Autor desconocido. **"Información sobre WebML"**. Disponible en: dataware.nce.ufrj.br:8080/dataware/fisico/seminariosAlunos/questoes_modelagem/WEBML.ppt. Consultado el 14 de Diciembre de 2007.

[2] **"Microsoft Diccionario de Informática e Internet"**. Editorial McGraw-Hill. Primera Edición. Madrid. (2001).

[3] CEDEÑO, J. **"Desarrollo de una aplicación Web para el registro, manejo y control de eventos organizados por la Unidad de Calidad de Vida del departamento de Recursos Humanos de PDVSA – Refinación en Puerto La Cruz"**. Trabajo de Grado. Universidad de Oriente, Núcleo Anzoátegui - Venezuela. (2007).

[4] Ceri S., Fraternali P., Bongio A. y Brambilla M., **"Designing Data-Intensive Web Applications"**. Morgan Kaufmann Publishers. Primera Edición. (2003).

[5] Ceri, S., Fraternali, P. y Bongio, A. Dipartimento di Elettronica e Informazione, Politecnico di Milano. **"Web Modeling Language (WebML): a modeling language for designing Web sites"**. Disponible en: http://www9.org/w9cdrom/177/177.html Consultado el 20 de Enero de 2008.

[6] IBM IT EDUCATION SERVICES WORLDWIDE CERTIFIED MATERIAL. **"Guía de Estudiante del Curso Análisis y Diseño de Sistemas Orientados a Objetos Libro Nº. 1"**. Código del Curso: CY450. Versión 5.0. (2006).

[7] IBM IT EDUCATION SERVICES WORLDWIDE CERTIFIED MATERIAL. **"Guía de Estudiante del Curso Base de Datos Libro Nº. 1"**. Código del Curso: TWB22B. Versión 4.0. (2006).

[8] IBM IT EDUCATION SERVICES WORLDWIDE CERTIFIED MATERIAL. **"Guía de Estudiante del Curso de Java Empresarial Libro Nº. 1"**. Curso: CY760. Versión 4.0. (2006).

[9] IBM LEARNING SERVICES WORLDWIDE CERTIFIED MATERIAL. **"Guía del Estudiante del Curso de Ingeniería de Software (Volumen 1: Fundamentos de Análisis y Diseño)"**. Código del Curso: CY440. Versión 4.1. (2006).

[10] IBM LEARNING SERVICES WORLDWIDE CERTIFIED MATERIAL. **"Guía del Estudiante del Curso de Ingeniería de Software (Volumen 2: Métricas, Calidad y Pruebas)"**. Código del Curso: CY440. Versión 4.1. (2006).

[11] IBM LEARNING SERVICES WORLDWIDE CERTIFIED MATERIAL. **"Guía del Estudiante del Curso de Programación III Libro N°. 1: Core Java"**. Código del Curso: CY440. Versión 4.0. (2006).

[12] JACOBSON, I. y BOOCH, G. **"El proceso Unificado Racional de Desarrollo de Software RUP"**. Editorial Pearson Educación, S.A. Madrid - España. (2000).

[13] Manual de Repuestos OPW para Dispensadores de gasolina, dispensadores de gasoil y bombas sumergibles Marca GILBARCO. Propiedad de HIDROCONS, C. A. (2003).

[14] MEDINA, J. **"Desarrollo de un Sistema basado en Aplicaciones Web para la Automatización del Control de Pedidos asociados al Proceso de Ventas de una empresa cafetalera"**. Trabajo de Grado. Universidad de Oriente, Núcleo Anzoátegui - Venezuela. (2007).

[15] MILLÁN, L. y GARELLI, L. **"Desarrollo de un Software que permita la Automatización de las Actividades asociadas al Departamento de Admisión y Control de Estudios de la Extensión Centro / Sur del Núcleo de Anzoátegui de la Universidad de Oriente"**. Trabajo de Grado. Universidad de Oriente, Núcleo Anzoátegui - Venezuela. (2007).

[16] NÚÑEZ, L. **"Desarrollo de una Aplicación Web para la Visualización en Tiempo Real de los Parámetros Operacionales de Producción de la empresa PDVSA"**. Trabajo de Grado. Universidad de Oriente, Núcleo Anzoátegui - Venezuela. (2007).

[17] NÚÑEZ, L. "**Programación Orientada a Objeto"**. Universidad Tecnológica de Santiago de Chile. Disponible en: http://www.monografias.com/trabajos4/basesdatos/basesdatos.shtml. Consultado el 23 de Noviembre de 2007.

[18] Politecnico di Milano. **"Modelado de Hipertexto"**. Disponible en: http://www.webml.org/webml/page18.do?dau70.oid=12&UserCtxParam=0&GroupCtxParam=0&ctx1=EN. Consultado el 23 de Noviembre de 2007.

[19] TENIAS, J. **"Desarrollo de un software basado en Aplicaciones Web para el Monitoreo de Dispositivos de la Plataforma de Telecomunicaciones de PDVSA - GAS"**. Trabajo de Grado. Universidad de Oriente, Núcleo Anzoátegui - Venezuela. (2007).

[20] Templeman J., Olsen A. "Microsoft Visual C++.NET Aprenda Ya". Primera Edición. Editorial McGraw – Hill. Estados Unidos. (2002).

[21] ZAVALA, R. "**Ingeniería de Software**". Disponible en: www.angelfire.com/scifi/jzavalar/apuntes/IngSoftware.html Consultado el 14 de Diciembre de 2007.

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

TÍTULO	"DESARROLLO DE UN SOFTWARE EN AMBIENTE WEB DESTINADO A GESTIONAR LOS PROCESOS DEL DEPARTAMENTO DE ADMINISTRACIÓN DE UNA EMPRESA ENCARGADA DEL MANTENIMIENTO DE EQUIPOS DISPENSADORES DE COMBUSTIBLE, UBICADA EN PUERTO LA CRUZ – ESTADO ANZOÁTEGUI"
SUBTÍTULO	

AUTOR (ES):

APELLIDOS Y NOMBRES	CÓDIGO CVLAC / E MAIL
Guevara F., Luis C.	CVLAC: 13.167.548 E MAIL: carlosfranco2020@hotmail.com

PALÁBRAS O FRASES CLAVES:

Desarrollo de Software.

Aplicación Web.

Diseño de Base de Datos (BDD).

Lenguaje de Modelado Web (WebML).

Lenguaje de Modelado Unificado (UML).

Establecimientos dispensadores de combustible.

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

ÀREA	SUBÀREA
Ingeniería y Ciencias Aplicadas	Ingeniería en Computación

RESUMEN (ABSTRACT):

HIDROCONS, C.A. es una compañía contratista cuya principal actividad económica es la construcción de obras civiles en general, electricidad, plomería, pintura, instalación, reconstrucción, mantenimiento y reparación de equipos de estaciones de servicios dispensadoras de combustible. Igualmente compra, vende e importa todo material de construcción destinado para los fines económicos descritos anteriormente. Debido a problemas como el retraso en que llegan los presupuestos a mano de los directivos de las empresas clientes, aunado al desconocimiento del estado de cuenta de la facturación por parte de los mismos, e inclusive de los servicios y repuestos que ofrece la empresa, se propuso el proyecto denominado "Desarrollo De Un Software En Ambiente Web Destinado A Gestionar Los Procesos Del Departamento De Administración De Una Empresa Encargada Del Mantenimiento De Equipos Dispensadores De Combustible, Ubicada En Puerto La Cruz – Estado Anzoátegui", que permite el flujo de información de una manera más rápida, oportuna y segura, además de fidedigna y veraz. Para diseñar la aplicación se utilizó el Proceso Unificado Racional de Desarrollo de Software (RUP) en combinación con los lenguajes de modelado UML y WebML. La construcción del mismo fue realizada mediante la herramienta de computación distribuida IBM RATIONAL 6.0. Debido a que es un sistema con entorno Web, la arquitectura se construyó utilizando el patrón cliente-servidor y se implementó IBM DB2 Versión 9 como sistema manejador de Base de Datos Relacionales. Para la codificación se usó Java como lenguaje de programación del lado del servidor, JavaScript (lenguaje interpretado orientado a páginas Web) del lado del cliente, ambos acompañados con lenguaje HTML. La visualización de la aplicación se encuentra a cargo del navegador Web Internet Explorer 6, y el Sistema Operativo a utilizar para el mismo es el Windows XP. La aplicación se denominó SIADCON, por las siglas de "Sistema Integral de ADministración para CONtratistas" .

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

CONTRIBUIDORES:

APELLIDOS Y NOMBRES	ROL / CÓDIGO CVLAC / E_MAIL				
Torrealba, Aquiles.	**ROL**	**CA**	**AS** X	**TU**	**JU**
	CVLAC:	7.385.840			
	E_MAIL	tmar1966@hotmail.com			
	E_MAIL				
Veracierta, Gabriela.	**ROL**	**CA**	**AS**	**TU**	**JU** X
	CVLAC:	14.616.683			
	E_MAIL	gveracierta@hotmail.com			
	E_MAIL				
Saettone, Mónica.	**ROL**	**CA**	**AS**	**TU**	**JU** X
	CVLAC:	10.948.010			
	E_MAIL	msaettone@yahoo.com			
	E_MAIL				
	ROL	**CA**	**AS**	**TU**	**JU**
	CVLAC:				
	E_MAIL				
	E_MAIL				

FECHA DE DISCUSIÓN Y APROBACIÓN:

2009	10	16
AÑO	**MES**	**DÍA**

LENGUAJE. SPA

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

ARCHIVO (S):

NOMBRE DE ARCHIVO	TIPO MIME
TESIS.SistDeInformWeb_SIADCON.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:

Ingeniero en Computación

NIVEL ASOCIADO CON EL TRABAJO:

Pre-Grado

ÁREA DE ESTUDIO:

Departamento de Computación y Sistemas

INSTITUCIÓN:

Universidad de Oriente – Núcleo Anzoátegui

METADATOS PARA TRABAJOS DE GRADO, TESIS Y ASCENSO:

DERECHOS

De acuerdo al artículo 44 del Reglamento de Trabajos de Grado

"Los Trabajos de Grado son exclusiva propiedad de la Universidad de Oriente y sólo podrán ser utilizadas para otros fines con el consentimiento del Consejo de Núcleo respectivo, quien lo participará al Consejo Universitario"

Luis Carlos Guevara Franco

AUTOR

Ing. Aquiles Torrealba	Ing. Gabriela Veracierta	Ing. Mónica Saettone
TUTOR	**JURADO**	**JURADO**

Ing. José Luis Bastardo

POR LA SUBCOMISION DE TESIS