INSTITUTO SUPERIOR POLITÉCNICO "JOSÉ ANTONIO ECHEVERRÍA"
FACULTAD DE INGENIERÍA INDUSTRIAL
CENTRO DE ESTUDIOS DE INGENIERÍA DE SISTEMAS (CEIS)

GESTOR DE RECUPERACIONES DINAMICAS PARA APLICACIONES WEB

TRABAJO PARA OPTAR POR EL TÍTULO DE INGENIERÍA EN INFORMÁTICA

AUTORES:
Michel López Camino
Raydel Muñoz Vidal

TUTOR:
Ing. Raquel María Vidal Soto

Ciudad de la Habana
Junio 2006

AGRADECIMIENTOS

La amistad y el cariño que se recibe de los que te rodean son factores que determinan de forma muy decisiva el desarrollo y equilibrio de una persona.

No es tarea fácil poder escribir en un papel a todos los que transitaron contigo en los incontables momentos felices y no tan felices a través de toda una vida.

Si siembras frutos que perduren, al menos te queda el consuelo de agradecer a todos aquellos que caminaron siempre junto a ti, en nuestro caso nos sentimos inmensamente dichosos y agradecemos de la forma más grandiosa posible a nuestros creadores en todo sentido, de forma más sencilla, nuestros padres.

Quisiéramos expresar nuestras más sincera admiración y agradecimiento a todos los profesores que aportaron todo su esfuerzo en la preparación de quienes somos hoy en día.

Sin olvidarnos jamás de nuestros familiares que siempre juntos nos brindaron todo el universo de amor y apoyo incondicional que nos hacía falta para no claudicar en nuestro empeño.

A todos nuestros amigos que nunca nos dieron la espalda y siempre tuvieron una sonrisa en sus rostros a la hora de contar con ellos, nos vienen a la mente gente linda como José Manuel, Yamel, Figueras, Yoangel, Marín, Lázaro, Karel, Arcel, Reinier, Javier, Gerardo y todos nuestros compañeros de la Universidad de Holguín.

No pueden quedar exentos nuestros queridos compañeros de batalla, que siempre estuvieron con nosotros a pesar de los pesares, ese piquete que nos hacíamos llamar el Clan, las locuras de Mr Zenel, el ánimo de Franky DJ, la constancia del Yord, la persistencia de Andy Lucas y por supuesto nosotros como parte indisoluble.

También aportaron mucho a nuestra formación y estuvieron ahí en los últimos tiempos nuestros colegas de la Facultad 4 de la UCI, guiados por su decana Ivonne, el profesor Leonardo, Anelys, Yoysi y muchos otros que igual nos ayudaron.

Como eje esencial de nuestro desempeño nuestra tutora y amiga Raquelita y a su familia por demostrarnos que sí se podía.

Este es el comienzo de un nuevo camino en nuestras vidas y los conceptos que hasta ahora nos eran difíciles de concebir comenzarán a hacerse realidad.

Quiero destacar que no por dejar de mencionar a muchas de esas personas que contribuyeron de una forma u otra con nuestro desarrollo, son menos importantes. A todos ustedes MUCHAS GRACIAS.

Espero humildemente que lo que se expresa en este papel les llegue a todos y toque de muy cerca su corazón.

DEDICATORIA

A mi padre, infinitamente le dedico este triunfo en mi vida.

A mi madre Annia Camino, por ser mi fiel amiga, acompañante y consejera. Por ser lo que soy. Y porque este, es su sueño.

A mis hermanos por formar parte de mí y darme fuerzas para seguir.

A mis tíos y abuelos por dejarme seguir su ejemplo.

A la vida que tengo y a mis amigos de siempre Raydel, Harold y Rubiel, que sin ellos nada de esto hubiese sido posible.

Michel

A mi padre Jorge Muñoz Dieguez por haberme sabido guiar y trasmitirme su experiencia a lo largo de todos estos años.

A mi madre Lilian Vidal Labrada por su confianza y cariño como madre, y ser amiga indisoluble de mi vida.

A mi hermana Raiza Muñoz Vidal por brindarme en todo momento su lealtad, su confianza y su amor.

A toda mi familia por depositar en mí toda la confianza que un ser humano puede llevar consigo.

A mis amigos Michel y Franky por haber sabido compartir toda su vida universitaria, y que sin ellos no habría sido posible realizar este trabajo.

Raydel

RESUMEN

El presente Trabajo de Diploma está dirigido al proceso de búsqueda y recuperación de información en los Sistemas de Gestión de Base de Datos del Ministerio de las Fuerzas Armadas Revolucionarias (MINFAR). Para ello se ha propuesto llevar a cabo la automatización de los principales procesos que se realizan en esta actividad.

Debido a la importancia del proceso de Gestión de Información, incluido en toda entidad social, no caben dudas de que la infraestructura creada para soportar las acciones comprendidas en el ejercicio de esta actividad, genera documentos oficiales de apoyo, informes, reportes de suma importancia, para los que se hace imprescindible llevar a cabo un adecuado tratamiento, pues de estos depende el correcto funcionamiento de la dirección. Actualmente, existe una necesidad evidente de establecer un mecanismo que permita obtener datos, que sea de fácil acceso y capacidades avanzadas de búsqueda con completa seguridad, pues resulta considerable la cantidad de información generada sistemáticamente, la cual se encuentra constantemente propensa a errores.

Con el objetivo de proporcionarle al MINFAR dicho mecanismo, se propone la implementación e implantación en los sistemas de gestión de bases de datos de una aplicación que brinde al usuario total seguridad, protección y recuperación de los datos.

Al final de todo este proceso se deben obtener como resultados relevantes:

- Proveer al MINFAR de una herramienta de apoyo a la toma de decisiones.
- Brindar información rápida, de forma segura y eficiente de los elementos de carácter general y particular que se manipulan en los sistemas de gestión de información.

En este documento se plasman los resultados de un estudio realizado a algunas de las principales aplicaciones de recuperación existentes; se incluyen los conceptos relacionados con la recuperación de información y el fruto de las investigaciones realizadas durante todo el proyecto. Finalmente se muestran los resultados del análisis y diseño de la propuesta del sistema, y se proponen algunas recomendaciones para el desarrollo futuro del mismo.

INDICE

Recuperaciones
Dinámicas

INTRODUCCION

INTRODUCCION

La recuperación de información constituye una parte fundamental en las aplicaciones de software que se desarrollan, ya que permite a las entidades acceder de forma rápida y objetiva a los datos almacenados.

A lo largo de los años la cantidad de Información se va haciendo cada vez más amplia y gracias al desarrollo de las tecnologías de la Información y las Telecomunicaciones, hoy contamos con los medios necesarios para poner a disposición de los usuarios todo tipo de recursos, tanto físicos como digitales y en los más disímiles formatos.

Durante la etapa de análisis y diseño de las aplicaciones, por diferentes razones no se pronostican todos los requerimientos informativos que se deben tener en cuenta. Las necesidades que surgen al respecto a medida que se utilicen las aplicaciones son mayores de las que se le brindan al usuario. Por esta razón es muy usual que requieran mantenimientos que atrasan el funcionamiento óptimo del sistema, trayendo como consecuencia un menor rendimiento de las tareas y de manera inmediata el trabajo del desarrollador.

Este trabajo surge como necesidad de dar solución a las situaciones antes expuestas; por lo que el **problema** a solucionar en él, consiste en:

> ¿Cómo obtener la información almacenada en los Sistemas de Gestión de Base de Datos del MINFAR de manera rápida, dinámica y eficiente?

Conjuntamente con el desarrollo de las redes de computadoras e Internet se han implementado diversas aplicaciones cuyo propósito es la exposición, construcción y actualización de la información para presentarla a través de la red con algún formato determinado previamente. La recuperación de información es el conjunto de tareas mediante las cuales el usuario localiza y accede a los recursos de información que son pertinentes para la resolución de un problema planteado. Es una tarea que ocupa a todos en la actualidad; desde la aparición de Internet se ha ido formando una revolución en la sociedad, incrementando vertiginosamente en un proceso sin precedentes las comunicaciones; suministrando un mecanismo

muy eficaz para la difusión de la información y la colaboración e interacción entre individuos sin importar su localización geográfica, posibilitando la globalización del conocimiento. Cada día se adicionan nuevas posibilidades a la red de redes, nuevos recursos se hacen disponibles y nuevos servicios comienzan a operar. Sin embargo, para poder sacar provecho de todas estas posibilidades que se brindan es necesario encontrar el conocimiento en la red, y es aquí donde la variedad es un problema. La gran cantidad y diversidad de recursos disponibles se convierte en el principal obstáculo para obtener la información deseada en el menor tiempo posible y una de las variantes más utilizadas para resolver este problema es la creación de herramientas de búsqueda y recuperación.

Con el presente trabajo se propone construir una herramienta de recuperaciones dinámicas de información que permita un mayor aprovechamiento de la jornada laboral, logrando asimismo que el sistema se convierta en una ayuda indispensable y útil para todos; propiciando de esta forma la colaboración e incremento de la funcionalidades del sistema, y haciendo partícipes a los usuarios de la entidad involucrada en el proceso para la posterior utilización de los recursos de información.

Por tanto el **objeto de estudio** de este trabajo es el proceso de gestión de información y organización de todo el universo de datos que existen en las bases de datos del MINFAR.

De ello se deriva que el **campo de acción** que abarca este trabajo, es la automatización de la recuperación de información que existe en las bases de datos relacionales del MINFAR.

Como **hipótesis** se parte de la idea que, si se desarrolla una aplicación Web, basada en un gestor de Bases de Datos como el PostGreSQL, y un intérprete como PHP; debe mejorar la captura de datos con más calidad, celeridad y rapidez.

El **objetivo general** de este trabajo será: desarrollar una aplicación Web que permita la recuperación de información existente en las bases de datos del MINFAR, mediante avanzadas técnicas de gestión de búsqueda para facilitar el proceso de toma de decisiones.

De acuerdo con esta propuesta se derivan los siguientes **objetivos específicos**:

- Realizar un estudio sobre formas y mecanismos de recuperación de información. Y proponer uno para desarrollar la aplicación.
- Analizar y Diseñar una aplicación Web que pueda acoplarse fácilmente a un sistema de gestión de base de datos para la búsqueda de información.
- Proponer un proceso que garantice la integridad de la información contenida en las bases de datos y a la vez permita la mayor colaboración posible en las tareas de recuperación por parte de los usuarios que utilicen la aplicación.

Para cumplir con estos objetivos y resolver la situación problémica planteada, se proponen las siguientes **tareas**:

1. Estudio y descripción de los sistemas y formas de recuperación existentes en el MINFAR actualmente.
2. Análisis de cómo se encuentran en la arena internacional las tecnologías que se utilizan para llevar a cabo sistemas como el que se pretende desarrollar.
3. Selección de la metodología de Análisis y Diseño de sistemas informáticos que facilite la creación y garantice la calidad del sistema.
4. Selección de las herramientas para llevar a cabo el proyecto y la elección de la plataforma en la que se desarrollará la aplicación. Fundamentando su elección.

La realización y puesta en práctica del Gestor de Recuperaciones Dinámicas para aplicaciones Web resolverá el problema existente en estos momentos en el ámbito para el cual está destinado.

Las **ventajas** más inmediatas que posee son:

- Acoplar a cualquier sistema del MINFAR una herramienta para recuperar la información de forma dinámica.
- Establecer una forma más eficiente de acceder a los datos.
- Facilitar al usuario una búsqueda por diferentes conceptos, por listado de peticiones o por estadística.

- Generar plantillas de reportes según los requisitos seleccionados por los usuarios.
- Permitir la realización de las consultas de acceso a las bases de datos sin necesidad de programarlas.

Este documento consta de cinco capítulos en los cuales se describen paso a paso todo el proceso o etapas por la que transitó nuestro trabajo.

El Capítulo 1 describe detalladamente proceso de recuperación actual, incluyendo los sistemas automatizados que existen y que están vinculados a este trabajo.

El Capítulo 2 trata la situación de las tecnologías a utilizar en el desarrollo de la aplicación, se comparan y seleccionan las mejores propuestas para el trabajo, y se explican los conceptos principales que se van abordar.

El Capítulo 3 describe el negocio a través de un modelo de Dominio, y se hace el análisis del sistema a desarrollar. Se definen las funcionalidades del sistema y se describen detalladamente, utilizando herramientas de modelación.

El Capítulo 4 enfoca la construcción de la solución mediante diagramas de clases, de datos, y se plantean los principios para el diseño y la implementación. Aquí se construyen las funcionalidades que se definieron en el capítulo anterior.

El Capítulo 5 es un estudio de factibilidad del sistema, obteniendo los beneficios tangibles e intangibles y analizando los costos del desarrollo de nuestra propuesta.

Capítulo

I

FUNDAMENTACION TEORICA

1.1 Introducción

En el presente capítulo se brinda una visión general de los aspectos relacionados con los sistemas de recuperación de información y los conceptos necesarios para el estudio y clasificación de los mismos. También se mencionan las características de cada tipo de herramienta, así como la descripción de las principales definiciones asociadas al dominio del problema y que son necesarias para entender el negocio y la propuesta de solución.

Además se definen los conceptos más importantes a manipular en el resto del documento y se hace una valoración de softwares similares existentes en el mundo y una comparación entre estos. Se da una breve descripción de la metodología de análisis y diseño escogida y de las herramientas de desarrollo usadas para la confección del sistema.

1.2 ¿Qué es la recuperación de Información?

El proceso conocido como Recuperación de Información es la base de las máquinas de búsqueda y trata los aspectos relacionados con el almacenamiento, organización y representación de la información, algoritmos de búsquedas, y acceso a la misma. La recuperación de información es el conjunto de tareas mediante las cuales el usuario localiza y accede a los recursos de información que son pertinentes para la resolución del problema planteado.

En principio, la recuperación de información engloba las acciones encaminadas a identificar, seleccionar y acceder a los recursos de información útiles al usuario, sin perjuicio de otras acepciones del concepto.

1.3 El objeto de estudio

El Ministerio de las Fuerzas Armadas Revolucionarias es una institución, que se propone, mediante la actividad científico-técnica, contribuir de forma significativa al desarrollo sostenible de la sociedad cubana y velar por la defensa de la patria; es por esto, que debe mantener un liderazgo nacional en el campo de la tecnología, y con soluciones creativas y autóctonas, ser competitivos internacionalmente, para lo cual hace suyas las aspiraciones más legítimas.

1.3.1 Flujo actual de los procesos.

Actualmente en el MINFAR existe gran cantidad de información almacenada de manera digital, muchas veces de carácter confidencial, por lo que la recuperación de este tipo de fuentes es una tarea que va más allá de las actividades propias de los archiveros y documentalistas, y se convierte más bien en trabajo de todos los usuarios recopiladores de documentos. Además de recopilar los datos es necesario crear formatos adecuados para la empresa en cuestión, que permitan organizar y actualizar la información y recuperarla en el momento preciso para facilitar la toma de decisiones. En estos momentos los procesos de recuperación de información se desarrollan de forma independiente, por lo que no todos los sistemas poseen una herramienta común para la obtención y análisis de la información, es decir, la búsqueda se hace de manera ajena al usuario dependiendo del criterio del especialista informático encargado de la materia a recuperar.

El flujo de información que se maneja es muy grande y cambiante por lo que surgen nuevas necesidades informativas constantemente. Los procesos existentes para recuperar la información no se encuentran preparados para darle respuesta a las nuevas necesidades y poseen un nivel de dinamismo demasiado bajo que no permite la interacción con otros sistemas, por tanto se hace muy engorroso todo el mecanismo que se lleva a cabo para satisfacer los nuevos requerimientos.

De este modo el proceso se desarrolla de forma lenta, y no permite la comunicación e integridad entre los datos que son manejados por estas aplicaciones. Esto trae como consecuencia la intervención de un especialista

informático que le de solución a la actividad que se desea desarrollar, con el fin de obtener la información que se solicite. Debido a esto se produce un retraso en la recuperación de la información que se desea llegue al cliente y esto a su vez, obstaculiza el proceso de toma de decisiones.

Los procesos que se automatizarán van a ser todos los procesos de recuperación y análisis de información que se utilizan en la mayoría de los sistemas de gestión de base de datos.

1.4 Sistemas automatizados existentes vinculados al campo de acción

A nivel mundial existen diversos gestores para la recuperación de la información, uno de los más comunes que se utiliza es el Microsoft Query que conforma el paquete de herramientas de la compañía Microsoft. También podemos encontrar herramientas más potentes para realizar recuperaciones como el Crystal Report, siendo éste uno de los más populares y el Agata Report en sus versiones iniciales.

Microsoft Query es una herramienta que permite incorporar datos de orígenes externos a otros programas de Microsoft Office, especialmente a Microsoft Excel. Si utiliza Query para recuperar datos de las bases de datos y/o de los archivos corporativos, no es necesario que vuelva a escribir en Excel los datos que desee analizar. También puede actualizar los informes y resúmenes de Excel automáticamente de la base de datos de origen inicial siempre que la base de datos se actualice con información nueva.

Es posible recuperar datos de varios tipos de bases de datos, incluidos Microsoft Access, Microsoft SQL Server y los servicios OLAP (Online Analytical Procesing - Procesamiento Analítico en Línea) de Microsoft SQL Server. También puede obtener datos de las listas de Excel y de archivos de texto.

Puede recuperar datos de una base de datos creando una consulta, que es una pregunta que se hace acerca de los datos almacenados en una base de datos externa. Por ejemplo: si los datos están almacenados en una base de datos Access, puede que desee conocer las cifras de ventas de un producto determinado por regiones. Es posible seleccionar parte de los datos seleccionando

sólo los datos del producto y la región que desee analizar y omitir los datos que no necesite.

Microsoft Query no es una aplicación gratuita que te permitirá usar la conectividad de bases de datos ODBC, es decir establecer conexiones, a fin de poder importar datos provenientes de diversas bases de datos a Microsoft Excel. Ideal para incorporar datos de origen externo a Microsoft Excel o recuperar datos de bases variadas sin tener que volver a introducir todos los datos que deseas analizar, también se pueden actualizar los informes a partir de la base de datos de origen.

Crystal Reports es una poderosa herramienta que puede almacenar sentencias SQL, bitmaps, funciones de usuario y objetos de texto para reutilizarlos y compartirlos entre múltiples informes.

Permite integrar contenidos dinámicos en sus aplicaciones Web y con la puesta en el mercado de su última versión Crystal Reports 11 ya no existen restricciones a la hora de realizar consultas sobre cualquier base de datos.

Dispone de plantillas de usuario que le permiten rápidamente aplicar un aspecto y estilo uniformes sobre múltiples informes, así como crear atractivas vistas de datos de forma más rápida y sencilla. Soporta las actuales exigencias de rendimiento de las aplicaciones Web, con una conectividad e integración mejorada con Java, .NET y COM.

Necesita determinados requisitos para su aplicación, los cuales se mencionan a continuación:

- ✓ Microsoft Windows XP o superior.
- ✓ Computador Pentium o superior.
- ✓ Memoria RAM 128 o superior.

Agata Report es un generador de reportes multi-plataforma, una herramienta de consulta y generación de gráficos como el Crystal Reports que se conecta a varias Bases de Datos, como PostgreSQL, MySQL, Oracle, DB2, MS-SQL, Informix, InterBase, Sybase, o Frontbase y permite exportar los reportes en formatos como PostScript, plain text, HTML, XML, PDF o CSV (StarCalc, Excel).

Permite definir niveles de datos, subtotales y totales para el informe. Permite crear documentos como cartas y conjugar dinámicamente con los datos provenientes del reporte, así como crear etiquetas de direccionamiento y hasta generar un diagrama Entidad-Relación completo a partir de su banco de datos. Agata Report está hecho en PHP-GTK y se ejecuta como una aplicación Visual Basic.

Sin duda alguna los gestores de información han dejado de ser simples herramientas operativas para convertirse en verdaderos aliados de las organizaciones en etapas de modernización.

1.5 Propuesta de Solución

Como hemos podido apreciar estos servicios son realmente efectivos, pero poseen varios inconvenientes a la hora de su aplicación; en el caso del Crystal Reports en su más avanzada versión, nos encontramos con una herramienta sumamente cara y que requiere de tecnología muy avanzada para su aplicación, por lo que no es factible utilizarla en nuestro caso, ya que, además de ser costosa, no todas las instituciones que pudieran emplearla poseen los recursos necesarios para su implantación.

En el caso de Agata Report sí es una aplicación gratuita, pero su interfaz es complicada, cargada de muchas operaciones, y sin una organización normalizada. Además se considera un software complicado, por lo que se necesita de mucha preparación para utilizarlo y existe muy poca bibliografía, ya que está ahora en las primeras fases de su desarrollo.

EL Microsoft Query es el más comúnmente utilizado puesto que es el más conocido y fácil de utilizar, viene como parte del paquete de herramientas del Microsoft Excel. Puede ser muy rápido y cómodo cuando se quiere realizar recuperaciones a una base de datos, pero posee inconvenientes a la hora de realizar la conexión entre campos de diferentes tablas, además de ser un software comercial.

Después de realizar un análisis sobre algunos de los tipos de recuperadores de información y determinar claramente cuál es la situación actual sobre el objeto de

estudio que tiene este trabajo, se concluye que se hace necesario desarrollar una herramienta que pueda recuperar toda la información existente en las bases de datos del MINFAR, además que pueda identificar los tipos de información pertenecientes a cada usuario y efectuar las búsquedas de una forma eficiente y lo más exacta posible.

1.6 Conclusiones

En este capítulo se detallaron las condiciones y problemas que rodean el objeto de estudio, y a través de los conceptos y definiciones planteadas, se determinaron las condiciones específicas que rodean al problema y en base a esto se obtuvieron los objetivos generales y específicos para este trabajo; aunque en esta etapa solo se habla de ideas, es correcto que estén bien fundamentadas, porque estas constituyen la base para el posterior desarrollo de este trabajo.

Capítulo

II

TENDENCIAS Y TECNOLOGIAS ACTUALES

2.1 Introducción

La revolución de las Tecnologías de la Información y las Comunicaciones (TIC), con la incorporación de la computadora a los medios electrónicos, los sistemas de comunicación por satélite, el teléfono, el fax y el celular, no acaban de asombrarnos. En el presente siglo otras novedades de comunicación e información se desarrollan y tendrán aplicación social. Se anuncian ya las redes de telecomunicación multimedia, que darán lugar al cambio más grande de todos los tiempos.

Entre las TIC tenemos: la realidad virtual, que puede catalogarse como la multimedia interactiva en su máxima expresión; la formación de redes que pudiéramos nombrarla como la tendencia fundamental de las nuevas tecnologías y donde se destaca la red de redes **Internet**. El número de computadoras que se venden cada año en todo el mundo es creciente por lo que el mercado en general de las TIC apunta a un crecimiento vertiginoso en el uso de las nuevas tecnologías.

La incorporación de las computadoras a la velocidad de los negocios y la posibilidad de que en segundos cualquier información dé la vuelta al mundo varias veces obligan a un constante movimiento de ésta para que sea válida, ya que cuando no está actualizada no sirve para la toma de decisiones.

El desarrollo de las TIC ha llamado la atención de todas las empresas necesitadas de sistemas de procesamiento de información para acelerar sus procesos y elevar la calidad de los mismos.

2.2 ¿Qué es Internet?

El fenómeno social, cultural, sociológico y comercial de Internet tiene su origen en la década del 60 y se relaciona con un proyecto de defensa financiado por el gobierno de Estados Unidos. Gracias a esta iniciativa, hoy es posible buscar, crear y transferir información en tiempo real para 6 mil millones de personas.

Desde sus inicios, el crecimiento de Internet ha sido fenomenal, especialmente en la década del 90, época en que la red se convirtió en una herramienta fundamental de comunicación, información e integración que permite a los usuarios ahorrar tiempo y dinero, además de tener a su alcance todos los productos y servicios que requieran sin fronteras de espacio o tiempo.

2.3 ¿Cómo funciona Internet?

Para que dos computadoras se comuniquen vía Internet, debe existir un camino físico que los una (líneas telefónicas, conmutadas, redes digitales, enlaces satelitales, microondas, fibra óptica, cable coaxial, etc.) y un mismo protocolo de comunicación entre ellos (TCP/IP).

El desarrollo de Internet no sólo se ha traducido en beneficios para los usuarios, sino también para las empresas, organismos, instituciones, etc. Dentro de este ámbito el comercio electrónico ha tenido un crecimiento constante. Los principales productos que se trasmiten a través de la red son: información, música, viajes, libros, hardware y software.

2.4 La Web

World Wide Web (WWW), o simplemente Web, es el universo de información accesible a través de Internet, una fuente inagotable del conocimiento humano.

Es un sistema de información global, interactivo, dinámico, distribuido, gráfico, basado en Hipertexto, con plataforma de enlaces cruzados, que se ejecuta en Internet.

El componente más usado en el Internet es, definitivamente la Web. Su característica sobresaliente es el texto remarcado, un método para referencias

cruzadas instantáneas. Usando la Web, se tiene acceso a millones de páginas de información. La exploración se realiza por medio de un software especial denominado "Browser" o Explorador. La apariencia de un Sitio Web puede variar ligeramente dependiendo del explorador que use. Así mismo, las versiones más recientes disponen de una funcionalidad mucho mayor tal como animación, realidad virtual, sonido y música. El protocolo que se utiliza para la comunicación en la Web es el HTTP (Hypertext Transfer Protocol) y el formato que se utiliza para la transferencia es el HTML (Hypertext Markup Language).

2.5 Tecnología Cliente/Servidor

El concepto de cliente/servidor proporciona una forma eficiente de utilizar todos estos recursos de máquina, de tal forma que la seguridad y fiabilidad que proporcionan los entornos mainframe se traspasa a la red de área local. A esto hay que añadir la ventaja de la potencia y simplicidad de los ordenadores personales.

La arquitectura cliente/servidor es un modelo para el desarrollo de sistemas de información, en el que las transacciones se dividen en procesos independientes que cooperan entre sí para intercambiar información, servicios o recursos. Se denomina cliente al proceso que inicia el diálogo o solicita los recursos y servidor, al proceso que responde a las solicitudes.

Es el modelo de interacción más común entre aplicaciones en una red. No forma parte de los conceptos de Internet como los protocolos IP, TCP o UDP, sin embargo todos los servicios estándares de alto nivel propuestos en Internet funcionan según este modelo.

Los principales componentes del esquema cliente/servidor son entonces los Clientes, los Servidores y la infraestructura de comunicaciones.

En este modelo, las aplicaciones se dividen de forma que el servidor contiene la parte que debe ser compartida por varios usuarios, y en el cliente permanece sólo lo particular de cada usuario.

Los Clientes interactúan con el usuario, usualmente en forma gráfica. Frecuentemente se comunican con procesos auxiliares que se encargan de

establecer conexión con el servidor, enviar el pedido, recibir la respuesta, manejar las fallas y realizar actividades de sincronización y de seguridad.

2.6 Lenguajes de Programación para la Web

Uno de los ejes fundamentales que diferencian a Internet de otros medios de comunicación es la interacción y personalización de la información con el usuario. Esto se logra por medio de algunos de los diferentes lenguajes de programación para Web que existen hoy en día. Dichos lenguajes se clasifican en dos partes fundamentales que reconocen la propia arquitectura Cliente/Servidor de esta plataforma de desarrollo: los lenguajes del lado del Servidor y los lenguajes del lado del Cliente.

Entre los lenguajes del lado del servidor podemos encontrar entre los más sobresalientes por el auge que estos han tenido, algunos como PERL, ASP, PHP, Java, JSP, los módulos CGIs e ISAPIs etc. Estos se caracterizan por desarrollar la lógica de negocio dentro del Servidor, además de ser los encargados del acceso a Bases de Datos, tratamiento de la información etc. Del lado del cliente se encuentran principalmente el JavaScript y el Visual Basic Script, que son los encargados de aportar dinamismo a la aplicación en los navegadores.

Esta distinción en los lenguajes ha sido necesaria debido a que la Web funciona en modo "Desconectado", o sea, un usuario a través de un navegador hace una petición de una página Web a un Servidor Web (Request); el Servidor recepciona la petición, la procesa y le envía la respuesta al Cliente(Response), este la recepciona y se desconecta.

2.6.1 Practical Extraction and Report Language (Perl)

Es un lenguaje de programación muy utilizado para construir aplicaciones CGI para el Web. Perl es un acrónimo de Practical Extracting and Reporting Language, que viene a indicar que se trata de un lenguaje de programación muy práctico para extraer información de archivos de texto y generar informes a partir del contenido de los ficheros.

Es un lenguaje libre de uso, eso quiere decir que es gratuito. Antes estaba muy asociado a la plataforma Unix, pero en la actualidad está disponible en otros sistemas operativos como Windows. Perl es un lenguaje de programación interpretado, al igual que muchos otros lenguajes de Internet como JavaScript o ASP.

2.6.2 Active Server Pages (ASP).

ASP (Active Server Pages) es la tecnología desarrollada por Microsoft para la creación de páginas dinámicas del servidor. ASP se escribe en la misma página Web, utilizando el lenguaje Visual Basic Script o Jscript (de Microsoft).

La mayor desventaja que presenta este lenguaje es que solo se puede implementar en los Servidores Web de su desarrollador: Microsoft. Actualmente se ha presentado ya la segunda versión de ASP: el ASP.NET, que comprende algunas mejoras en cuanto a posibilidades del lenguaje y rapidez con la que funciona. ASP.NET tiene algunas diferencias en cuanto a sintaxis con el ASP, de modo que se ha de tratar de distinta manera uno de otro. Para implementarlo es necesario montar en el Servidor la Plataforma .NET.

2.6.3 Personal Home Page (PHP).

PHP es el acrónimo recursivo de Hypertext Preprocessor. Es un lenguaje de programación del lado del servidor gratuito e independiente de plataforma, rápido, con una gran librería de funciones y mucha documentación. Es también un lenguaje interpretado y embebido en el HTML.

PHP, en el caso de estar montado sobre un servidor Linux o Unix, es más rápido que ASP, dado que se ejecuta en un único espacio de memoria y esto evita las comunicaciones entre componentes COM[1] que se realizan entre todas las tecnologías implicadas en una página ASP.

[1] Iniciales de Common Objet Model. Un Componente desarrollado por Microsoft para el trabajo con Aplicaciones Web.

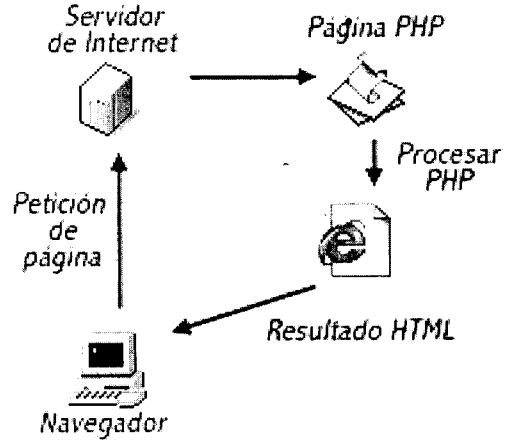


Figura 1.1 Esquema de representación del funcionamiento del PHP.

Fue creado originalmente en 1994 por Rasmus Lerdorf, pero como PHP está desarrollado en política de código abierto, a lo largo de su historia ha tenido muchas contribuciones de otros desarrolladores. Actualmente PHP se encuentra en su versión 5, que utiliza el motor Zend, desarrollado con mayor meditación para cubrir las necesidades de las aplicaciones Web actuales.

PHP es la gran tendencia en el mundo de Internet. Últimamente se puede observar un ascenso imparable, ya que cada día son muchísimas más las páginas Web que lo utilizan para su funcionamiento, según las estadísticas, PHP se utiliza en más de 10 millones de páginas, y cada mes realiza un aumento del 15%.

Resumiendo, el PHP corre en 7 plataformas, funciona en 11 tipos de servidores, ofrece soporte sobre unas 20 Bases de Datos y contiene unas 40 extensiones estables sin contar las que se están experimentando, además de que:

- Es software libre y abierto, lo que implica menos costo y servidores más baratos que otras alternativas.
- Es muy rápido. Su integración con la base de datos PostGreSQL y el servidor Apache, le permite constituirse como una de las alternativas más atractivas del mercado.
- Su sintaxis está inspirada en C, ligeramente modificada para adaptarlo al entorno en el que trabaja, de modo que si se está familiarizado con esta sintaxis, resultara muy fácil aprender PHP.

- Su librería estándar es realmente amplia, lo que permite reducir los llamados "costos ocultos", uno de los principales defectos de ASP.
- PHP tiene una de las comunidades más grandes en Internet, por lo que no es complicado encontrar ayuda, documentación, artículos, noticias y más recursos.
- Posee una potente variedad de extensiones para el acceso a la mayoría de los sistemas de gestión de bases de datos, por lo que una migración a otro sistema de gestión es mucho menos costosa que en otras plataformas.

2.6.4 Java Server Pages (JSP).

JSP es un acrónimo de Java Server Pages, que en castellano vendría a decir algo como Páginas de Servidor Java. Es pues, una tecnología orientada a crear páginas Web con programación en Java.

Con JSP podemos crear aplicaciones Web que se ejecuten en variados servidores Web, de múltiples plataformas, ya que Java es en esencia un lenguaje multiplataforma. Las páginas JSP están compuestas de código HTML/XML mezclado con etiquetas especiales para programar scripts de servidor en sintaxis Java. Por tanto, las JSP podremos escribirlas con nuestro editor HTML/XML habitual.

2.6.5 Selección de lenguaje a utilizar.

Hasta el momento se han analizado las características fundamentales de los lenguajes de programación candidatos para la implementación de la propuesta de este trabajo, para fundamentar nuestra elección haremos una comparación teniendo en cuenta algunas características que influyen directamente en el ambiente de trabajo donde se va a desarrollar la propuesta. En cuanto a:

- Características multiplataformas: Menos el ASP, que es solamente soportado por la plataforma Windows, los demás lenguajes están soportados en múltiples plataformas.
- Velocidad de ejecución: la velocidad es mayor en PHP, seguidos por PERL y JSP.

- Disponibilidad de recursos: actualmente los más utilizados en la Internet son el PHP y el JSP, siendo más utilizado en la publicación de artículos y códigos de ejemplos. PHP tiene una de las comunidades más grandes en Internet, al igual que la de Java.
- Familiaridad con el lenguaje: En la universidad los lenguajes más utilizados por los programadores es el ASP y el PHP.

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

2.7 Sistemas de Gestión de Bases de Datos (SGBD)

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

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

Una Base de Datos (BD) es un conjunto de datos interrelacionados, almacenados con carácter más o menos permanente en la computadora, puede ser considerada una colección de datos variables en el tiempo.

Un Sistema de Gestión de Base de Datos (SGBD) es el software que permite la utilización y/o la actualización de los datos almacenados en una (o varias)

base(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez.

El objetivo fundamental de un SGBD consiste en suministrar al usuario las herramientas que le permitan manipular, en términos abstractos, los datos, o sea, de forma que no le sea necesario conocer el modo de almacenamiento de los datos en la computadora, ni el método de acceso empleado.

Un SGBD tiene los siguientes objetivos específicos:

- Independencia de los datos y los programas de aplicación
- Minimización de la redundancia
- Integración y sincronización de las bases de datos
- Integridad de los datos
- Seguridad y protección de los datos
- Facilidad de manipulación de la información
- Control centralizado

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

2.7.1 MySQL

MySQL es un sistema de administración de Base de Datos. Opera en una arquitectura cliente/servidor. Es un proyecto "open sourse". Permite la fácil conectividad, alta velocidad de respuesta a solicitudes y gran seguridad, por ello se utiliza para acceder a Bases de Datos desde Internet. [MySQL-a] [MySQL-b]

MySQL es muy rápido, confiable y fácil de usar, es multiplataforma, multiusuario y permite elaborar consultas con el robusto SQL, además no tiene valor monetario, es un software que se puede adquirir libremente, la licencia es completamente libre.

El lenguaje PHP es altamente compatible con MySQL, por el amplio conjunto de comandos definidos para el tratamiento de este.

2.7.2 SQL Server

Microsoft SQL Server, propietario de Microsoft, pertenece a la familia de los sistemas de administración de base de datos, operando en una arquitectura cliente/servidor de gran rendimiento. Su desarrollo fue orientado para hacer posible manejar grandes volúmenes de información, y un elevado número de transacciones. SQL Server es una aplicación completa que realiza toda la gestión relacionada con los datos. El servidor sólo tiene que enviarle una cadena de caracteres (la sentencia SQL) y esperar a que le devuelvan los datos.

SQL Server permite la creación de procedimientos almacenados, los cuales consisten en instrucciones SQL que se almacenan dentro de una base de datos de SQL Server, realizados en lenguaje SQL, se trata de procedimientos que se guardan semicompilados en el servidor y que pueden ser invocados desde el cliente. Se ejecutan más rápido que instrucciones SQL independientes.

SQL Server puede manejar perfectamente bases de datos de TeraBytes con millones de registros y funciona sin problemas con miles de conexiones simultáneas a los datos, sólo depende de la potencia del hardware del equipo en el que esté instalado y solamente corre sobre Windows NT- 2000 Server.

2.7.3 PostgreSQL

PostgreSQL es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (ORDBMS) que ha sido desarrollado de varias formas desde 1977. Comenzó como un proyecto denominado Ingres en la Universidad Berkeley de California. Ingres fue más tarde desarrollado comercialmente por la Relational Technologies/Ingres Corporation.

En 1986 otro equipo dirigido por Michael Stonebraker de Berkeley continuó el desarrollo del código de Ingres para crear un sistema de bases de datos objeto-relacionales llamado Postgres. En 1996, debido a un nuevo esfuerzo de código abierto y a la incrementada funcionalidad del software, Postgres fue renombrado a PostgreSQL, tras un breve periplo como Postgres95. El proyecto PostgreSQL

sigue actualmente un activo proceso de desarrollo a nivel mundial gracias a un equipo de desarrolladores y contribuidores de código abierto.

PostgreSQL está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo. Posee muchas características (las cuales serán discutidas más adelante) que tradicionalmente sólo se podían ver en productos comerciales de alto calibre.

2.7.4 Selección del Sistema de Gestión de Bases de Datos (SGBD)

Luego de un estudio detallado de los SGBD antes mencionados, hemos seleccionado el **PostgreSQL** como herramienta para almacenar y gestionar los datos de nuestro sistema.

¿Qué características tiene PostgreSQL que lo hace nuestra elección?

PostgreSQL ofrece muchas ventajas para una compañía o negocio, respecto a otros sistemas de bases de datos, en cuanto a:

1- *Instalación ilimitada*

Es frecuente que las bases de datos comerciales sean instaladas en más servidores de lo que permite la licencia. Algunos proveedores comerciales consideran a esto la principal fuente de incumplimiento de licencia. Con PostgreSQL, nadie puede demandarlo por violar acuerdos de licencia, puesto que no hay costo asociado a la licencia del software.

Esto trae consigo ventajas adicionales como:

✓ Modelos de negocios más rentables con instalaciones a gran escala.

✓ No existe la posibilidad de ser auditado para verificar cumplimiento de licencia en ningún momento.

✓ Flexibilidad para hacer investigación y desarrollo sin necesidad de incurrir en costos adicionales de licenciamiento.

2- *Ahorros considerables en costos de operación*

Este software ha sido diseñado y creado para tener un mantenimiento y ajuste mucho menor que los productos de los proveedores comerciales, conservando todas las características, estabilidad y rendimiento.

Además de esto, los programas de entrenamiento son reconocidamente mucho más costo-efectivos, manejables y prácticos en el mundo real que aquellos de los principales proveedores comerciales.

3- *Estabilidad y confiabilidad legendarias*

En contraste a muchos sistemas de bases de datos comerciales, es extremadamente común que compañías reporten que PostgreSQL nunca ha presentado caídas en varios años de operación de alta actividad.

4- *Extensible*

El código fuente está disponible para todos sin costo. Si un equipo necesita extender o personalizar PostgreSQL de alguna manera, pueden hacerlo con un mínimo esfuerzo, sin costos adicionales. Esto es complementado por la comunidad de profesionales y entusiastas de PostgreSQL alrededor del mundo que también extienden PostgreSQL todos los días.

5- *Multiplataforma*

PostgreSQL está disponible en casi cualquier Unix (34 plataformas en la última versión estable), y una versión nativa de Windows está actualmente en estado beta de pruebas.

6- *Diseñado para ambientes de alto volumen*

PostgreSQL usa una estrategia de almacenamiento de filas llamada MVCC para conseguir una mejor respuesta en ambientes de grandes volúmenes. Los principales proveedores de sistemas de bases de datos comerciales usan también esta tecnología, por las mismas razones.

7- *Herramientas gráficas de diseño y administración de bases de datos.*

Existen varias herramientas gráficas de alta calidad para administrar las bases de datos (pgAdmin, pgAccess) y para hacer diseños de bases de datos (Tora, Data Architect).

Además PostgreSQL ofrece una serie de características técnicas, al igual que otros gestores, que permiten un mejor trabajo con las bases de datos; entre estas podemos encontrar:

✓ Replicación (soluciones comerciales y no comerciales) que permiten la duplicación de bases de datos maestras en múltiples sitios de réplica.

- ✓ Interfaces nativas para ODBC, JDBC, C, C++, PHP, Perl, TCL, ECPG, Python y Ruby.
- ✓ Reglas
- ✓ Vistas
- ✓ Triggers
- ✓ Unicode
- ✓ Secuencias
- ✓ Herencia
- ✓ Outer Joins
- ✓ Sub-selects
- ✓ Una API abierta
- ✓ Procedimientos almacenados
- ✓ Soporte nativo SSL
- ✓ Lenguajes procedurales
- ✓ Índices parciales y funcionales
- ✓ Soporte para consultas con UNION, UNION ALL y EXCEPT
- ✓ Extensiones para SHA1, MD5, XML y otras funcionalidades
- ✓ Herramientas para generar SQL portable para compartir con otros sistemas compatibles con SQL
- ✓ Sistema de tipos de datos extensible para proveer tipos de datos definidos por el usuario y rápido desarrollo de nuevos tipos.
- ✓ Funciones de compatibilidad para ayudar en la transición desde otros sistemas menos compatibles con SQL.

2.8 Metodología a utilizar

La calidad en el desarrollo y mantenimiento del software se ha convertido hoy en día en uno de los principales objetivos estratégicos de las organizaciones, debido a que cada vez más, los procesos principales dependen de los sistemas informáticos para su buen funcionamiento. En los últimos años se han publicado diversos estudios y estándares en los que se exponen los principios que se deben seguir para la mejora de los procesos de software.

Una metodología para el desarrollo de un proceso de software es un conjunto de filosofías, fases, procedimientos, reglas, técnicas, herramientas, documentación y aspectos de formación para los desarrolladores de Sistemas Informáticos. Por ello escoger la metodología que va a guiar el proceso de desarrollo del sistema es un paso tan importante.

2.8.1 Lenguaje Unificado de Modelado (UML).

El desarrollo del Unified Modeling Lenguaje, (UML) empezó en octubre de 1994, cuando Grady Booch y Jim Rumbaugh en la Rational Software Corp. empezaron a trabajar para unificar el Booch (Metodología de Grady Booch) y la OMT (*Object Modeling Techniques*). Un proyecto versión 0.8 del Método Unificado (UML), como se llamó desde un comienzo, salió al público en octubre de 1995. En el otoño de 1995, Ivar Jacobson se unió a la compañía y unió su esfuerzo al nuevo modelo, uniendo el OOSE (*Object Oriented Software Engineering*) al UML.

En resumen UML es el resultado de la experiencia sumada, anotaciones, y conceptos. Ya que todas las metodologías bases han tenido una aplicación extensa en el campo de la POO, ha sido desarrollado en la práctica, tienen su historia, y han sido aplicados en una gran variedad de industrias y problemas por lo que pueden ser clasificadas como muy maduras. UML no es una salida revolucionaria de Booch, OMT, y OOSE, sino una evolución y síntesis de estos tres.

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

2.8.2 El Proceso Unificado de Modelado (RUP).

El objetivo final de cualquier aplicación, es un software robusto, flexible y escalable, por lo que es necesario tanto un lenguaje como un proceso para poder obtenerlo.

El *Proceso Unificado de Rational* (RUP), es un proceso de ingeniería de software planteado por Kruchten (1996) cuyo objetivo es producir software de alta calidad, es decir, que cumpla con los requerimientos de los usuarios dentro de una planificación y presupuesto establecido.

RUP toma en cuenta las mejores prácticas en el modelo de desarrollo de software en particular las siguientes:

- ✓ Desarrollo de software en forma iterativa (repite una acción).
- ✓ Manejo de requerimientos.
- ✓ Utiliza arquitectura basada en componentes.
- ✓ Modela el software visualmente (modela con UML).
- ✓ Verifica la calidad del software.
- ✓ Controla los cambios.

El Proceso Unificado de Rational (RUP) consta de cuatro fases o etapas:

- ✓ Fase de comienzo o inicio.
- ✓ Fase de Elaboración.
- ✓ Fase de Construcción.
- ✓ Fase de Transición.

2.9 Herramientas CASE

¿Por qué deberíamos usar herramientas CASE de modelado con UML?

A medida que los sistemas que hoy se construyen se tornan más y más complejos, las herramientas de modelado con UML ofrecen muchos beneficios para todos los involucrados en un proyecto, por ejemplo, administrador del proyecto, analistas, arquitectos, desarrolladores y otros. Las herramientas CASE de modelado con UML nos permiten aplicar la metodología de análisis y diseño orientados a objetos y abstraernos del código fuente, en un nivel donde la

arquitectura y el diseño se tornan más obvios y más fáciles de entender y modificar. Cuanto más grande es un proyecto, es más importante utilizar una herramienta CASE.

Algunos ejemplos de herramientas CASE:

- ASADAL - Herramienta CASE especializada en Sistemas de Tiempo Real.
- CASE GENEXUS Tool.
- System Architect, herramientas CASE para Análisis y Diseño, incluye técnicas estructuradas y orientadas a objetos.
- Win A&D, herramientas CASE para Análisis y Diseño, incluye técnicas estructuradas y orientadas a objetos.
- CRADLE, conjunto de herramientas CASE integradas que dan soporte a la Planificación estratégica, Análisis y Diseño.
- PowerDesigner 7.0: herramienta CASE de Análisis y Diseño incluye capacidades de generación relacional y con orientación a objetos.
- SilverRun: Conjunto integrado de herramientas CASE para el modelado de negocios.
- Racional Rose, herramienta CASE para Análisis y Diseño basándose en el Proceso Unificado de Rational (RUP).
- Visual Paradigm, herramienta CASE para Análisis y Diseño, utiliza el Lenguaje Unificado de Modelado (UML)

Selección de la herramienta CASE a utilizar:

La herramienta CASE seleccionada fue el Visual Paradigm ya que nos ofrece un entorno de creación de diagramas para UML; diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad; uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación; capacidades de ingeniería directa (versión profesional) e inversa; modelo y código que permanece sincronizado en todo el ciclo de desarrollo; disponibilidad de múltiples versiones, para cada necesidad; disponibilidad de integrarse en los principales IDEs y disponibilidad en múltiples plataformas.

2.10 Otras Herramientas necesarias

Como se pretende implementar una aplicación Web para confeccionar la propuesta de este trabajo, se hace necesario tener en cuenta la utilización de un editor de páginas Web, y una herramienta para el trabajo con las imágenes.

Para estas funciones la elección no ha sido muy difícil, ya que la herramienta de creación de sitios Web más utilizada en la actualidad es el *Macromedia Dreamweaver 8*. Con esa herramienta se podrá desarrollar cualquier sitio Web personal con características de sitio profesional y utilizar casi todos los recursos de la Web, así como realizar aplicaciones que se ejecuten en servidor y vinculaciones dinámicas de datos; además de contar con un soporte para aplicaciones PHP y utilización de bases PostgreSQL. También cuenta con un amplio soporte para la creación y utilización de CSS (*Cascading Style Sheets*) para lograr un diseño fácil y óptimo.

Finalmente se escogió el *Adobe Photoshop 7* como herramienta principal para crear las imágenes del Sistema, ya que se considera la aplicación estándar para el tratamiento digital de imágenes. Las continuas mejoras han hecho de este programa uno de los más profesionales para la edición y retoque fotográfico. Tiene un enfoque dirigido hacia los gráficos para la Web, y posee una total integración con su avanzada herramienta de producción Web: *Adobe ImageReady 3.0*.

También utilizamos *Smarty* que es un motor de plantillas para PHP. La finalidad de trabajar con plantillas es la de separar el código PHP del código HTML, con la ventaja de que un diseñador pueda trabajar en su ámbito sin tener que saber PHP. Por consiguiente, el programador puede hacer los cambios a la lógica de la aplicación sin la necesidad de reestructurar el diseño, y el diseñador puede hacer los cambios a las plantillas sin romper la lógica de la aplicación. Algunos de los principales aspectos de SMARTY son:

- Es sumamente rápido.
- Ninguna plantilla se analiza dos veces, sólo compila una vez.
- Tiene inteligencia para recompilar sólo los archivos de las plantillas que han cambiado.

- Se pueden hacer funciones personalizadas y personalizar las variables, por lo que el idioma de la plantilla es sumamente extensible.
- Se puede configurar los delimitadores que etiquetan la sintaxis, se puede usar {}, {{}}, <!--{}-->, <% %>,etc.
- Se pueden anidar ilimitadas secciones de if/else, for, foreach, etc.
- El uso arbitrario de las fuentes de la plantilla, o sea que una plantilla puede ser usada por varias pág. PHP, siempre que muestren el mismo contenido.

2.11 Propuesta solución

Después de realizar un análisis sobre algunas de las herramientas que se utilizan para el proceso de recuperación, y determinar claramente cual es la situación actual sobre el objeto de estudio que tiene este trabajo, se concluye que se hace necesario implementar una herramienta que sea capaz de acoplarse a una aplicación donde se gestionen datos y extraer toda la información que existe en su base de datos, y efectuar las recuperaciones de una forma eficiente y lo más exacta posible.

Basado en los argumentos antes expuestos y dado que se hará uso de tecnologías cliente/servidor sobre plataforma Web para implementar este trabajo, se ha escogido al *PHP* como lenguaje de programación producto a su portabilidad y eficiencia, y para una mayor facilidad de trabajo se usará el motor de plantillas de *SMARTY*, y como SGBD el *PostGreSQL*.

La implantación y adquisición en nuestro país del software libre es una de nuestras principales metas, se pretende ayudar a convencer a nuestra comunidad informática radicada en las distintas empresas, de que con software libre bajo la licencia GNU/GPL se le pueden dar solución a disímiles problemas que nos encontramos hoy en día sin necesidad de invertir en software y sistemas operativos propietarios.

2.12 Conclusiones

En este capítulo se detallaron las condiciones y problemas que rodean el objeto de estudio; y a través de los conceptos y definiciones planteadas, se determinaron las condiciones específicas que envuelven al mismo. En base a esto se obtuvieron los objetivos generales y específicos para este trabajo, se planteó una solución al problema, se realizó un análisis completo de las tecnologías que serán utilizadas a lo largo del desarrollo del sistema propuesto, y se fundamentaron las elecciones del lenguaje, el sistema gestor de base de datos y la metodología a utilizar. Una vez conocidas las herramientas óptimas y los conceptos a utilizar se puede empezar a desarrollar la propuesta de sistema.

Capítulo

III

DESCRIPCIÓN DE LA SOLUCION PROPUESTA

3.1 Introducción

En el presente capítulo se hace la descripción de la propuesta que trae este trabajo, para ello se describen los procesos del negocio que tiene que ver con el objeto de estudio, de acuerdo a esto se llega a la conclusión que debido a la poca estructuración de esos procesos, para poder entender el contexto en que se emplaza el sistema, necesitamos definir conceptos que podemos agrupar en un Modelo de Dominio para capturar correctamente los requisitos y poder construir un sistema correcto.

Además se enumeran los requisitos funcionales y no funcionales que debe tener el sistema que proponemos, lo que permite hacer una concepción general del sistema, e identificar mediante un Diagrama de Casos de Uso, las relaciones de los actores que interactúan con el sistema y las secuencias de acciones con las que interactúan.

El objetivo del Proceso Unificado, dirigido por los casos de uso, centrado en la arquitectura, iterativo e incremental, es guiar a los desarrolladores de cualquier sistema software en la implementación y distribución eficiente de sistemas que se ajusten a las necesidades de los clientes.

En este capítulo que comienza aparece información respecto al Modelo de Dominio de la aplicación, específicamente en el Proceso Unificado para la definición del dominio de la aplicación y sus conceptos asociados; los requerimientos del sistema, tanto funcionales como no funcionales; los conceptos asociados al dominio, descripción de los procesos y eventos principales, así como

la representación del problema en términos informáticos, y por ultimo, los respectivos *casos de uso* en cada una de estas estructuras.

3.2 Descripción de los procesos del negocio propuestos

Para obtener una descripción detallada de los procesos del negocio que se relacionan con el campo de acción, se hace necesario centrar la atención en los procesos de recuperación y la relación entre las aplicaciones de recuperación o mediante las cuales se realiza la recuperación que existe en el MINFAR.

A través de las técnicas de modelado que propone UML, se puede comprender mejor como se lleva a cabo la recuperación de información en el entorno al cual hacemos referencia. El primer paso dentro del modelado del negocio es la identificación de los diferentes procesos del negocio que existen en la organización dirigidos al tema en cuestión. La obtención de un adecuado conjunto de procesos del negocio es una cuestión crucial puesto que establece los límites del modelado.

Actualmente cualquier persona previamente autorizada que desee recuperar algún tipo de dato de la información que se encuentra almacenada, necesita acceder a esta a través de la realización de consultas realizadas directamente al sistema de almacenamiento que se esta empleando. En caso de aparecer complicaciones o presentar un bajo nivel de conocimientos de manera que no pueda realizar la consulta, existe un encargado o técnico informático al cual debe consultar.

El mantenimiento de la información almacenada es realizado por los administradores o determinada persona con plenos permisos para guardar, transformar, insertar o eliminar cualquier dato.

3.3 Modelo del Dominio

Considerando las descripciones de los procesos anteriormente comentados, se llega a la conclusión de que el negocio que se está estudiando tiene muy bajo nivel de estructuración, con soluciones muy diversas y dispersas, aunque todas llevan el mismo propósito de satisfacer una necesidad de recuperar información.

Para ello se utilizará un modelo del dominio, ya que permite de manera visual mostrar los principales conceptos que se manejan en el dominio del sistema en desarrollo. Esto ayuda a los usuarios, clientes y desarrolladores e interesados a utilizar un vocabulario común para poder entender el contexto en que se emplaza el sistema. Para capturar correctamente los requisitos y poder construir un sistema correcto se necesita tener un firme conocimiento del funcionamiento del objeto de estudio. Este modelo va a contribuir posteriormente a identificar algunas clases que se utilizarán en el sistema.

3.3.1Conceptos principales

Usuarios: cualquier persona autorizada que trabaje o pertenezca a la institución y solicite información almacenada.

Almacenero: es aquella persona que manipula la información que no es almacenada en formato digital (guardar, almacenar, ubicar).

Datos: conjunto organizado e integrado de información almacenada y clasificada en computadora o papel para su posterior consulta, actualización o cualquier tarea de mantenimiento mediante operaciones específicas.

Datos Digitales: representan datos almacenados en formato digital.

Datos Papel: representan datos almacenados en papel y otros componentes sólidos de grabados.

Técnico Informático: es el especialista que trabaja con la base de datos y se encarga del manejo de la información que se encuentra almacenada en formato digital.

Entidad: organización a la que pertenece el usuario.

3.3.2 Principales Eventos

Recuperar información: es el proceso mediante el cual el usuario respondiendo a una solicitud necesita obtener información de los datos almacenados.

Gestiona: proceso mediante el cual una persona encargada maneja la información que existe, es decir provee, inserta, modifica, elimina y manipula la información almacenada.

Pertenece: proceso que identifica al usuario especificando la entidad a la que pertenecen.

3.3.3 Diagrama de clases del dominio

El modelo del dominio se describe mediante diagramas UML, específicamente con un diagrama de clases conceptuales significativas en el dominio del problema.

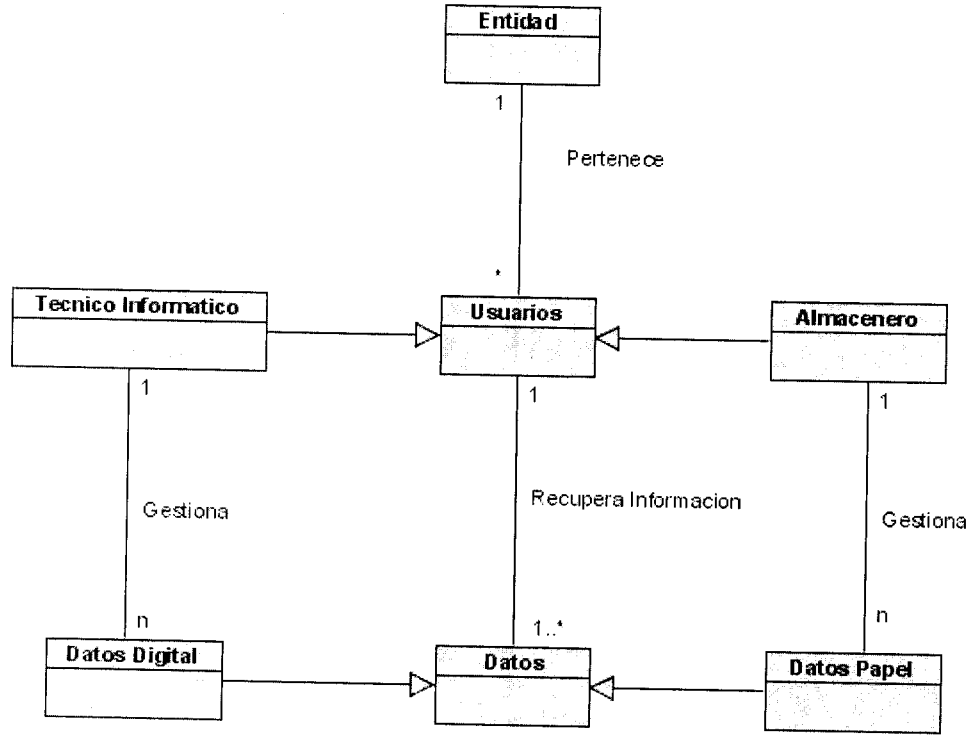


Figura 3.1: Diagrama de Clases del Dominio.

3.4 Requerimientos funcionales

Una vez conocidos los conceptos que rodean al objeto de estudio, se debe analizar: ¿Qué debe hacer el sistema para que se cumplan los objetivos planteados al inicio de este trabajo?, para ello se enumeran, a través de requerimientos funcionales, las acciones que el sistema deberá ser capaz de realizar. Dentro de ellos se incluyen las acciones que podrán ser ejecutadas por el usuario, las acciones ocultas que debe realizar el sistema y las condiciones extremas a determinar por el sistema.

R1. Actualizar sistemas.

1.1. Mostrar sistemas existentes.

1.2. Permitir al configurador insertar un nuevo sistema.

1.3. Permitir al configurador modificar un sistema ya existente.

1.4. Permitir al configurador eliminar sistemas.

R2. Actualizar subsistemas.

2.1. Mostrar todos los subsistemas existentes.

2.2. Permitir al configurador insertar un nuevo subsistema.

2.3. Permitir al configurador modificar un subsistema ya existente.

2.4. Permitir al configurador eliminar subsistemas.

R3. Actualizar campos

3.1. Mostrar campos existentes.

3.2. Permitir al configurador insertar un nuevo campo.

3.3. Permitir al configurador modificar un campo ya existente.

3.4. Permitir al configurador eliminar campos.

R4. Definir relaciones

4.1. Mostrar todas las relaciones existentes.

4.2. Permitir al configurador insertar una nueva relación.

4.3. Permitir al configurador modificar una relación ya existente.

4.4. Permitir al configurador eliminar relaciones.

R5. Definir dependencias

5.1. Mostrar todas las relaciones existentes.

5.2. Permitir al configurador insertar una nueva dependencia.

5.3. Permitir al configurador modificar una dependencia ya existente.

5.4. Permitir al configurador eliminar dependencias.

R6. Clasificar las tablas de los esquemas.

6.1. Permitir seleccionar el esquema que desea clasificar.

6.2. Permitir seleccionar la clasificación de la tabla.

6.3. Permitir salvar las clasificaciones de las tablas definidas.

R7. Actualizar la base de datos.

7.1. Permitir ver los reportes de las actualizaciones realizadas.

7.2. Permitir al configurador eliminar cualquier reporte de actualización de la base de datos que se visualizan.

7.3. Permitir realizar una nueva actualización.

7.3.1. Visualizar el reporte de la nueva actualización.

7.4. Salir de actualizar arquitectura de la base de datos.

R8. Definir condiciones.

8.1. Mostrar condiciones por campos.

8.2. Agregar una nueva condición.

8.3. Modificar una condición existente.

8.4. Eliminar.

8.5. Mostrar condiciones formadas.

R9. Definir campos a mostrar

9.1. Mostrar campos que se van a mostrar en el informe recuperado.

9.2. Agregar un nuevo campo.

9.3. Modificar campos mostrados.

9.4. Eliminar campos mostrados.

R10. Definir formato del informe.

10.1.Definir formato del titulo que se mostrara en el informe de la recuperación.

10.2.Definir formato del cuerpo del informe que se mostrará como resultado de la recuperación.

R11. Generar plantilla.

11.1.Permitir al recuperador abrir una plantilla almacenada con anterioridad.

11.2.Permitir eliminar alguna(s) de las plantillas almacenadas.

11.3.Permitir que el recuperador salve como plantilla cualquier configuración de recuperación que haya realizado.

11.4.Permitir el recuperador realizar un nuevo informe.

R12. Visualizar reporte de la recuperación.

16.1 Visualizar reporte por Listado.

16.2 Visualizar reporte por Estadística.

R13. Definir encabezados.

13.1.Mostrar campos a seleccionar.

13.2.Definir mostrar totales.

R14. Permitir personalizar el campo seleccionado.

14.1.Insertar condiciones por etiquetas del campo.

14.2.Modificar condiciones por etiquetas del campo.

14.3.Eliminar condiciones por etiquetas del campo.

14.4. Mostrar condiciones por etiquetas.

R15. Definir campos por columnas.

15.1.Mostrar campos a seleccionar.

15.2.Definir mostrar totales.

R16. Definir campos por filas.

16.1.Mostrar campos a seleccionar por filas y que aparecerán en el reporte de la recuperación.

16.2.Agregar campo.

16.3.Eliminar campo.

16.4.Mostrar campos seleccionados.

16.5.Definir totales.

3.5 Requerimientos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable.

Apariencia o interfaz externa:

✓ Diseño sencillo, con pocas entradas, permitiendo que no sea necesario mucho entrenamiento para utilizar el sistema.

✓ Empleo de los colores: gris, blanco, azul y rojo principalmente, que son los definidos en los estándares del proyecto.

Usabilidad:

✓ El sistema podrá ser usado por cualquier persona que posea conocimientos mínimos de base de datos y conocimientos medios en el manejo de la computadora.

✓ El software tendrá siempre la posibilidad de ayuda disponible para cualquier tipo de usuario, lo que le permitirá un avance considerable en la explotación de la aplicación en todas sus funcionalidades.

Rendimiento:

✓ Tiempos de respuestas rápidos al igual que la velocidad de procesamiento de la información, no mayor a los 5 segundos en las actualizaciones y no mayor de 20 para las recuperaciones.

Soporte:

✓ Se requiere un servidor de bases de datos con las siguientes características:

✓ Soporte para grandes volúmenes de datos y velocidad de procesamiento.

✓ Tiempo de respuesta rápido en accesos concurrentes.

✓ Versión de PHP 5.0.

✓ Por parte del cliente se requiere un navegador capaz de interpretar JavaScript.

Portabilidad:

✓ Necesidad de que el sistema sea multiplataformas.

Seguridad:

✓ Autenticación (contraseña de acceso)

✓ Garantizar que las funcionalidades del sistema se muestren de acuerdo al nivel de usuario que este activo.

✓ Protección contra acciones no autorizadas o que puedan afectar la integridad de los datos.

✓ Verificación sobre acciones irreversibles (eliminaciones).

Legales:

✓ El sistema se basa en el manual de normas y principios establecidos por el MINFAR.

Confiabilidad:

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

Funcionalidad:

✓ Mínima cantidad de páginas para ejecutar todas las funciones posibles (preferentemente que estén relacionadas).

Implantación:

- ✓ Entregar toda la documentación asociada al proyecto.
- ✓ Organizar el adiestramiento de los usuarios.

Software:

En secciones anteriores se ha mencionado que la construcción de nuestra aplicación funcionará bajo los conceptos de arquitectura cliente/servidor. Por tanto el servidor del usuario final debe tener como requerimientos mínimos de software:

- ✓ Una computadora personal con plataforma del sistema operativo Windows Advancer Server 2000 o superior; o Linux.
- ✓ Apache 2.0 o superior como servidor Web, con módulo PHP 5 disponible y debe estar configurado con la extensión pgsql incluida.
- ✓ PostgreSQL como Sistema Gestor de Base de Datos.

Y la máquina cliente del usuario debe tener como requerimiento mínimo:

- ✓ El navegador Mozilla FireFox.

Hardware:

Partiendo del mismo supuesto que los requerimientos de software, nuestro modelo ideal (cliente/servidor), para los requerimientos mínimos de hardware, el usuario final debe tener un servidor con las siguientes características:

- ✓ Tarjeta de red.
- ✓ 128 Mb. de RAM o superior.
- ✓ 40 Gb. de disco duro o superior.
- ✓ Pentium II a 133 MHz de velocidad en su procesador o más.

Una computadora que sirva de cliente:

- ✓ Pentium a 200 MHz. de velocidad de procesamiento o superior.
- ✓ 32 Mb. de memoria RAM superior.
- ✓ Tarjeta de red.

3.6 Modelo de casos de uso del sistema

Utilizando las facilidades que brinda el UML, se representarán los requisitos funcionales del sistema mediante un diagrama de casos de uso. Para ello hay que definir de acuerdo a lo planteado en los epígrafes anteriores, cuales serían los

actores que van a interactuar con el sistema y los casos de uso que van a representar las funcionalidades.

Un caso de uso es un documento narrativo que describe la secuencia de un actor (agente externo) que utiliza un sistema para completar un proceso. Un actor no es parte del sistema, sino un rol que se juega dentro del sistema, que puede intercambiar información o puede ser un recipiente pasivo de información y representa a un ser humano, a un software o a una máquina que interactúa con el sistema. En este caso interactúan tres actores que se definen a continuación.

Teniendo en cuenta todos los requerimientos planteados y para cumplimentar los objetivos propuestos al inicio de este trabajo, el sistema que se propone se ha dividido en tres paquetes para una mejor compresión, un paquete que contiene todos los casos de usos relacionados con la configuración de la recuperación, otro paquete que agrupa todos los casos de usos relacionados con la recuperaciones dinámicas que se pueden obtener a partir de los dos subpaquetes contenidos en este último, uno para la recuperación por listado y el otro para la recuperación por estadísticas.

Representación por paquetes:

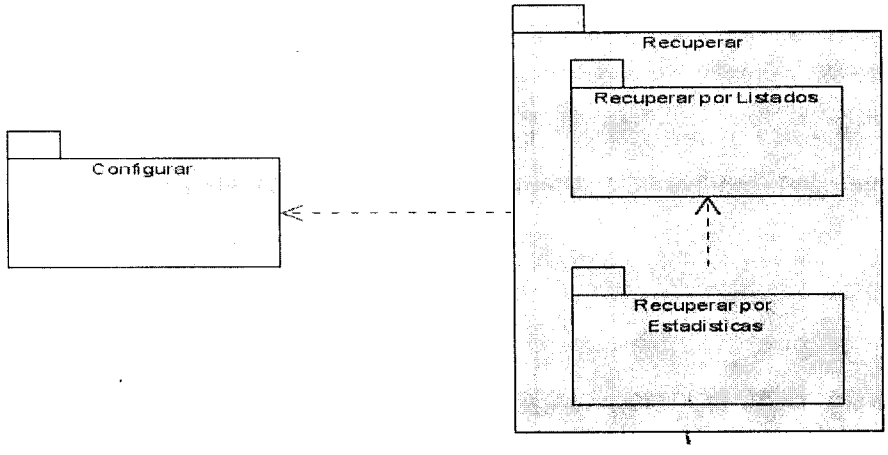


Figura 3.2 Diagrama de paquetes del sistema.

Se considera la existencia de dos roles, el primero se le llamará con el nombre de Configurador, que será el encargado de utilizar el paquete de Configuración y el segundo se denomina con el nombre de Recuperador que utilizará el paquete de Recuperar.

Actores	Justificación
Configurador	Representa a una persona con acceso solamente al módulo configuración donde podrá definir diferentes eventos que serán utilizados para la recuperación.
Recuperador	Representa a una persona que tiene permisos para recuperar la información a la que tiene acceso y conformar un informe. Puede utilizar el sistema para recuperar información por listado y por estadística.

A continuación se presentan los casos de uso determinados para satisfacer los requerimientos funcionales del sistema:

CU-1	Actualizar Sistema
Actor	Configurador
Descripción	El configurador solicita la actualización del sistema con que se trabajara para la recuperación de información.
Referencia	R1

CU-2	Actualizar Subsistema
Actor	Configurador
Descripción	El configurador solicita la actualización del subsistema con que se trabajara para la recuperación de información.
Referencia	R2

CU-3	Actualizar Campos del Subsistema
Actor	Configurador
Descripción	El configurador define los campos que van a estar contenidos en su subsistema
Referencia	R3

CU-4	Generar Relaciones
Actor	Configurador
Descripción	El configurador solicita definir relaciones entre los campos que conforman las tablas de los esquemas.
Referencia	R4

CU-5	Generar Dependencias
Actor	Configurador
Descripción	El configurador solicita definir dependencias entre los campos que conforman las tablas de los esquemas.
Referencia	R5

CU-6	Clasificar Tablas.
Actor	Configurador
Descripción	El configurador solicita la clasificación de las tablas de los esquemas de la base de datos.
Referencia	R6

CU-7	Actualizar Arquitectura de la Base de Datos
Actor	Configurador
Descripción	El configurador solicita la actualización de la arquitectura de la base de datos.
Referencia	R7

CU-8	Definir Condiciones
Actor	Recuperador.
Descripción	El recuperador desea definir las condiciones que se tomarán en cuenta para la recuperación.
Referencia	R8

CU-9	Definir Campos a Mostrar.
Actor	Recuperador.
Descripción	El recuperador escoge los campos que se van a mostrar en informe la recuperación.
Referencia	R9

CU-10	Crear Formato del Informe.
Actor	Recuperador.
Descripción	El recuperador solicita crear el informe en el que se mostrarán los resultados de la recuperación.
Referencia	R10

CU-11	Generar Plantilla
Actor	Recuperador.
Descripción	El recuperador desea abrir una de las plantillas almacenadas con anterioridad.
Referencia	R11

CU-12	Visualizar Reporte
Actor	Recuperador.
Descripción	El recuperador desea visualizar los reportes de su recuperación, puede ser por listado o por estadística.
Referencia	R12

CU-13	Definir Encabezados.
Actor	Recuperador.
Descripción	El recuperador selecciona los encabezados que van a tener los diferentes

	reportes por tablas que se obtendrán en la recuperación.
Referencia	R13

CU-14	Personalizar.
Actor	Recuperador.
Descripción	El recuperador desea definir diferentes condiciones por campo para su recuperación.
Referencia	R16

CU-15	Definir Campos por columnas.
Actor	Recuperador.
Descripción	El recuperador define los campos que se mostrarán en las columnas.
Referencia	R15

CU-16	Definir Campos por Filas
Actor	Recuperador
Descripción	El recuperador define los campos que se mostrarán en las filas.
Referencia	R16

3.6.1 Representación gráfica de los casos de uso del sistema

El diagrama donde se representa la relación existente entre los actores y los casos de uso se representa a continuación:

Paquete de Configuración:

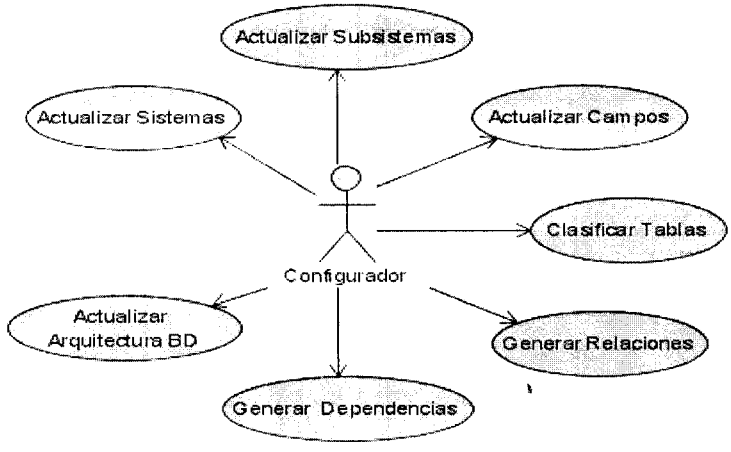


Figura 3.3 Casos de uso del paquete Configurar.

Paquete de Recuperación por Listado:

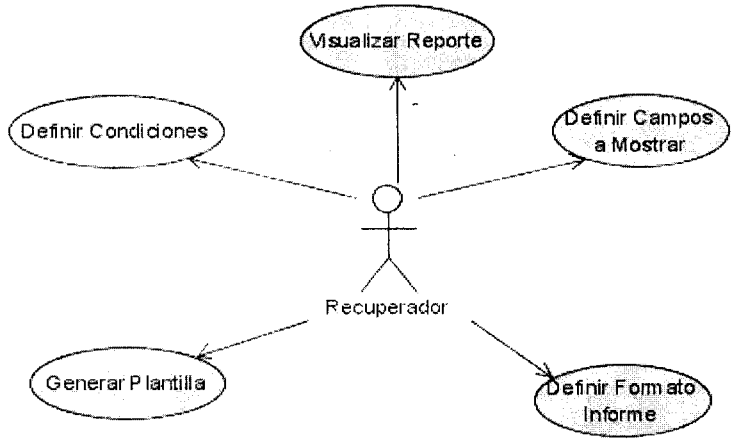


Figura 3.4 Casos de uso del paquete Recuperar por Listados.

Paquete de Recuperación por Estadísticas:

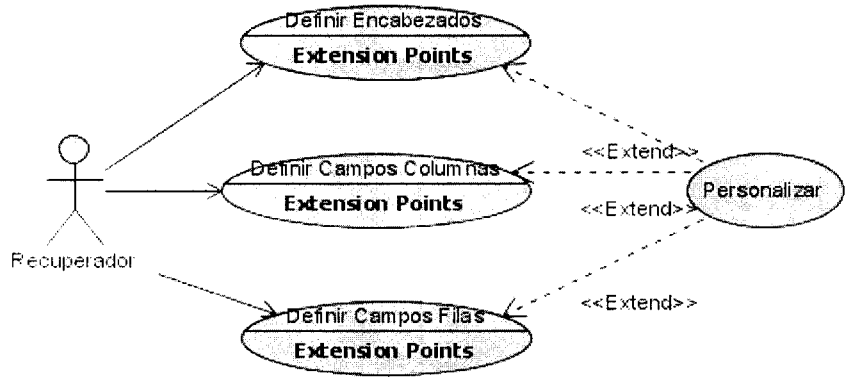


Figura 3.5 Casos de uso del paquete Recuperar por Estadísticas.

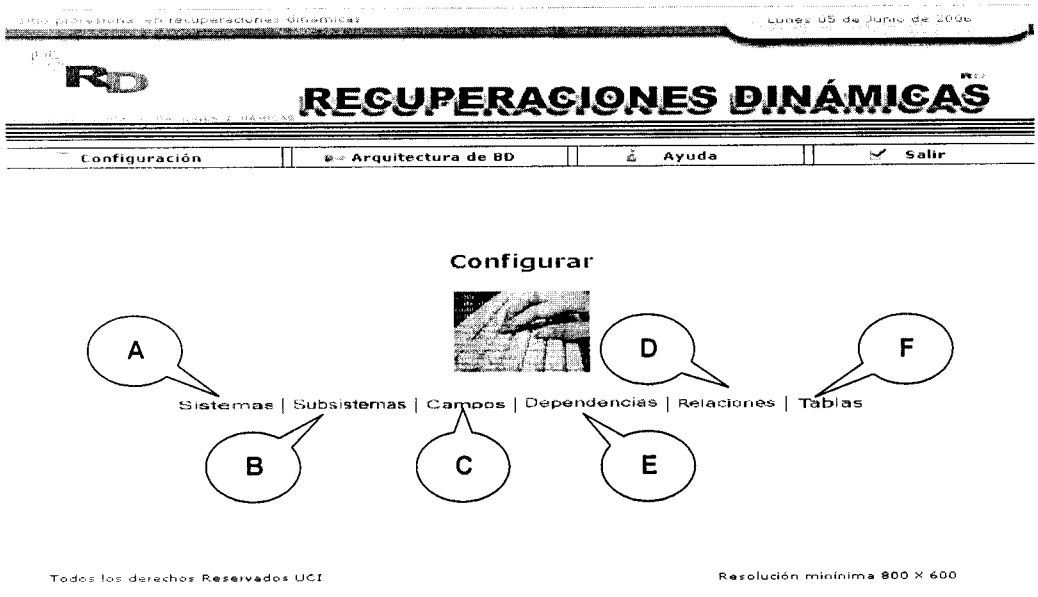
3.7 Expansión de los Casos de Uso

Mediante los casos de uso expandidos se describe paso a paso la secuencia de eventos que los actores utilizan para completar un proceso a través del sistema. Este sería el último paso en el análisis para pasar a la construcción de la solución propuesta. En este caso se van a describir los casos de uso que representan a los módulos principales que se detallan en el Epígrafe 1.6.

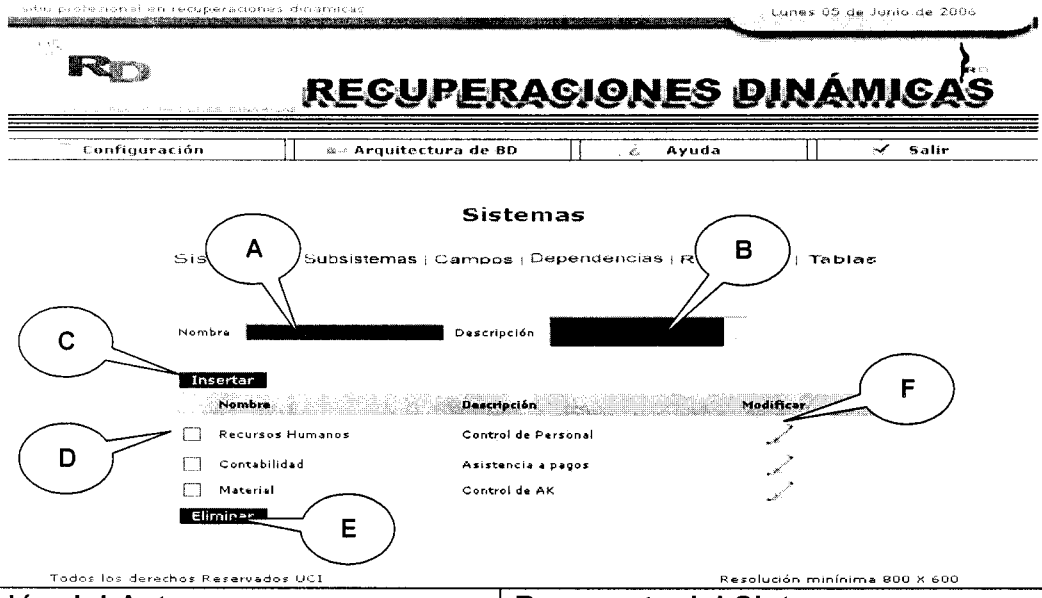
Caso de Uso:	**Actualizar Sistema**
Actor(es):	Configurador (inicia)
Propósito:	Definir sistema con el que trabajará el recuperador.
Resumen:	El caso de uso inicia cuando el configurador requiere actualizar la configuración del sistema, de acuerdo a su requerimiento puede redefinir, insertar o eliminar. El caso de uso termina cuando el configurador cierra el sistema o accede a otras opciones de la aplicación.
Referencias:	R1
Precondiciones:	Debe haber sido autenticado como configurador. En caso de que quiera eliminar o modificar un sistema, este debe estar previamente almacenado en la base de datos.

Curso normal de eventos para el caso de uso

Pantalla 1



Pantalla 2

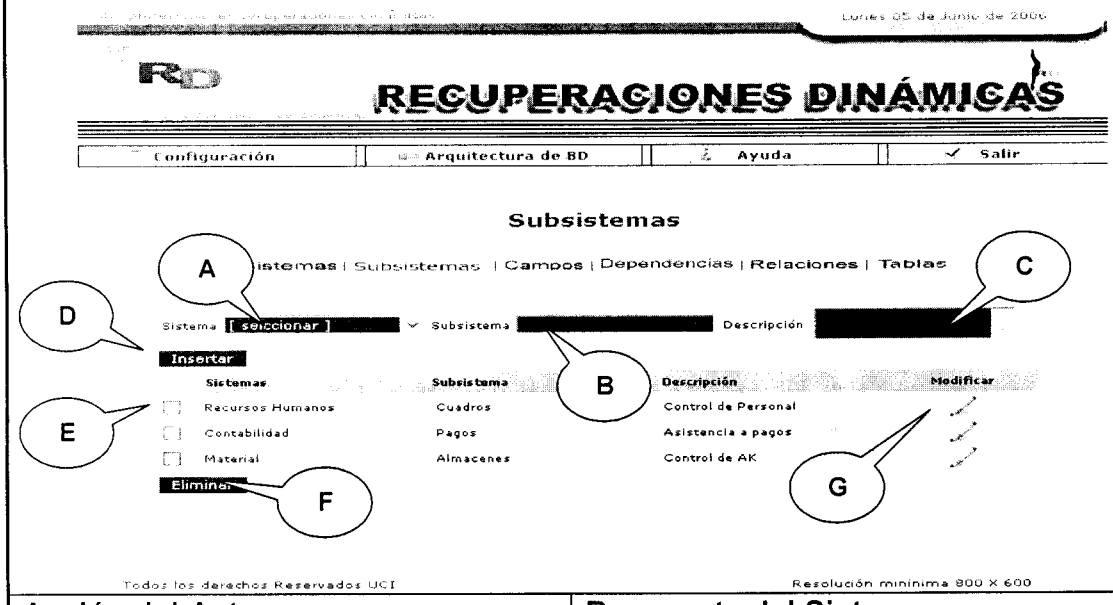


Acción del Actor	Respuesta del Sistema
1. Solicita configurar el sistema.(Pantalla 1-A) 3. Escribe el nombre del nuevo sistema (Pantalla 2 - A). 5. Escribe la descripción del sistema (Pantalla 2 - B). 4. Acciona el botón Insertar (Pantalla 2 - C).	2. El sistema muestra una interfaz que contiene todos los sistemas existentes y sus opciones (Pantalla 2). 5. Se inserta un nuevo sistema en la base de datos y se muestra en la interfaz.
Evento: "Eliminar sistema"	
1. Desea eliminar sistemas existentes. 2. Selecciona los sistemas que desea eliminar (Pantalla 2 - D). 3. Acciona el botón Eliminar(Pantalla 2 - E) 5.El actor responde afirmativamente.	4 El sistema pregunta al actor: "Está seguro que desea eliminar el(los) Sistema(s)" <Aceptar> <Cancelar> 6. Elimina el(los) sistema(s) de la base de datos 7. Muestra interfaz sin el sistema eliminado.
Evento: "Modificar sistema"	
1. Desea modificar sistemas existentes. 2. Selecciona el sistema que desea modificar (Pantalla 2 - F).	3. Se muestran los datos del sistema seleccionado (Pantalla 2 – A, B).

4. Modifica los datos del sistema. 5. Acciona el botón Insertar (Pantalla 2 – C) 7. El actor responde afirmativamente.	6. El sistema pregunta al actor: "Está seguro que desea modificar el Sistema" <Aceptar> <Cancelar> 8. Modifica el sistema en la base de datos. 9. Muestra sistema modificado en la interfaz.
Cursos Alternos	
En el curso normal de eventos en la Línea 5: El sistema muestra mensaje de error porque ya existe otro con el mismo nombre, "Imposible almacenar el nuevo sistema ". **Sección Eliminar** Línea 4: Si el actor responde negativamente, la aplicación no elimina. **Sección Modificar** Línea 6: Si ya existe un sistema con el mismo nombre, se mostrará un mensaje de error "Ya existe un sistema con ese nombre."	
Poscondiciones: La estructura de la base de datos de la aplicación se ha actualizado.	

Caso de Uso:	**Actualizar Subsistema**
Actor(es):	Configurador (inicia)
Propósito:	Definir subsistema con el que trabajará el recuperador.
Resumen:	El caso de uso inicia cuando el configurador requiere insertar, modificar o eliminar elementos en la configuración del subsistema. El caso de uso termina cuando el configurador cierra el sistema o accede a otras opciones de la aplicación.
Referencias:	R2
Precondiciones:	Debe haber sido autenticado como configurador. Para insertar un nuevo subsistema debe existir al menos un sistema. En caso de que quiera eliminar o modificar un subsistema, este debe estar previamente almacenado en la base de datos.
Curso normal de eventos para el caso de uso	

Pantalla 3



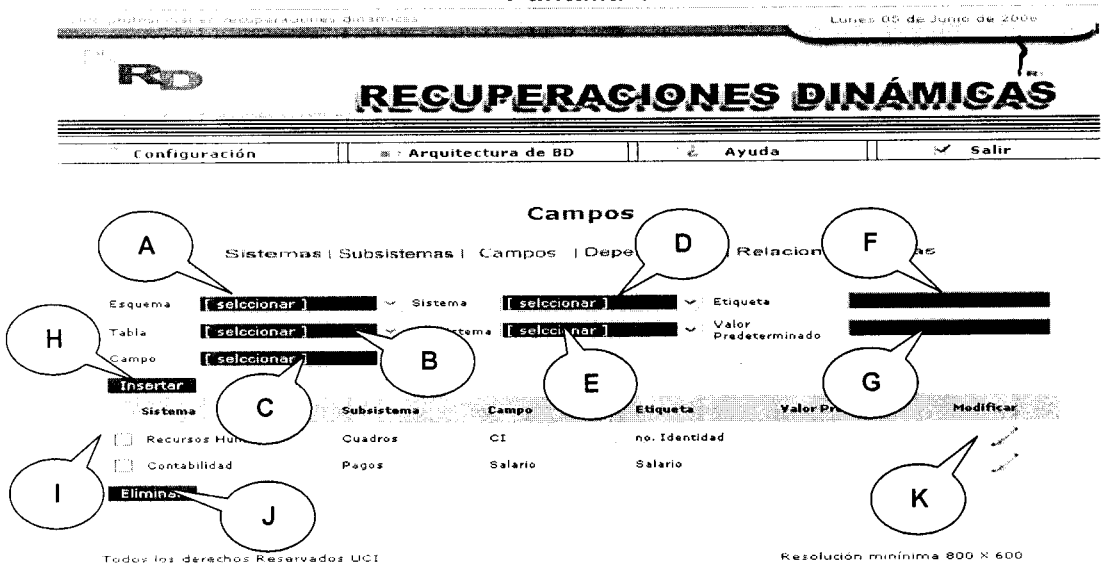
Acción del Actor	Respuesta del Sistema
1. Solicita configurar el subsistema.(Pantalla 1-B) 3. Selecciona el sistema al que pertenece el nuevo subsistema (Pantalla 3-A). 4. Introduce el nombre del nuevo subsistema. (Pantalla 3-B) 6. Inserta la descripción(Pantalla 3-C) 5. Acciona el botón Insertar (Pantalla 3 - D).	2. El sistema muestra una interfaz que contiene todos los sistemas existentes, sus subsistemas y sus opciones (Pantalla 3). 6. Se inserta un nuevo subsistema en la base de datos y se muestra en la interfaz.
Evento: "Eliminar Subsistema"	
1. Desea eliminar subsistemas existentes. 2. Selecciona los subsistemas que desea eliminar (Pantalla 3 - E). 3. Acciona el botón Eliminar(Pantalla 3 - F) 5. El actor responde afirmativamente.	4. El sistema pregunta al actor: "Está seguro que desea eliminar el(los) Subsistema(s)" <Aceptar> <Cancelar> 6. Elimina el(los) subsistema(s) de la base de datos. 7. Muestra interfaz sin el subsistema eliminado.

Evento: "Modificar subsistema"	
1. Desea modificar subsistemas existentes. 2. Selecciona el subsistema que desea modificar (Pantalla 3 - G). 4. Modifica los datos del subsistema seleccionado (Pantalla 3 – A, B, C). 5. Acciona el botón Insertar (Pantalla 3 – D) 7. Responde afirmativamente.	3. Se muestran los datos del subsistema seleccionado (Pantalla 3 - A). 6. El sistema pregunta al actor: "Está seguro que desea modificar el Subsistema" <Aceptar> <Cancelar> 8. Modifica el subsistema en la base de datos. 9. Muestra subsistema modificado en la interfaz.
Cursos Alternos	
En el curso normal de eventos **Línea 6:** Se muestra mensaje de error porque ya existe otro con el mismo nombre, "Imposible almacenar el nuevo subsistema ". **Sección Eliminar** **Línea 4:** Si el actor responde cancelar, el subsistema no se elimina. **Sección Modificar** **Línea 6:** Si el actor responde cancelar, el subsistema no se modifica. **Línea 8:** Si ya existe un subsistema con el mismo nombre, se mostrará un mensaje de error "Ya existe un subsistema con ese nombre."	
Poscondiciones: La estructura de la base de datos de la aplicación se ha actualizado.	

Caso de Uso:	**Actualizar Campos**
Actor(es):	Configurador (inicia)
Propósito:	Definir campos del subsistema con que trabajará el recuperador.
Resumen:	El caso de uso inicia cuando el configurador requiere realizar alguna modificación, inserción o eliminación de algún campo de los subsistemas que están determinados. El caso de uso termina cuando el configurador cierra el sistema o accede a otras opciones de la aplicación.
Referencias:	R3
Precondiciones:	Debe haber sido autenticado como configurador. Para insertar un nuevo campo debe existir al menos un sistema y al menos un subsistema. En caso de que quiera eliminar o modificar un campo, este debe estar previamente almacenado en la base de datos.

Curso normal de eventos para el caso de uso

Pantalla 4



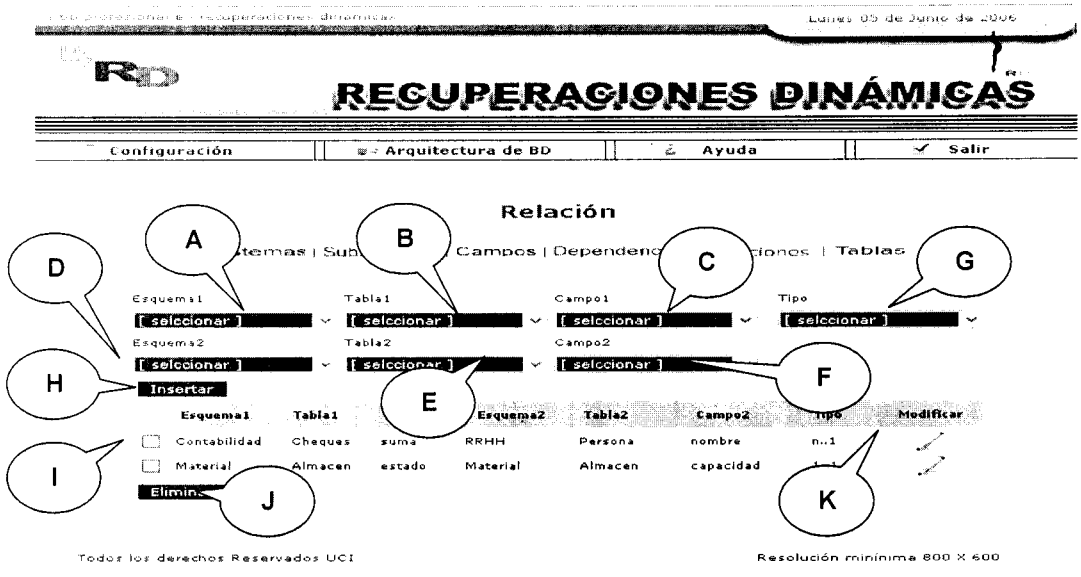
Acción del Actor	Respuesta del Sistema
1. Solicita configurar campos.(Pantalla 1-C)	2. El sistema muestra una interfaz que contiene todos los sistemas existentes, sus subsistemas y sus campos (Pantalla 4).
3. Selecciona el esquema al que pertenece la tabla donde se encuentra el campo a insertar (Pantalla 4-A).	4. Muestra todos los esquemas que existen en la base de datos.
5. Selecciona la tabla a la que pertenece el campo (Pantalla 4-B).	6. Muestra todos los campos de la tabla seleccionada.
7. Selecciona el campo a insertar (Pantalla 4-C).	
8. Selecciona el sistema donde se insertará el nuevo campo (Pantalla 4-D).	9. Muestra todos los subsistemas del sistema seleccionado.
10. Selecciona el subsistema donde se insertará el nuevo campo (Pantalla 4-E).	
11. Introduce los datos del nuevo subsistema (Pantalla 4 – F, G).	
12. Acciona el botón Insertar (Pantalla 4 -H).	13. Inserta el campo en la base de datos y se muestra en la interfaz.
Evento: "Eliminar campos"	
1. Desea eliminar campos de subsistemas existentes.	
2. Selecciona los campos que desea eliminar (Pantalla 4 - I).	
3. Acciona el botón Eliminar(Pantalla	4. El sistema pregunta al actor: "Está

4 - J) 5. El actor responde afirmativamente.	seguro que desea eliminar el(los) campo(s)" <Aceptar> <Cancelar> 6. Elimina el(los) campos(s) de la base de datos. 7. Muestra interfaz sin los campos eliminado.
Evento: "Modificar campos"	
1. Desea modificar algún campo existente. 2. Selecciona el campo a modificar (Pantalla 4 - K). 4. Modifica los datos del campo seleccionado. 5. Acciona el botón Insertar (Pantalla 4 - G). 7. Responde afirmativamente.	 3. Se muestra en Pantalla 4 – A, B, C, D, E, F los datos del campo seleccionado. 6. Pregunta al actor: "Está seguro que desea modificar el campo --------". <Aceptar> <Cancelar> 8. Modifica el campo en la base de datos. 9. Muestra en la interfaz el campo modificado.
Cursos Alternos	
En el curso normal de eventos **Línea 13:** Se muestra mensaje de error porque ya existe otro con la misma etiqueta, "Imposible almacenar el nuevo campo ". **Sección Eliminar** **Línea 5:** Si el actor responde cancelar, el campo no se elimina. **Sección Modificar** **Línea 7:** Si el actor responde cancelar, el campo no se modifica. **Línea 8:** Si ya existe un campo con la misma etiqueta, se mostrará un mensaje de error "Ya existe un campo con esa etiqueta".	
Poscondiciones: Los campos han sido actualizados.	

Caso de Uso:	**Generar Relaciones**
Actor(es):	Configurador (inicia).
Propósito:	Crear nueva relación con la que trabajará el recuperador.
Resumen:	El caso de uso inicia cuando el configurador desea crear una nueva relación, de acuerdo a su requerimiento puede redefinir, insertar o eliminar alguna relación que él haya creado con anterioridad. El caso de uso termina cuando el configurador cierra el sistema o accede a otras opciones de la aplicación.
Referencias:	R4
Precondiciones:	Debe haber sido autenticado como configurador. En caso de que quiera eliminar o modificar una relación, esta debe estar previamente almacenada en la base de datos y debe haber sido creada por él mismo.

Curso normal de eventos para el caso de uso

Pantalla 5

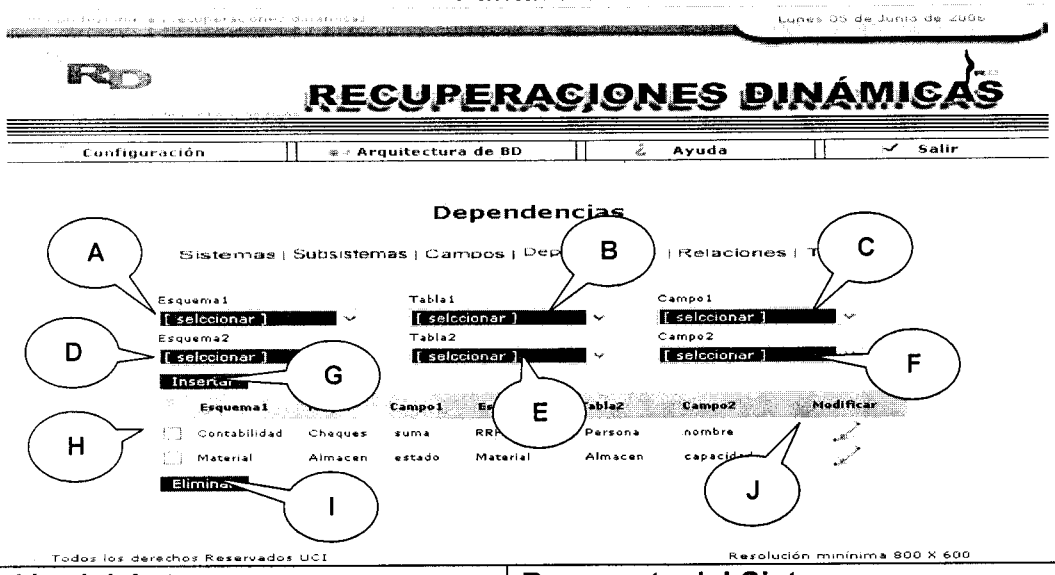


Acción del Actor	Respuesta del Sistema
1. Desea crear una nueva relación y accede la opción (Pantalla 1- D). 3. Selecciona el esquema, la tabla y el campo del primer miembro de la relación que desea crear (Pantalla 5 – A, B, C). 4. Selecciona el esquema, la tabla y el campo del segundo miembro de la relación (Pantalla 5- D, E, F). 5. Selecciona el tipo de relación que desea insertar (Pantalla 5-G). 6. Acciona el botón Insertar (Pantalla 5 - H).	2. Muestra la interfaz de Relaciones con las que existen en ese momento (Pantalla 5). 7. Inserta la nueva relación en la base de datos. 8. Muestra la relación creada en la lista de relaciones en la interfaz.
Evento: "Eliminar Relaciones"	
1. Desea eliminar alguna(s) relación(es). 2. Selecciona la(s) relación(es) que desea eliminar (Pantalla 5 - I). 3. Acciona el botón Eliminar (Pantalla 5 - J). 5. El actor responde afirmativamente.	4. El sistema pregunta al actor: "Está seguro que desea eliminar la(las) relación(es)" <Aceptar> <Cancelar> 6. Elimina la(s) relación(es) de la base de datos. 7. Muestra interfaz sin la(s) relación(es)

	eliminada(s).
Evento: "Modificar Relaciones"	
1. Desea modificar alguna relación existente. 2. Selecciona la relación a modificar (Pantalla 5-K). 4. Modifica los datos de la relación seleccionada. 5. Acciona el botón Insertar (Pantalla 5-H). 7. Responde afirmativamente.	3. Se muestra en Pantalla 5 – A, B, C, D, E, F los datos de la relación seleccionada. 6. Pregunta al actor: "Está seguro que desea modificar la relación". <Aceptar> <Cancelar> 8. Modifica la relación en la base de datos y la muestra. 9. Muestra en la interfaz la relación modificada.
Cursos Alternos	
En el curso normal de eventos en la **Línea 7:** El sistema muestra mensaje de error porque ya existe otra con el mismo nombre, "Imposible almacenar la nueva relación ". **Sección Eliminar** **Línea 6:** Si el actor responde negativamente, la aplicación no elimina **Sección Modificar** **Línea 8:** Si ya existe una dependencia como los mismos elementos, se mostrará un mensaje de error "Ya existe esa dependencia". **Línea 8:** Si el actor responde negativamente, la aplicación no modifica. .	
Poscondiciones: Las relaciones han sido actualizadas.	

Caso de Uso:	**Generar Dependencia**
Actor(es):	Configurador (inicia).
Propósito:	Crear nueva dependencia con el que trabajará el recuperador.
Resumen:	El caso de uso inicia cuando el configurador desea crear una nueve dependencia, de acuerdo a su requerimiento puede redefinir, insertar o eliminar alguna dependencia que el haya creado con anterioridad. El caso de uso termina cuando el configurador cierra el sistema o accede a otras opciones de la aplicación.
Referencias:	R5
Precondiciones:	Debe haber sido autenticado como configurador. En caso de que quiera eliminar o modificar una dependencia, esta debe estar previamente almacenada en la base de datos y debe haber sido creada por el mismo.
Curso normal de eventos para el caso de uso	

Pantalla 6

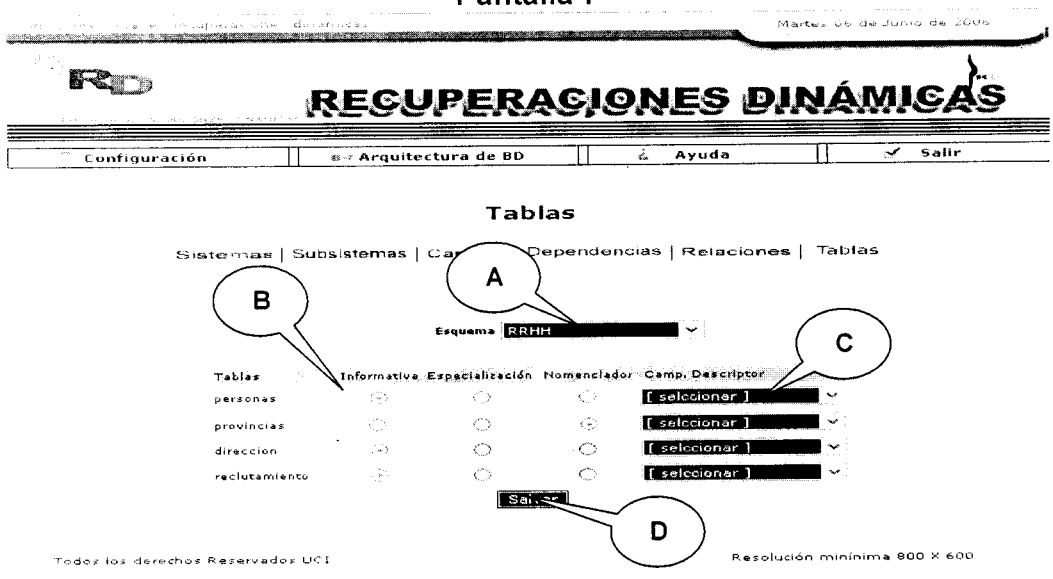


Acción del Actor	Respuesta del Sistema
1. Desea crear una nueva dependencia y accede la opción (Pantalla 1- E). 3. Selecciona el esquema, la tabla y el campo del primer miembro de la dependencia que desea crear (Pantalla 6–A, B, C). 4. Selecciona el esquema, la tabla y el campo del segundo miembro de la dependencia (Pantalla 6-D, E, F). 5. Acciona el botón Insertar (Pantalla 6-G).	2. Muestra la interfaz de Dependencias con las que existen en ese momento (Pantalla 6). 6. Inserta la nueva dependencia en la base de datos. 7. Muestra la dependencia creada en la lista de dependencias en la interfaz.
Evento: "Eliminar Dependencias"	
4. Desea eliminar alguna(s) dependencia(s). 5. Selecciona la(s) dependencia(s) que desea eliminar (Pantalla 6-H). 6. Acciona el botón Eliminar (Pantalla 6-I). 5. El actor responde afirmativamente.	4. El sistema pregunta al actor: "Está seguro que desea eliminar la(las) dependencia(s)" <Aceptar> <Cancelar> 6. Elimina la (s) dependencia(s) de la base de datos. 7. Muestra interfaz sin la(s) dependencias eliminada(s).
Evento: "Modificar Dependencias"	
1. Desea modificar alguna	

dependencia existente. 2. Selecciona la dependencia a modificar (Pantalla 6-J). 4. Modifica los datos de la dependencia seleccionada. 5. Acciona el botón Insertar (Pantalla 6-G). 7. Responde afirmativamente.	3. Se muestra en Pantalla 6 – A, B, C, D, E, F los datos de la dependencia seleccionada. 6. Pregunta al actor: "Está seguro que desea modificar la dependencia". <Aceptar> <Cancelar> 8. Modifica la dependencia en la base de datos. 9. Muestra en la interfaz la dependencia modificada.
Cursos Alternos	
En el curso normal de eventos en la **Línea 6:** El sistema muestra mensaje de error porque ya existe otra con el mismo nombre, "Imposible almacenar la nueva dependencia ". **Sección Eliminar** **Línea 6:** Si el actor responde negativamente, la aplicación no elimina **Sección Modificar** **Línea 6:** Si ya existe una dependencia como los mismos elementos, se mostrará un mensaje de error "Ya existe esa dependencia". **Línea 8:** Si el actor responde negativamente, la aplicación no modifica .	
Poscondiciones: Las dependencias han sido actualizadas.	

Caso de Uso:	**Clasificar Tablas**
Actor(es):	Configurador (inicia)
Propósito:	Definir de qué tipo son las tablas que componen los campos de los subsistemas creados.
Resumen:	El caso de uso inicia cuando el configurador desea definir las diferentes tablas que existen en los diferentes esquemas de la base de datos y que son las que contienen los campos existentes en el subsistema creado. El caso de uso termina cuando el configurador cierra el sistema o accede a otras opciones de la aplicación.
Referencias:	R6
Precondiciones:	Debe haber sido autenticado como configurador. Debe existir al menos un esquema.
Curso normal de eventos para el caso de uso	

Pantalla 7

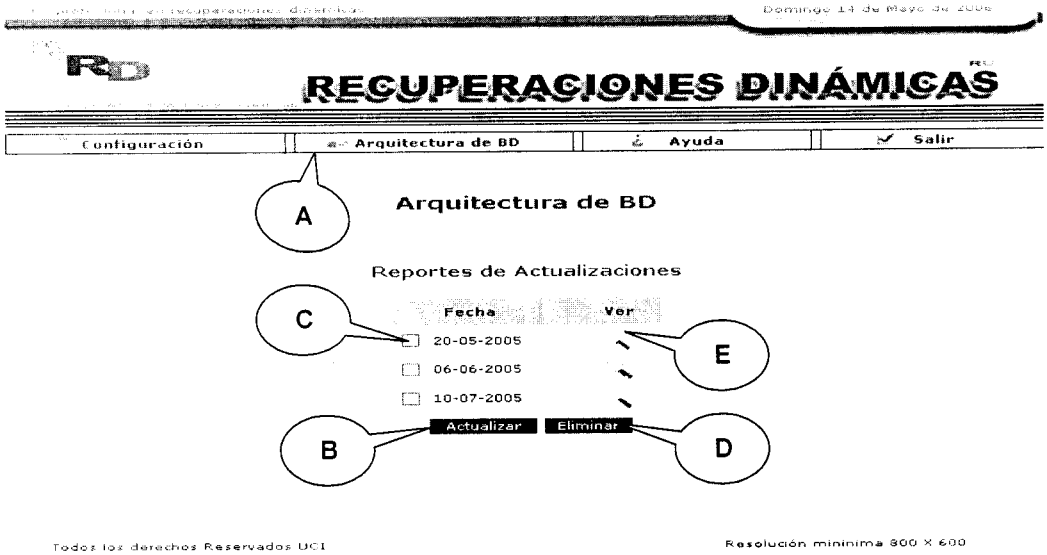


Acción del Actor	Respuesta del Sistema
1. Accede a la opción Tablas(Pantalla 1-F)	2. El sistema muestra una interfaz donde se pueden definir las tablas de los esquemas existentes en la base de datos (Pantalla 7).
3. Selecciona el esquema al que pertenecen los campos que agrego a un subsistema determinado (Pantalla 7-A). 5. Selecciona de que tipo es la tabla, si informativa, nomenclador o especificación (Pantalla 7-B). 6. Selecciona el campo descriptor por el que será identificad la tabla (Pantalla 7-C).	4. Muestra todas las tablas por la que esta compuesto el esquema seleccionado.
7. Acciona el botón salvar (Pantalla 7-D).	8. Se almacenan las tablas con su clasificación definida.
Cursos Alternos	
En el curso normal de eventos **Línea 8:** Se muestra mensaje de error porque no se han clasificados todas las tablas, "Imposible almacenar las tablas definidas ".	
Poscondiciones: La estructura de la base de datos de la aplicación se ha actualizado.	

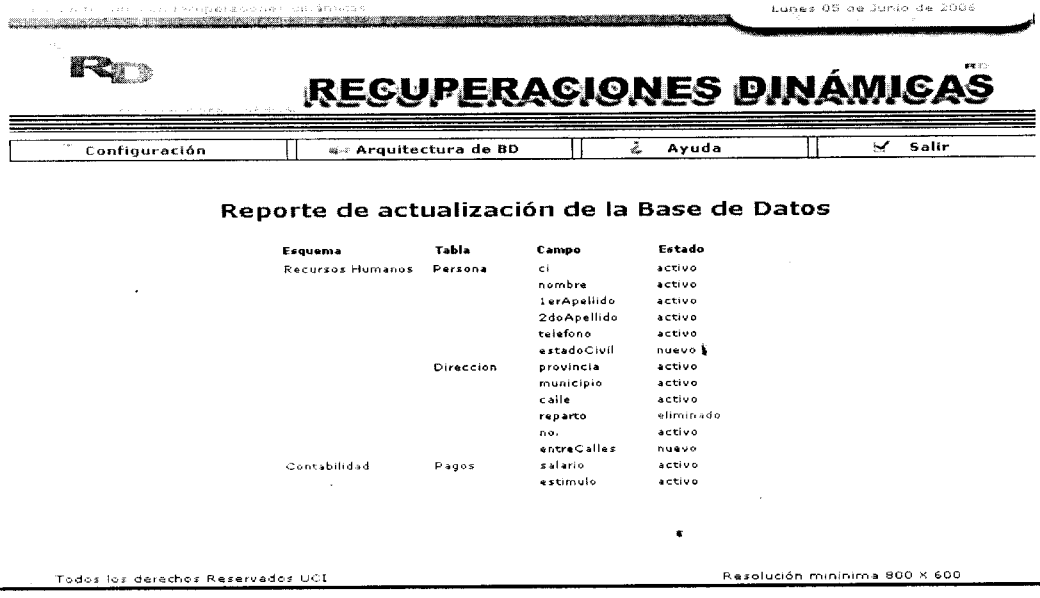
Caso de Uso:	**Actualizar Arquitectura de Base de Datos**
Actor(es):	Configurador (inicia)
Propósito:	Definir la estructura de la aplicación con respecto a la Base de

	Datos original.
Resumen:	El caso de uso inicia cuando el configurador requiere actualizar la arquitectura de la base de datos, de acuerdo a su requerimiento puede visualizar reportes de actualizaciones ya existentes. El caso de uso termina cuando el configurador cierra el sistema o accede a otras opciones de la aplicación.
Referencias:	R7
Precondiciones:	Debe haber sido autenticado como configurador. Debe existir la base de datos a la cual se conectará nuestra aplicación y actualizará la arquitectura.
Curso normal de eventos para el caso de uso	

Pantalla 8



Pantalla 9

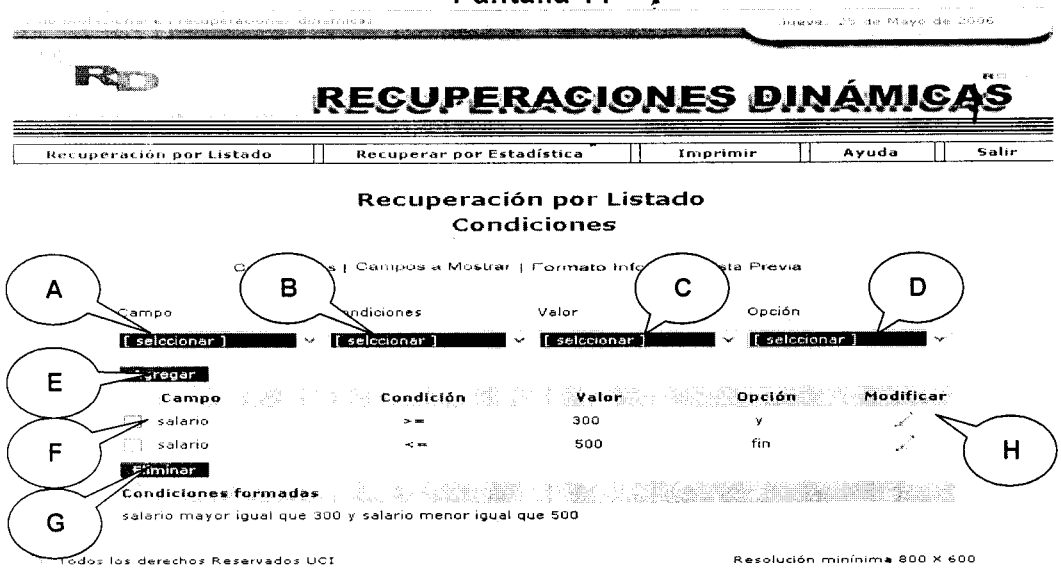


Acción del Actor	Respuesta del Sistema
1. Solicita actualizar arquitectura de la base de datos (Pantalla 8-A).	2. El sistema muestra la interfaz de Actualizar arquitectura de la base de datos (Pantalla 8).
	3. En caso de que se haya actualizado el sistema con anterioridad se mostraran las actualizaciones anteriores y sus diferentes opciones, si es la primera vez que se actualiza la aplicación no existen reportes y por lo tanto no se muestran las actualizaciones.
4. El configurador escoge la opción de realizar nueva actualización (Pantalla 8-B).	
	5. El sistema se conecta a la base de datos y realiza una copia de su estructura.
	6. Va realizando una comparación con la estructura que existe actualmente y creando un reporte dinámicamente.
	7. Luego compara tabla por tabla la cantidad de campos que existen y si han sido modificados; en caso afirmativo los modifica o elimina de los subsistemas donde se encontraban configurados.
	8. Compara las relaciones entre tablas y las dependencias funcionales y actualiza.
	9. Muestra el reporte con todos los esquemas, las tablas, los campos, las relaciones y las dependencias, así como el estado en que se encuentran, que pueden ser nuevo, eliminado, modificado o activo (Pantalla 9).
10. Cierra el reporte.	11. Vuelve a la pantalla de Actualizar arquitectura de la base de datos y se incluye la nueva actualización.
13. Desea visualizar reportes anteriores.	
14. Selecciona el reporte que desea consultar (Pantalla 8-E).	15. Muestra reporte seleccionado.
16. Desea eliminar reportes.	
17. Selecciona los reportes que desea eliminar (Pantalla 7-C).	19. Elimina el(los) reportes de las actualizaciones anteriores.
20. Acciona el botón Eliminar (Pantalla 8-D).	21. Muestra interfaz sin los reportes eliminados.
Cursos Alternos	
Línea 5: No se ha actualizado nunca la base de datos y se copia completamente la estructura sin realizar la comparación.	
Poscondiciones: La estructura de la base de datos de la aplicación ha quedado actualizada.	

Caso de Uso:	**Definir Condiciones**
Actor(es):	Recuperador (inicia)
Propósito:	Definir las condiciones que cumplirán los campos de la recuperación que se va a realizar.
Resumen:	El caso de uso inicia cuando el recuperador desea definir las condiciones que tendrá su recuperación. El caso de uso finaliza cuando el recuperador termina la definición de condiciones y accede a otras opciones de la aplicación.
Referencias:	R8
Precondiciones:	Debe haber sido autenticado como recuperador. Debe existir al menos un campo para definir las condiciones.

Pantalla 10

Pantalla 11

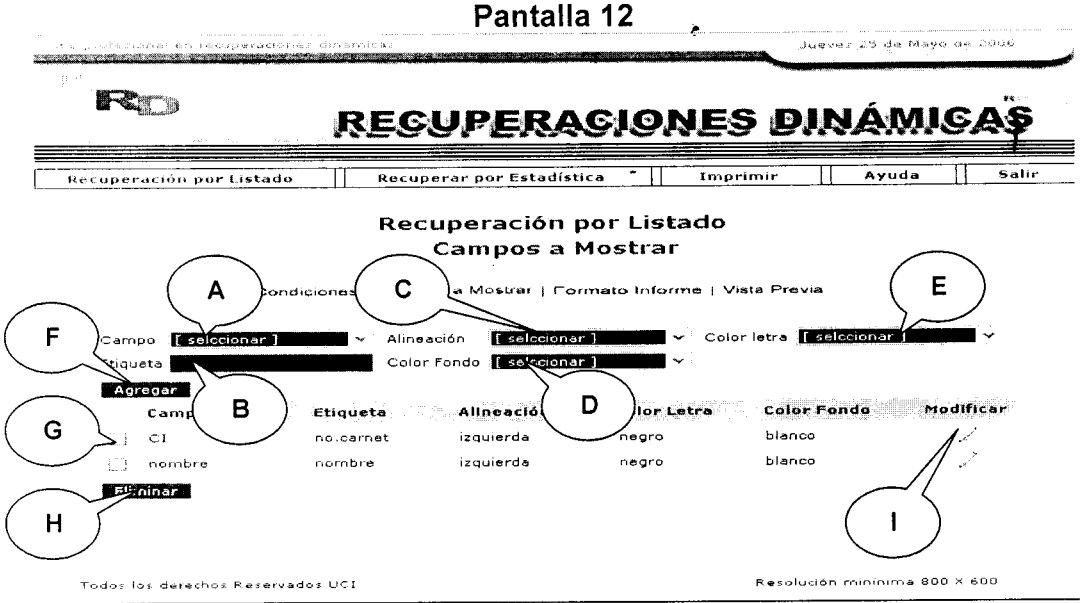


Acción del Actor	Respuesta del Sistema
1. Solicita configurar recuperación por listado (Pantalla 10-A). 3. Selecciona el sistema y subsistema con el que va a trabajar (Pantalla 10-F, G). 4. Solicita definir condiciones de la recuperación (Pantalla 10-B). 6. Escoge el campo al cual le aplicará una determinada condición (Pantalla 11-A). 7. Selecciona la condición que se aplicará (Pantalla 11-B). 8. Inserta el valor que tendrá esa condición (Pantalla 11-C). 9. Escoge la opción que realizará (Pantalla 11-D). 10. Acciona el botón Agregar (Pantalla 11-E).	2. El sistema muestra la interfaz de recuperación por listado (Pantalla 10). 5. Se visualiza la interfaz de condiciones de la recuperación (Pantalla 11). 11. El sistema va mostrando las condiciones que se van creando en la parte inferior de la interfaz en caso contrario el sistema conforma la consulta a la base de datos y la almacena.
Evento "Eliminar Condición"	
1. Desea eliminar una o varias condiciones que ha creado. 2. Escoge la(s) condición(es) que desea eliminar (Pantalla 11-F). 3. Acciona el botón Eliminar (Pantalla 11-G).	4. Muestra mensaje diciendo al usuario si esta seguro que desea eliminar la

	selección. 5. Elimina la(s) condición(es) que seleccionó y actualiza la interfaz sin la(s) condición(es) eliminada(s).
Evento "Modificar Condición"	
1. Desea modificar una condición que creó. 2. Selecciona la condición que desea modificar (Pantalla 11-H). 4. Modifica la condición. 5. Acciona el botón Agregar (Pantalla 11-E).	3. Se muestran los requisitos de la condición a modificar (Pantalla 11–A, B, C, D). 6. Muestra en la interfaz la condición modificada y la almacena.
Cursos Alternos	
Curso normal de eventos **Línea 10:** El sistema muestra mensaje de error porque la condición ya ha sido creada. **Evento "Eliminar Condición"** **Línea 5:** El usuario responde negativamente y el sistema no elimina. **Evento "Modificar Condición"** **Línea 6:** El sistema muestra mensaje de error porque la condición ya ha sido creada	
Poscondiciones: Las condiciones que cumplirán los campos del subsistema han sido creadas.	

Caso de Uso:	**Definir Campos a Mostrar**
Actor(es):	Recuperador (inicia)
Propósito:	Definir los campos que mostrara la recuperación que va a realizar.
Resumen:	El caso de uso inicia cuando el recuperador requiere definir los campos que tendrá su recuperación. El caso de uso termina cuando el recuperador termina y accede a otras opciones de la aplicación.
Referencias:	R9
Precondiciones:	Debe haber sido autenticado como recuperador. Deben existir al menos un campo.

Pantalla 12



Acción del Actor	Respuesta del Sistema
1. Solicita definir campos a mostrar en la recuperación (Pantalla 10-C). 3. Escoge el campo que quiere mostrar en la recuperación (Pantalla 12-A). 4. Inserta la etiqueta con que nombrará al campo (Pantalla 12-B). 6. Selecciona la alineación que tendrá el campo en el informe de la recuperación (Pantalla 12-C). 7. Selecciona el color de fondo y el color de la letra (Pantalla 12–D, E). 8. Acciona el botón Agregar Pantalla 12-F).	2. Se visualiza la interfaz de campos de la recuperación (Pantalla 12). 9. Se muestran los campos seleccionados en la parte inferior de la interfaz.
Evento "Eliminar Campo"	
1. Desea eliminar uno o varios campos de su recuperación. 2. Escoge el(los) campo(s) que desea eliminar (Pantalla 12-G). 3. Acciona el botón Eliminar (Pantalla 12-H).	4. Pregunta si desea eliminar la consulta. 5. Elimina el(los) campos(s) que seleccionó y actualiza la interfaz sin el(los) campo(s) eliminado(s).
Evento "Modificar Campo"	
1. Desea modificar un campo de su recuperación. 2. Selecciona el campo que desea modificar (Pantalla 12-I). 4. Modifica los datos.	3. Se muestran los datos del campo a modificar (Pantalla 12–A, B, C, D, E).

5. Acciona el botón Agregar (Pantalla12-F).	6. Muestra en la interfaz el campo modificado y almaceno la modificación.
Cursos Alternos	
En el curso normal de eventos **Línea 9:** El sistema muestra mensaje de error porque ya existe otra condición igual u otro campo con la misma etiqueta. **Evento "Eliminar Condición"** **Línea 5:** Si el actor responde negativamente, la aplicación no elimina. **Evento "Modificar Condición"** **Línea 6:** El sistema muestra mensaje de error porque ya existe otra condición igual u otro campo con igual etiqueta.	
Poscondiciones: Los campos que se mostrarán en la recuperación han sido seleccionados.	

Caso de Uso:	**Definir Formato del Informe**
Actor(es):	Recuperador (inicia)
Propósito:	Definir el formato que tendrá el informe y que se mostrará en la recuperación.
Resumen:	El caso de uso inicia cuando el recuperador requiere definir el formato del informe. Tiene la opción de seleccionar el formato del título y del cuerpo del informe que va a utilizar. También puede seleccionar si quiere ordenar o agrupar el informe por un campo determinado. El caso de uso termina cuando el recuperador cierra el sistema o accede a otras opciones de la aplicación.
Referencias:	R10
Precondiciones:	Debe haber sido autenticado como recuperador.
Curso normal de eventos para el caso de uso	

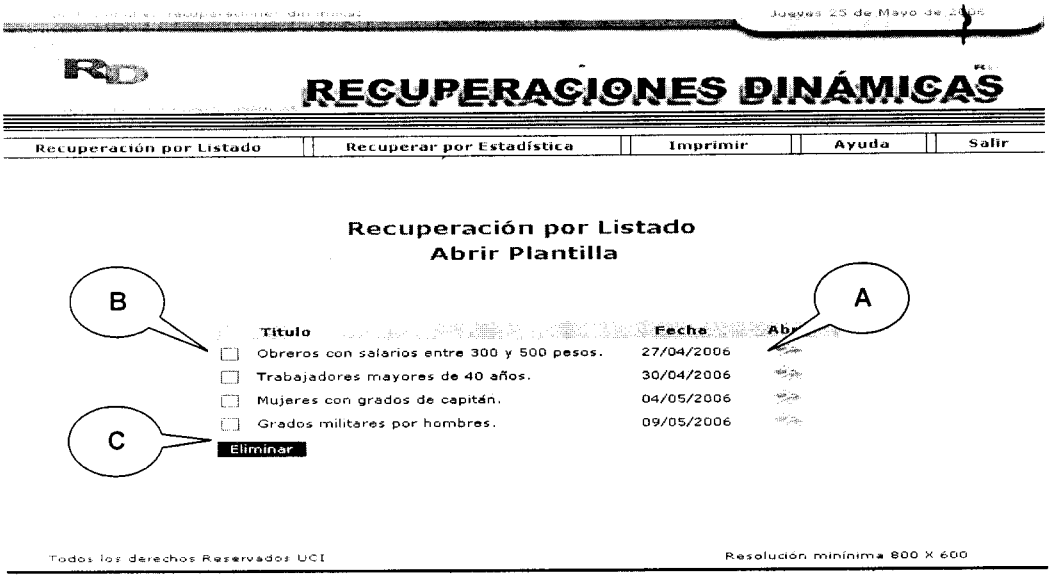
Pantalla 13

Acción del Actor	**Respuesta del Sistema**
1. Solicita definir el formato del informe para su recuperación (Pantalla 10-D). 3. Inserta el título que quiere mostrar en el informe de la recuperación (Pantalla 13-A). 4. Selecciona la fuente, el color de fondo, el color de letra y la alineación que desea utilizar para el título del informe (Pantalla 13–B, C, D, E). 5. Selecciona la fuente, el color de fondo, el color de letra y la alineación que desea utilizar para el cuerpo del informe (Pantalla 13–F, G, H, I). 6. Selecciona el campo por el cual quiere ordenar el informe y su orden (Pantalla 13–J, K). 8. Selecciona el campo por el cual quiere agrupar el informe (Pantalla 13–L).	2. Se visualiza la interfaz de Formato Informe (Pantalla 13). 7. Muestra los campos que se seleccionaron en la interfaz Campos a Mostrar (Pantalla 13- J). 9. Muestra los campos que se seleccionaron en la interfaz Campos a Mostrar (Pantalla 13- J).
Cursos Alternos	
En caso de que no se defina ningún formato, se utilizará uno predeterminado.	
Poscondiciones: Se ha creado el informe deseado.	

Caso de Uso:	**Generar Plantilla**
Actor(es):	Recuperador (inicia)
Propósito:	Abrir como plantilla los procesos de recuperación de información que se hayan realizado y salvado con anterioridad.
Resumen:	El caso de uso comienza cuando el recuperador desea abrir algunos de los procesos de recuperación de información que se hayan realizado con anterioridad. Luego éste selecciona de la interfaz correspondiente, la plantilla que desea abrir. El caso de uso finaliza cuando se actualizan las diferentes pantallas que intervienen en la plantilla cuyos datos se encuentran almacenados en esta.
Referencias:	R11
Precondiciones:	Debe estar autenticado como recuperador. Debe existir al menos una plantilla. Para salvar un informe por listado debe definir al menos un campo y en el caso de estadística un campo por columna y al menos uno por listado

Curso normal de eventos para el caso de uso(Abrir Plantilla)

Pantalla 14



Acción del Actor	Respuesta del Sistema
1. Desea abrir una plantilla de las recuperaciones anteriores que se hayan salvado (Pantalla 10-I). 3. Selecciona la plantilla que va a abrir (Pantalla 14-A).	2. Muestra la interfaz de abrir plantilla con el título y la fecha en la que fue realizada (Pantalla 14). 4. Realiza actualizaciones de datos en las pantallas 9, 10 y 11 en el caso que la plantilla sea de recuperación por listado, y, pantalla 13, 14, 15 y 16 en caso que se por estadísticas haciendo corresponder cada componente de la pagina con sus datos relacionados. 5. Muestra la pantalla 8 si la plantilla fue almacenada por listado y pantalla 12 si fue almacenada por estadísticas.
Evento "Eliminar Plantilla"	
1. Desea eliminar una(s) de las plantilla(s) existentes y que se muestra en la pantalla 14. 2. Selecciona la(s) plantilla(s) que desea eliminar (Pantalla 14-B). 3. Acciona el botón Eliminar (Pantalla 14-C) 5.En caso afirmativo	4. Pregunta si esta seguro que desea eliminar la(s) plantilla(s) seleccionadas. 6. Elimina la(s) plantilla(s) y muestra la lista de plantillas si la(s) eliminada(s) (Pantalla 14).

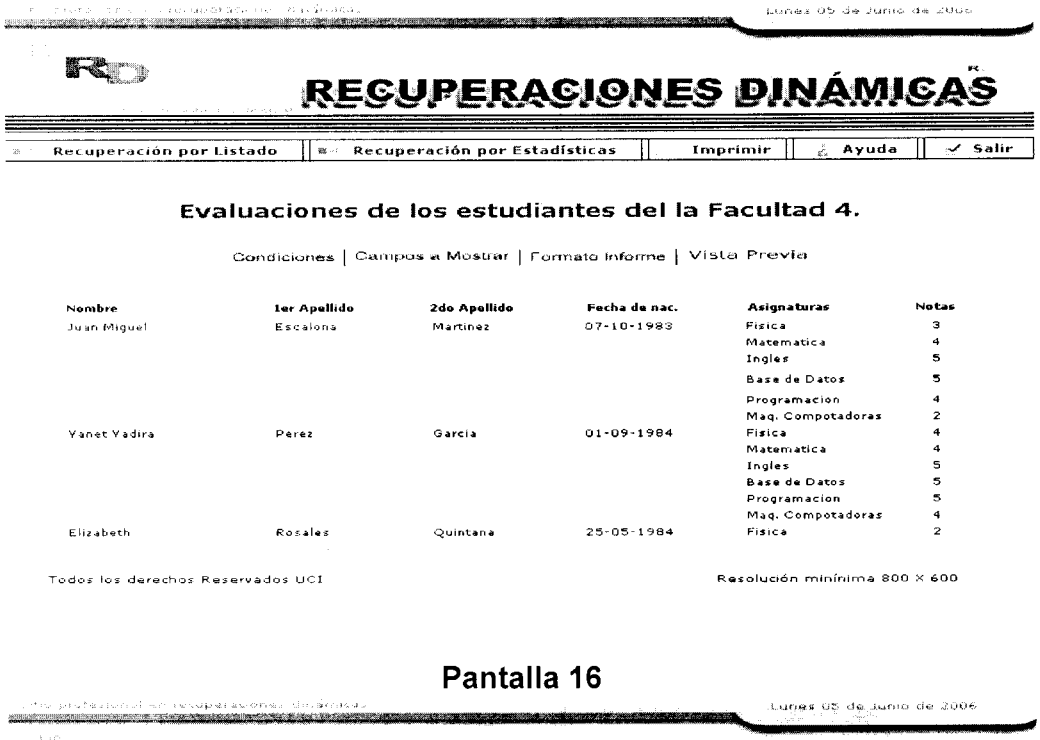
Evento "Salvar Plantilla"	
1. Desea salvar como plantilla las definiciones hechas en las diferentes pantallas de recuperación, ya se por listado o por estadísticas (Pantalla 10-J).	2. Guarda las diferentes definiciones hechas en las tablas correspondientes de la base de datos. Si es por listado guarda la configuración de: -Campos a mostrar en la tabla de la base de datos Campos por Listados. -Condiciones en la tabla Condiciones. -Formato del informe en las tablas Titulo, Informe, Grupo y Orden. Si es por estadísticas se guarda la configuración de: -Condiciones en la tabla de la base de datos Condiciones. -Campo de encabezado en la tabla Campos por estadísticas. -Campos por filas en la tabla Campos por estadísticas. -Campos por columnas en la tabla Campos por estadísticas. -Formato del informe en las tablas Título, Informe, Grupo y Orden. 3. Muestra nuevamente la pantalla 10 si es por listado y pantalla 17 si la salva se hizo por estadísticas.
Evento "Nuevo Informe"	
1. Desea comenzar de nuevo las operaciones en vista a una nueva recuperación (Pantalla 10-H).	2. Limpia todas las diferentes pantallas de definición de parámetros que fueron establecidos en la recuperación anterior.
Cursos Alternos	
Evento "Eliminar Plantilla": **Línea 4:** Si el recuperador responde negativamente, no se elimina ninguna plantilla.	
Poscondiciones: Se han realizado las operaciones correspondientes a la plantilla.	

Caso de Uso:	**Visualizar Reporte**
Actor(es):	Recuperador (inicia).
Propósito:	Visualizar el informe de la recuperación.
Resumen:	El caso de uso inicia cuando el recuperador desea visualizar el reporte de la recuperación que define anteriormente, de acuerdo a su requerimiento puede ser un Informe por Listado o por Estadística. El caso de uso termina cuando el recuperador cierra el sistema o accede a otras opciones de la aplicación.

Referencias:	R12
Precondiciones:	Debe haber sido autenticado como configurador. Para visualizar un informe por listado debe definir al menos un campo y en el caso de estadística un campo por columna y al menos uno por listado

Curso normal de eventos para el caso de uso

Pantalla 15



Pantalla 16

Acción del Actor	**Respuesta del Sistema**
1. Desea visualizar el reporte de la recuperación ya sea por listado o por	

estadística. 2. Selecciona la opción de vista previa (Pantalla 10-E).	3. En caso de que estemos en Recuperación por Listado el sistema llamará a la función CrearReportePorListado y si se encuentra en Recuperación por Estadística llama a la función CrearReportePorEstadistica. Ambas funciones realizan la recuperación a la base de datos con las opciones que seleccionaron en las pantallas anteriores y muestran el resultado en la interfaz correspondiente. Si es un informe por listado (Pantalla 15) y un informe por estadística (Pantalla -16).
Poscondiciones: El reporte de la recuperación ha sido visualizado.	

Caso de Uso:	**Definir Encabezados**
Actor(es):	Recuperador (inicia)
Propósito:	Definir los encabezados que aparecerán en la tabla de la recuperación que se va a realizar.
Resumen:	El caso de uso inicia cuando el recuperador accede a definir los encabezados que aparecerán en la tabla que tendrá su recuperación, de acuerdo a su requerimiento puede definirlos o no. El caso de uso termina cuando el recuperador termina y accede a otras opciones de la aplicación.
Referencias:	R13
Precondiciones:	Debe haber sido autenticado como recuperador. Debe existir el campo a seleccionar.

Pantalla 17

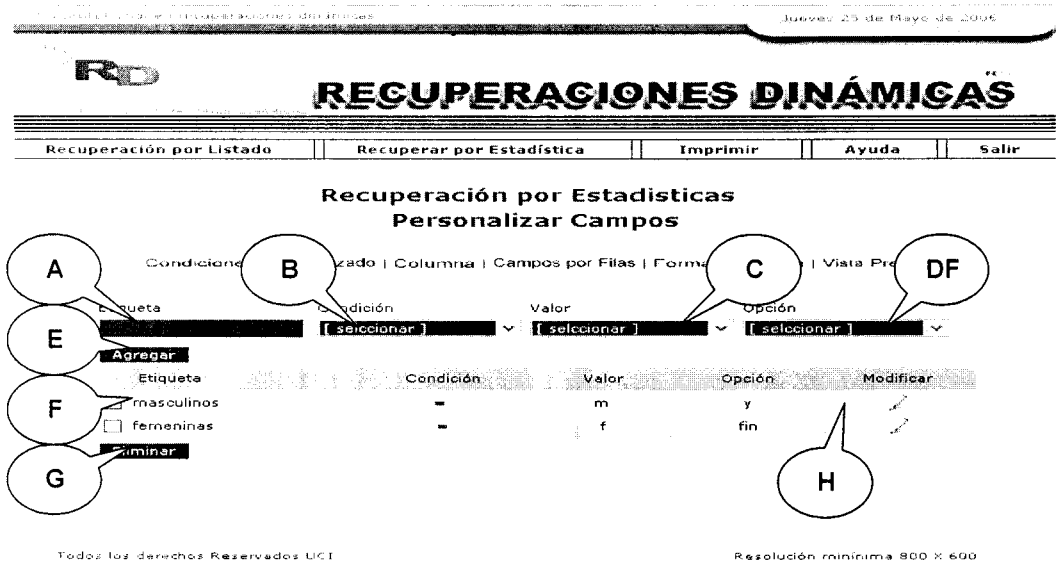
Pantalla 18

Acción del Actor	Respuesta del Sistema
1. Solicita configurar recuperación por estadísticas (Pantalla 17-A). 3. Define el sistema y subsistema con el que trabajará (Pantalla 17-E, F) 4. Solicita definir encabezados de la recuperación (Pantalla 17-B). 6. Selecciona el campo que aparecerá como encabezado en la recuperación (Pantalla 18-A). 7. Selecciona si desea visualizar los totales o no (Pantalla 18-B).	2. Muestra la interfaz de recuperación por estadísticas (Pantalla 17). 5. Se visualiza la interfaz de encabezados de la recuperación (Pantalla 18). 9. Almacena el campo seleccionado.
Cursos Alternos:	
Poscondiciones: El encabezados que tendrá la tabla que ha sido creados.	

Caso de Uso:	Personalizar
Actor(es):	Recuperador (inicia)
Propósito:	Definir las condiciones por campos que aparecerán en la recuperación.
Resumen:	El caso de uso inicia cuando el recuperador accede a definir las diferentes condiciones por campos que han sido seleccionados previamente por el recuperador. El caso de uso termina cuando el recuperador termina y accede a otras

	opciones de la aplicación.
Referencias:	R14
Precondiciones:	Debe haber sido autenticado como recuperador. Debe haber seleccionado un campo.

Pantalla 19

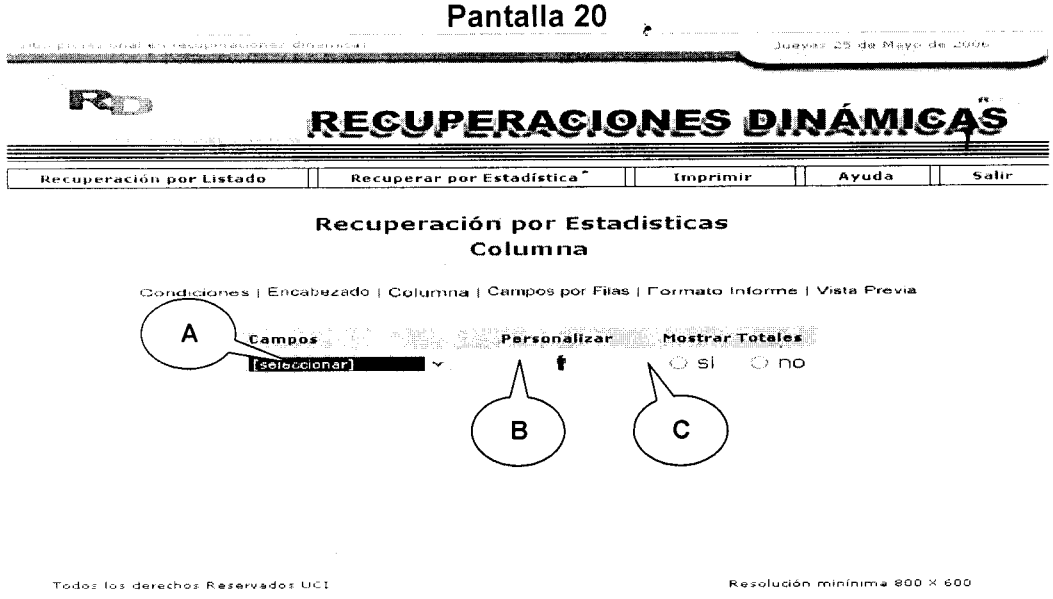


Evento "Agregar Condición"	
1. Desea personalizar un campo que seleccionó (Pantalla 18-C, 20-E, 21-B). 3. Inserta la etiqueta del campo seleccionado (Pantalla 19-A). 4. Selecciona la condición que utilizará (Pantalla 19-B). 5. Selecciona el valor y la opción (Pantalla 14-C, D). 6. Acciona el botón Agregar (Pantalla 19-E).	2. Muestra interfaz de personalizar (Pantalla 19). 7. Agrega y muestra la condición formada.
Evento "Eliminar Condición"	
1. Desea eliminar una o varias condiciones que ha creado. 2. Escoge la(s) condición(es) que desea eliminar (Pantalla 19-F). 3. Acciona el botón Eliminar (Pantalla 19-G).	4. Pregunta si esta seguro que desea eliminar las condicione(s) seleccionada(s). 5. Elimina la(s) condición(es) que seleccionó y actualiza la interfaz sin la(s) condición(es) eliminada(s).

Evento "Modificar Condición"	
1. Desea modificar una condición que creó. 2. Selecciona la condición que desea modificar (Pantalla 19-H). 4. Modifica los datos. 5. Acciona el botón Agregar (Pantalla 19-E).	3. Muestran los datos a modificar (Pantalla 19–A, B, C, D). 6. Agrega y muestra en la interfaz la condición modificada.
Cursos Alternos	
Evento "Agregar Condición" **Línea 7:** El sistema muestra mensaje de error porque ya existe otra condición igual, "Imposible agregar la nueva condición". **Evento "Eliminar Condición"** **Línea 5:** Si el actor responde negativamente, la aplicación no elimina. **Evento "Modificar Condición"** **Línea 6:** El sistema muestra mensaje de error porque ya existe otra condición igual, "Imposible agregar la nueva condición".	
Poscondiciones: El campo seleccionado ha sido personalizado.	

Caso de Uso:	**Definir columna**
Actor(es):	Recuperador (inicia)
Propósito:	Definir los campos que aparecerán por columnas en la tabla de la recuperación que se va a realizar.
Resumen:	El caso de uso inicia cuando el recuperador accede a definir el campo columna que aparecerán en la tabla que tendrá su recuperación, de acuerdo a su requerimiento. El caso de uso termina cuando el recuperador de definir los campos y accede a otras opciones de la aplicación.
Referencias:	R15
Precondiciones:	Debe haber sido autenticado como recuperador. Deben existir los campos que se van a seleccionar.

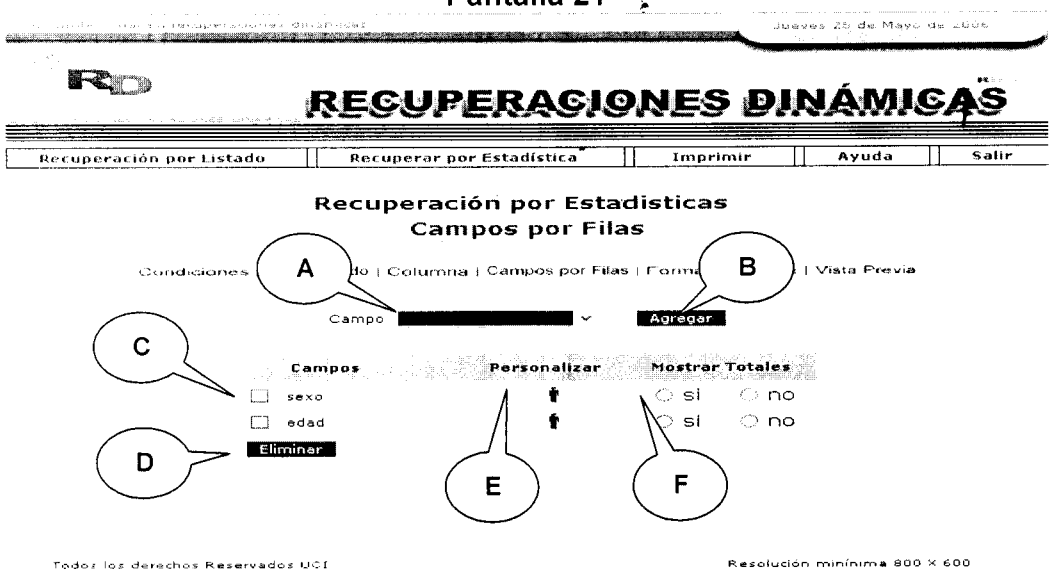
Pantalla 20



Acción del Actor	Respuesta del Sistema
1. Solicita definir campos por filas de la recuperación (Pantalla 17-C). 3. Escoge el campo que desea que aparezca (Pantalla 20-A). 4. Selecciona si se mostraran los totales (Pantalla 20-F).	2. Muestra la interfaz de campos columna (Pantalla 20).
Cursos Alternos:	
Poscondiciones: El campo columna ha sido definido.	

Caso de Uso:	**Definir campos por filas**
Actor(es):	Recuperador (inicia)
Propósito:	Definir los campos que aparecerán por filas en la tabla de la recuperación que se va a realizar.
Resumen:	El caso de uso inicia cuando el recuperador accede a definir los campos por filas que aparecerán en la tabla que tendrá su recuperación, de acuerdo a su requerimiento. El caso de uso termina cuando el recuperador de definir los campos y accede a otras opciones de la aplicación.
Referencias:	R16
Precondiciones:	Debe haber sido autenticado como recuperador. Deben existir los campos que se van a seleccionar.

Pantalla 21



Acción del Actor	Respuesta del Sistema
1. Solicita definir campos por filas de la recuperación (Pantalla 17-D). 3. Escoge el campo que desea que aparezca (Pantalla 21-A). 4. Acciona el botón Agregar el campo seleccionado (Pantalla 21-B). 6. Selecciona si se mostrarán los totales (Pantalla 21-F).	2. Muestra la interfaz de campos por filas (Pantalla 21). 5. Agrega el campo seleccionado y lo muestra.
Evento "Eliminar Campo"	
1. Desea eliminar uno o varios campos que ha seleccionado. 2. Escoge el(los) campo(s) que desea eliminar (Pantalla 21-C). 3. Acciona el botón Eliminar (Pantalla 21-D).	4. Pregunta si está seguro que desea eliminar el(los) campo(s) seleccionado(s). 5. Elimina el(los) campo(s) que seleccionó y actualiza.
Cursos Alternos	
En el curso normal de eventos **Línea 5:** El sistema muestra mensaje de error porque ya existe el campo seleccionado, "Imposible agregar el campo". **Evento "Eliminar Campo"** **Línea 5:** Si el actor responde negativamente, la aplicación no elimina.	
Poscondiciones: Los campos por filas ya están definidos.	

3.8 Conclusiones

En este capítulo se comenzó a desarrollar la propuesta de solución, obteniéndose a partir del análisis de los procesos del dominio, un listado con las funciones que debe tener el sistema, que se representaron mediante un Diagrama de Casos de Uso, y finalmente se describieron paso a paso todas las acciones de los actores del sistema con los casos de uso con los que interactúan. Con lo ya establecido se puede empezar con la construcción del sistema, tratando de que se cumplan todos los requerimientos y las funciones que han sido consideradas necesarias en este capítulo.

Capítulo

IV

CONSTRUCCION DE LA SOLUCION PRACTICA

4.1 Introducción

El modelo de análisis y diseño es muy importante en el desarrollo de software dirigido por modelos ya que constituye la vista lógica de la arquitectura del software. Es en este capítulo donde se ajusta el resultado del análisis a las tecnologías y lenguajes que serán utilizados, además de darle forma a la arquitectura del sistema. Para ello los componentes de la aplicación se tratan como clases, y utilizando las extensiones del UML, se pueden presentar a través de diagramas de clases Web.

4.2 Mecanismos de Diseño

Los mecanismos de diseño se modelan para comunicar mejor la manera en que debe darse solución a problemas recurrentes en la aplicación. Aunque su desarrollo y mantenimiento es opcional, se recomienda su uso en entornos de desarrollo complejos. Los mecanismos de diseño están más relacionados con la tecnología de implementación. Con su elaboración, se modelaría un conocimiento que ayudará tanto al desarrollo de la aplicación actual como a construcciones futuras, además de las labores de mantenimiento. En los mecanismos de diseño intervienen diversos elementos de la aplicación (clases, subsistemas, etcétera).

Un ejemplo típico de mecanismos son los patrones de diseño. En la actualidad son muchos los patrones de diseños utilizados en la construcción de una aplicación, el Facade (Fachada), Singletons (Instancia Única), Strategy (Estrategia), Factory (Factoría) y Abstract Factory (Factoría Abstracta) entre otros; en nuestro sistema usamos los conceptos de Factory y Abstract Factory.

El patrón factoría es uno de los varios patrones creadores definidos por la GoF (Gans of Four, Grupo de los Cuatro). La idea que se esconde detrás de este patrón es la de centralizar el sitio donde se crean los objetos, normalmente donde se crean objetos de una misma "familia", sin dar una definición clara de lo que nuestro software puede entender como familia, como podría ser componentes visuales, componentes de la lógica del negocio o objetos concurrentes en el tiempo.

La clase factoría devuelve una instancia de un objeto según los datos que se le pasan como parámetros. Para que la creación centralizada de objetos sea lo más "útil y eficaz" posible, es de esperar que todos los objetos creados desciendan de la misma clase o implementen la misma interfase (es decir, hagan una operación similar pero de distintas formas), así podemos usarlos todos de la misma manera, con los mismos métodos (gracias al polimorfismo), sin importarnos que clase concreta estamos tratando en cada momento.

El patrón Factoría Abstracta es muy sencillo si se ha entendido el patrón factoría. Como la palabra abstracta nos puede hacer suponer, este patrón lleva al de la factoría un punto más lejos en la idea de abstraer el código de creación de objetos del resto de la aplicación. ¿Cómo debemos entender esto?, pues una factoría abstracta es una clase factoría, pero que los objetos que devuelve son a su vez factorías. Por este motivo, para que sea efectiva, estas factorías que devuelve, deben ser de la misma familia (es decir, tener antecesores comunes), como ocurría con las factorías normales.

Los mecanismos se pueden subdividir para organizarlos mejor. Definir los artefactos que se necesitan para cada uno es una decisión del Arquitecto; se pueden construir mecanismos para cada servicio que se necesite dentro de la aplicación. En nuestro caso utilizamos uno para controlar la seguridad y otro para el acceso a datos concurrentes.

4.2.1 Mecanismo de diseño de Seguridad

La seguridad es un aspecto crítico de las aplicaciones Web. Las aplicaciones Web, por definición, permiten el acceso de usuarios a recursos centrales, el

servidor Web y, a través de éste, a otros como los servidores de base de datos. Con los conocimientos y la implementación correcta de medidas de seguridad, puede proteger sus propios recursos así como proporcionar un entorno seguro donde los usuarios trabajen cómodos con su aplicación.

El proceso administrativo centralizado de la seguridad en los sistemas es un elemento fundamental para un control riguroso de los accesos a las aplicaciones Web, dado que es mas fácil actuar ante cualquier violación.

Los sistemas realizan el control de la seguridad de manera semejante, usando un servicio Web encargado del control de los accesos, autenticación y registro de los eventos que ocurren, por lo cual se propone un mecanismo de diseño que sirva de manera general a todas las aplicaciones que usan dicho servicio, garantizando así los requerimientos necesarios para su correcto funcionamiento.

Básicamente los servicios Web permiten que diferentes aplicaciones, realizadas con diferentes tecnologías, y ejecutándose en toda una variedad de entornos, puedan comunicarse e integrarse, lo cual es muy importante.

Por lo explicado anteriormente se propone el siguiente mecanismo de diseño para seguridad basado en el uso de servicios Web.

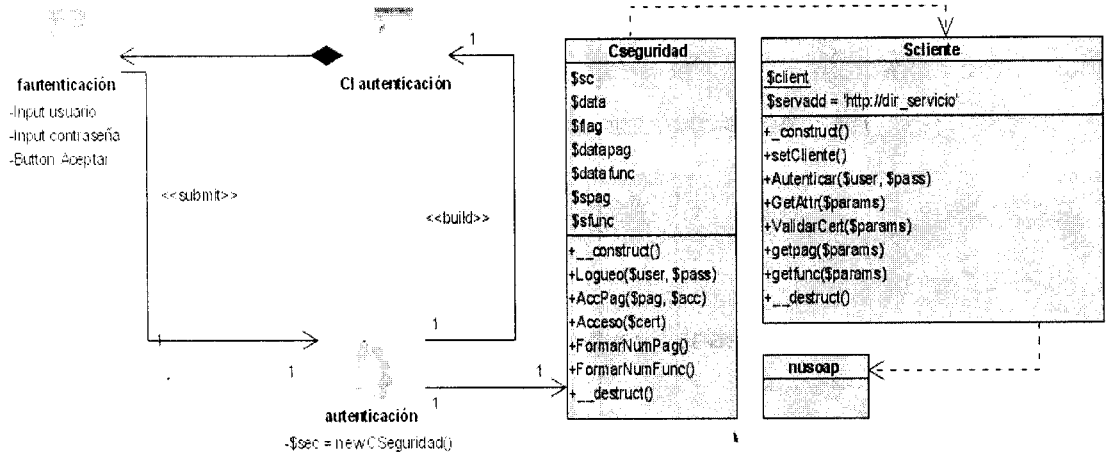


Figura 4.1 Diagrama de Clases del Mecanismo de Diseño de Seguridad.

El mecanismo anterior constituye la parte cliente del servicio, con la que contarán todas las aplicaciones y mediante la cual se hará el acceso al componente servidor del servicio Web. La clase Scliente es la encargada de la

comunicación con la parte servidora del servicio Web de seguridad. La clase Cseguridad es la intermediaria entre los sistemas y la clase Scliente, siendo transparente el servicio Web a los sistemas que lo usan. La clase nusoap incluye todas las clases necesarias para el funcionamiento del servicio en la parte cliente. Además de las clases principales se brinda la interfaz de autenticación, que es el elemento fundamental e inicio del mecanismo, así como la clase autenticación que es la que regula el proceso e instancia a la clase Cseguridad.

4.2.2 Mecanismo de diseño de Acceso a Datos

Buscar mecanismos para modelar el acceso a datos siempre ha sido un meta a lograr, es por eso que un mecanismo aplicado para resolver esta situación nunca será igual a otro definido. Muchos son los especialistas que buscan un modelo ideal que sirva para todos los casos y se pueda emplear sin pensarlo dos veces, pero es una realidad que el desarrollo de hoy en día es tan vertiginoso que nos permite buscar soluciones tan buenas como otras ya definidas, convirtiéndose a su vez en mecanismos aplicables bajo ciertas circunstancias. Como todos sabemos el acceso y la manipulación de los datos es algo realmente indispensable a la hora de desarrollar un sistema informático.

Por todo esto, a raíz de la aparición del concepto de patrones, se han definido algunos que abordan esta problemática, por lo cual nos resulta un punto de partida a la hora de modelar el "acceso a datos".

¿Por qué definir un mecanismo para el acceso a datos?

Para acceder a los datos siempre están involucrados los mismos objetos y se efectúan un conjunto de operaciones comunes en las realizaciones de algunos casos de uso. Por todo esto la necesidad de documentar un mecanismo que simplifique el modelado y que quede como punto de referencia para los desarrolladores. Esto nos permitirá obtener diagramas más entendibles, que nos permitan una mayor comunicación con nuestro equipo de desarrollo; pero lo más importante es que nos trazará una línea común, una política a seguir, fomentando algo muy indispensable para lograr eficiencia, la *reutilización*.

Para nuestro sistema en cuestión se plantea el siguiente mecanismo de diseño para modelar el "acceso a datos".

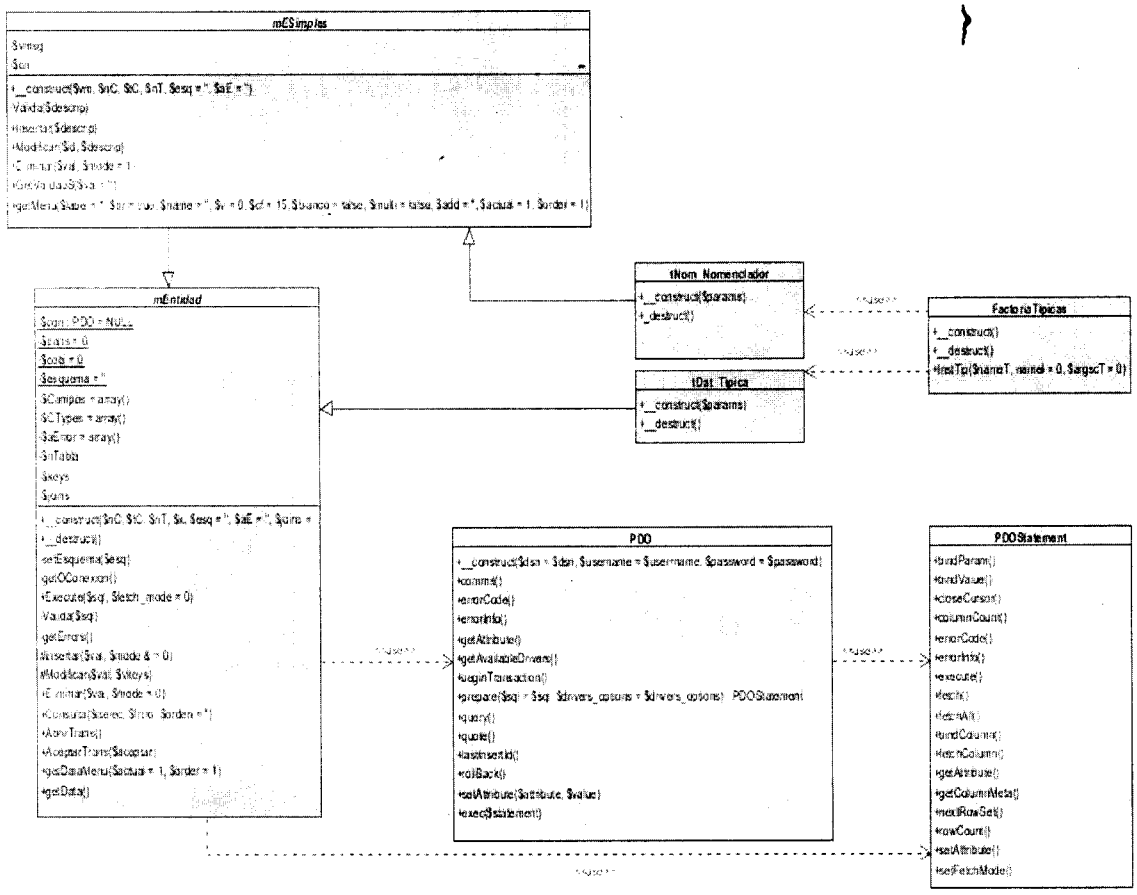


Figura 4.2 Vista estática del mecanismo de diseño para persistencia. Conectividad usando entorno desarrollo brindado por el lenguaje PHP.

La vista estática de este mecanismo de acceso a datos muestra un conjunto de clases que interactúan para dar acceso y manipulación de los datos de la persistencia desde el nivel más bajo, es decir, utilizando los objetos nativos brindados por el entorno de desarrollo PHP como son PDO y PDStatement, siguiendo así hasta la abstracción del acceso a datos a través de MEntidad de la cual heredan las clases particulares de nuestro sistema como Típicas y MSimples.

Para dar la responsabilidad a una clase que encapsulara las instancias de estos objetos se definió la clase FactoriaTipicas ya antes mencionada en el patrón de diseño aplicado a este funcionamiento.

4.3 Diagrama de clases del diseño

Para obtener un nivel correcto de abstracción y detalle que permita obtener un resultado final, es mejor modelar los artefactos del sistema, es decir, modelar las páginas, los enlaces entre ellas, todo el código que irá creando las páginas, así como el contenido dinámico de estas una vez que estén en el navegador del cliente; estos son los artefactos que se necesitan modelar para que el desarrollador los implemente luego y obtener así el producto final.

Para nuestro sistema por cada caso de uso tenemos concebido lo siguiente:

- ✓ Una clase de lógica de negocio (ln_**NombreCU**) por cada caso de uso.
- ✓ Una clase aportadora de contenido para todas las paginas clientes involucradas (ac_**NombreCU**). La misma tendrá un método (getInt**PaginaCliente1**, get**IntPaginaCliente2**... get**IntPaginaClienteN**) donde n es la cantidad de paginas clientes asociadas al caso de uso
- ✓ Una clase **GLOBAL** que centraliza a los objetos principales para manejar la seguridad y el acceso a datos: **Cseguridad** y **FactoriaTipicas**.
- ✓ Un paquete **Smarty** que gestiona a las plantillas, donde tendrá asociado tantas plantillas como páginas clientes tenga asociado el caso de uso.
- ✓ La página servidora tendrá las instancias del los objetos ln_**NombreCU** y ac_**NombreCU.**

Para entender el funcionamiento de la seguridad y el acceso a datos ver los mecanismos antes mencionados.

De acuerdo a la forma en que se ha organizado el contenido del trabajo se deben presentar los modelos organizados por paquetes y subpaquetes, de forma que pueda entenderse mejor la lógica del negocio.

Paquete 1 Configurar Recuperación: A - Actualizar Sistema, B - Actualizar Subsistema, C - Actualizar Campos del Subsistema, D - Generar Dependencias, E - Generar Relaciones, F - Actualizar Arquitectura de la Base de Datos.

Paquete 2 Recuperar: dentro este paquete ubicamos dos subpaquetes que poseen relación entre si, ya que algunos casos de uso se utilizan y se definen en ambos.

✓ Subpaquete Recuperación por Listado:

A - Definir Condiciones, B - Definir Campos a Mostrar, C - Crear Formato del Informe, D - Generar Plantilla, E – Visualizar Reporte.

✓ Subpaquete Recuperación por Estadística.

A - Definir Encabezados, B - Definir Campos por Filas, C - Definir Campos por columnas, D – Personalizar.

4.3.1 Configurar la recuperación

En este paquete se describe todo el proceso de configuración del sistema que luego es utilizado por los otros paquetes.

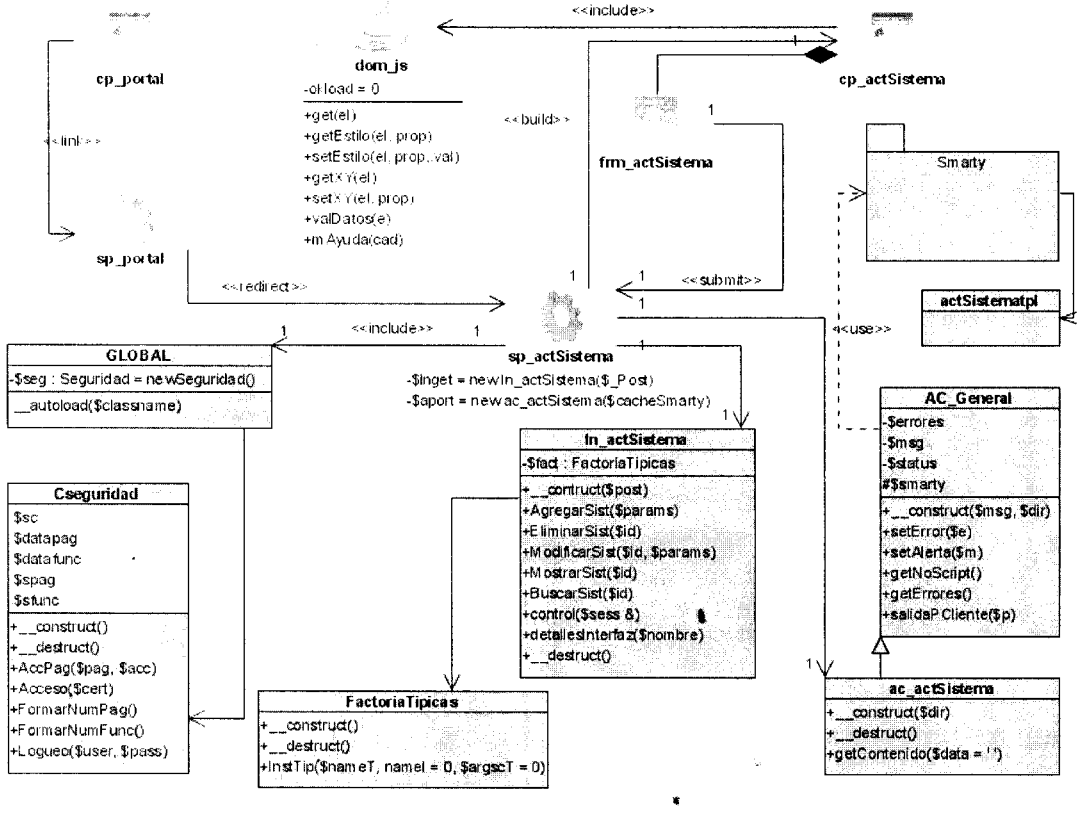


Figura 4.3 Diagrama de Clases A – Actualizar Sistema.

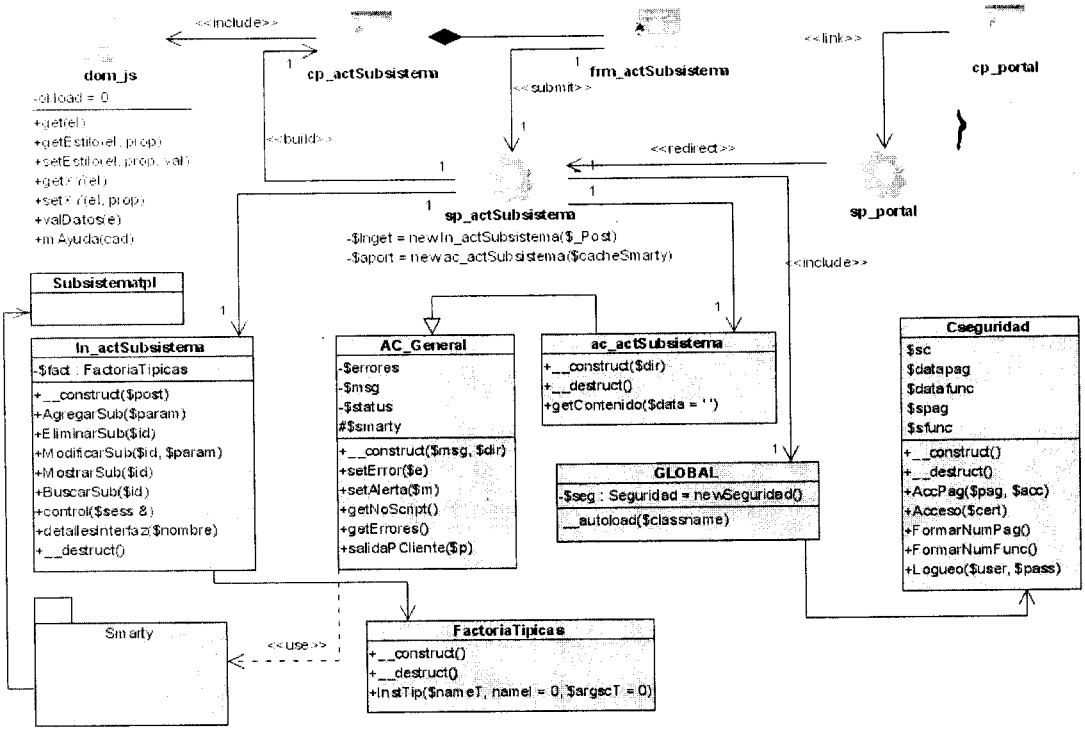


Figura 4.4 Diagrama de Clases B – Actualizar Subsistema.

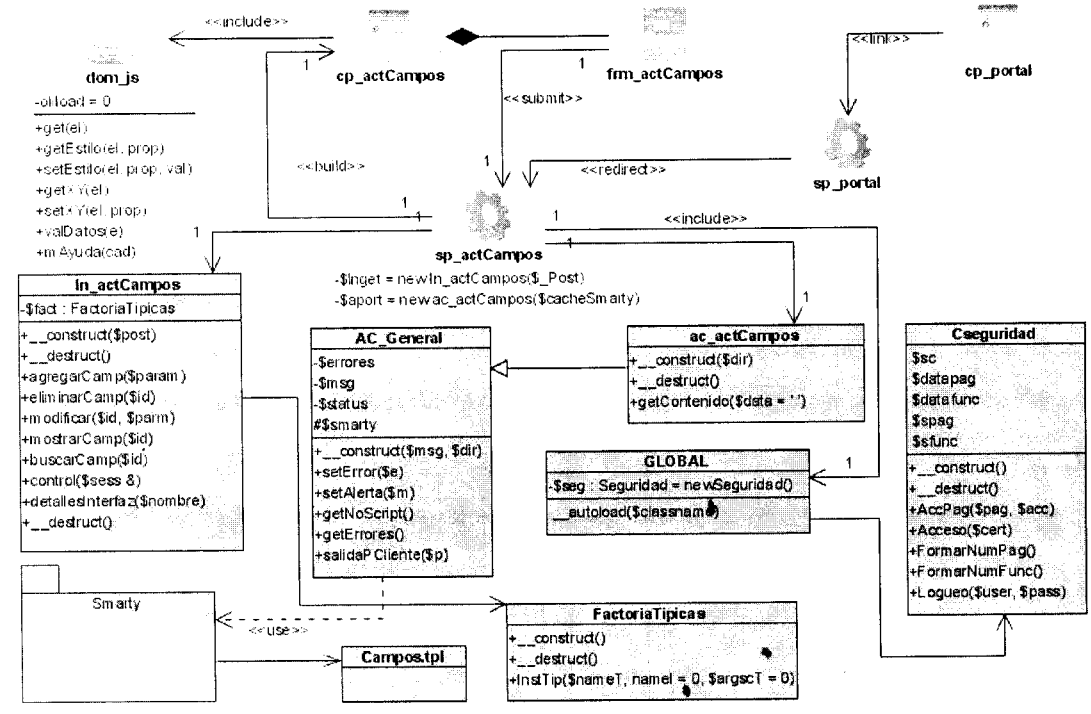


Figura 4.5 Diagrama de Clases C – Actualizar Campos del Subsistema.

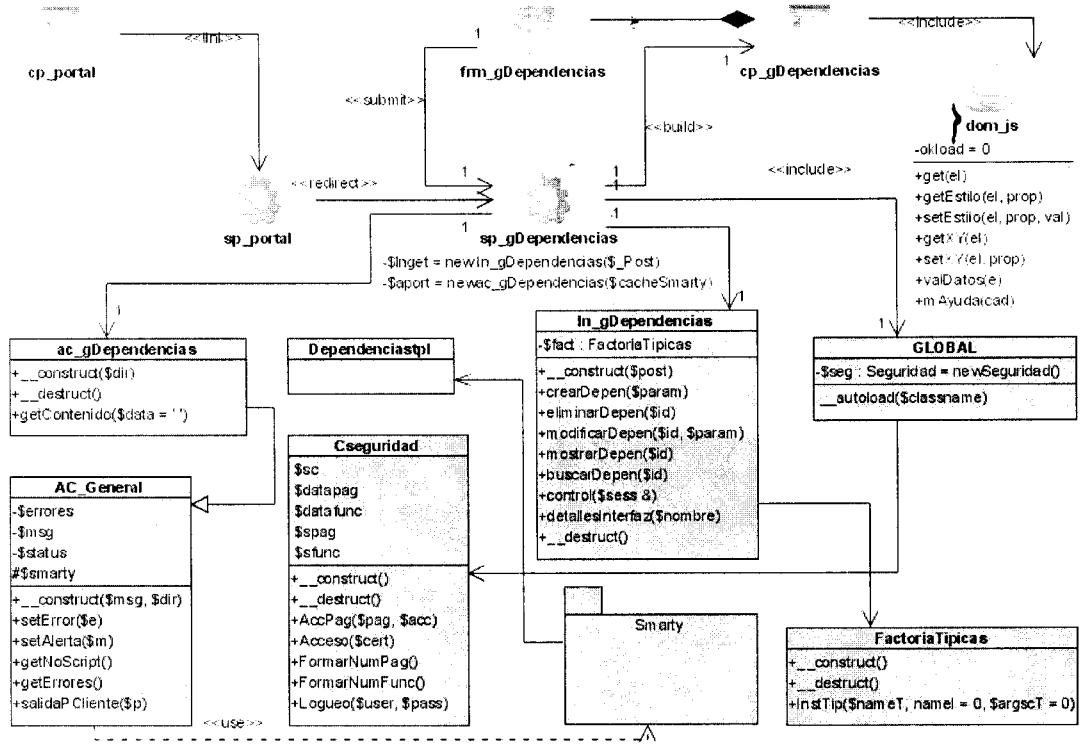
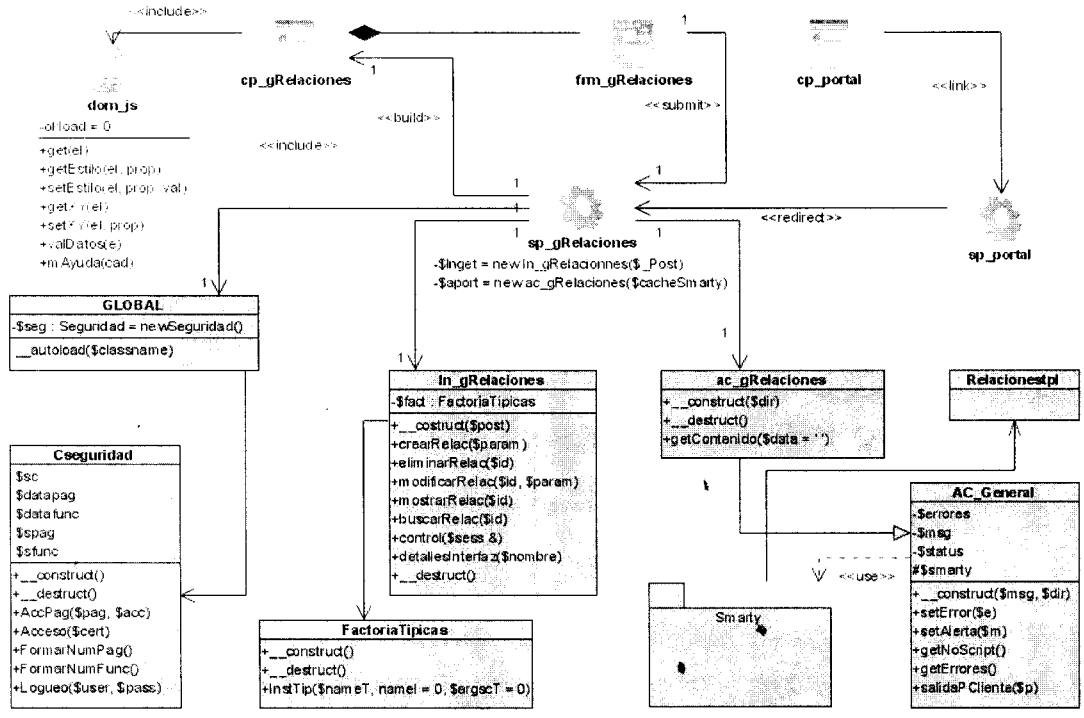
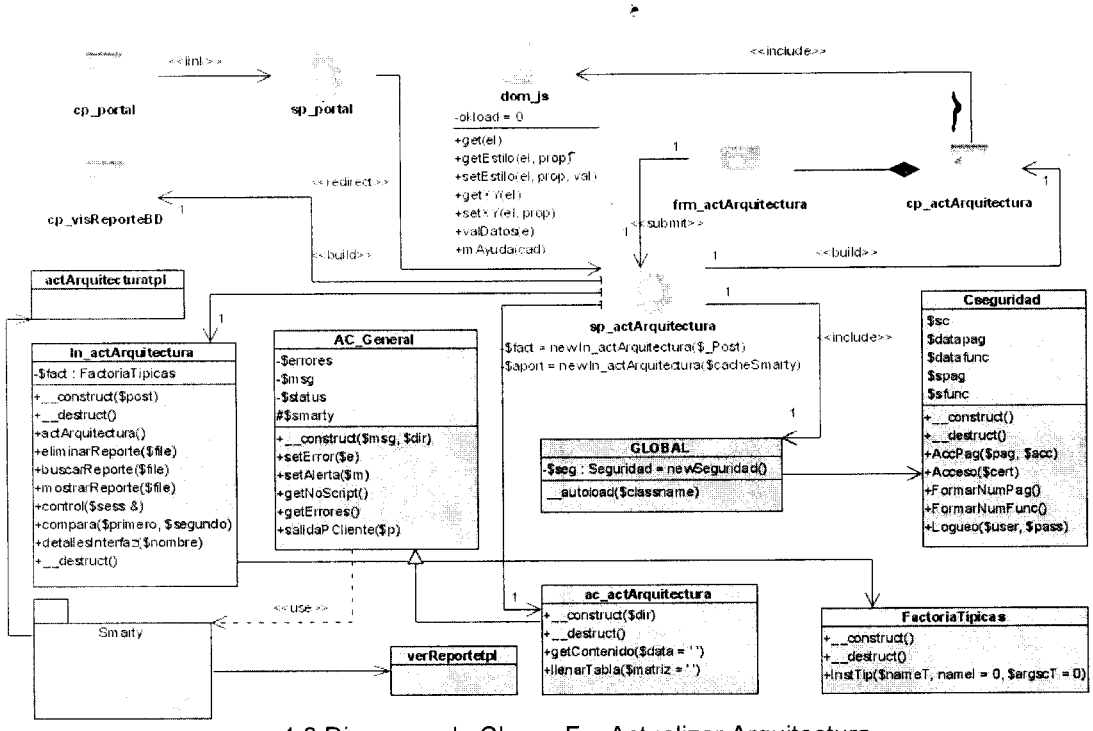


Figura 4.6 Diagrama de Clases D – Generar Dependencias.



4.7 Diagrama de Clases E – Generar Relaciones.



4.8 Diagrama de Clases F – Actualizar Arquitectura.

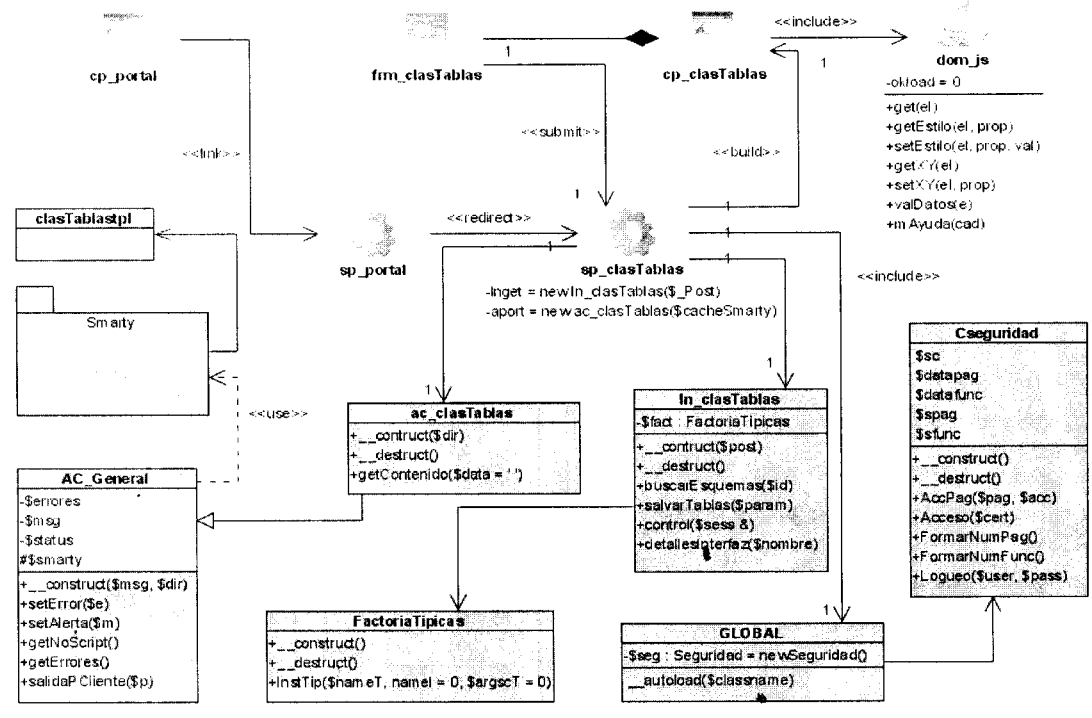


Figura 4.9 Diagrama de Clases G – Clasificar Tablas.

4.3.2 Recuperaciones Dinámicas

Subpaquete Recuperación por Listado

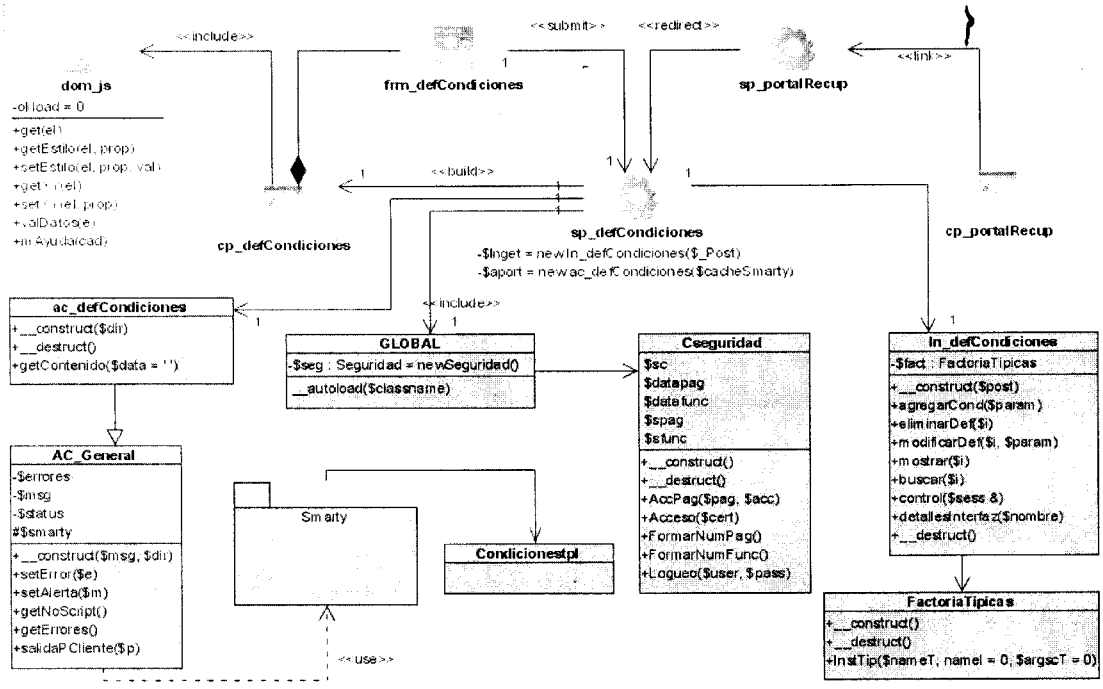
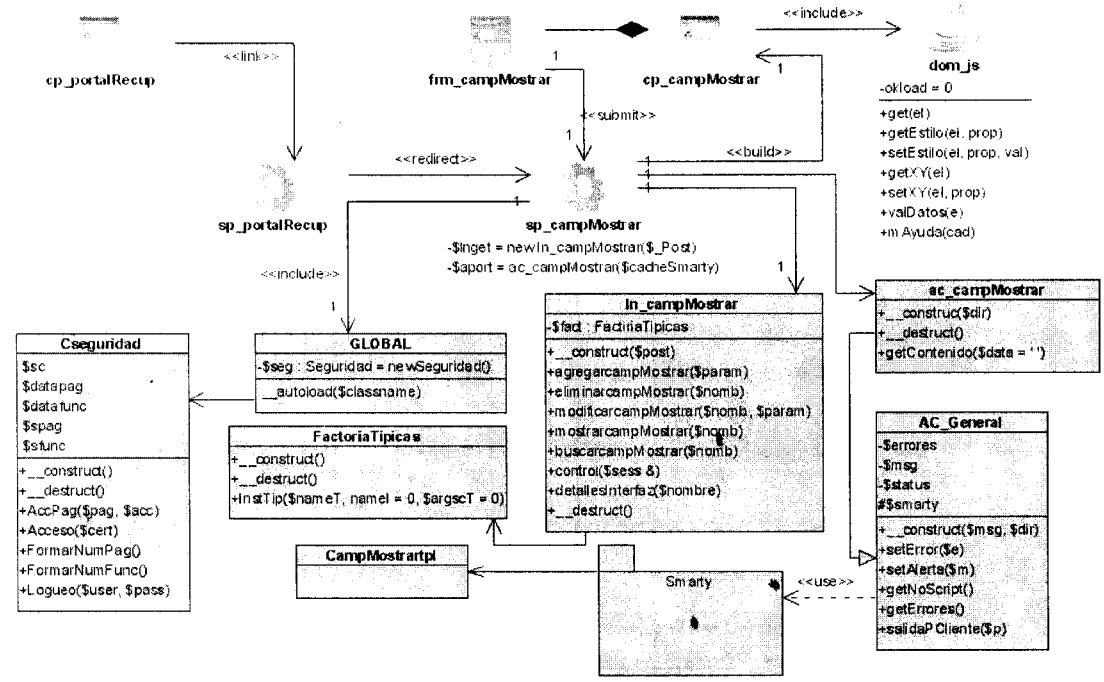


Figura 4.10 Diagrama de Clases A – Definir Condiciones.



4.11 Diagrama de Clases B – Definir Campos a Mostrar.

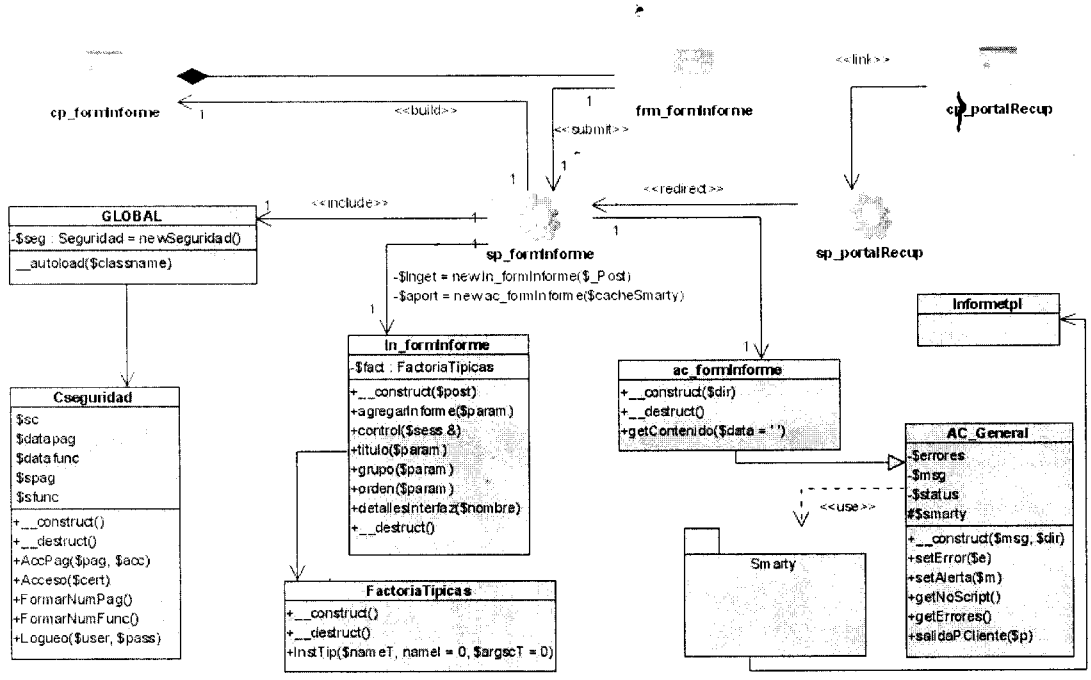


Figura 4.12 Diagrama de Clases C – Definir Formato de Informe.

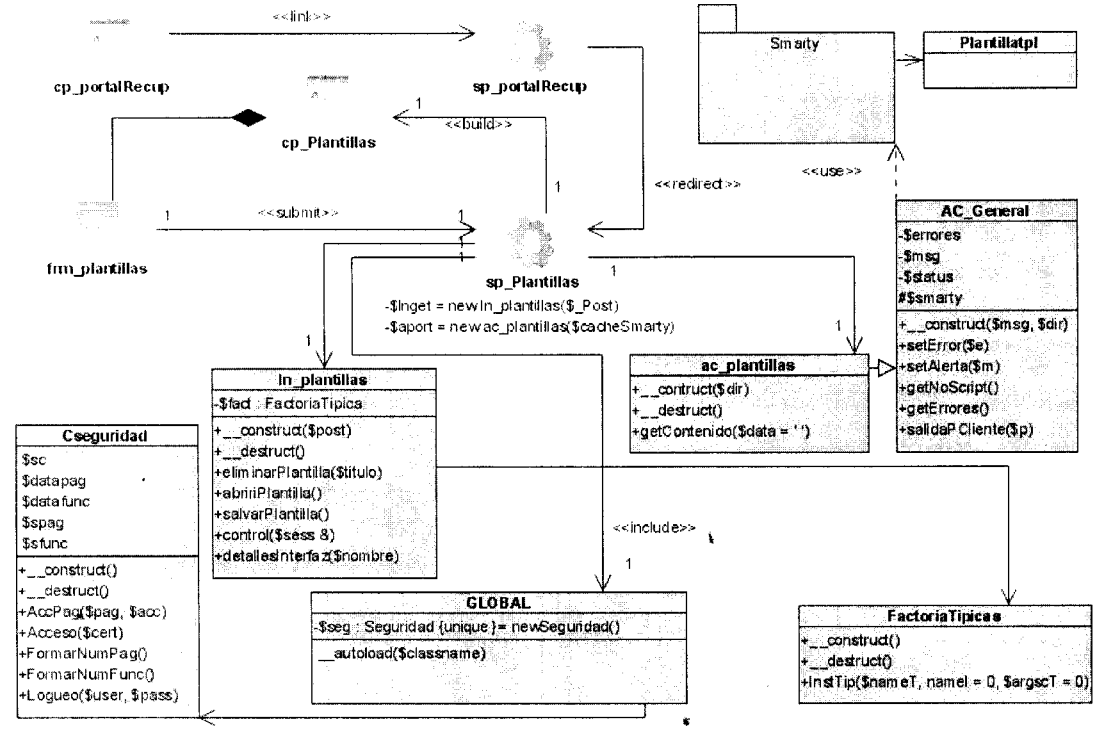


Figura 4.13 Diagrama de Clases D – Generar Plantilla.

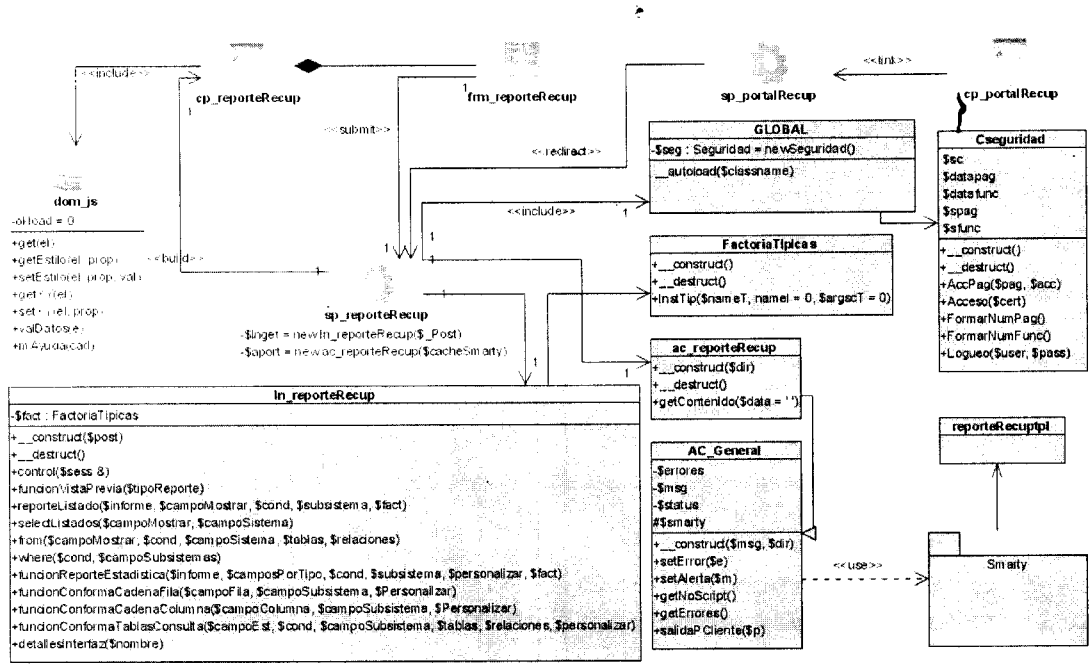


Figura 4.14 Diagrama de Clases E – Visualizar Reporte

Subpaquete Recuperación por Estadística

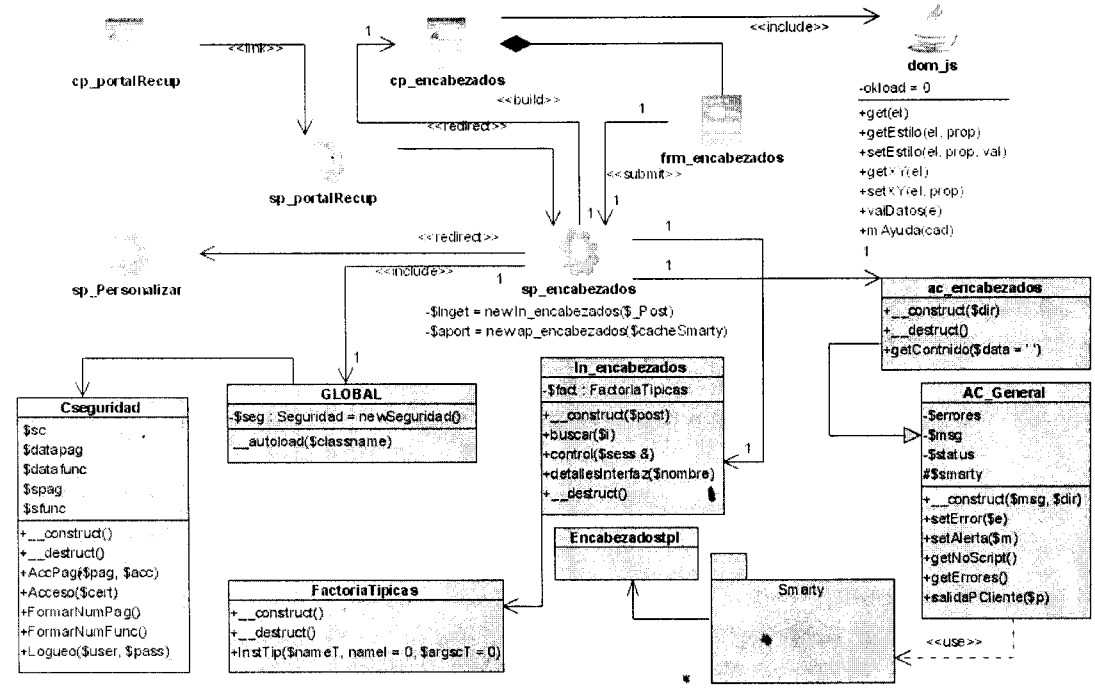


Figura 4.15 Diagrama de Clases A – Definir Encabezados.

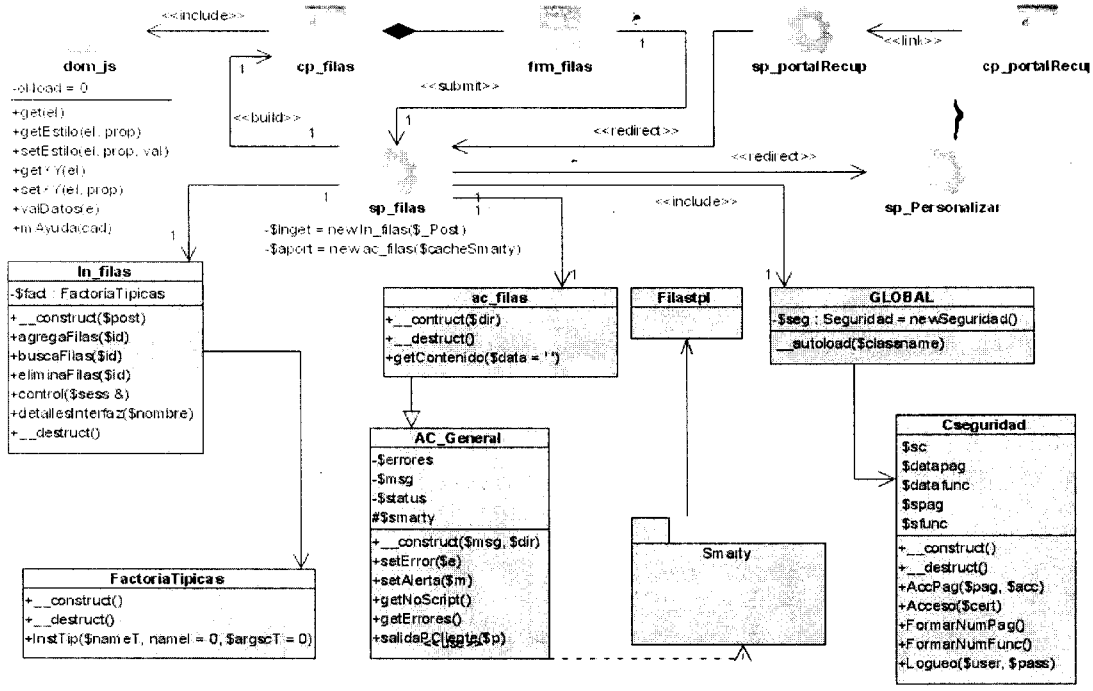


Figura 4.16 Diagrama de Clases B – Definir Campos por Filas.

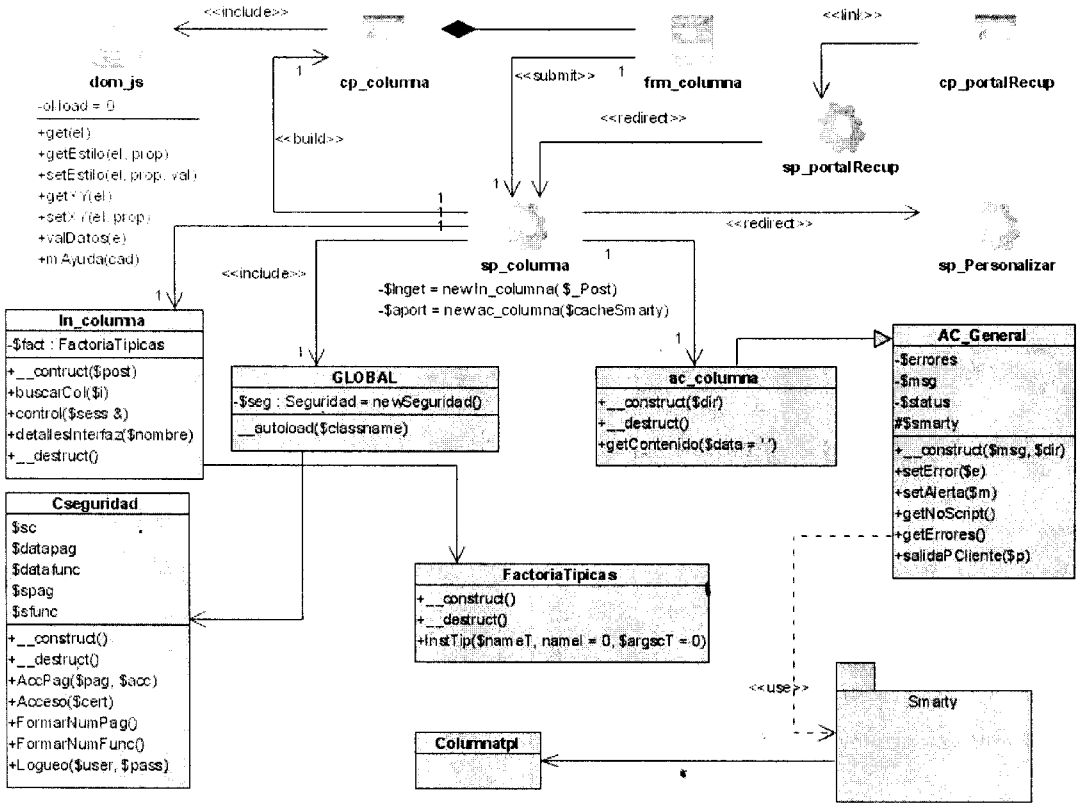


Figura 4.17 Diagrama de Clases C – Definir Campos por Columnas.

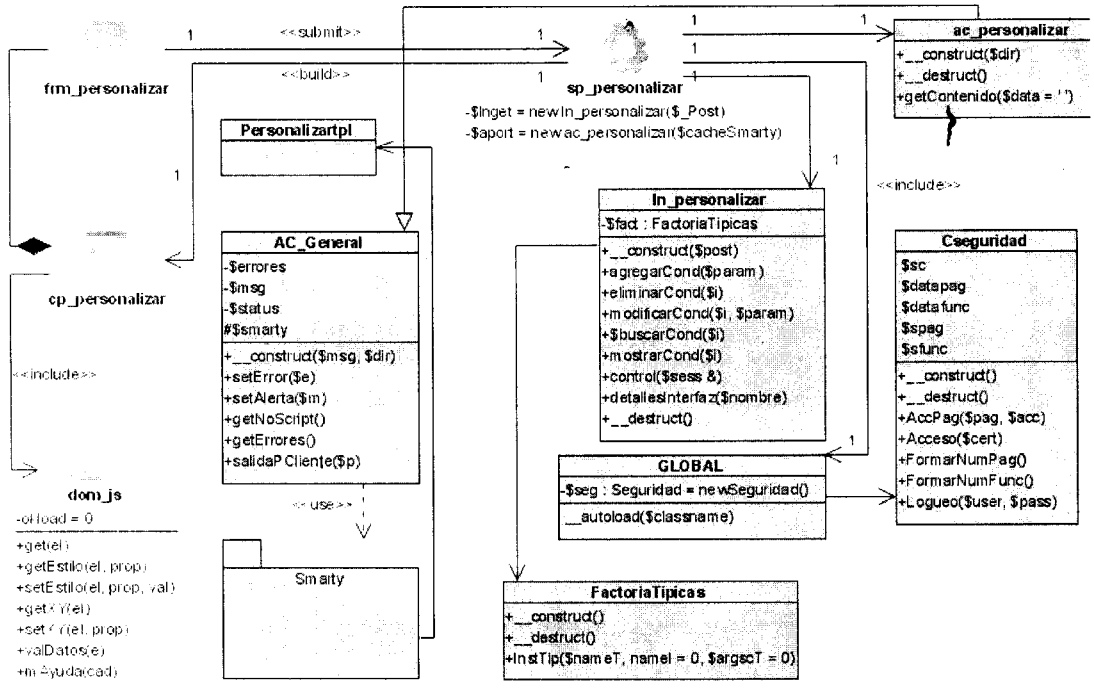


Figura 4.18 Diagrama de Clases D – Personalizar

4.4 Diseño de la Base de Datos

En el diseño de la base de datos se modela el tratamiento de la información con carácter persistente dentro del sistema.

Varios son los métodos y alternativas para modelar la persistencia de los datos, incluyendo una gran variedad de herramientas de modelado. La propuesta actual es modelar la persistencia de los datos a partir de los diagramas de clase, con herramientas modernas que realizan una traducción del modelo de clases a un modelo de datos relacional.

Se propone construir dos modelos para la representación de los datos persistentes: el Modelo Lógico de Datos y el Modelo Físico de Datos. Estos dos modelos proporcionan una flexibilidad óptima para el soporte de la automatización entre el Modelo de Análisis y Diseño, y la Base de Datos Física.

4.4.1 Diagrama de Clases Persistentes

Este modelo debe ser usado siempre que se modelen datos persistentes, en sistemas que manipulan gran cantidad de información, contenida en medios de almacenamiento persistente.

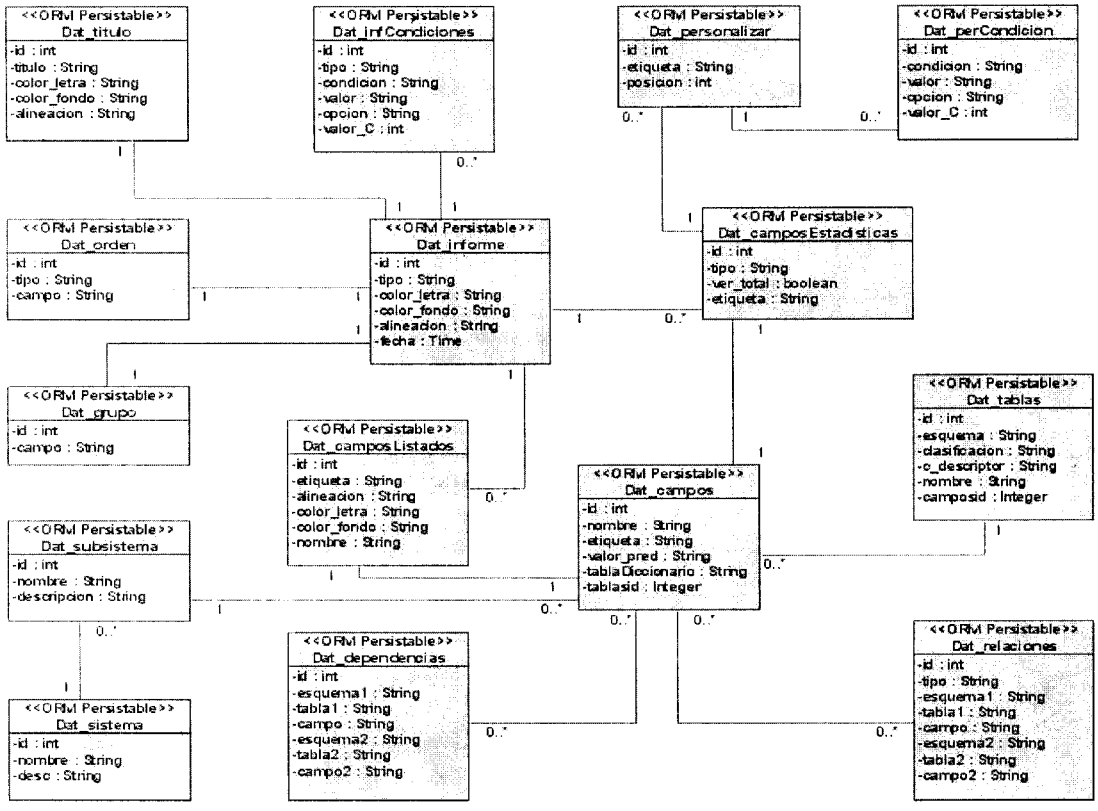


Figura 4.19 Diagrama del Modelo Lógico de los Datos.

4.4.2 Modelo de Datos

El modelo físico de los datos contiene un conjunto de tablas que conforman la base de datos. Este modelo constituye entonces la representación física del modelo de clases persistentes visto anteriormente. Las herramientas de modelado permiten la obtención de este modelo directamente a partir de las clases de los modelos anteriores.

Cuando se crea el Modelo Físico de la Base de Datos el diseñador debe seleccionar el sistema apropiado de Base de Datos, en nuestro caso el

PostGreSQL. No obstante, para lograr que la aplicación sea fácil de migrar, debería construirse utilizando un estándar que sirva para múltiples bases de datos.

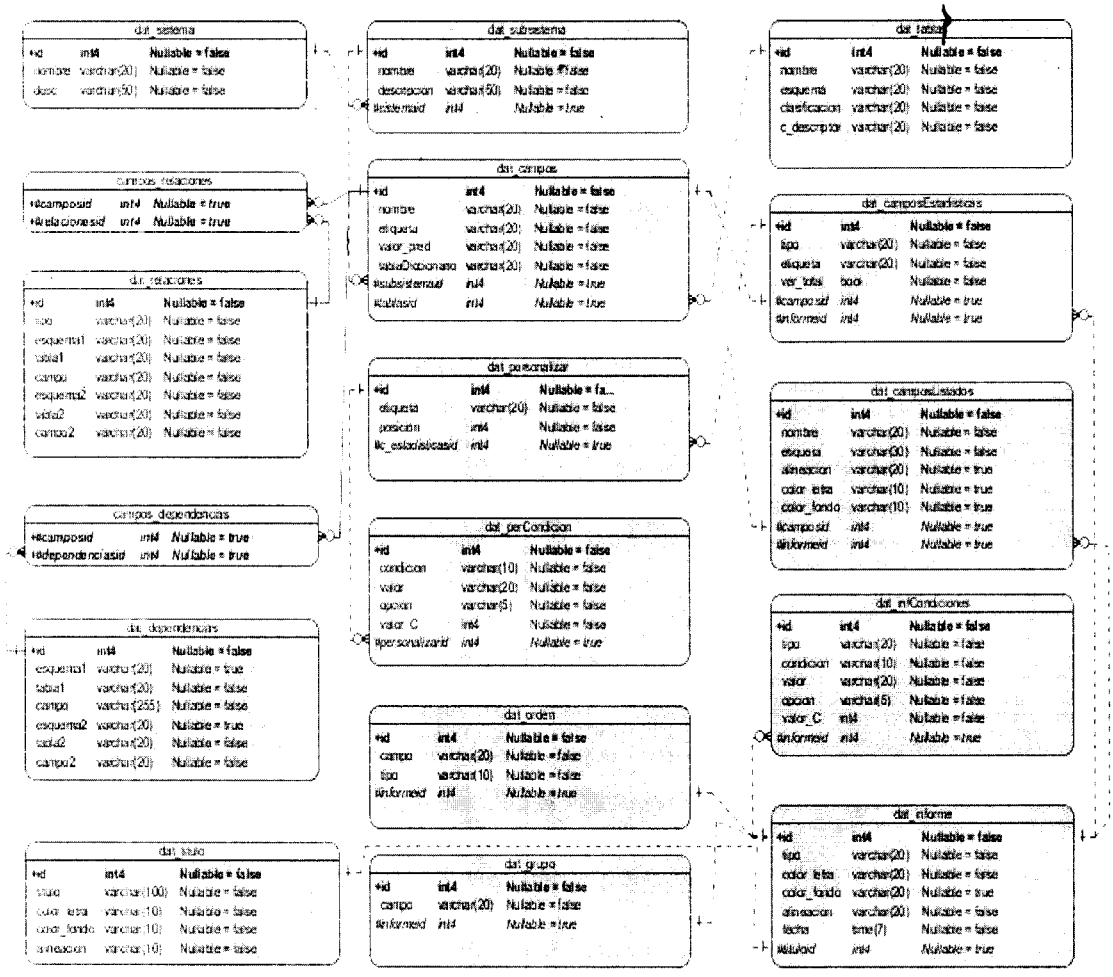


Figura 4.20 Diagrama del Modelo Físico de los Datos.

4.5 Principios de Diseño y Factores de Calidad

El diseño, sea cual sea el objeto del mismo, tiene que basarse en el usuario, muchos de ellos sin una preparación en las cuestiones de la informática. Este sirve para permutar los elementos de una página y ayùda al usuario a comprender los contenidos informativos con una presentación cómoda, eficaz y hermosa. Para ello, este sistema utiliza ciertos principios generales que garantizan la usabilidad en los diseños para aplicaciones Web.

1. Principio de uso equiparable: donde las características de privacidad, garantía y seguridad estén igualmente disponibles para todos los usuarios, y que el diseño sea atractivo para todos los usuarios.

2. Principio de la flexibilidad: donde se ofrezcan posibilidades de elección en los métodos de uso, que facilite al usuario la exactitud y precisión, y se adapte al paso o ritmo del usuario.

3. Principio de la Información perceptible: donde se usen diferentes modos para presentar de manera redundante la información esencial (gráfica y verbal), se proporcione contraste suficiente entre la información esencial y sus alrededores, se amplíe la legibilidad de la información esencial.

4. Principio de tolerancia al error: donde se dispongan los elementos para minimizar los riesgos y errores, por ejemplo utilizando elementos comunes; y los elementos peligrosos eliminados, aislados o tapados, que se proporcionen advertencias sobre peligros y errores. Hay que posibilitar el descubrimiento interactivo y la reversibilidad y recuperabilidad de las acciones.

5. Principio de esfuerzo de acceso y uso: que minimicen las acciones repetitivas, y que proporcione una línea de visión clara hacia los elementos importantes tanto para un usuario sentado como de pie.

Pueden distinguirse dos tipos de factores de calidad: los factores de calidad externos, aquellos que son perceptibles por los usuarios: corrección, robustez, extensibilidad, reutilización, compatibilidad y los factores de calidad internos que son los perceptibles por los especialistas en informática: modularidad y legibilidad.

Factores de calidad externos:

Corrección: el sistema tiene la habilidad para desempeñar las funciones, exactamente como le fueron definidas en los requisitos y especificaciones.

Solidez o robustez: puede funcionar en condiciones anormales, es decir, con aquellos casos no explícitos en las especificaciones. Si se presentan el sistema termina limpiamente.

Extensibilidad: posee la facilidad para adaptarse a los cambios en las diversas especificaciones.

Reutilización: capacidad para utilizar nuevos productos de software completos o partes de ellos en nuevas versiones de la aplicación.

Compatibilidad: puede combinarse con otros productos de software. Se logra homogeneidad en el diseño y estandarización en la comunicación entre aplicaciones.

Factores de calidad internos

Modularidad: existe independencia funcional entre los componentes de la aplicación.

Legibilidad: existe facilidad de lectura e interpretación del código del programa.

4.6 Estándares de codificación

Las principales ventajas de utilizar un estándar para escribir el código de las aplicaciones son:

1. Reduce los errores.
2. Garantiza la obtención de un código claro y comprensible.
3. Garantiza una buena comunicación entre los programadores del equipo
4. Facilita el mantenimiento del software.

Con el fin de unificar los esfuerzos y mejorar los rendimientos de la aplicación, se utilizó el paradigma de la programación orientada a objetos, creando clases para todas las funcionalidades. El manejo de la base de datos, de las búsquedas, y los recursos; todos están implementados como clases.

4.6.1 Estándares de codificación

A continuación se muestra el estándar de codificación a seguir para implementar el sistema.

Apariencia de clases y objetos	Primera letra en mayúscula	Los nombres de las clases y las instancias de las mismas deben comenzar con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación PascalCasing. Ejemplo: MiClase ().

Nombre de clases y objetos	Relacionados al propósito	El nombre empleado, debe permitir que con sólo leerlo se conozca el propósito de la clase o instancia de la misma. Para el caso de las instancias es recomendable que se denoten así: Para la clase: Nomcliente su instancia será $Ocliente, de forma tal que la primera letra indique que es un objeto y el resto, la clase a la que pertenece.
Apariencia de atributos	Primera letra en minúscula	El nombre que se le da a los atributos de las clases debe comenzar con la primera letra en minúscula, en caso de que sea un nombre compuesto se empleará notación CamelCasing.
Nombre de atributos	Nemotécnicos	El nombre empleado, debe permitir que con sólo leerlo se conozca el propósito del mismo dentro de la clase. Ejemplo: $nTabla, este atributo denota el nombre de una tabla.
Apariencia de las funciones	Primera letra en mayúscula	Los nombres de las funciones deben comenzar con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación PascalCasing. Si son funciones que obtienen un dato se emplea el prefijo **get** y si fijan algún valor se emplea el prefijo **set.**
Nombre de las funciones	Nemotécnicos	El nombre empleado, debe permitir que con sólo leerlo se conozca el propósito de la misma dentro de la clase.
Declaración de parámetro en funciones	Agrupados por tipos primero los string, los numéricos y valores por defecto.	Los parámetros que se le pasan a las funciones se recomienda sean declarados de forma tal que estén agrupados por el tipo de dato que contienen. Ejemplo: BuscaUnidad $nTabla(string),$nCampos(string),$kIndice (entero)).
Variables y constantes		
Apariencia de constantes	Todas sus letras en mayúscula	Se deben declarar las constantes con todas sus letras en mayúscula.
Nombres de las variables y constantes	Nemotécnicos	El nombre empleado, debe permitir que con sólo leerlo se conozca el propósito de la misma. Ejemplo: $nFields.
Declaración de constantes y	Una por cada línea	Se recomienda declarar una constante por cada línea y con las asignaciones a las

asignación a variables		variables sucede lo mismo.
		Ejemplo:
		define("CONSTANT1","value1");
		define("CONSTANT2","value2");
		$nTabla='nomproducto';
		$kIndice=0;

Identación **Objetivo:** Lograr una estructura uniforme para los bloques de código así como para los diferentes niveles de anidamiento.		
0 espacios en blanco desde la izquierda en	Require Include Class	No se empleará ningún espacio en blanco desde la izquierda para las instrucciones antes mencionadas. Se tomará como inicio de la página el tag PHP <?
2 espacio en blanco desde la izquierda en	Function Define	Se dejarán dos espacios en blanco desde la izquierda en las instrucciones antes mencionadas.
2 espacio en blanco desde la referencia en	Inicio y fin de bloque	Se recomienda dejar dos espacios en blanco desde la instrucción anterior para el inicio y fin de bloque {}. Lo mismo sucede para el caso de las instrucciones If, else, For, While, Do While, Switch, Foreach.
Niveles de anidación	Hasta 5 niveles	Se recomienda emplear hasta 5 niveles de anidación en instrucciones If, For, While.

Ejemplo de identación

```
<?
require ('class/Interface.php');
class MiClase
{
  function BuscaUnidad($nTabla, $nFields, $kIndice)
  {
    if ($nTabla)
    {
      ...
    }
    else (...)
    {
      ...
    }
  }
}
?>
```

Comentarios, separadores, líneas y espacios en blanco

Objetivo: Establecer un modo común para comentar el código de forma tal que sea comprensible con sólo leerlo una vez.

Ubicación de comentarios	Al inicio de cada clase o función y al final de cada bloque de código.	Se recomienda comentar al inicio de la clase o función especificando el objetivo de la misma así como los parámetros que usa (especificar tipos de dato, y objetivo del parámetro) entre otras cosas. Y se comenta también cuando se cierran los ciclos, clases, instrucciones if y otras.
Separador de instrucciones	Se emplea el punto y coma.	Se recomienda usar el separador al final de cada instrucción y no en la línea de abajo. Ejemplo: define ("CONSTANT","value1");
Líneas en blanco	Se emplean antes de cada función.	Se recomienda dejar una línea en blanco antes de la definición de cada función para dar claridad al código.
Espacios en blanco	Entre operadores lógicos y aritméticos.	Se recomienda usar espacios en blanco entre estos operadores para lograr una mayor legibilidad en el código. Ejemplo: $nTabla = 'nomproducto'; if (($nTabla) && ($nFields))

4.6.2 Estándares para la BD.

Apariencia de la BD	Primera letra en mayúscula	Los nombres de las BDs deben comenzar con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación PascalCasing.
Nombres de las BDs	Nemotécnicos y relacionados al propósito.	El nombre empleado, debe permitir que con sólo leerlo se conozca el propósito de la misma.
Apariencia de los esquemas	Todas las letras en minúscula.	El nombre a emplear para los esquemas debe escribirse con todas las letras en minúscula para evitar problemas con el Case Sensitive del gestor. Ejemplo: create schema 'finanzas';
Nombres de los esquemas	Nemotécnicos y relacionados al propósito.	El nombre empleado, debe permitir que con sólo leerlo se conozca el propósito del

		mismo.
Apariencia de las tablas	Todas las letras en minúscula.	El nombre a emplear para las tablas debe escribirse con todas las letras en minúscula para evitar problemas con el Case Sensitive del gestor. Ejemplo: create table 'nom_producto';
Nombres de las tablas	Nemotécnicos y relacionados al propósito. Además clasificando las tablas por su tipo.	El nombre empleado, debe permitir que con sólo leerlo se conozca el propósito del mismo. Se deben clasificar las tablas por su tipo, es decir por los datos que contienen se le coloca un prefijo, que se puede clasificar en: Ejemplo: Nomencladores nom_... Auxiliares aux_... Datos dat_... Históricas his_... Seguridad seg_... Temporales tmp_... Configuración cfg_...
Apariencia de los campos	Todas las letras en minúscula.	El nombre a emplear para los campos debe escribirse con todas las letras en minúscula para evitar problemas con el Case Sensitive del gestor. Ejemplo: add field 'idproducto';
Nombre de los campos	Nemotécnicos En caso de identificadores, emplear id, este seria igual en la tabla de datos que lo emplea.	El nombre empleado, debe permitir que con sólo leerlo se conozca el propósito del mismo. Además se debe incluir un comentario en la descripción del mismo.
Nombre de las llaves primarias	Nemotécnicos empleando prefijos.	Se nombrarán las llaves primarias de forma que se vea de qué tabla es y que es primaria. Ejemplo: pk_cuenta. (Llave primaria de la tabla cuenta). Si es una llave compuesta se coloca el prefijo y en nemotécnico los campos que la forman.

Nombre de las llaves foráneas.	Nemotécnicos empleando prefijos.	Se nombrarán las llaves foráneas de forma que se vea de qué tabla es y que es foránea. Ejemplo: fk_cuenta. (Llave foránea de la tabla cuenta). Si es una llave compuesta se coloca el prefijo y en nemotécnico los campos que la forman.
Nombre de las secuencias	Nemotécnicos empleando prefijos.	Se nombrarán las secuencias de forma que se vea de qué campo es y que es una secuencia. Ejemplo: seq_idcuenta. (Secuencia del campo idcuenta).
Restricciones Únicas y de Chequeo	Nemotécnicos empleando prefijos.	Ejemplo: (u_ o c_) + nombre del campo que la emplea.
Nombres de las funciones, triggers, y vistas	Prefijos + Nemotécnicos	El nombre empleado, debe permitir que con sólo leerlo se conozca el propósito del mismo. Ejemplo: ft_ Funciones de triggers.

4.7 Modelo de despliegue

El diagrama de despliegue representa la arquitectura de tiempo de ejecución de los procesadores, dispositivos y los componentes de software que se ejecutan en esa arquitectura. Es la última descripción física de la topología del sistema y describe la estructura de las unidades de hardware.

En nuestro caso el diagrama representa un *modelo ideal* que está en correspondencia con la arquitectura cliente-servidor en tres capas, compuesto por cuatro nodos y la conexión entre ellos.

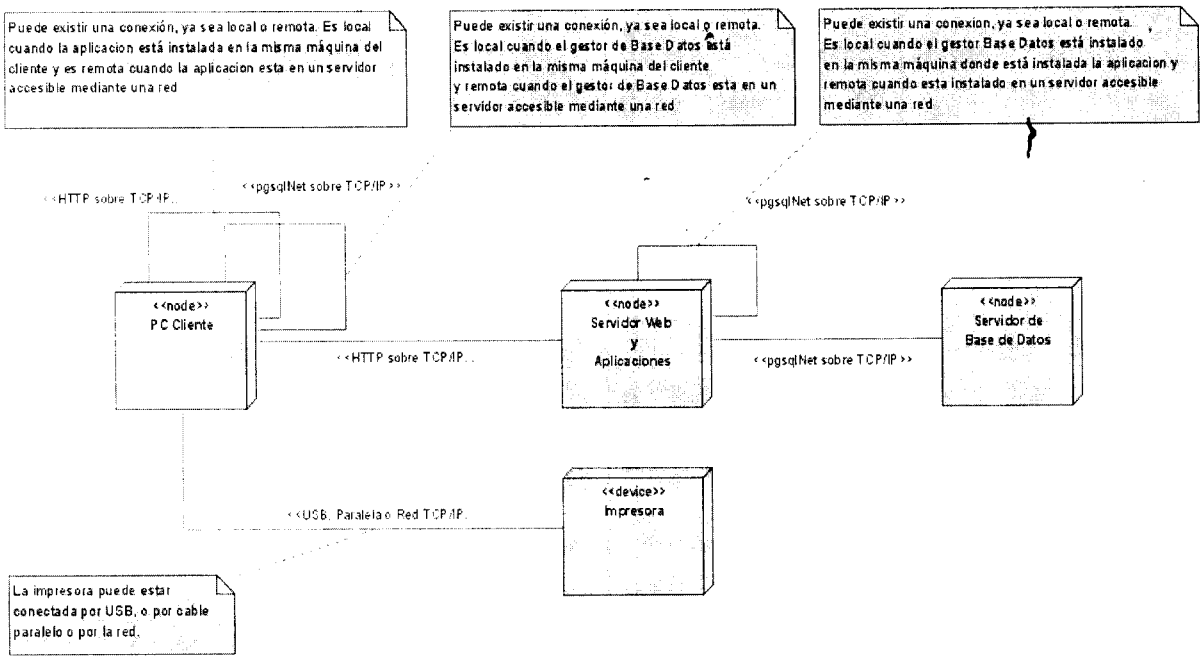


Figura 4.21 Diagrama de Despliegue.

Ahora nos preguntamos: ¿Por qué en el modelo se representan ----líneas que inciden sobre los mismos nodos de los que parten?

Primeramente debemos ubicarnos en que nuestra aplicación no solo será utilizada por una sola institución, sino por un conjunto de ellas, donde no todas cuentan con los recursos necesarios para poder aplicar este modelo ideal.

Existen empresas en nuestro país que necesariamente tienen que aplicar otros modelos. En el caso de aplicar el *modelo semi-ideal*, la topología del sistema se verá representada en dos nodos: uno corresponderá al terminal cliente y en el otro estarán contenidos el servidor Web y el servidor de Base de Datos.

Por otra parte se podría aplicar también el *modelo crítico* en empresas con un bajo nivel de desarrollo. En este caso el terminal cliente, el servidor Web y el de Base de Datos estarán contenidos en un solo recurso y la topología del sistema se representará entonces por un solo nodo.

4.8 Modelo de implementación

El modelo de implementación constituye la vista de Implementación de la arquitectura, y como tal guía las labores de construcción del sistema. Este contiene fundamentalmente los subsistemas de implementación, incluyendo las dependencias y otras informaciones necesarias para su utilización.

Para logar una mejor comprensión de los componentes que forman nuestro sistema, presentaremos primeramente un diagrama de paquetes de componentes los cuales representan una división lógica de varios componentes según su funcionalidad.

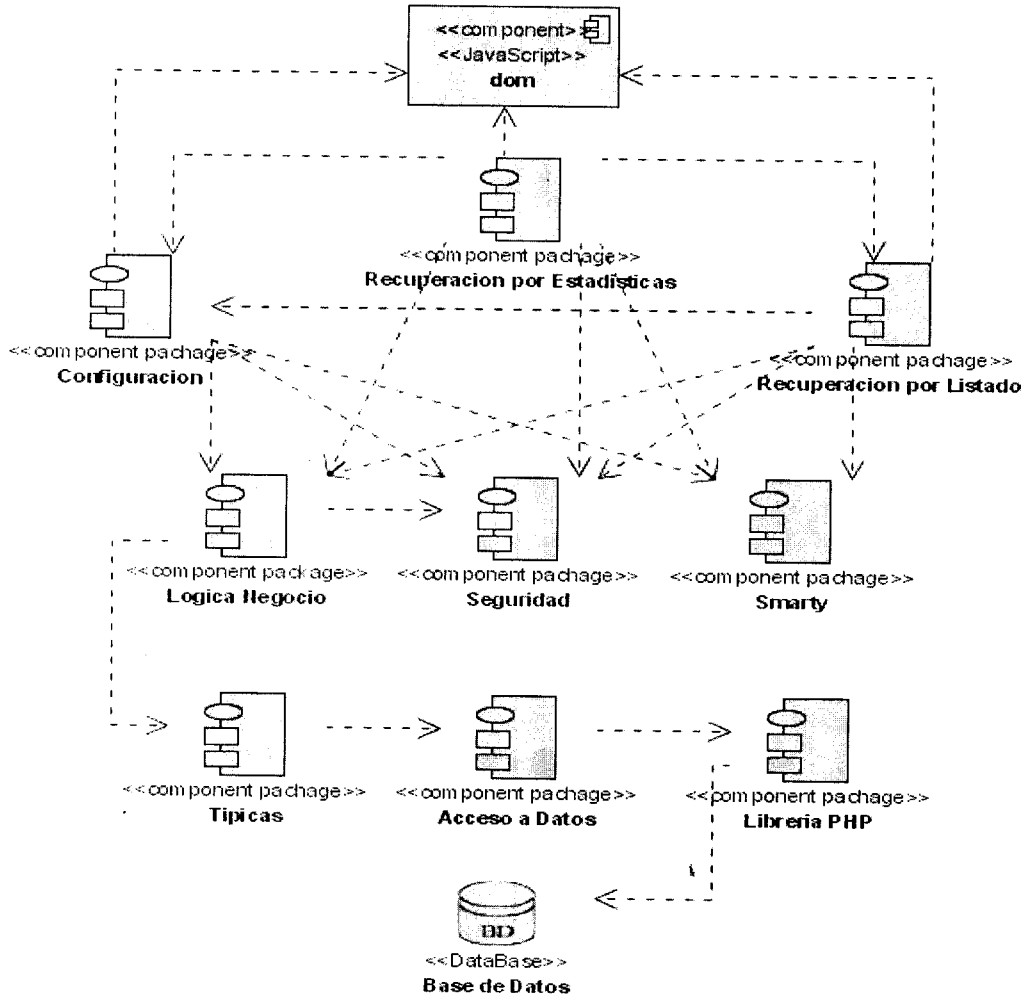


Figura 4.22 Diagrama de Componentes.

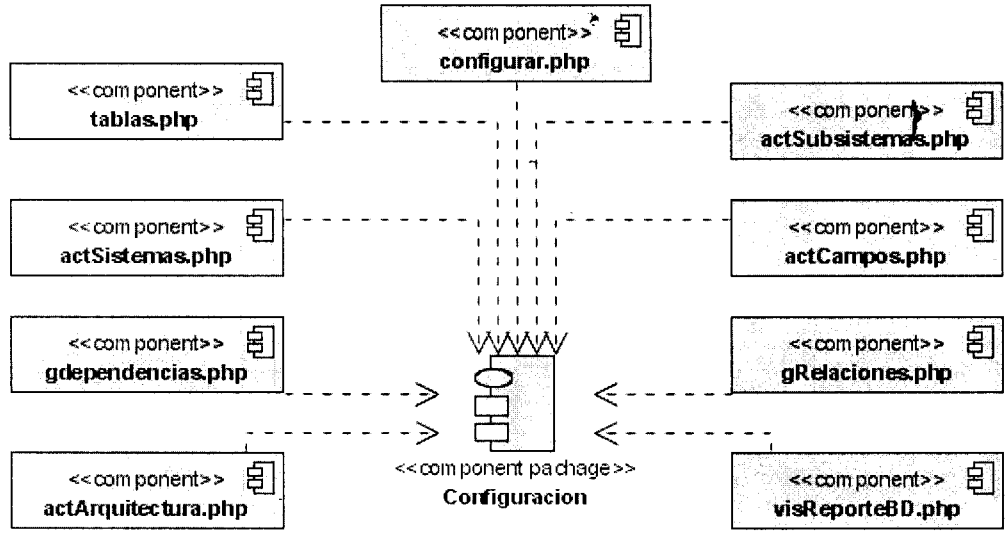


Figura 4.23 Representación de los Componentes del paquete Configuración.

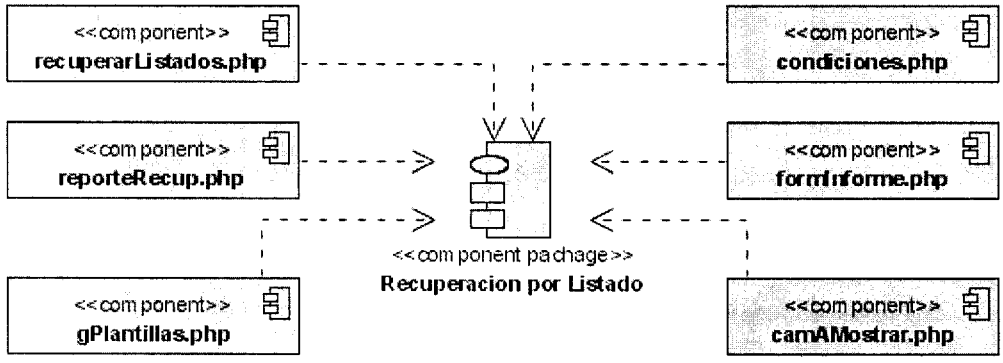


Figura 4.24 Representación de los Componentes del paquete Recuperación por Listados.

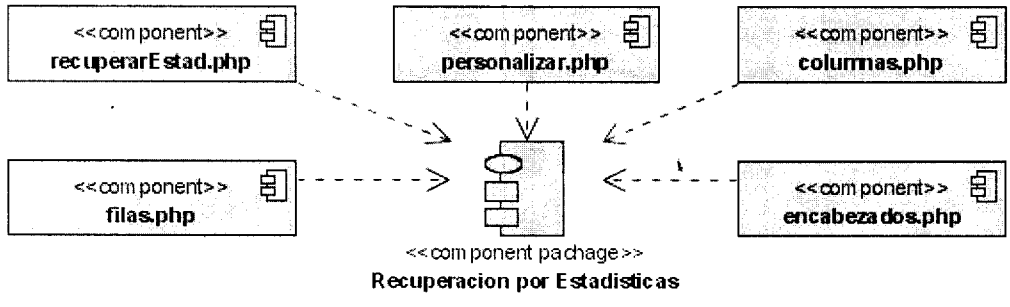


Figura 4.25 Representación de los Componentes del paquete Recuperación por Estadísticas.

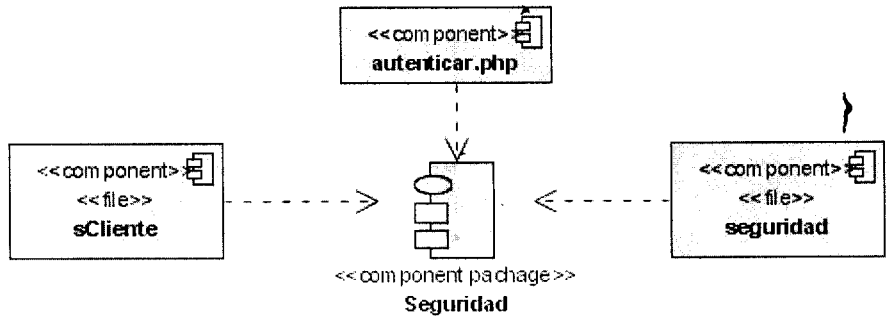


Figura 4.26 Representación de los Componentes del paquete Seguridad.

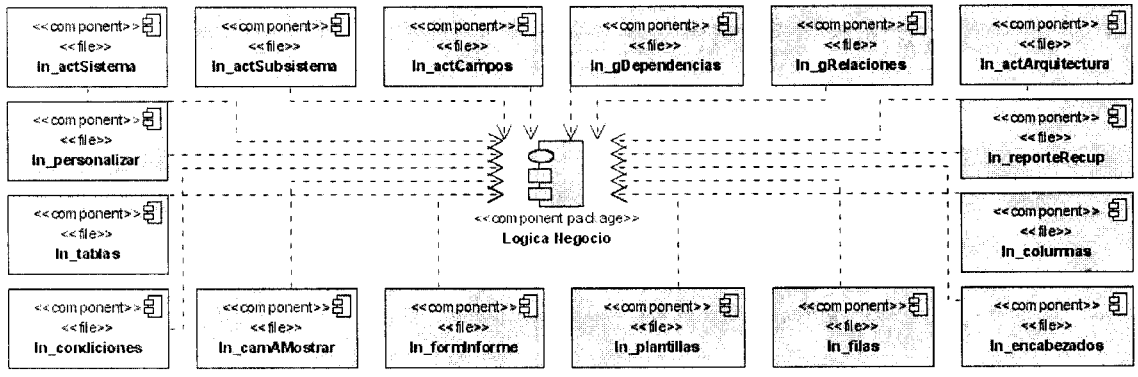


Figura 4.27 Representación de los Componentes del paquete Lógica de Negocio.

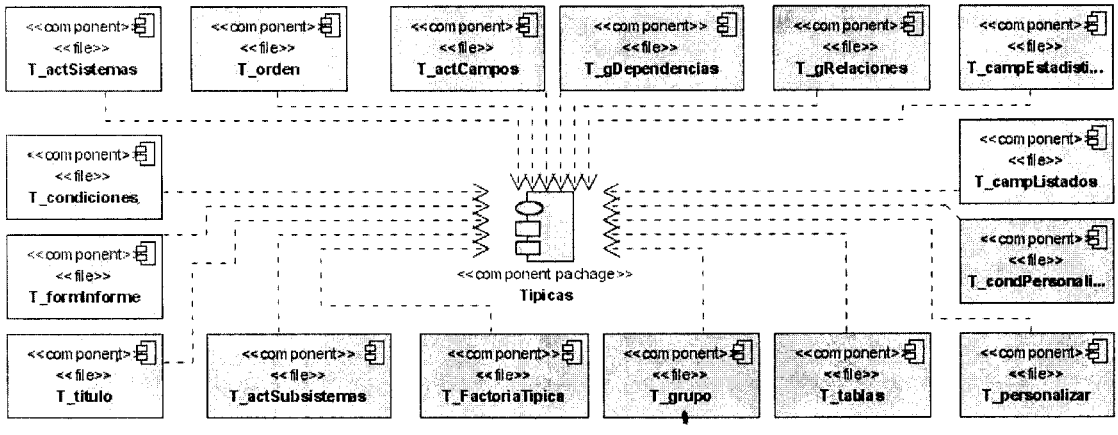


Figura 4.28 Representación de los Componentes del paquete Tipicas.

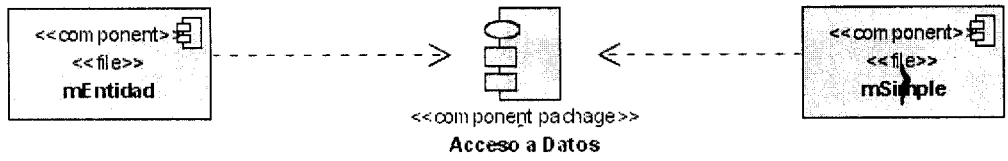


Figura 4.29 Representación de los Componentes del paquete Acceso a Datos.

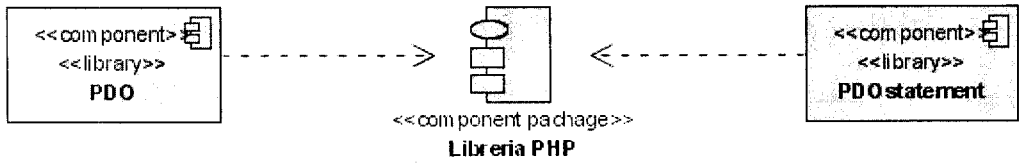


Figura 4.30 Representación de los Componentes del paquete Librería PHP.

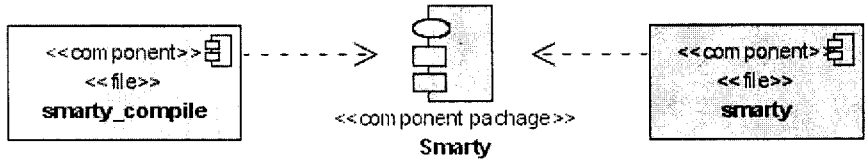


Figura 4.31 Representación de los Componentes del paquete Smarty.

4.9 Conclusiones

En este capitulo se abarcó lo perteneciente a las vistas estáticas y de implementación correspondientes a la notación UML. De igual manera fueron señalados los contenidos que sobre los principios de diseño debían añadirse a los requisitos funcionales mencionados en el capítulo anterior, para de esta forma completar los análisis al respecto. Con este capítulo culmina la modelación completa del Gestor de Recuperaciones Dinámicas y se ha hecho alusión a todas las vistas del Lenguaje Unificado de Modelado (UML).

Capítulo

V

ESTUDIO DE FACTIBILIDAD

5.1 Introducción

La estimación de costo y esfuerzos sigue siendo una de las tareas más difíciles en la gestión de un proyecto de software. El análisis del costo de un proyecto es imprescindible a la hora de acometer una tarea, es la forma que se tiene de saber si la realización del mismo es factible o no. Esta actividad es realizada por el jefe de proyecto, quien es responsable de hacer dichas estimaciones lo más precisas posible. En este capítulo hacemos un análisis de costo y beneficios que tendría la realización del proyecto, para esto aplicamos una técnica de estimación de esfuerzo y tiempo de desarrollo llamada Análisis por Puntos de Casos de Uso que predice el tamaño de un sistema a partir de las características de sus requisitos, expresados en los casos de uso.

La especificación de los requerimientos mediante Casos de Uso ha probado ser uno de los métodos más efectivos para capturar la funcionalidad de un sistema. Este hecho se puede apreciar en algunas metodologías actuales ampliamente difundidas, como es el Proceso Unificado de Rational (Rational Unified Process), en el cual se propone especificar la funcionalidad de los sistemas mediante la utilización de Casos de Uso, ya que este permite documentar los requerimientos de un sistema en términos de Actores y Casos de Uso, donde un actor típicamente representa a un usuario humano o a otro sistema que interactúa con el sistema bajo análisis.

Existe una relación natural entre los Puntos de Función y los Casos de Uso. Los Puntos de Función permiten estimar el tamaño del software a partir de sus requerimientos, mientras que los Casos de Uso permiten documentar los

requerimientos del software. Ambos tratan de ser independientes de las tecnologías utilizadas para la implementación.

5.2 Planificación

Todo proyecto necesita una planificación. Esta, básicamente, debe incluir las actividades a realizar durante la ejecución del proyecto, los responsables y las fechas de cumplimiento. Su objetivo fundamental es establecer planes razonables para desarrollar la Ingeniería de Software y manejar los cambios de los proyectos de Software. En ella se establecen los pasos necesarios a seguir y se define el plan de desarrollo del software.

La planificación del proyecto proporciona un marco de trabajo que permite al gestor hacer estimaciones razonables de recursos, costos y planificación temporal. Estas estimaciones se hacen dentro de un marco de tiempo limitado al comienzo de un proyecto de software, y deben actualizarse regularmente a medida que progresa el proyecto. Además las estimaciones deben definir los escenarios del mejor caso y peor caso, de modo que los resultados del proyecto pueden limitarse.

Sin la existencia de una planificación bien fundamentada casi siempre se cometen los siguientes errores:

1. Mal análisis en los requerimientos.
2. No tener una negociación (documento, contrato) con el cliente.
3. No hacer un análisis costo beneficio.
4. Desconocer el ambiente de trabajo de los usuarios.
5. Desconocer los usuarios que trabajan con el sistema.
6. Mala elección de recursos (hardware, software, humanos).

En la planificación de nuestro trabajo siempre se tuvo en cuenta que existieran dependencias entre las tareas identificadas y a cada una de ellas se le asignó un cierto número de unidades de trabajo, así como fecha de inicio y de fin, no se sobreasignaron recursos y a cada miembro del equipo se le dieron tareas específicas con un resultado bien definido.

Para controlar la planificación realizamos una guía para la administración del proyecto y sus actividades. Para esto utilizamos una de las herramientas

automáticas más conocida y fácil de utilizar Microsoft Project y en ella realizamos el plan del proyecto (Ver Anexo).

5.3 Estimación de esfuerzo y costo

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

Aquí se describe la estimación de costo del sistema propuesto y sus beneficios.

Aplicando la técnica de Análisis de Puntos de Casos de Uso obtenemos:

Paso 1. Cálculo de Puntos de Casos de Uso sin ajustar.

Se calcula a partir de la siguiente ecuación:

$$UUCP = UAW + UUCW$$

Donde:

UUCP: Puntos de casos de uso sin ajustar.

UAW: Factor de peso de los actores sin ajustar.

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

Tipo de actor	Descripción	Factor de peso	Actores	Total
Simple	Sistema con sistema a través de interfaz de programación.	1	0	0
Medio	Sistema con sistema mediante protocolo de interfaz basada en texto.	2	0	0
Complejo	Persona que interactúa con el sistema mediante interfaz gráfica.	3	2	6

Calculando el factor de peso de los actores sin ajustar nos queda que:

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

$$UAW = 6$$

Para obtener el factor de peso de los casos de uso sin ajustar utilizamos la siguiente tabla:

Tipo de CU	Descripción	Peso	Cantidad de CU	Total
Simple	El caso de uso tiene de 1 a 3 transacciones.	5	3	15
Medio	El caso de uso tiene de 4 a 7 transacciones.	10	6	60
Complejo	El caso de uso tiene más de 8 transacciones.	15	5	75

Una transacción es cada interacción del usuario que necesita una respuesta del sistema, y que está representada por uno o más pasos del flujo de eventos principal del Caso de Uso, pudiendo existir más de una transacción dentro del mismo Caso de Uso.

Por tanto:

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

$$UUCW = 150$$

Finalmente, los Puntos de Casos de Uso sin ajustar resultan

$$UUCP = UAW + UUCW$$

$$UUCP = 6 + 150$$

$$UUCP = 156$$

Paso 2. Cálculo de Puntos de Casos de Uso ajustados.

Una vez que se tienen los Puntos de Casos de Uso sin ajustar, se debe acoplar este valor a la siguiente ecuación:

$$UCP = UUCP * TCF * EF$$

Donde:

UCP: Puntos de casos de uso ajustados.

UUCP: Puntos de casos de uso sin ajustar.

TCP: Factor de complejidad técnica.

EF: Factor de ambiente.

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

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

El Factor de complejidad técnica se calcula mediante la siguiente ecuación:

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

TCF = 0.6 + 0.01 * 58.5
TCF = 0.6 + 0.585
$TCF = 1.185$

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

Factor	Descripción	Peso	Valor asignado	Total
E1	Familiaridad con el modelo de proyecto utilizado	1,5	4	6
E2	Experiencia en la aplicación	0,5	3	1.5
E3	Experiencia en la orientación a objetivos.	1	4	4
E4	Capacidad del analista líder.	0,5	5	2.5
E5	Motivación.	1	5	5
E6	Estabilidad de requerimientos	2	4	8
E7	Personal Part–Time	-1	2	-2
E8	Dificultad del lenguaje de programación	-1	4	-4

El Factor de ambiente se calcula mediante la siguiente ecuación:

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

EF = 1.4 – 0, 03 * 21
EF = 1.4 – 0.63
$EF = 0.77$

Finalmente, los Puntos de Casos de Uso ajustados resultan:

UCP = UUCP * TCF * EF
UCP = 156 *1.185 * 0.77
$UCP = 142.34$

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

El esfuerzo en horas-hombre viene dado por:

$$E = UCP * CF$$

Donde:

E: Esfuerzo estimado en horas hombres.

UCP: Punto de casos de usos ajustados.

CF: Factor de conversión.

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

Este método proporciona una estimación del esfuerzo en horas-hombre contemplando sólo el desarrollo de la funcionalidad especificada en los casos de uso.

En este caso se puede decir que:

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

$E = 142.34 * 20$

$E = 2846.84$ Horas-Hombre

Paso 4. Estimación de esfuerzo a través de los puntos de casos de uso.

Para una estimación completa de la duración total del proyecto, se le agrega a la estimación del esfuerzo obtenida por los Puntos de Casos de Uso, las estimaciones de esfuerzo de las demás actividades relacionadas con el desarrollo de software.

Para ello planteamos el siguiente criterio que estadísticamente se considera aceptable. El criterio plantea la distribución del esfuerzo entre las diferentes actividades de un proyecto, según la siguiente aproximación:

Actividad	Porcentaje %	Horas-Hombres
Análisis	10	711,71
Diseño	20	1423,42
Implementación	40	2846,84
Pruebas	15	1067,56
Sobrecarga (otras actividades)	15	1067,56
Total	100	7117,11

Si E_T = **7117,11 horas-hombre** cada mes tiene como promedio 240 horas, eso daría un E_T = **29,654 mes-hombre.**

Esto quiere decir que 1 persona puede realizar el problema analizado en 30 meses.

Podríamos decir que si una persona puede realizar el trabajo en 30 meses aproximadamente y el equipo está compuesto por 4 personas se concluiría diciendo que el trabajo se puede realizar en aproximadamente 7 meses y medio aunque esta deducción matemática no es recomendable.

Con este criterio, y tomando como entrada la estimación de tiempo calculada a partir de los Puntos de Casos de Uso, se pueden calcular las demás estimaciones para obtener el costo total del proyecto.

-Costo del Proyecto.

Se asume como salario promedio mensual $50, debido a que los integrantes del equipo son estudiantes.

CHM = 4 * Salario Promedio

CHM = 200.00 $/mes

Costo = CHM * E_T

Costo = 200.00 * 7.50

Costo=$ 1500.00

5.4 Beneficios tangibles e intangibles

El Gestor de Recuperación Dinámicas no es un producto con fines comerciales, aunque puede adjuntarse por sus características a cualquier sistema que necesite recuperar información, así como obtener reportes. Su principal objetivo es resolver uno de los problemas actuales que existen en el mundo que es recuperar dinámicamente la información almacenada en una base de datos relacional.

El problema de la recuperación de información es de mucha actualidad no solo en el MINFAR, sino a nivel mundial, es por tanto RD una solución a esa disyuntiva, da respuestas a sus usuarios en el desarrollo de las distintas actividades relacionadas con la búsqueda y recuperación de información necesaria para el proceso de toma de decisiones.

El beneficio fundamental del sistema es contar con una aplicación Web flexible, dinámica y de interfaz agradable que permita ver, guardar y realizar reportes de una forma más precisa y en el menor tiempo posible, accediendo a datos de su interés.

Por tanto, los beneficios inmediatos son generalmente intangibles:

1. Ahorro de tiempo en la búsqueda de cualquier recurso de información necesario para cualquier área de interés.

2. Evita la necesidad de agrupar la información para que esta sea accesible, es decir, se puede quedar donde esta y de esta manera no hay necesidad de compras de nuevos recursos para llevar a cabo una tarea de esta magnitud.

3. A partir de un solo sistema se puede tener acceso a cualquier tipo de información, sin necesidad de conocer aspectos como: protocolo necesario para su acceso, nombre exacto del recurso, localización exacta del recurso, formato exacto del recurso, lenguaje exacto del recurso, etc.

4. Fácil procesamiento de la información y obtención dinámica de reportes en cualquier momento.

5. Centralización de los mecanismos de búsqueda para el MINFAR.

5.5 Análisis de costos y beneficios

El desarrollo de este sistema no supone grandes gastos de recursos, ni tampoco de mucho tiempo; la base de datos que contiene la organización de la información puede ser alojada en los Sistemas de Gestión existentes en la entidad ya que los mismos tienen buenas prestaciones y acceso rápido. La tecnología utilizada para el desarrollo del sistema es totalmente libre, por tanto no hay que incurrir en gastos en el pago de licencias de uso. El sistema es portable por lo que un cambio de plataforma para la implantación del mismo es viable y factible, y no hay que incurrir en muchos cambios debido a la estructuración en capas de los procesos que se diseñaron.

5.6 Valoración de sostenibilidad

Si analizamos en detalle cada una de las implicaciones que traerá la realización de un sistema informático, nos percatamos que repercute en lo económico y en lo social.

Cuando hablemos de un sistema informático, debemos estar convencidos que este repercutirá de forma positiva y negativa en el personal que esté en intercambio con él. En este caso se analizan cada uno de los impactos que se ocasionaron con el desarrollo e implantación de la aplicación, los cuales están estrechamente vinculados.

El impacto económico es muy seguido por estadistas y especialistas en el tema informático. En relación con la aplicación desarrollada, podemos destacar que con la implantación del sistema, la entidad no tiene que mejorar el equipamiento, y con ello no tiene gastos económicos, pues el gestor fue realizado en función de los

requerimientos técnicos que poseían, no es necesaria la contratación de personal para el trabajo con éste, ya que el mismo actualmente dispone de los conocimientos necesarios para su uso.

Con este software no se pretende la eliminación de especialistas en las distintas áreas, pues a pesar del nivel de informatización logrado, la mejora en los tiempos de respuestas, la no presencia de especialistas en todo momento por la facilidad de estar habilitado el sistema para funcionar en Red, sí son necesarios los especialistas para el control de la información y el procesamiento detallado de los resultados que se obtienen.

5.7 Conclusiones

A lo largo de este capítulo se ha dedicado la exposición a los aspectos que de una forma u otra influyen en la ejecución o no de este proyecto. Se ha detallado de una forma clara los costos a incurrir, los recursos materiales necesarios, los recursos humanos implicados, el tiempo de desarrollo, su planificación y los beneficios que aporta el producto en cuestión.

Es menester señalar nuevamente las potencialidades, que sobre la base del prestigio de la educación y la sociedad cubana, tienen productos de este tipo para comercializarse en cualquier lugar del mundo y obtener de esta forma beneficios tangibles de forma rápida, objetivo primordial de la industria cubana del software y de la que también formamos parte importante.

CONCLUSIONES

A modo de conclusión podemos decir que, con la realización de este proyecto, hoy contamos con una vía más fácil, rápida y económica para familiarizar a los usuarios con los nuevos conceptos de recuperaciones dinámicas de la información.

Nos queda claro que cualquier esfuerzo por progresar en la calidad de la automatización de nuestra sociedad pasa primeramente por una mejora en la forma de educación.

En la realización de este trabajo se investigó de forma exhaustiva los procesos que se llevan a cabo para realizar la recuperación, áreas de conocimiento de ingeniería de software, metodologías ágiles y soluciones prácticas de desarrollo.

Con todos los problemas que actualmente existen en la entidad MINFAR a la hora de realizar las recuperaciones, debemos señalar que el surgimiento de esta aplicación trae consigo incontables beneficios, citando como principales: la estabilidad y actualidad de la información que en ella se manipula; así como la integración de la información de los distintos departamentos, áreas, etc.; trayendo esto consigo el logro de un eficiente manejo de la información.

El sistema se desarrolló siguiendo la metodología RUP, y se utilizaron representaciones para la modelación de todas las fases del proyecto. El análisis y diseño fue desarrollado en aproximadamente 3 meses y el costo total es de $ 1500.00 pesos.

El sistema resultante está provisto de un ambiente cómodo, fácil de entender, que cumple los estándares de diseño y utiliza técnicas modernas de programación orientada a objetos.

Se cumplieron satisfactoriamente los objetivos propuestos para el trabajo y se analizó el proceso de recuperación como elemento importante en el desarrollo de un software, así como las tendencias actuales en el área. Además se expuso una breve reseña sobre algunos sistemas de recuperación existentes.

RECOMENDACIONES

Sin duda no es éste un trabajo estático; los modelos y la solución que se propone habrán de revisarse y actualizarse periódicamente para reflejar los cambios que tan dinámicamente se dan en las disciplinas abordadas, ha quedado claro que la propuesta es sólo la primera fase de un proyecto que puede ser mucho más ambicioso. Por ello se recomienda:

✓ La adaptación de las cuestiones planteadas en nuestro trabajo a las diferentes aplicaciones Web que hoy se utilizan en la entidad MINFAR.

✓ Instamos a todos quienes crean poder colaborar con el desarrollo de este software expresen sus ideas y colaboren con los autores en la regulación e implantación del mismo.

✓ Las tareas de organización y recuperación son actividades conscientes por lo que cada vez es más importante la superación del trabajador de la información.

✓ La información se ha convertido en un recurso vital dentro de cualquier organización por lo tanto se deben hacer continuas referencias e investigaciones al valor de la gestión de información como factor crítico del éxito.

✓ Estudiar con más profundidad las necesidades de los usuarios para que estos estén representados en los sistemas de recuperación de información porque es en función de satisfacer sus demandas informativas que trabajamos.

✓ Perfeccionar el mecanismo que se utiliza para realizar las consultas teniendo en cuenta el tratamiento del lenguaje utilizado en la formulación de la búsqueda.

✓ Hacer este trabajo extensible a todas las aplicaciones Web que necesiten recuperar información y que utilicen como SGBD PostGreSQL.

BIBLIOGRAFIA

Pressman, R. *Software Engineering. A Practitioner's Approach*. Fourth Edition. McGraw – Hill. USA, 1999.

Booch, G., Rumbaugh, J., Jacobson, I. *El Lenguaje Unificado de Modelado*. Addison-Wesley. 1999.

Larman, C. *UML Y PATRONES, Introducción al análisis y diseño orientado a objetos*. La Habana. Cuba 2004.

Booch, G., Rumbaugh, J., Jacobson, I. *El Proceso Unificado de Desarrollo de Software*. La Habana. Cuba 2004.

Franco, J A. *UML en acción. Modelando Aplicaciones Web*. La Habana. Cuba 2006.

Muñoz, J. *Un Framework basado en OSGi para el Desarrollo de Sistemas Pervasivos. mayo 2006.*

Méndez G. *Construcción de Aplicaciones Web con UML. Diseño. Mayo 2006.*

Méndez G. *Construcción de Aplicaciones Web con UML.* Conceptos Generales. Mayo 2006.

Mexica J. *Un patrón arquitectónico para la creación de cursos WBT.mayo 206.*

Herrera R A., Caldera R J., Martínez M. tema: *Análisis y Diseño de Sistemas con el Lenguaje de Modelaje Unificado (UML). Univeridad Catolica Redemptoris Mater".* Proyecto Monografico. Managua, abril de 1999.

Soto Lopez N M., Saborit Ramírez Y. *Sistema de Catalogación y Recuperación de Recursos de Información, HUBBLE,* trabajo de diploma para optar por el titulo de Ingeniería en Informática. Instituto Superior Politécnico "José Antonio Echeverría". Ciudad de la Habana, junio del 2004.

Espinosa Hidalgo A. *Sistema para la Administración Unificada de usuarios.* Trabajo de diploma para optar por el titulo de Ingenieria en Informatica. Instituto Superior Politécnico "José Antonio Echeverría". Ciudad de la Habana, junio del 2004.

Franco Navarro, J. *Acceso a Datos Objeto – Relacional; Patrones y Mecanismos. EJB - CMP, una solución "Alguien lo hace por mí" sobre J2EE.* Consultado en mayo del 2006.

Grupo de Ingeniería del software, Universidad de Sevilla, *Introducción al Análisis de Requisitos.* Publicado en Octubre del 2005. Consultado en mayo del 2006.

Grupo de Ingeniería del software, Universidad de Sevilla, *Elicitación de Requisitos: modelado del Negocio. (Diagramas de Actividades).* Publicado en Octubre del 2005. Consultado en mayo del 2006.

Grupo de Ingeniería del software, Universidad de Sevilla, *Documentación de Casos de Uso. Publicado en* Octubre del 2005. *Consultado en mayo del 2006.*

Grupo de investigación Ingeniería de Software. Universidad EAFIT. *Tertulia de Ingeniería de Software. La Importancia de la Arquitectura en el desarrollo de software de calidad.* Publicado febrero 17 de 2005. Consultado mayo 26, 2006.

Isla Monte, j. *Modelado Estructural de Patrones de Diseño.* Cadiz, España, 2003.

Peralta M. *Estimación del esfuerzo basada en casos de usos.pdf.* http://www.itba.edu.ar/capis/webcapis/planma.html , mayo del 2006.

http://msdn.microsoft.com/architecture/ (mayo 10, 2006)
http://www.dm.univaq.it/~muccini (mayo 10, 2006)

http://www.sparxsystems.cl/uml-tutorial.html (mayo 28, 2006)

http://www.sparxsystems.cl/resources/uml_datamodel.html (mayo 28, 2006)

http://www.sparxsystems.cl/resources/developers/uml_patterns.html (mayo 28 2006)

http://www.mozilla-europe.org/es/about/ (mayo 22, 2006)

http://kusor.net/article/584/control-de-versiones-con-subversion (mayo 22, 2006)

http://www.capisol.net/formacion/mod/book/view.php?id=52&chapterid=25 (mayo 22,2006)

http://www.lug.fi.uba.ar/sistemas-de-control-versiones.mht (mayo 22, 2006)

http://www.monografias.com/trabajos15/estimacion-hipermedia/estimacion-hipermedia.shtml#INTRO (mayo 23, 2006)

http://es.wikipedia.org/wiki/Patr%C3%B3n_de_dise%C3%B1o (mayo 29. 2006)

http://es.tldp.org/Tutoriales/doc-modelado-sistemas-UML/multiple-html/x219.html (mayo 29, 2006)

http://www.elrincondelprogramador.com/articulos/puntuar.asp?puntos=4&id=45">4ptos.</A> (mayo 29, 2006)

http://www.fi.uba.ar/~dmontal/ (mayo 29, 2006)

http://www3.uji.es/~mmarques/f47/apun/node79.html (junio 4, 2006)

http://www.monografias.com/trabajos10/recped/recped.shtml#intro(marzo 14, 2006)

http://www_php_net-pgsql.htm/ (marzo15, 2006)

GLOSARIO DE TERMINOS

Gestor: Programa que se encarga de una tarea específica, como el gestor de ficheros, el de impresión, o el de memoria, dentro del sistema operativo de un ordenador.

Recuperación de información: es la acción que permite gestionar datos para que puedan someterse a un proceso de aprovechamiento.

Reporte: Informe que se emite o presenta con base en la realización de una actividad o tarea.

Software: Conjunto de instrucciones escritas en un determinado lenguaje, que dirigen a un ordenador para la ejecución de una serie de operaciones, con el objetivo de resolver un problema que se ha definido previamente.

Sistemas de Gestión de Base de Datos: Herramienta que proporciona una interfaz entre los datos almacenados y los programas de aplicación que acceden a éstos y que se caracteriza fundamentalmente por permitir una descripción centralizada de los datos y por la posibilidad de definir vistas parciales de los mismos para los diferentes usuarios.

Actor: Abstracción de las entidades externas a un sistema, subsistemas o clases que interactúan directamente con el sistema. Un actor participa en un caso de uso o en conjunto coherente de casos de usos para llevar acabo un propósito global.

Arquitectura: Estructura organizativa de un sistema que incluye su descomposición en partes, su conectividad, mecanismos de interacción y principios de guía que proporciona información sobre el diseño del mismo.

Casos de uso: Especificación de las secuencias de acciones, incluyendo secuencias variantes y secuencias de errores, que pueden ser efectuadas por un sistema, subsistema o clase por interacción con autores externos

Clase: Descriptor de un conjunto de objetos que comparten los mismos atributos., operaciones, métodos, relaciones y comportamientos. Una clase representa un concepto dentro del sistema que se está modelando.

Componente: Una parte física reemplazable de un sistema que empaqueta su implementación, y es conforme a un conjunto de interfaces a las que proporciona su realización.

Interfaz: Un conjunto de operaciones que posee un nombre y que caracteriza el comportamiento de un elemento.

Modelo: Es una abstracción semánticamente completa de un sistema.

Paquete: Término que denota un mecanismo de propósito general para organizar en grupos los elementos. Se pueden anidar dentro de otros paquetes, y en el pueden aparecer tanto elementos del modelo como diagramas.

Requisito o Requerimiento: Una característica, propiedad o comportamiento que se desea para el sistema. Sistema: Colección de unidades conectadas que se organiza para lograr un propósito. El sistema es el "modelo completo".

Subsistema: Paquete de elementos que se tratan como una unidad, incluyendo una especificación de todo el contenido del paquete, que se trata como una unidad coherente. Se modela simultáneamente como paquete y para clase. Tiene un conjunto de interfaces que describen su relación con el resto del sistema y las circunstancias en que se puede utilizar.

ODBC son las siglas de Open DataBase Connectivity, que es un estándar de acceso a Bases de Datos desarrollado por Microsoft Corporation.

ANEXOS

ANEXO I. FASES Y ARTEFACTOS DEL PROCESO UNIFICADO DE DESARROLLO.

FASE	SUBPRODUCTOS	
PREPARACIÓN INICIAL	1.-	Alcance del Sistema
		1.- Lista de Características
		2.- Modelo del Dominio o Modelo del Negocio (1ª. versión)
		3.- Modelo de Casos de Uso, de Análisis y de Diseño (1ª. versión)
	2.-	4.- Requerimientos Suplementarios (1ª. Versión) Arquitectura Inicial
	3.-	Lista Inicial de Riesgos (riesgos críticos más importantes) y Lista Priorizada de los Casos de Uso
	4.-	Prototipo para Validación de Conceptos (prototipo de descarte)
	5.-	Entorno de Desarrollo Configurado (proceso y herramientas) inicial
	6.-	Plan Inicial del Proyecto.
	7.-	Caso Inicial del Negocio (1ª. versión) (contexto del negocio y criterios de éxito) (costo, tiempos, calidad, utilidades)
PREPARACIÓN DETALLADA	1.-	Contexto del Sistema (Modelo del Dominio o Modelo del Negocio preferiblemente completo)
	2.-	Captura del 80% de los Requerimientos Funcionales
		1.- Modelo de Casos de Uso (aprox. el 80%) y Modelo de Análisis (realización de los casos de uso más significativos)
		2.- Modelo de Diseño, Modelo de Despliegue y Modelo de Implementación (menos del 10%)
		3.- Niveles para los Atributos de Calidad y Requerimientos Suplementarios Actualizados
		4.- Manual Preliminar de Usuario
	3.-	Arquitectura de Referencia (línea de base) (descripción de las vistas arquitecturales de los modelos del sistema)
	4.-	Lista Actualizada de Riesgos (críticos y significativos) y Riesgos Críticos Mitigados
	5.	Plan del Proyecto para las fases de Construcción y Transición
	6.	Entorno de Desarrollo Adecuado (proceso y herramientas)
	7.	Caso del Negocio Completo (y "Contrato" o declaración del negocio)
CONSTRUCCIÓN	1.-	Modelos Completos (Casos de Uso, Análisis, Diseño, Despliegue e Implementación)
	2.-	Arquitectura Integra (mantenida y mínimamente actualizada)
	3.-	Riesgos Presentados Mitigados
	4.-	Plan del Proyecto para la fase de Transición
	5.-	Manual Inicial de Usuario (con suficiente detalle)
	6.-	Prototipo Operacional – beta
	7.-	Caso del Negocio Actualizado
TRANSICIÓN	1.-	Prototipo Operacional
	2.-	Documentos Legales
	3.- 4.-	Caso del Negocio Completo Línea de Base del Producto completa y corregida que incluye todos los modelos del sistema
	5.-	Descripción de la Arquitectura completa y corregida
	6.-	Manuales para Usuario Final, Operador y Administrador del Sistema, y Materiales para Entrenamiento

Tabla I Fases y artefactos del Proceso Unificado de Desarrollo.

ANEXO II. TABLA DE CONTENIDO DE LA BASE DE DATOS.

Tabla BD	Clasificación	Descripción
dat_sistemas	Informativa	Esta tabla almacena los sistemas que crea el configurador como parte de la organización que se le da a la información en un lugar.
dat_subsistemas	Informativa	Esta tabla almacena los subsistemas que crea el configurador y que están en correspondencia con el sistema al que pertenece.
dat_campos	Informativa	Esta tabla almacena los campos que selecciona el configurador de la base de datos los cuales se utilizarán para recuperar información de la misma.
dat_relaciones	Informativa	Esta tabla almacena las nuevas relaciones que crea el configurador y que como necesidad de la recuperación necesitará el recuperador.
dat_dependencias	Informativa	Esta tabla almacena las dependencias que crea el configurador como necesidad de la recuperación.
dat_tablas	Informativa	Esta tabla almacena las tablas que contienen los diferentes esquemas de la base de datos de la que se va a recuperar y que intervienen en la recuperación. En ella se almacena la clasificación de estas tablas en informativas, especialización o nomenclador.
dat_camposlistados	Informativa	Esta tabla almacena los campos que selecciona el recuperador para recuperar por listados.
dat_camposestadisticas	Informativa	Esta tabla almacena los campos por listados que selecciona el recuperador para recuperar por estadísticas.
dat_infcondiciones	Informativa	Esta tabla almacena las condiciones por estadísticas y listados que son definidas por el recuperador y que tiene que cumplir la recuperación.
dat_informe	Informativa	Esta tabla almacena los datos de los diferentes formatos de los informes que son definidos.
dat_titulo	Informativa	Esta tabla almacena los títulos de los diferentes informes que son creados.
dat_grupo	Informativa	Esta tabla almacena la agrupación por campos que tiene el reporte final y que es definido con anterioridad.
dat_orden	Informativa	Esta tabla almacena el orden en el que aparecerán los campos en el reporte.
dat_personalizar	Informativa	Esta tabla almacena los campos que son personalizados cuando se define un campo por estadísticas.
dat_percondiciones	Informativa	En esta tabla se almacenan las condiciones que son definidas por el recuperador en la personalización de los campos escogidos.

Tabla II Contenido de la Base de Datos.

ANEXO III. TABLA DE ASIGNACION DE TAREAS Y RECURSOS.

Id	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras	Nombres de los recursos
1	**Modelamiento del negocio**	**16 días?**	**mié 01/02/06**	**mié 22/02/06**		**Insumos de oficina;Computad**
2	Identificación de Procesos de Negocio	6 días	mié 01/02/06	mié 08/02/06		
3	Identificación de Roles del Entorno del Negocio	3 días	jue 09/02/06	lun 13/02/06	2	
4	Especificación de Reglas del Negocio	1 día?	mar 14/02/06	mar 14/02/06	3	
5	Obtencion de los modelos principales del negocio	3 días	mié 15/02/06	vie 17/02/06	4	
6	Descripción de los Casos de Uso del Negocio	3 días	lun 20/02/06	mié 22/02/06	5	
7	**Captura de requisitos**	**30 días**	**jue 16/02/06**	**mar 28/03/06**		
8	Obtención requerimientos funcionales del sistema	5 días	jue 16/02/06	mié 22/02/06		
9	Obtención requerimientos no funcionales del sistema	5 días	mié 22/02/06	mar 28/02/06		
10	Encontrar los actores y casos de uso	2 días	mar 28/02/06	mié 01/03/06		
11	Obtención del modelo de caso de uso del sistema	5 días	jue 02/03/06	mié 08/03/06	10	
12	Descripción de los casos de usos	15 días	jue 09/03/06	mar 28/03/06		
13	**Análisis y Diseño**	**48 días**	**sáb 18/03/06**	**mar 23/05/06**		
14	Identificación de clases del análisis por CU	6 días	jue 30/03/06	jue 06/04/06		
15	Obtención del diagrama de clases del análisis por CU	4 días	vie 07/04/06	mié 12/04/06	14	
16	Obtencion del diagrama de colaboración por CU	4 días	jue 13/04/06	mar 18/04/06	15	
17	Obtención de los mecanismos de diseño	30 días	mié 05/04/06	mar 16/05/06		
18	Identificación de las clases del diseño por CU	10 días	mié 19/04/06	mar 02/05/06	16	
19	Obtención del diagrama de clase del diseño por CU	5 días	mié 03/05/06	mar 09/05/06	18	
20	Obtención del diagrama de secuencia por realizaciones de CU	10 días	mié 10/05/06	mar 23/05/06	19	
21	Obtención del diagrama de clases persistentes	4 días	sáb 18/03/06	mié 22/03/06		
22	**Implementación**	**115 días**	**sáb 18/03/06**	**jue 24/08/06**		
23	Definir arquitectura	5 días	sáb 18/03/06	jue 23/03/06		
24	Diseñar componentes	5 días	mié 19/04/06	mar 25/04/06	16	
25	Diseñar Base de Datos	5 días	jue 23/03/06	mié 29/03/06	21	
26	Obtener diagrama de componentes	1 día	mié 26/04/06	mié 26/04/06	24	
27	Obtener diagrama de despliegue	1 día	vie 24/03/06	vie 24/03/06	23	
28	Generacion de codigos a partir de diagrama de clases	1 día	mié 03/05/06	mié 03/05/06	18	
29	Implementar los elementos de diseño	60 días	mié 24/05/06	mar 15/08/06	20	
30	Integrar los resultados en un sistema ejecutable	7 días	mié 16/08/06	jue 24/08/06	29	
31	**Prueba**	**10 días**	**vie 25/08/06**	**jue 07/09/06**		
32	Encontrar y documentar los defectos del software	10 días	vie 25/08/06	jue 07/09/06		
33	Probar que el software trabaje como fue diseñado	10 días	vie 25/08/06	jue 07/09/06		
34	Probar los requisitos que debe cumplir el software	10 días	vie 25/08/06	jue 07/09/06		
35	Probar que los requisitos fueron implementados correctamente	10 días	vie 25/08/06	jue 07/09/06		

Figura I. Tareas de asignación de tareas y recursos.

ANEXO IV. DIAGRAMA DE GANT

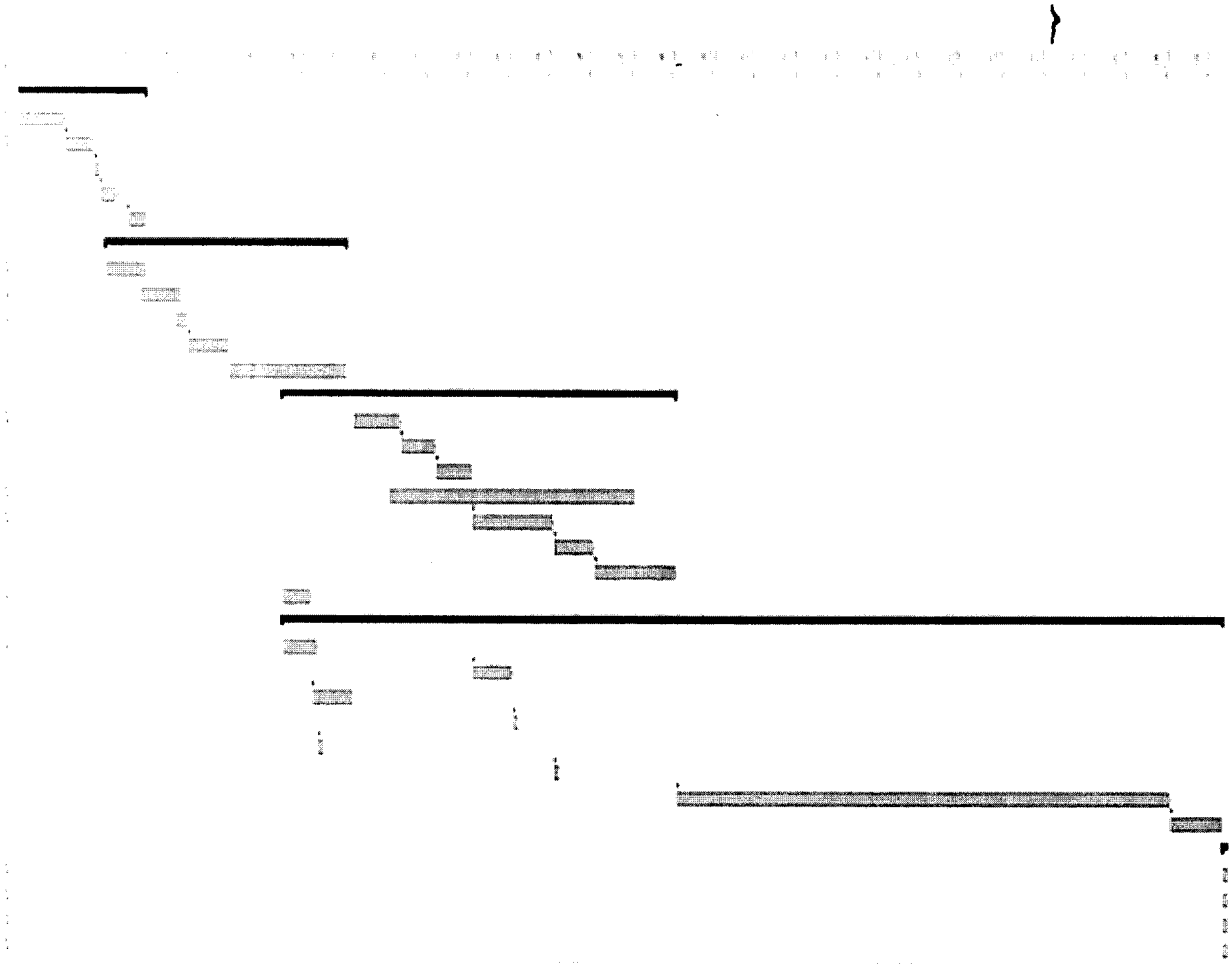


Figura II. Diagrama de Gant correspondiente a la asignación de tareas y recursos.