

República de Cuba

Universidad de las Ciencias Informáticas



Facultad 2

*Sistema de Gestión de Emergencias de Seguridad Ciudadana.*

*“Módulo de Edición de Reportes”.*

Trabajo de Diploma

Presentado para optar por el título de  
Ingeniero en Ciencias Informáticas

Autor (es): Evelyn Guindo Betancourt

Yadier Mesa Pérez

Tutor (es): Ing. Yordanis Tornos Medina

Ing. Javier Fernández Cruz

“Año 53 de la Revolución”

Ciudad de la Habana, Cuba. 20 de junio de 2011.

# *Declaración de Autoría*

Por este medio declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los \_\_\_\_ días del mes de \_\_\_\_\_ del año \_\_\_\_\_.

**Evelyn Guindo Betancourt**

\_\_\_\_\_  
Firma del Autor

**Yadier Mesa Pérez**

\_\_\_\_\_  
Firma del Autor

**Ing. Yordanis Tornes Medina**

\_\_\_\_\_  
Firma del Tutor

**Ing. Javier Fernández Cruz**

\_\_\_\_\_  
Firma del Tutor

## *Dedicatoria*

*Quiero dedicar este trabajo a los seres más queridos que tengo en mi vida: a mi mamá Vilma Betancourt Colás por ser la madre más linda del mundo, a mi papá Francisco Guindo Iznaga por ser tan especial, a mi hermana querida por ser hermana y amiga al mismo tiempo y por traer al mundo dos preciosuras Marelis y Melissa, para ellas también esta dedicatoria.*

*Dedicárselo a toda mi familia por brindarme su apoyo y estar pendientes de mí en todo momento, disculpen las tantas carreras que les he hecho hacer.*

*Evelyn*

## *Dedicatoria*

*A mis padres Efraín y María Luz que con mucho amor y entrega han sabido educarme correctamente y siempre han estado ahí para lo que necesite. A ellos les debo la vida y todo lo que soy. Muchas gracias por haber confiado en mí, espero no defraudarlos nunca.*

*A mi hermana Yadialys por ser la inspiración de muchas cosas en mi vida, te quiero.*

*A mi familia en general por brindarme su apoyo.*

*Yadier*

# *Agradecimientos*

*Insuficientes son estas líneas para agradecerles a tantas personas lindas que han marcado mi vida de estudiante.*

*Primero que nada gracias a Dios por hacer realidad este sueño. Agradecerles a mami y a papi por traerme al mundo y hacer de mi vida una gran felicidad.*

*Agradecerle al Comandante por crear este proyecto futuro en el que hoy, yo me gradúo.*

*Gracias a mi compañero de tesis Yadier por entenderme y aguantarme todo este tiempo. “Lo logramos...”.*

*A mi prima Caty por estar pendiente de mí y mis cosas, “te quiero mucho primi”.*

*A mis amigas de la infancia Vanessa y Greter por darme tanta alegría y fiestas; por demostrarme que puedo contar con ustedes pase lo que pase.*

*A mis compañeras de cuarto Yaineris, Yulissa, Dailenis y Gleidis muchas gracias por estar ahí cuando más lo necesité, gracias por sus consejos y comprensión. A todo el piquete del 4103 al mando de Yanet.*

*Gracias a mis primeras amistades en la UCI: Lisbet y Dairys; por hacer de ese primer año un momento inolvidable en mi vida.*

*Agradecerle a Wilson por llegar a mi vida hace muy poco, pero en tan poco tiempo me has demostrado tu apoyo, gracias por todo nene.*

*A Yari y Dayana que llegaron a mi vida para quedarse, las quiero y saben que pueden contar conmigo para lo que sea. A Rainer por escucharme y comprenderme.*

*A todos los profes que me han ayudado a realizar este sueño: Yovanotti, Yadira, Manfred, Michael, Idalys y Adrian; muchas gracias por su comprensión, apoyo y paciencia. Agradecimiento a mis tutores.*

*Gracias a todos los que deseaban al igual que yo ver este sueño realidad.*

*Evelyn*

# *Agradecimientos*

*A todas aquellas personas que han estado presentes a lo largo de esta carrera tanto profesores como compañeros de aula.*

*A Danae por demostrarme su incondicionalidad en todo momento, gracias por ser mi amiga.*

*A mi compañera de tesis por todo el esfuerzo y sacrificio realizado durante toda la carrera y en especial durante esta última etapa.*

*A Yadira por ser tan paciente y generosa conmigo, por estar ahí siempre que la necesité, siempre serás mi camaroncito duro.*

*A Manfred, Adrián, Michel, Idalis por brindarme sus conocimientos y responder sin vacilar ante una duda.*

*A mis amigos de fiestas: Chao, Yanelis, Danelis, Yanesota, Ingris, Dailenis, Enrique, Leo, Jorge.*

*A todas las personas del apto 4104: Yaineris, Gleidis, Yula, Yanko.*

*A mis amistades desde preescolar a quienes aprecio muchísimo y con quienes realicé mis primeras trastadas.*

*A mis compañeros y amigos del Preuniversitario Carlos Marx "Lo logré".*

*A la Universidad por formarme como Ingeniero y por darme la posibilidad de conocer a muchas personas maravillosas.*

*A grandes personas que no están físicamente entre nosotros pero hubiesen disfrutado mucho de este momento.*

*A mis abuelos, tíos, tías, primos y primas y en especial a Yiri por quererme mucho y estar pendiente de mí, gracias chuchi.*

*A mis tutores por la ayuda brindada.*

*A todas aquellas personas que quisieron verme algún día graduado.*

*Yadier*

## *Resumen*

La seguridad ciudadana es de vital importancia para cualquier nación porque implica la protección al ser humano contra la violencia, el hurto y cualquier otro tipo de incidente. Para que un país funcione correctamente teniendo todas sus esferas sincronizadas y su población en constante armonía y seguridad deben estar controladas todas estas situaciones de emergencia. El Gobierno Bolivariano de Venezuela no ha olvidado este tema que es de vital importancia, la creación de los Centros de Gestión de Emergencias 171 por parte del Ministerio del Poder Popular para Relaciones Interiores y Justicia es una de las medidas encaminadas a mantener altos niveles de seguridad en cada región del país.

Con el surgimiento de estos centros surge el Sistema de Gestión de Emergencias de Seguridad Ciudadana (SIGESC), encargado de gestionar los procesos necesarios para la atención a las emergencias reportadas a los Centros de Gestión de Emergencias 171.

El objetivo de esta investigación es desarrollar un Módulo de Edición de Edición de Reportes, específicamente un Editor de Consulta Simple como parte de SIGESC para el control y gestión de la información estadística que incluya funcionalidades de edición de procedimientos, además de realizar el diseño de funcionalidades que permitan la creación y aplicación de plantillas a reportes estadísticos y la utilización de varios orígenes de datos durante el diseño de los reportes.

De esta manera, los Centros de Gestión de Emergencias 171 que decidan implantar el SIGESC, contarán con un módulo que permitirá la edición de consultas que finalmente se utilizarán en la confección de los reportes estadísticos. Brindarán además información del funcionamiento del centro de manera tanto general como específica, posibilitando el uso de esta información de forma directa en la toma de decisiones y brindando indicadores referentes a la oportuna y correcta atención de las solicitudes de emergencia por parte de estos centros.

## Índice

Introducción .....	1
Capítulo 1. Fundamentación Teórica .....	5
1.1 Introducción.....	5
1.2 Centro de Gestión de Emergencia.....	5
1.3 SIGESC.....	6
1.4 Entorno Integrado de Gestión Estadística .....	7
1.5 Editor de Procedimientos.....	8
1.5.1 Editores de Consultas Simples.....	8
1.5.2 Gestionar Plantillas y Diseño de Reportes.....	9
1.6 Metodología Seleccionada Rational Unified Process.....	11
1.7 Lenguaje Unificado de Modelado v2.0.....	12
1.8 Procedimiento de Modelado de Procesos IDEF0 .....	12
1.9 Tecnología y Herramientas de Desarrollo seleccionadas .....	13
1.9.1 Visual Paradigm v3.4.....	13
1.9.2 Framework DMAX .....	14
1.9.3 Plataforma de Desarrollo .NET .....	14
1.9.4 Entorno Integrado de Desarrollo.....	15
1.9.5 .NET Framework v1.1 .....	15
1.9.6 Lenguaje de Programación CSharp.....	16
1.9.7 Active Reports .....	17
1.9.8 Sistema Gestor de Base de Datos Oracle 10g .....	17
1.10 Conclusiones Parciales .....	18
Capítulo 2. Características del Sistema .....	19

2.1 Introducción.....	19
2.2 Problema y situación problemática .....	19
2.3 Propuesta del sistema .....	19
2.3.1 Gestionar plantillas .....	20
2.3.2 Diseñar Reportes.....	20
2.3.3 Editor.....	20
2.4 Modelo de Negocio.....	23
2.4.1 Descripción del Modelo de Negocio.....	25
2.5 Modelo de Dominio.....	26
2.5.1 Identificar las Clases Conceptuales .....	26
2.6 Arquitectura de la solución propuesta.....	28
2.7 Especificación de los requisitos de software.....	30
2.7.1 Requisitos No Funcionales .....	30
2.8 Definición de los actores.....	33
2.9 Diagrama de Paquete.....	33
2.10 Diagrama de Casos de Uso.....	35
2.11 Descripción de Casos de Uso.....	39
2.12 Conclusiones Parciales .....	39
Capítulo 3. Diseño del Sistema .....	40
3.1 Introducción.....	40
3.2 Patrones de Diseño .....	41
3.3 Diagrama de Paquetes del Diseño .....	43
3.4 Diagrama de Clases del Diseño .....	45
3.5 Conclusiones Parciales .....	46

Capítulo 4. Implementación .....	47
4.1 Introducción.....	47
4.2 Diagrama de Despliegue .....	47
4.3.1 Nodo Servidores de BD .....	48
4.3.3 Nodo PC Estadísticas .....	50
4.4 Diagrama de Componentes.....	51
4.5 Conclusiones Parciales .....	53
Conclusiones Generales.....	54
Recomendaciones .....	55
Referencias Bibliográficas.....	56
Bibliografía.....	59
Glosario de Términos.....	61

# *Introducción*

En el último decenio, la seguridad ciudadana ha pasado a ser un tema central para muchos países, instituciones y actores sociales en busca de métodos innovadores que permitan hacer frente a las amenazas contra la paz y la seguridad. La seguridad ciudadana enfatiza la protección de los individuos, de las comunidades locales y las instituciones de los desafíos internos y externos que los afectan. Por esto, la seguridad ciudadana requiere de la participación de los gobiernos a nivel nacional y local, además de una visión amplia para enfrentar las raíces que causan la violencia. El crimen, la agresividad y el conflicto siembran el miedo y la ansiedad hacia la seguridad personal dificultando el desarrollo económico transformando ciertas áreas en menos atractivas para la inversión (1).

A la llegada del presidente Hugo Rafael Chávez al poder en Venezuela el país se encontraba inmerso en serios problemas sociales tales como altos niveles de violencia, drogadicción, pobreza, insalubridad y discriminación de la mujer, por tan sólo mencionar algunos. Desde ese momento se han buscado nuevas alternativas en aras de erradicar estos problemas, demostrando democracia y preocupación del gobierno por sus ciudadanos. El mantenimiento de la seguridad ciudadana se convirtió en una de las principales misiones de todos los estados territoriales de Venezuela. El Ministerio del Poder Popular para Relaciones Interiores y Justicia (MPPRIJ) como parte de la dirección del Estado de la República Bolivariana de Venezuela promovió la modernización de los Centros de Atención de Emergencias 171 (Centro 171) en un intento por mantener el bienestar de la población en materia de seguridad ciudadana en todo el territorio nacional.

Como parte del proceso de modernización de los Centros 171 surge el Sistema de Gestión de Emergencias de Seguridad Ciudadana (SIGESC) el cual se encarga de gestionar los procesos necesarios para la atención a las emergencias reportadas a estos centros, independientemente de donde esté funcionando. El SIGESC está compuesto por una aplicación Web, un sistema de sincronización y nueve aplicaciones de escritorio dentro de las cuales se encuentra la aplicación de Estadística. La misma genera reportes estadísticos sobre la información almacenada en la base de datos del SIGESC, reportes que

apoyan la evaluación de los Centros 171 y de los órganos de seguridad ciudadana que intervienen en la atención de las emergencias.

Actualmente en los Centros 171 donde se encuentra desplegado el SIGESC, el personal responsable del trabajo con la aplicación de Estadística depende del equipo de desarrollo y/o soporte para realizar nuevos reportes que requieren de conocimientos específicos de programación y bases de datos. Los desarrolladores que forman parte del equipo, en caso de no solucionar los pedidos de reporte en un momento dado, imposibilitan al Centro 171 del trabajo con estos reportes que se desean generar. La no respuesta oportuna ante una solicitud de reporte estadístico provoca que desde el Centro 171 se deba generar la información requerida manualmente o en el peor de los casos no disponer de la misma. Otro problema a reflejarse es la no existencia de una gestión eficiente relacionada con el pedido de los reportes hacia el equipo de desarrollo y/o soporte, las vías utilizadas actualmente pueden ser varias: teléfono, correo electrónico, entre otros y a veces iterativas e insistentes.

Por otra parte cada reporte creado incluye un diseño del encabezado y pie de página con elementos comunes, pudiendo convertir estos en una plantilla y aplicarlo luego al cuerpo de los mismos sin necesidad de diseñar una y otra vez las mismas partes, agilizando el diseño de nuevos reportes.

Sumado a lo anterior, la confección de un reporte necesita tener asociado algún origen de datos al cuerpo del mismo. Actualmente la aplicación de Estadística solo permite asociar un Origen de datos por reporte o uno por cada subreporte asociado al reporte principal, imposibilitando la creación de un reporte que contenga asociado varios orígenes de datos, donde uno de ellos dependa de la ejecución de otro.

Después del análisis de la situación anterior se formula el siguiente **problema a resolver**:

Las posibilidades de la versión actual del Módulo de Edición de Reportes del SIGESC son insuficientes para generar reportes complejos.

Según el problema identificado anteriormente se plantea como **objeto de estudio** los sistemas de generación de reportes, quedando enmarcado el **campo de acción** en los sistemas de generación de reportes vinculados a soluciones de Gestión de Emergencias, para la edición de consultas simples, gestión de plantillas y asignación de varios orígenes de datos.

Para dar solución al problema existente se plantea como **objetivo general**:

Desarrollar un editor de consultas simples para incorporarlo al Módulo de Edición de Reportes del SIGESC de manera que facilite la edición de reportes complejos.

Por lo anteriormente expuesto la **idea a defender** se formula de la siguiente forma:

El desarrollo de un editor de consultas simples incorporado al Módulo de Edición de Reportes del SIGESC facilitará la creación de reportes complejos.

Para dar cumplimiento a los objetivos planteados se planificaron una serie de **tareas a desarrollar**, tales como:

1. Estudio del estado del arte acerca de las principales herramientas de edición de consultas simples, y de generación de reportes para el análisis de su funcionamiento.
2. Comprensión de la arquitectura del SIGESC para entender los basamentos arquitectónicos y de distribución del sistema.
3. Profundización en el estudio de la estructura de un compilador que sea capaz de interpretar consultas simples para validar la edición de consultas.
4. Diseño de funcionalidades que permitan gestionar plantillas para la creación de reportes.
5. Diseño de funcionalidades para asignar varios orígenes de datos a un mismo reporte.
6. Diseño de funcionalidades para editar consultas simples.
7. Implementación de un editor de consulta simple para la edición de las mismas.

Después de haber dado cumplimiento a las tareas se pretende obtener un producto de software ajustado a las necesidades de los clientes, brindando una solución adecuada al problema existente. Con esta solución los estados venezolanos que implanten el SIGESC contarán con un sistema para crear consultas, las cuales se utilizarán como parte de los reportes.

## **Métodos Científicos**

Los **métodos teóricos** se utilizan en la construcción y desarrollo de la teoría científica y en el enfoque general para abordar los problemas de la ciencia. Los utilizados en el siguiente trabajo de diploma fueron:

**Análisis y síntesis:** Posibilita el procesamiento de la información necesaria de forma particular para guiar la investigación sobre editores de consultas simples, así como de diseño de funcionalidades, sumado a esto se encuentra la relación entre cada una de las partes involucradas.

**Histórico lógico:** Permite realizar un estudio de los antecedentes y tendencias actuales de los editores de consultas así como de la gestión de plantillas y de la asignación de varios orígenes de datos a un reporte.

El presente documento consta de 4 capítulos, estructurados de la siguiente forma:

### **Capítulo 1. Fundamentación teórica**

Incluye un estado del arte del tema tratado, a nivel internacional, nacional y de la Universidad, así como las metodologías y software utilizados en la actualidad o los que se analizarán para dar solución al problema planteado.

### **Capítulo 2. Características del sistema**

Este capítulo aborda la situación problemática, la información a manejar en el sistema así como la propuesta de solución, el modelo de dominio, las especificaciones de los requisitos de software, definición de los actores, diagrama de paquetes y de casos de uso.

### **Capítulo 3. Diseño del Sistema**

Este capítulo aborda todo lo referente al diseño del Módulo de Edición de Reportes, el diagrama de paquetes que fue necesario realizar para una mejor comprensión y claridad en la modelación, los diagramas de clases del diseño asociados a cada paquete y los patrones de diseño utilizados en la solución propuesta.

### **Capítulo 4. Implementación**

Este capítulo incluye el diagrama de despliegue, así como la explicación de cada nodo presente en dicho diagrama, además contiene el diagrama de componentes.

# *Capítulo 1. Fundamentación Teórica*

## **1.1 Introducción**

Para que un sistema informático funcione correctamente necesita que sus partes y componentes estén integrados completamente, además debe estar soportado por un conjunto de herramientas bien definidas y justificadas con las cuales el equipo de desarrollo sea capaz de desarrollar el producto final.

En este capítulo se abordará el estado del arte de los diferentes sistemas de reportes existentes, la definición de algunos conceptos relevantes para el entendimiento y asimilación del problema; así como una descripción de las herramientas a utilizar en la solución propuesta. Incluye también la metodología de desarrollo de software a utilizar y los términos importantes para la comprensión del tema que se aborda.

## **1.2 Centro de Gestión de Emergencia**

Los Centros de Gestión de Emergencias son instituciones que brindan servicios integrados de seguridad y atención a emergencias de la población, en un período de tiempo dado. Estos centros reciben llamadas las 24 horas, lo que requiere la integración y coordinación de acciones para que las entidades competentes solucionen las situaciones de emergencias o solicitudes de servicios realizadas por parte de la población. Los Centros 171 mantienen un servicio de comunicaciones que garantiza una adecuada supervisión contribuyendo a mejorar la capacidad de respuesta de los organismos, brindando de esta forma una mayor seguridad. (2)

Los Centros de Gestión de Emergencia de Venezuela, mejor conocidos como Centros 171, siendo este número el modo de acceso por vía telefónica cuentan con tres áreas fundamentales: Recepción de Llamadas, Despacho y Supervisión. El área de Recepción de Llamadas se encarga de recibir todas las llamadas realizadas por la población, se clasifican y luego se envían al área de Despacho donde se coordina con las unidades y los órganos asociados que darán atención a la emergencia o servicio solicitado. El área de Supervisión se encarga de mantener el control y buen funcionamiento del Centro

171, dígase las respuestas de las solicitudes en el tiempo previsto, el cumplimiento efectivo de los turnos de trabajo, la correcta asignación de las unidades involucradas, entre otros.

Estos centros utilizan sistemas informáticos para agilizar los procesos de la atención a las emergencias y los servicios; en este sentido en el centro de Informatización para la Seguridad Ciudadana (ISEC) se desarrolla el Sistema de Gestión de Emergencias de Seguridad Ciudadana (SIGESC). (3)

### **1.3 SIGESC**

El Sistema de Gestión de Emergencias de Seguridad Ciudadana (SIGESC) es un sistema que proporciona aplicaciones informáticas con una alta integración entre ellas, las cuales permiten gestionar las actividades relacionadas con la atención de emergencias.

El SIGESC está formado por nueve aplicaciones de escritorio, una aplicación Web y un sistema de sincronización. La aplicación Web permite llevar el control de todos los organismos que coordine un Centro 171, así como de los recursos que estos ponen a disposición del mismo para la atención de las situaciones de emergencias. El sistema de sincronización está formado por la aplicación de Sincronización, la cual cuenta con un servicio de Windows y una interfaz de administración para su configuración. Esta aplicación gestiona la sincronización del estado de las solicitudes de emergencias que se atienden en un Centro 171 y de los recursos móviles asignados a cada solicitud en todas las aplicaciones que utilicen esta información. (4)

El núcleo central del sistema está compuesto por las aplicaciones Recepción de Llamadas, Despacho, Supervisión de Operadores, Supervisión de Despacho y Supervisión General. A través de estas aplicaciones se realiza el flujo completo de atención a las solicitudes de emergencias, desde que se recibe la información de la situación hasta que se le da una solución. Las aplicaciones de Administración Informática y Configuración de Operaciones son las que permiten la configuración básica necesaria para que todo el sistema funcione correctamente. (5)

Paralelamente a la gestión de las emergencias, debe existir algún mecanismo para la toma de decisiones basándose en análisis estadísticos generados a partir de diferentes criterios (rendimiento de los operadores, eficiencia en la atención de alguna solicitud en específico, por tan solo mencionar algunos).

Una vez analizada toda la información se pueden arribar a conclusiones concretas que permitan medir de alguna manera la eficiencia de cada centro y la prestación de sus servicios.

La aplicación Estadísticas está compuesta por dos módulos: Editor de Reportes y Visor de Reportes. Permite realizar análisis sobre toda la información almacenada referente a las emergencias que se atienden en un Centro 171, genera reportes que permiten ser analizados sobre el mapa digital del área de atención.

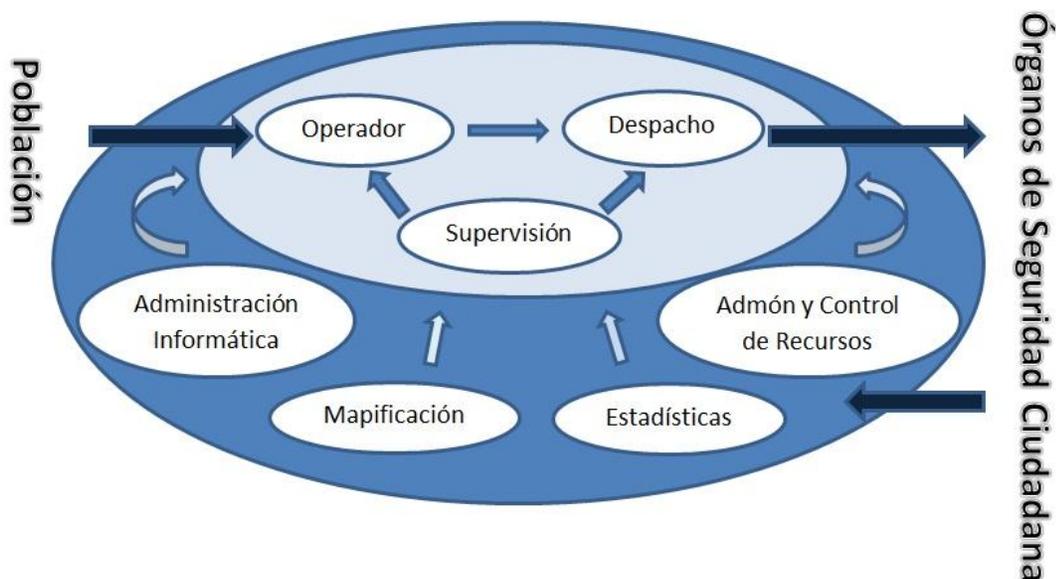


Figura 1.1 Esquema general de un Centro 171.

#### 1.4 Entorno Integrado de Gestión Estadística

El SIGESC cuenta con un gran volumen de información referente al proceso de atención de las emergencias formuladas por la población a los Centros 171. La correcta manipulación y análisis de esta información favorece al buen funcionamiento del centro y a la seguridad de la población venezolana. A partir de su estudio se puede evaluar el trabajo de los operadores, despachadores, organismos de seguridad y unidades, además de apreciar la aceptación de la población. (2)

El Módulo Visor de Reportes se encarga de visualizar reportes estadísticos que han sido definidos con anterioridad. A través de esta aplicación se puede exportar un determinado reporte a diferentes formatos:

Adobe PDF<sup>1</sup>, Archivo RTF<sup>2</sup>, Archivo TIFF<sup>3</sup>, Microsoft Office Excel, Archivo de Texto y Página Web. Es posible además imprimir los reportes visualizados.

Para facilitar la edición de los reportes; en el módulo Editor de Reportes se quiere agregar nuevas funcionalidades:

- Gestionar plantillas para la creación de reportes donde los usuarios podrán diseñar sus propias plantillas y luego asignarlas en la creación de nuevos reportes. Esto es útil cuando muchos de los reportes tienen características comunes, como los encabezados y pie de páginas.
- Asignar varios orígenes de datos a un mismo reporte y crear consultas desde una interfaz de diseño, así como un editor de procedimientos almacenados en el lenguaje PL/SQL<sup>4</sup>.

## 1.5 Editor de Procedimientos

### 1.5.1 Editores de Consultas Simples

Se tuvieron en cuenta algunos editores de consultas simples existentes en el ámbito informático, siendo de gran ayuda para el diseño del módulo Editor de Consulta. Todos presentan características semejantes referentes al diseño de las consultas conllevando a la usabilidad de las aplicaciones.

#### Microsoft Access 2007 (sección de Diseño de Consulta)

Es un programa para la gestión de bases de datos, componente de la suite Microsoft Office, desarrollado y mantenido por Microsoft.

El diseño de las consultas en Microsoft Access se realiza por un método llamado QBE<sup>5</sup> que consiste en la construcción de la consulta en un modo gráfico. De manera sencilla se diseñan las consultas, a partir de un cuadro de diálogo se seleccionan las tablas y luego los campos, así como los restantes miembros de las consultas están previamente definidos y el usuario selecciona lo deseado. (6)

---

<sup>1</sup> Formato de documento portátil

<sup>2</sup> Formato de texto enriquecido

<sup>3</sup> Formato de archivo para almacenar imágenes

<sup>4</sup> Lenguaje de procedimiento/lenguaje de consulta estructurado

<sup>5</sup> Consultas por ejemplos

Esta herramienta es privativa por lo que si se desea hacer uso de ella hay que pagar una licencia, pero la estructura y la forma de edición de las consultas ha servido de guía para el presente trabajo de diploma.

### **Web Ad Hoc Query and Reporting Client (WAQR)**

La solución proporcionada por la plataforma de inteligencia de negocio de código abierto Pentaho e integrada en su suite para el desarrollo de informes se nombra Pentaho Reporting.

Pentaho Reporting proporciona una forma integrada de realizar consultas de forma intuitiva y sencilla a través de WAQR, herramienta integrada en el portal. (7)

El asistente dispone de 4 etapas:

1. Selección del origen de datos que se utilizará.
2. Realizar selección, se seleccionan los campos oportunos de cada una de las vistas de negocio disponibles incluyendo 3 posibles secciones: Grupos, Detalles, Filtros.
3. Personalizar las selecciones, se configuran los aspectos del formato de cada una de las secciones del informe.
4. Configuración del reporte, se indica la orientación del informe, el tamaño de papel y la configuración de la cabecera y pie de página de este. (7)

Ambos editores de consulta presentan una forma sencilla de edición, sirviendo de guía para la solución propuesta; además estas aplicaciones mantienen informado visualmente al usuario de los datos seleccionados para la edición de la consulta, característica que se mantuvo en la solución.

Se definió que el término Consulta Simple está asociado a la sintaxis básica de una consulta lo que incluye selección de las tablas y los campos incluyendo además la creación de filtros, agrupación de los campos a mostrar y ordenación de los resultados según el criterios de ascendente o descendente.

### **1.5.2 Gestionar Plantillas y Diseño de Reportes**

En la Universidad de las Ciencias Informáticas (UCI) existen diferentes proyectos productivos, los cuales tienen incluidos sistemas de reportes. A continuación se reflejan algunos de esos sistemas.

## **Sistema de Investigación e Información Policial (SIIPOL)**

SIIPOL es un software informático desarrollado para el Cuerpo de Investigaciones Científicas Penales y Criminalísticas (CICPC) de la República Bolivariana de Venezuela. Este sistema permite la creación de reportes estadísticos, que brindan indicadores, datos sintetizados y procesados referentes a la información con la que trabaja este cuerpo policial, mejorando el rendimiento del mismo. La herramienta utilizada para la generación de reportes estadísticos es JasperReport, una herramienta para soluciones sobre la plataforma Java. El Módulo de Estadística tiene los reportes ya definidos a partir de procedimientos almacenados, el usuario solo debe especificar el rango de fecha. Lo que aún no tienen implementado es el mecanismo para poder seleccionar más de un origen de datos en el reporte principal y sumado a esto no utilizan la misma herramienta para la generación de los reportes que SIGESC.

## **Proyecto Identidad**

El proyecto Identidad mantiene una arquitectura muy semejante al SIGESC, cuenta con un Módulo de Estadística, que se encarga de gestionar los reportes desde su solicitud, pasando por el diseño hasta su visualización e impresión. Utiliza como herramienta de generación de reporte ActiveReports. El proyecto Identidad maneja el trabajo de las plantillas en los reportes, permitiendo crear plantillas y guardarlas en un formato específico para luego asignarle a un reporte dicha plantilla. No presenta el diseño de varios orígenes de datos. A pesar de presentar muchas características comunes, el equipo de desarrollo de Identidad trabaja con el framework 2.0 de .NET lo que representa un problema de compatibilidad con SIGESC impidiendo de esta forma asimilar la solución planteada por ellos.

## **Reporteador Dinámico (DATEC)**

El Centro de Tecnologías de Gestión de Datos (DATEC) existente en la Universidad, cuenta con un Reporteador Dinámico como uno de sus tantos productos. El mismo está desarrollado con tecnología web y con el framework Spring en su capa de Presentación. Permite la creación de plantillas personalizadas y de reportes utilizando las plantillas previamente diseñadas. Durante la etapa del diseño se pueden incluir gráficos y tablas y al culminar se pueden exportar a diversos formatos como PDF<sup>6</sup>, Excel, entre otros.

---

<sup>6</sup> Formato de documento portátil

## 1.6 Metodología Seleccionada Rational Unified Process

Una metodología de desarrollo es una colección de documentación formal referente a los procesos, políticas y procedimientos que intervienen en el proceso de desarrollo del software.

La finalidad de una metodología de desarrollo es garantizar la eficacia (ejemplo cumplir los requisitos iniciales) y la eficiencia (ejemplo minimizar las pérdidas de tiempo) en el proceso de generación de software. (8)

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. Cubre el ciclo de vida y desarrollo de software. (9)

La razones por lo cual se seleccionó RUP como metodología de desarrollo son:

- RUP al ser una metodología tradicional posibilita el desarrollo del proyecto sin la presencia constante del cliente, el mismo solo participa en reuniones o entrevistas donde se puntualizan las precisiones del producto. En este caso el cliente se encuentra en Venezuela y el equipo de desarrollo en la UCI.
- RUP posee varios elementos de planificación (Plan de Desarrollo, Plan de Iteración, entre otros más) que permiten llevar el control del desarrollo del proyecto.
- La UCI está en constante retroalimentación de estudiantes, quienes constituyen una parte importante del equipo de desarrollo. La documentación detallada generada con RUP, sirve de base para futuros desarrolladores.
- En sus inicios fue la metodología seleccionada por el proyecto SIGESC para guiar el proceso de desarrollo.

Sumado a lo anterior vale destacar que durante la formación de los estudiantes en la UCI, el plan de estudios contempla la asignatura de Ingeniería de Software con la metodología RUP, factor que beneficiará al equipo de desarrollo.

RUP, la metodología de desarrollo de software seleccionada para guiar el proceso de desarrollo de la presente solución propone a UML<sup>7</sup> como su lenguaje de modelado.

### 1.7 Lenguaje Unificado de Modelado v2.0

El Lenguaje Unificado de Modelado es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Se usa para entender, diseñar, hojear, configurar, mantener, y controlar la información sobre los sistemas que se deben construir. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. El lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar. (10)

UML incluye conceptos semánticos, notación, y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. Puede ser utilizado en herramientas interactivas de modelado visual que tengan generadores de código así como generadores de informes. La especificación de UML no define un proceso estándar pero está pensado para ser útil en un proceso de desarrollo iterativo. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos. (11)

Se deben seleccionar además las tecnologías y herramientas de desarrollo que servirán de apoyo para la implementación de la solución final.

### 1.8 Procedimiento de Modelado de Procesos IDEF0

IDEF<sup>8</sup> consiste en una serie de normas que definen la metodología para la representación de funciones modeladas. La versión utilizada para esta representación es IDEF0 donde como concepto de modelación va introduciendo gradualmente más y más niveles de detalle a través de la estructura del modelo. De esta manera, la comunicación se produce dando al lector un tema bien definido con una cantidad de información detallada disponible para profundizar en el modelo. (12)

Las principales ventajas del uso de IDEF0 son:

- Es una forma unificada de representar funciones o sistemas.
- Lenguaje simple pero riguroso y preciso.

---

<sup>7</sup> Lenguaje Unificado de Modelado

<sup>8</sup> Definición de integración para el modelado de funciones

- Permite establecer unos límites de representación de detalles establecido universalmente.

La descripción de cada proceso o actividad es considerada la combinación de cinco magnitudes básicas: procesos o actividades, entradas, controles, sujeto y salidas o resultados conseguidos en el proceso. Cada una de estas magnitudes es representada en los diagramas IDEF0 de la siguiente manera:



Figura 1.2 Representación de magnitudes del modelo IDEF0.

## 1.9 Tecnología y Herramientas de Desarrollo seleccionadas

### 1.9.1 Visual Paradigm v3.4

El Visual Paradigm es una suite completa de herramientas CASE<sup>9</sup> que da soporte al modelado visual con UML<sup>10</sup> 2.0 ofreciendo distintas perspectivas del sistema. Es independiente de la plataforma y dotada de una buena cantidad de productos o módulos para facilitar el trabajo durante la confección de un software así como garantizar la calidad del producto final. (3)

Es profesional, fácil de entender, brinda un número considerable de estereotipos a utilizar así como facilidades para redactar especificaciones de casos de uso. (3)

<sup>9</sup> Ingeniería de software asistido por computadoras

<sup>10</sup> Lenguaje Unificado de Modelado

Se integra fácilmente con diferentes IDEs<sup>11</sup> (Visual Studio, NetBeans, Eclipse), generando código o realizando ingeniería inversa. Además genera código ORM<sup>12</sup> a partir de un Diagrama Entidad Relación, a una Base de Datos Relacional y el código necesario para acceder a esta base de datos utilizando Java, PHP<sup>13</sup> o Enterprise Object Framework.

### 1.9.2 Framework DMAX

El framework DMAX<sup>14</sup> brinda una arquitectura soportada en tecnologías basadas en el modelo n-capas, además de un núcleo de clases que engloban funcionalidades y que preparan condiciones para el diseño y desarrollo de aplicaciones.

La integración con el mencionado framework implica la utilización de mecanismos sencillos de relaciones entre clases y archivos de configuraciones que permiten la reutilización de funcionalidades y comportamientos. SIGESC realiza toda la gestión de los dominios de aplicación a través de los mecanismos definidos por el Framework DMAX, quien a su vez provee seguridad, auditoría, configuración y aislamiento bien definidos. (13)

Los mecanismos de integración puestos en práctica, constituyen básicamente la utilización de herencia y relaciones de composición y/o agregación entre clases, así como la utilización e implementación de patrones de diseño como son Fábrica Abstracta, Proxy y Puente para el acceso a clases de comportamiento y procesamiento.

Cada producto de software presenta su propia arquitectura, siendo definida por el equipo de desarrollo, el cual debe respetar las pautas y parámetros definidos. El presente trabajo de diploma se rigió por la arquitectura del Sistema de Emergencia de Seguridad Ciudadana.

### 1.9.3 Plataforma de Desarrollo .NET

En informática, una plataforma de desarrollo es el entorno de software común donde se desenvuelve la programación de un grupo definido de aplicaciones. Comúnmente se encuentra relacionada directamente

---

<sup>11</sup> Entorno Integrado de Desarrollo

<sup>12</sup> Mapeo Objeto-Relacional

<sup>13</sup> Lenguaje de programación interpretado

<sup>14</sup> Desarrollo Modular basado en Arquitectura Extensible

a un sistema operativo; sin embargo, es posible encontrarla ligada a una familia de lenguajes de programación o a una Interfaz de Programación de Aplicaciones (14).

La plataforma .NET es un conjunto de productos formado por una serie de herramientas y librerías con las que se pueden crear todo tipo de aplicaciones, desde las tradicionales aplicaciones de escritorio hasta aplicaciones para XBOX<sup>15</sup> pasando por desarrollo web, desarrollo para móviles (compact framework) y aplicaciones de servidor (Windows Presentation Foundation y Windows Communication Foundation). (14)

Esta plataforma brinda seguridad, robustez, soluciones rápidas, económicas, escalables y distribuidas. Además de un conjunto de funcionalidades a los desarrolladores como son: representar tipos de datos base y excepciones, encapsular estructuras de datos, realizar tareas de entrada/salida y lectura/escritura de datos, dibujo y renderización gráfica, interacción con bases de datos, manipulación de documentos XML<sup>16</sup>, entre otras muchas. (15)

#### **1.9.4 Entorno Integrado de Desarrollo**

Los Entornos Integrados de Desarrollo son programas informáticos que brindan una serie de herramientas de programación y que pueden estar orientados a un lenguaje en específico o pueden soportar el trabajo en varios lenguajes de programación.

La plataforma .NET, brinda un entorno de desarrollo integrado nombrado Visual Studio .NET que posibilita la creación de aplicaciones robustas y seguras, poniendo a disposición del usuario un potente sistema de desarrollo. (2)

#### **1.9.5 .NET Framework v1.1**

Microsoft .NET Framework es el núcleo de la plataforma Microsoft. NET teniendo que cumplir con la entrega a los desarrolladores de un entorno de programación orientado a objetos coherente, en el cual el código se pueda ejecutar tanto localmente como distribuido en Internet o de forma remota. Ofrece al desarrollador un ambiente conocido durante el desarrollo de distintos tipos de aplicaciones (Web, Windows Form, etc.), basa la comunicación en los estándares para garantizar que las aplicaciones. NET se podrán integrar con aplicaciones de otras plataformas. (16)

---

<sup>15</sup> Videoconsola

<sup>16</sup> Lenguaje de marcas extensibles

.NET Framework posee 2 componentes esenciales: Entorno Común de Ejecución para Lenguajes (CLR) y la Librería de Clases Base (BCL), el primero carga las aplicaciones, es el encargado de la administración del código en tiempo de ejecución, proporcionando servicios de administración de memoria, administración de subprocesos, interacción remota y seguridad de tipos entre otros. Mientras la biblioteca de clases proporciona un completo conjunto de tipos, que se pueden utilizar en el desarrollo de las aplicaciones. (16)

Actualmente el equipo de desarrollo de SIGESC utiliza la versión 1.1, la cual incluye soporte de ODBC<sup>17</sup> para Oracle.

Toda aplicación informática se desarrolla con un lenguaje de programación escogido por el cliente o por el equipo de desarrollo. Cada lenguaje tiene sus particularidades, que los hace únicos.

#### **1.9.6 Lenguaje de Programación CSharp**

CSharp (C#) es un lenguaje de programación orientado a objetos, desarrollado por Microsoft. Se caracteriza por sencillo, eficiente. Incorpora todas las características de POO<sup>18</sup>, encapsulamiento, herencia (simple) y polimorfismo. (2)

La selección de este lenguaje para implementar la solución se justifica principalmente en que C# es un lenguaje que explota a fondo la plataforma .NET pues fue diseñado específicamente para ella, careciendo de elementos heredados innecesariamente. Es moderno, de muchas potencialidades, sencillo e intuitivo. Es un lenguaje auto contenido por lo que no necesita ficheros adicionales al del código fuente para su ejecución.

Incluye mecanismos de seguridad de tipos, que permiten asegurar que los accesos a tipos de datos siempre se realicen correctamente, lo que permite evitar que se produzcan errores difíciles de detectar por acceso a memoria no perteneciente a ningún objeto y es especialmente necesario en un entorno gestionado por un recolector de basura.

El trabajo de diploma utiliza para su solución Active Reports, un sistema de generación de reportes que es respaldado por el lenguaje de programación CSharp.

---

<sup>17</sup> Abrir conectividad para base de datos

<sup>18</sup> Programación Orientada a Objetos

### 1.9.7 Active Reports

Active Reports es una herramienta de generación de reportes desarrollada por la compañía Data Dynamics. Presenta un diseñador potente y fácil de utilizar, completamente integrado a los lenguajes de programación de Visual Studio .NET. (17)

Tiene como característica principal la personalización de reportes y un rendimiento rápido de sus prestaciones. Permite exportar a formatos de archivos habituales como son PDF<sup>19</sup>, Excel, RTF<sup>20</sup> y TIFF<sup>21</sup>. Active Reports incluye el uso de tecnologías como XML<sup>22</sup>, lenguajes script y Hojas de Estilo en Cascada (CSS) con el uso de una arquitectura abierta para proveer un diseñador de reportes completamente integrado y amigable. (17)

### 1.9.8 Sistema Gestor de Base de Datos Oracle 10g

Un Sistema Gestor de Bases de Datos (SGBD) es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Un SGBD permite definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad, confidencialidad e integridad de los mismos. (18)

Oracle es considerado como uno de los sistemas de bases de datos más completos destacando en soporte de transacciones, estabilidad, escalabilidad (soporta grandes niveles de transformación) y multiplataforma (posee interacción con todas las plataformas). La versión 10g, ha podido ser extendida a muchas áreas, demostrando los mejores resultados en aspectos tan importantes como seguridad y confidencialidad, manteniendo a su vez, altos indicadores de rendimiento y disponibilidad.

Esta versión se ajusta a las necesidades de almacenamiento y gestión de la información del SIGESC, producto a la gran cantidad de información que se maneja. Esta información almacenada debe ser gestionada de forma eficiente, con la mayor rapidez posible y un tiempo de respuesta mínimo (2). Otro de los factores importantes en la elección es el soporte que brinda la compañía ante cualquier problema con

---

<sup>19</sup> Formato de documento portátil

<sup>20</sup> Formato de texto enriquecido

<sup>21</sup> Formato de archivo para almacenar imágenes

<sup>22</sup> Lenguaje de marcas extensibles

el gestor, contribuyendo a la seguridad sobre la información que persiste en la base de datos, donde la mayor cantidad de los datos que se almacenan es de carácter confidencial (19). Sumado a lo anterior Oracle incluye PL/SQL<sup>23</sup> como lenguaje de programación, con el cual se pudieron programar procedimientos almacenados y funciones que fueron utilizados en la propuesta de solución.

### **1.10 Conclusiones Parciales**

Al concluir este capítulo se estudiaron las principales herramientas de edición de consultas simples, procedimientos y de generación de reportes. Se definieron las principales herramientas y tecnologías a utilizar. Los sistemas estudiados permitieron entender las bases de la solución propuesta, contribuyendo en gran medida al desarrollo de la misma.

---

<sup>23</sup> Lenguaje de procedimiento/lenguaje de consulta estructurado

## *Capítulo 2. Características del Sistema*

### **2.1 Introducción**

Durante el transcurso de este capítulo se pretende abordar la solución propuesta, describiéndose los procesos que fueron posibles identificar. Como complemento se empleó un Modelo de Dominio, ayudando a los desarrolladores, usuarios e interesados en el resultado de este trabajo a utilizar un vocabulario común en aras de lograr una mayor comprensión del problema, contribuyendo a una correcta captura de requisitos.

Además se enumeran los requisitos funcionales y no funcionales a cumplir por el sistema, se definen los actores y el diagrama de caso de uso del sistema así como una descripción detallada de los mismos.

### **2.2 Problema y situación problemática**

Actualmente en los Centros 171 donde se encuentra desplegado el SIGESC, el personal responsable del trabajo con el Módulo Estadística depende del equipo de desarrollo para realizar nuevos reportes. La aplicación Visor de Reportes permite visualizar reportes, conteniendo algunos proveedores de datos definidos por el equipo de desarrollo, pero no son suficientes a la hora de realizar un análisis completo del sistema pues se necesitan otros proveedores de datos y vistas, los cuales no pueden ser confeccionados por la falta de conocimiento de los trabajadores.

Durante el proceso de diseño de los reportes en la aplicación Estadística se incluye la edición del encabezado, el pie de página y el cuerpo pudiéndose confeccionar previamente mediante una plantilla donde luego puede ser utilizada para el diseño de los reportes.

### **2.3 Propuesta del sistema**

La aplicación Estadística del SIGESC cuenta actualmente con un Editor de Reportes para la generación de información que posteriormente puede ser usada como evaluadora de los Centros 171. El mismo

permite la creación de diferentes reportes y la asignación de un origen de datos por cada reporte o por cada subreporte asociado al reporte principal.

### **2.3.1 Gestionar plantillas**

Como parte de la solución propuesta se pretende realizar el diseño de la funcionalidad Gestionar Plantilla para permitir que sean utilizadas para la confección de reportes, las mismas estarán localizadas en un directorio local y podrán ser asignadas durante la etapa de diseño. Se propone que futuras versiones del módulo implementen el diseño realizado.

### **2.3.2 Diseñar Reportes**

Diseñar la funcionalidad de añadir varios orígenes de datos a un mismo reporte, permitiendo de esta forma tener una visión más clara de la situación debido a la cantidad de información a mostrar. Se propone que futuras versiones del módulo implementen el diseño realizado.

### **2.3.3 Editor**

De forma paralela se realizará el diseño y la implementación de un Editor de Consulta que permita diseñar consultas y almacenarlas en la base de datos como procedimientos. El lenguaje a emplear para la confección de las mismas será PL/SQL<sup>24</sup>.

#### **Editor Consulta Simple**

El Editor de Consulta contará con varios menús donde el Diseñador Estadístico podrá seleccionar las tablas de la base de datos de las cuales desean obtener la información para la realización de las consultas, así como los campos dentro de las tablas y las secciones opcionales de filtrar, agrupar y ordenar. Estos menús responderán a las sentencias básicas para consultas simples.

Una consulta simple es un método que permite acceder a los datos de una base de datos y realizar diversas acciones, el editor solo contemplará acciones de selección y la definición de simple viene dada por el empleo de cláusulas sencillas. (Ver tabla 2.1)

---

<sup>24</sup> Lenguaje de procedimiento/lenguaje de consulta estructurado

**Tabla 2.1 Componentes de una Consulta.**

<b>Sentencias básicas</b>	<b>Nombre de los menús</b>
Select	Campo Selección
From	Tabla
Inner Join	Relación
Where	Filtro
Group by	Agrupador
Order by	Ordenar

Existen dos menús adicionales Compilar y Registrar. El primer menú contemplará el hecho de crear la consulta simple en una cadena, donde será un parámetro de entrada al compilador (Ver epígrafe Compilador).

### **Área de Modelado**

Lugar o espacio determinado por fronteras donde se pueden modelar de forma visual con determinados elementos contenidos dentro de la propia área. Siempre está asociado a una herramienta que gestiona los elementos modelados en el área de trabajo. (20) (21).

Se utilizará como área de modelado un control *PanelControlador.VistaComponente* incluido en el framework DMAX<sup>25</sup>, el cual gestionará todo el proceso de selección y ocultamiento de las tablas y sus campos. Este control estará presente dentro del formulario principal.

### **Editor de Procedimiento**

El Editor de Procedimientos del cual se realizará solo su diseño contará con componentes visuales: líneas, figuras e imágenes. Todos estos componentes visuales formarán parte de un diagrama de secuencia que generará el procedimiento en PL/SQL<sup>26</sup>, el lenguaje empleado en el SIGESC.

<sup>25</sup> Desarrollo Modular basado en Arquitectura Extensible

<sup>26</sup> Lenguaje de procedimiento/lenguaje de consulta estructurado

**Tabla 2.2 Algunos componentes visuales.**

Sentencias básicas (PL/SQL)	Componentes visuales
Inicio (BEGIN)	
Declaración nombre varchar2(200)	
Condicionales (IF, ELSE)	
Secuencias de ciclos (WHILE, LOOP)	
Asignación de valores nombre:= 'Paul'	
Fin (END)	

**Tabla 2.3 Complemento al Editor**

Nombre	Componente Visual
Conector	

### Compilador

Para realizar la validación de las consultas y los procedimientos se utilizarán tres de los módulos generales que participan en el proceso de compilación: Análisis lexicológico, Análisis sintáctico, Análisis semántico.

La entrada de un compilador es el código de un programa escrito en un lenguaje de programación. Dicho código no es más que una secuencia de símbolos pertenecientes al alfabeto del lenguaje de

programación. El analizador lexicológico o scanner se encarga de tomarlos y agruparlos en entidades sintácticas simples o elementales denominadas tokens o lexemas. (22) (Ver Anexo 1).

El análisis sintáctico es el proceso en el cual se examina la secuencia de tokens para determinar si el orden de esa secuencia es correcto de acuerdo a ciertas convenciones estructurales (reglas) de la definición sintáctica del lenguaje. (22) (Ver Anexo 2).

En el análisis semántico se detectan errores relacionados con la validez del programa. Se puede decir que estos errores son de tipo sintáctico semántico, pero no pueden ser detectados por el analizador sintáctico, ya que se relacionan con interdependencias entre las diferentes partes de un programa que no son reflejadas en un análisis gramatical.

El analizador semántico recibe la información resultado del análisis sintáctico que puede ser un árbol jerárquico con la información relativa a la organización de los tokens en la instrucción que se está analizando.

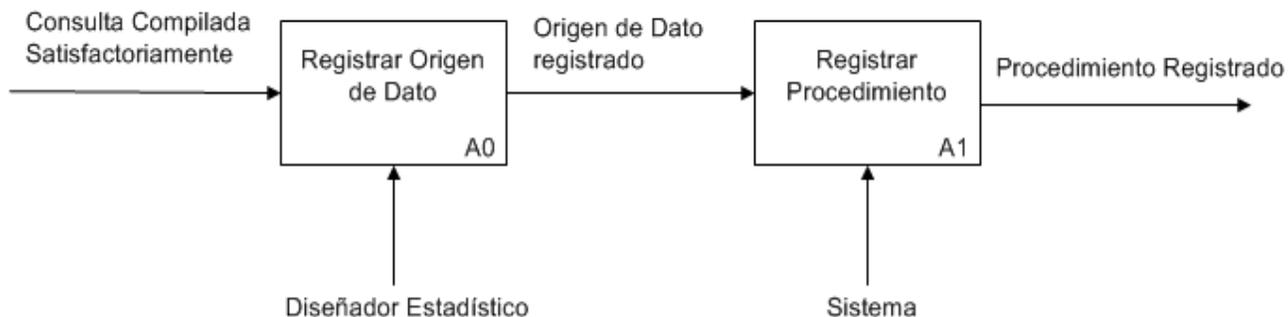
## **2.4 Modelo de Negocio**

Durante el desarrollo del presente trabajo de diploma fue necesaria la utilización de un Modelo de Negocio y de un Modelo de Dominio como complemento del primero al no estar bien definidos los procesos de Gestionar Plantillas y Diseñar Reporte con varios orígenes de datos.

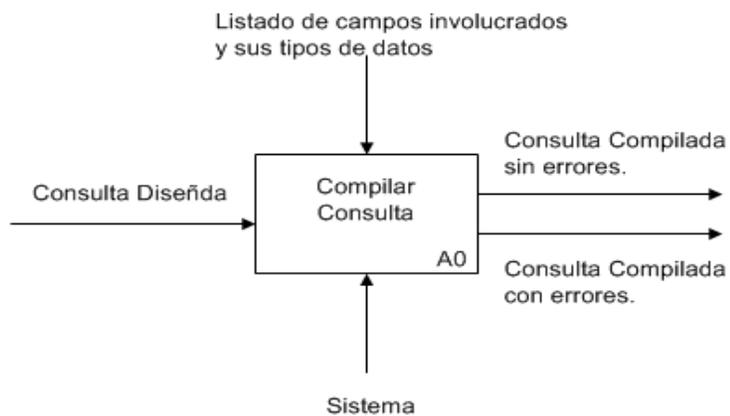
Con la ayuda del procedimiento para el modelado de procesos IDEF0<sup>27</sup> se muestra el negocio propuesto. Mediante la utilización de este método se identificaron tres procesos fundamentales, los cuales son: Registrar Consulta, Compilar Consulta y Editar Consulta.

---

<sup>27</sup> Definición de integración para el modelado de funciones



**Figura 2.1 Proceso Registrar Consulta.**



**Figura 2.2 Proceso Compilar Consulta.**

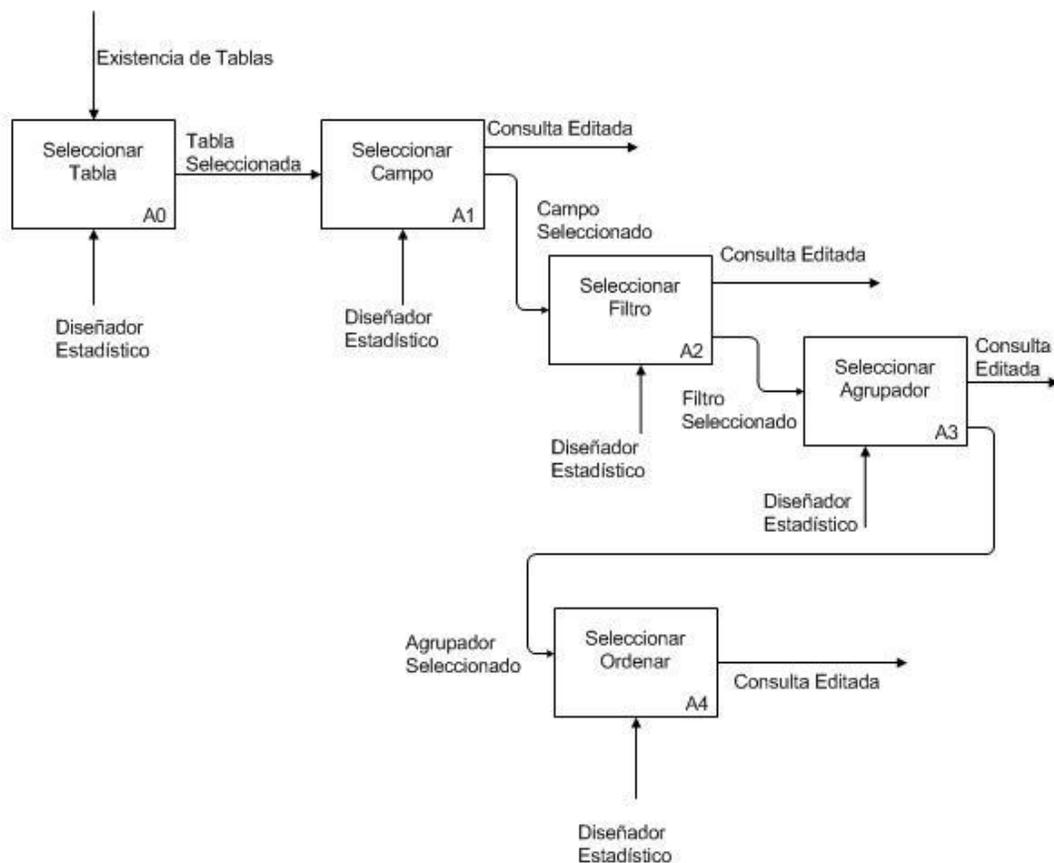


Figura 2.3 Proceso Editar Consulta.

### 2.4.1 Descripción del Modelo de Negocio

La descripción de los procesos de negocio modelados con IDEF0<sup>28</sup> se detalla a continuación.

**Registrar Consulta:** este proceso consiste en compilar la consulta diseñada, de no existir ningún error en la consulta se procede al registro del origen de dato. Como fase final del proceso se procede a la inserción en la base de datos de la consulta diseñada, la cual se registrará como un procedimiento en sí.

**Compilar Consulta:** consiste en la compilación de la consulta diseñada con la ayuda de un compilador. Se mostrará una notificación en caso de encontrarse errores.

<sup>28</sup> Definición de integración para el modelado de funciones

**Editar Consulta:** consiste en la confección de una consulta simple a través de la selección de las tablas de la base de datos así como de sus respectivos campos, es posible crear los filtros que desee, además de seleccionar los agrupadores y ordenadores. De esta forma quedará diseñada la consulta.

El sujeto en todos y cada uno de los procesos anteriormente descritos es el Diseñador Estadístico, siendo el responsable de realizar las actividades que se representan a través de estos procesos, y que será además el usuario a interactuar con el sistema.

## 2.5 Modelo de Dominio

El Modelo de Dominio ayudará a comprender los conceptos con los que se trabaja y con los que deberá trabajar la aplicación.

El Modelo de Dominio es una representación visual estática del entorno real objeto del proyecto. (10)

El proceso para su elaboración consta de tres pasos:

1. Identificar las Clases Conceptuales.
2. Dibujarlas en un Diagrama de Clases.
3. Añadir Relaciones y Atributos. (10)

Siguiendo la guía anterior se identifican las clases conceptuales, sus atributos y relaciones de las nuevas funcionalidades a realizar, además de tenerse en cuenta las clases conceptuales del módulo de Estadística del Proyecto las cuales están resaltadas en otro color (Rosado).

### 2.5.1 Identificar las Clases Conceptuales

**Consulta Simple:** sentencia SQL<sup>29</sup> generada por el Editor de Consulta.

**Consulta Procedimental:** sentencia SQL generada por el Editor de Procedimientos.

**Reporte:** es un contenido informativo que se genera referente a las atenciones y servicios de emergencia en los Centros 171.

---

<sup>29</sup> Lenguaje de consulta estructurado

**Plantilla:** un documento contenedor de varios componentes (líneas, imágenes, textos, con distintos márgenes y tamaños), que facilitan el desarrollo de los reportes. Es un estándar a utilizar para la creación de los reportes.

**Procedimiento:** un procedimiento almacenado es una sentencia de código que es almacenado físicamente en una base de datos. Su implementación varía de un manejador de bases de datos a otro. La ventaja de un procedimiento almacenado es que al ser invocado, en respuesta a una petición de usuario, es ejecutado directamente en el motor de bases de datos, el cual usualmente corre en un servidor separado.

**Origen de Dato:** procedimientos almacenados en la base de datos que son utilizados en la confección de los reportes.

**Active Reports:** herramienta para la generación de reportes.<sup>30</sup>

**Editor de Reportes:** esta aplicación se encarga de editar y diseñar los reportes.

**Visor de Reporte:** es la aplicación del SIGESC encargada de visualizar reportes estadísticos definidos previamente.

**Editor de Procedimientos:** es una aplicación que se diseñará con el objetivo de crear procedimientos almacenados simples a partir de una interfaz de diseño en PL/SQL<sup>31</sup>.

**Analista de Datos:** es la persona que visualiza reportes estadísticos con el fin de analizar la información mostrada en estos. Esta persona puede además exportar reportes a formatos conocidos para su mantenimiento y distribución física.

**Diseñador Estadístico:** es la persona encargada de diseñar los reportes y gestionar las plantillas.

Diagrama del Modelo de Dominio (Ver Anexo 3).

---

<sup>30</sup>Capítulo 1

<sup>31</sup> Lenguaje de procedimiento/lenguaje de consulta estructurado

## 2.6 Arquitectura de la solución propuesta

Cada módulo que integra el SIGESC, está compuesto y diseñado por agrupaciones lógicas de un conjunto de responsabilidades bien definidas, a las que se les llama capas, las que están divididas en: Presentación, Negocio y Acceso a Datos. (Ver figura 2.4).

- **Capa Presentación:** se divide en dos grupos de funcionalidades, Interfaz de Usuario y Acciones.  
Las Interfaces de Usuario presentan información a los usuarios y acepta entradas o respuestas del usuario para usarlas en el sistema. Estas no desarrollan ningún procesamiento de negocio o reglas de validación asociadas.  
Las Acciones representan peticiones que se realizan al negocio. Estas peticiones pueden tener una Interfaz de Usuario asociada o no, las que tienen Interfaz de Usuario asociada se les denomina clases controladoras de esa Interfaz de Usuario.
- **Capa Negocio:** es el núcleo del módulo puesto que encapsula la lógica de implementación fundamental para la informatización de los procesos del negocio en cuestión. Esta capa es determinante en el éxito del sistema y su modelación implica especial atención por ambas partes: los desarrolladores y el cliente.
- **Capa Acceso a Datos:** encapsula la lógica de implementación necesaria para gestionar los datos utilizados en uno o muchos procesos de negocio y abstraer así la forma en que los datos persisten o son obtenidos. (13)

### Dominio de un Módulo

Cada módulo del SIGESC requiere del paso de datos entre las capas. Estos datos suelen encapsularse en entidades informativas que no son más que clases, siendo su papel fundamental la información que almacena y no el comportamiento de esta. El conjunto de todas las clases entidades es denominado Dominio.



**Figura 2.4 Arquitectura de la solución propuesta.**

Existen diferentes mecanismos para la integración de las capas<sup>32</sup> y todos tienen el objetivo de resolver de una forma u otra un conjunto de problemáticas tales como:

- Lograr que las capas queden totalmente desacopladas con el objetivo de mejorar la capacidad de mantenimiento de la aplicación, facilitar el desarrollo paralelo y lograr una mayor cohesión de las funcionalidades de la capa.
- Que el desacople no provoque problemas de rendimiento.
- No provocar dependencias a otras tecnologías.
- Facilidad de uso e implementación. Si la integración entre capas requiere un esfuerzo significativo puede provocar atrasos y la posibilidad de cometer errores aumentaría considerablemente. (13)

<sup>32</sup> Explicación de la integración entre capas Capítulo 3

## 2.7 Especificación de los requisitos de software

El Estándar IEEE Glosario de Terminología de Ingeniería del Software define un requerimiento como condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo.

RUP<sup>33</sup> como metodología de desarrollo seleccionada propone para la especificación de los requisitos de software el Flujo de Trabajo Requisitos, el cual tiene como objetivo:

1. Definir el ámbito del sistema.
2. Definir una interfaz de usuarios para el sistema, enfocada a las necesidades y metas del usuario.
3. Establecer y mantener un acuerdo entre clientes y otros involucrados sobre lo que el sistema debería hacer.
4. Proveer a los desarrolladores un mejor entendimiento de los requisitos del sistema.
5. Proveer una base para estimar recursos y tiempo de desarrollo del sistema.
6. Proveer una base para la planeación de los contenidos técnicos de las iteraciones. (8)

Los requisitos funcionales han sido divididos y estructurados. (Ver Anexo 4).

### 2.7.1 Requisitos No Funcionales

Los requisitos 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. Normalmente están vinculados a requisitos funcionales, es decir una vez se conozca lo que el sistema debe hacer podemos determinar cómo ha de comportarse, qué cualidades debe tener o cuán rápido o grande debe ser. Los requisitos no funcionales forman una parte significativa de la especificación.

Para el desarrollo del trabajo de diploma se tuvieron en cuenta los requisitos no funcionales descritos en el documento Especificaciones Suplementarias – SIGESC v1.1, los cuales aplican para cada módulo a desarrollar dentro del proyecto. A continuación se nombran cada uno de ellos:

#### Portabilidad

El módulo de Edición de Reportes está diseñado para ser utilizado sobre el sistema operativo Microsoft Windows XP SP2. El servidor de bases de datos a utilizar debe ser Oracle 10g (recomendado) o superior, cual almacenará todos los datos que gestiona el módulo.

---

<sup>33</sup> Proceso Unificado de Rational

## **Usabilidad**

Constituye el requerimiento más importante durante el desarrollo de software. Para la utilización de la aplicación es necesario poseer conocimientos elementales de utilización del sistema operativo Microsoft Windows y aplicaciones informáticas. El módulo de Edición de Consultas podrá ser utilizado por la persona a la que se le asigne el rol de Diseñador Estadístico. El módulo cuenta con acceso de teclas rápido para acceder a las diferentes opciones del menú.

## **Escalabilidad**

El diseño del módulo contempla el uso óptimo de recursos tales como conexiones a la base de datos, contemplándose en el diseño la clara partición entre datos, recursos y aplicaciones optimizando de esta forma su escalabilidad. Nuevos usuarios pueden hacer uso del Editor de Consultas, asignándose previamente los permisos al rol específico y este a su vez puede ser instalado en varios puntos de trabajo.

## **Fiabilidad**

El módulo de Edición de Consultas está diseñado para funcionar bajo el régimen de 24 horas durante los 7 días de la semana, o sea, todos los días naturales del año, no deben realizarse servicios de reparaciones que provoquen su detención aunque esta sólo sea momentánea. Las reparaciones necesarias o actualizaciones deberán realizarse con el sistema funcionando.

Las principales condiciones que pudieran provocar fallos en el correcto funcionamiento del módulo son:

- Falta de fluido eléctrico en algunas zonas o en la totalidad del Centro de Gestión de Emergencias.
- Falla del servicio de red sobre el cual se soporta el sistema.
- Instalación o configuración incorrecta de algunos componentes del sistema.
- En caso que se den algunas de las condiciones anteriormente mencionadas, el módulo tiene la capacidad de reconectarse automáticamente al sistema con el objetivo de minimizar los errores que dichas situaciones pudieran provocar.

El módulo está concebido para registrar en la Base de Datos consultas que estén correctas (auxiliándose de la ayuda de un compilador), arrojando resultados por cada una de ellas.

## **Interfaz de usuario**

La interfaz de usuario del módulo es de fácil utilización por los usuarios finales, cumpliendo con los requisitos siguientes:

- Las ventanas del sistema contendrán claros y bien estructurados los datos, permitiendo la interpretación correcta e inequívoca de la información.
- El diseño responderá a la ejecución de acciones de manera rápida, minimizando los pasos a dar en cada proceso.
- El módulo usará una norma que permita la distinción visual entre los distintos elementos en las ventanas, a través del uso de colores, tamaño de las fuentes, así como otras técnicas.
- La corrección de errores en la introducción de los datos será clara y fácil de realizar. La entrada de datos incorrecta será detectada claramente e informada al usuario.
- Todos los textos y mensajes en pantalla aparecerán en idioma español.
- Su funcionamiento será intuitivo y requerirá de información mínima.

## **Seguridad**

La seguridad del módulo está basada en niveles de acceso sobre las funcionalidades y la información. Los principios básicos que determinan la seguridad del sistema son los siguientes:

- Seguridad y control a nivel de usuarios, contraseñas y roles, garantizando el acceso de los usuarios sólo a los niveles establecidos de acuerdo a la función que realizan. Las contraseñas sólo podrán ser cambiadas desde la aplicación de Administración Informática del Sistema.
- Seguridad a nivel del gestor de datos, estableciendo privilegios de acceso sobre los datos almacenados y garantizando el acceso a los datos sólo a aquellos usuarios con los permisos necesarios para hacerlo, y sólo a los datos que le esté permitido acceder de acuerdo a sus privilegios en el Sistema.
- Las excepciones que pueda generar el módulo de Edición de Consulta serán registradas para su posterior análisis desde la aplicación de Administración Informática.
- Seguridad basada en roles.

## **Mantenibilidad**

El código de la implementación está estructurado de manera consistente y predecible estableciendo un estándar común descrito en la arquitectura base del SIGESC (13). Un cambio en los parámetros del negocio no obliga a la generación de una nueva versión del módulo. (23)

### 2.8 Definición de los actores

Cada trabajador del negocio (inclusive si fuera un sistema ya existente) que tiene actividades a automatizar es un candidato a actor del sistema. Si algún actor del negocio va a interactuar con el sistema, entonces también será un actor del sistema. (11)

Se identificó un actor del sistema, que se muestra en la siguiente tabla.

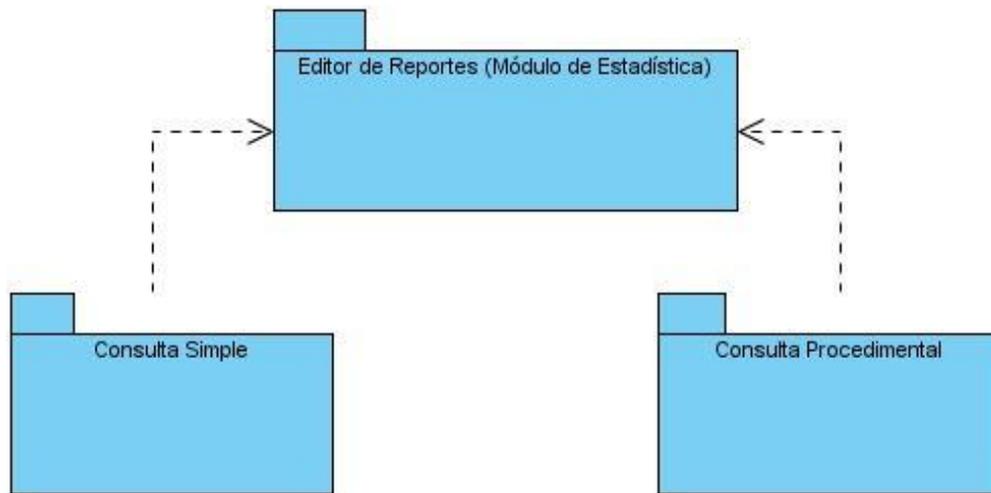
**Tabla 2.4 Actores del sistema.**

Actor	Justificación
Diseñador Estadístico	Es la persona encargada de gestionar las plantillas y diseñar reportes. Es el responsable de editar las consultas simples necesarias para la ejecución de los reportes estadísticos.

### 2.9 Diagrama de Paquete

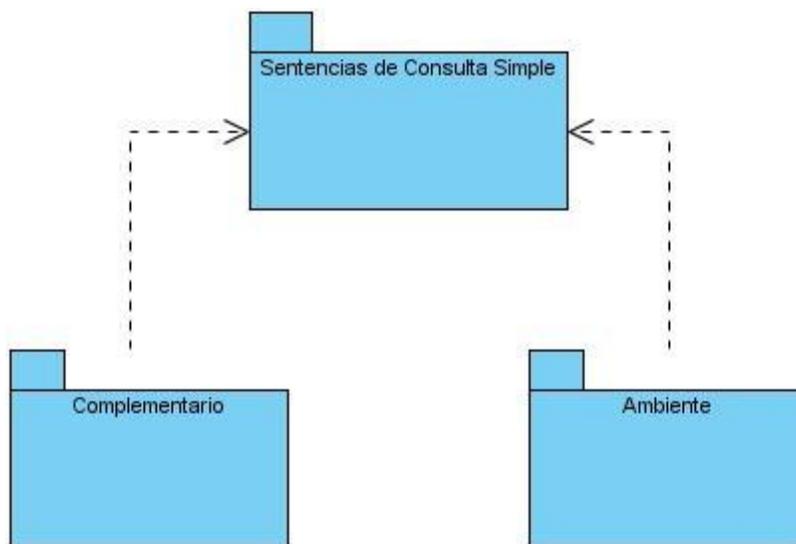
Estos paquetes están normalmente organizados para maximizar la coherencia dentro de cada paquete y minimizar el acoplamiento externo entre estos.

A continuación se muestra el diagrama de paquetes de la solución propuesta. Se han agrupado de esta manera los casos de uso por funcionalidades de cada una de las aplicaciones donde se trabajará. Fue necesario utilizar este diagrama para lograr entender mejor el funcionamiento de la solución.



**Figura 2.4 Diagrama de Paquete.**

Dentro del paquete Consulta Simple se han empaquetado las funcionalidades existentes para una mayor claridad en la modelación. (Ver figura 2.3)



**Figura 2.5 Diagrama de Paquete del paquete Consulta Simple.**

### 2.10 Diagrama de Casos de Uso

A continuación se exponen los diagramas de Caso de Uso del sistema y el paquete al que pertenece cada uno de ellos.

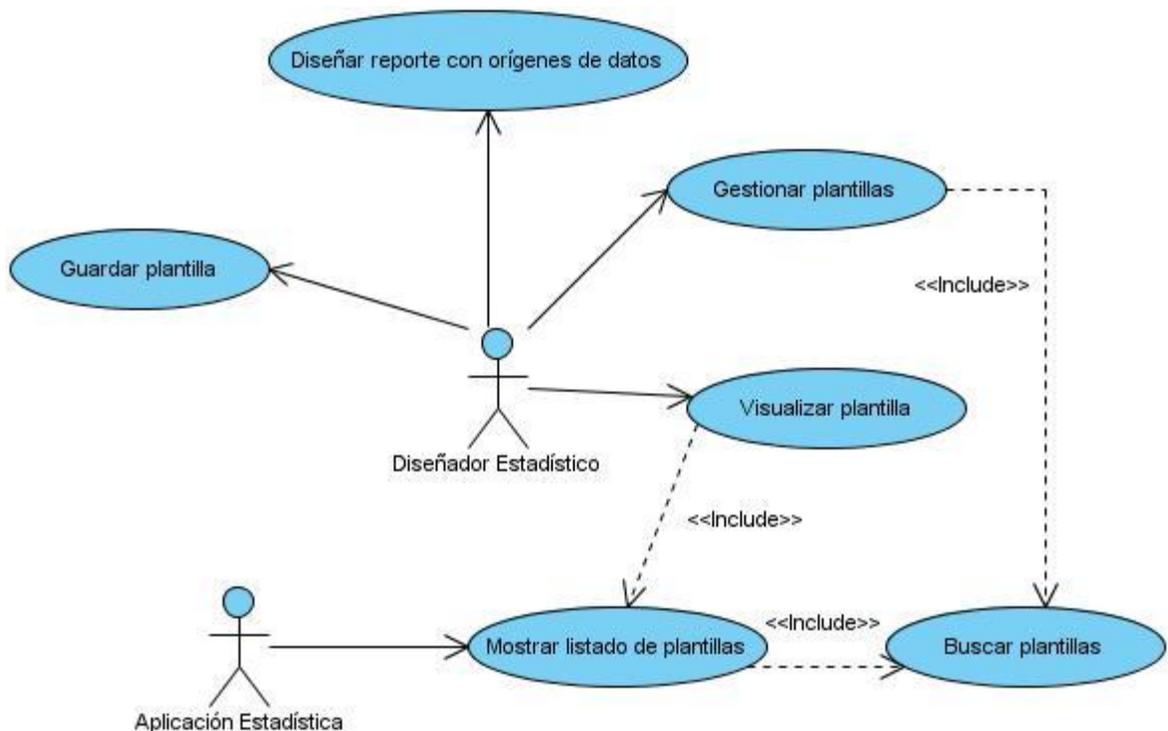


Figura 2.6 Diagrama de Caso de Uso del Paquete Editor de Reportes (Módulo de Estadísticas).

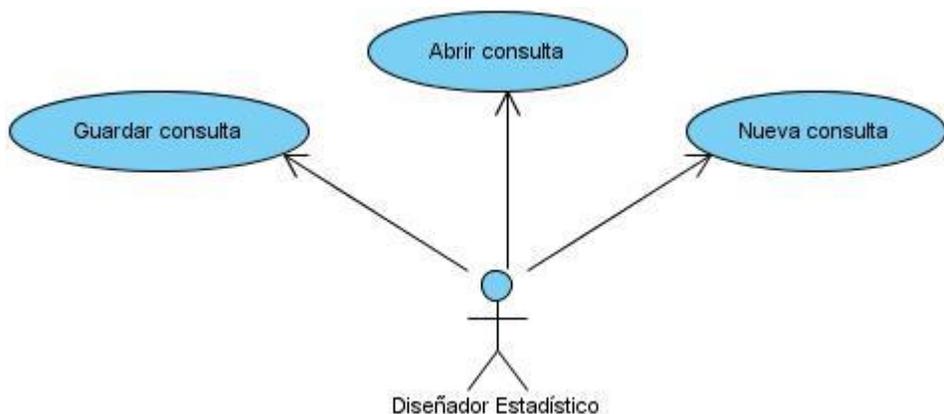


Figura 2.7 Diagrama de Caso de Uso del Paquete Ambiente.

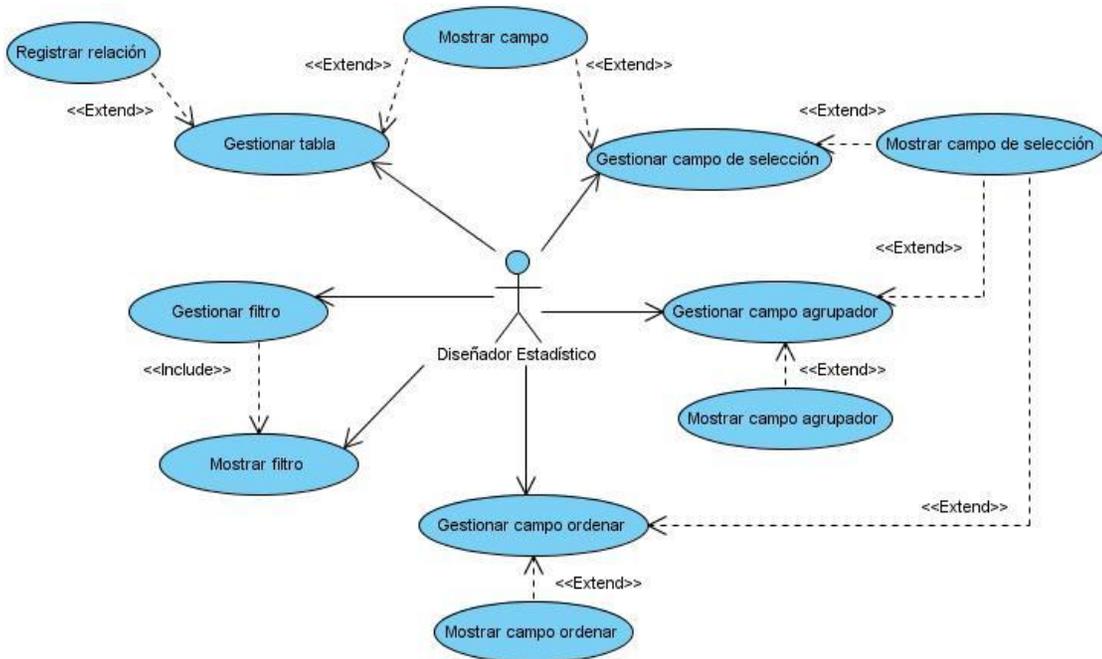


Figura 2.8 Diagrama de Caso de Uso del Paquete Sentencias de Consulta Simple.

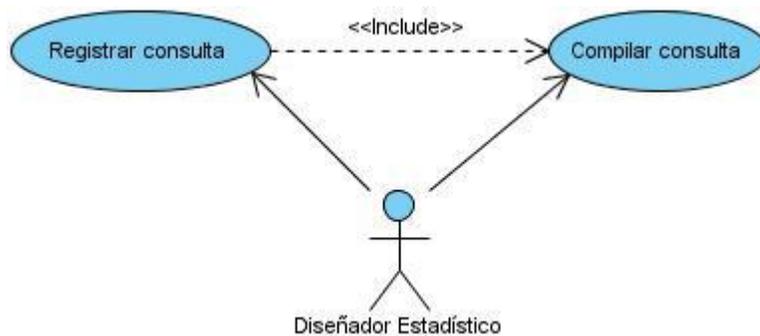
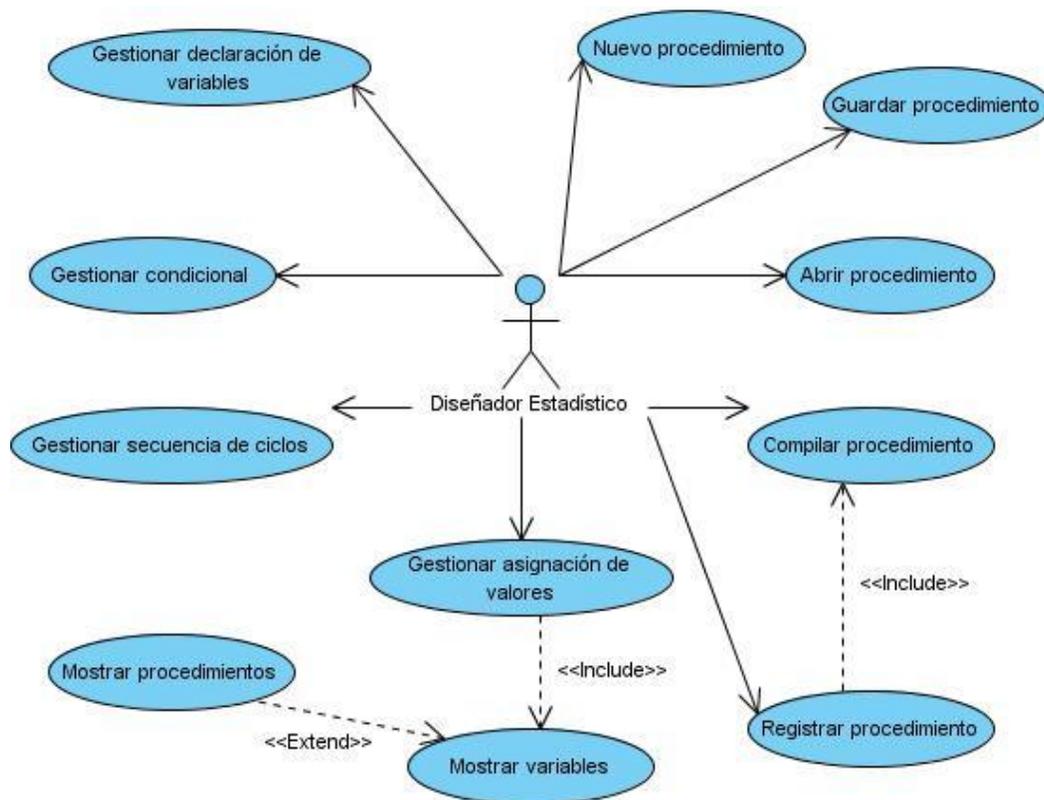


Figura 2.9 Diagrama de Caso de Uso del Paquete Complementario.



**Figura 2.10 Diagrama de Caso de Uso del Paquete Consultas Procedimentales.**

La implementación de los Casos de Uso se ha realizado por iteraciones. La tabla que se muestra a continuación representa los Casos de Uso y la iteración a la que pertenecen cada uno.

**Tabla 2.5 Casos de Uso por iteraciones.**

Nombre del Caso de Uso	Iteración
Gestionar tabla.	1.0
Mostrar campo.	
Gestionar filtro.	
Mostrar filtro.	
Gestionar campo ordenar.	
Mostrar campo ordenar.	
Mostrar campo de selección.	

Gestionar campo agrupador. Mostrar campo agrupador. Gestionar campo de selección. Registrar relación. Buscar campo de selección. Guardar consulta. Abrir consulta. Nueva consulta. Registrar consulta. Compilar consulta.	
Realizar declaración de variable. Realizar asignación de valores. Buscar variables. Buscar procedimientos. Seleccionar condicional. Estructurar secuencia de ciclos. Nuevo procedimiento. Compilar procedimiento. Registrar Procedimiento. Guardar Procedimiento. Abrir Procedimiento.	2.0
Gestionar Plantillas. Visualizar Plantilla. Guardar Plantilla. Mostrar Listado de Plantillas. Buscar Plantillas. Diseñar Reporte con orígenes de dato.	3.0

Durante el presente trabajo de diploma se procederá al Diseño de las iteraciones 1.0, 2.0 y 3.0, realizándose la implementación solamente de la primera iteración (1.0).

### **2.11 Descripción de Casos de Uso**

La descripción textual de los Casos de Uso del Sistema especificados anteriormente se expone en el Anexo 5 del presente documento.

### **2.12 Conclusiones Parciales**

En este capítulo se analizaron una serie de aspectos tales como: la información que se maneja, la propuesta del sistema, el modelo de dominio, la especificación de los requisitos de software, la definición de los actores, diagrama de paquete y de casos de uso y una descripción detallada de cada caso de uso, dando a conocer de esta forma las características del sistema.

# Capítulo 3. Diseño del Sistema

## 3.1 Introducción

En el presente capítulo se realizará el diseño del Módulo de Edición de Reportes, mostrando el diagrama de clases del diseño que se utilizará en la solución propuesta. Las clases se encuentran agrupadas por paquetes según su correspondencia con los casos de uso y como forma de lograr una mayor comprensión y claridad en la modelación.

Los propósitos del diseño son:

- Adquirir una comprensión en profundidad de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, componentes reutilizables, sistemas operativos, tecnologías de distribución y concurrencia, tecnología de interfaz de usuario, tecnologías de gestión de transacciones.
- Crear una entrada apropiada y un punto de partida para actividades de implementación subsiguientes capturando los requisitos o subsistemas individuales, interfaces y clases.
- Ser capaces de descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo, teniendo en cuenta la posible concurrencia. Esto resulta útil en los casos en que la descomposición no puede ser hecha basándose en los resultados de la captura de requisitos.
- Capturar las interfaces entre los subsistemas antes en el ciclo de vida del software. Esto ayuda cuando reflexionamos sobre la arquitectura y cuando utilizamos interfaces como elementos de sincronización entre diferentes equipos de desarrollo.
- Ser capaces de visualizar y reflexionar sobre el diseño utilizando una notación común.
- Crear una abstracción sin costuras de la implementación del sistema, en el sentido de que la implementación es un refinamiento directo del diseño que rellena lo existente sin cambiar la estructura. Esto permite la utilización de tecnologías como la generación de código y la ingeniería de ida y vuelta entre el diseño y la implementación. (8)

### 3.2 Patrones de Diseño

El término patrón surgió en 1977 cuando un arquitecto llamado Christopher Alexander publicó su primer libro "A Pattern Language" relacionado con la utilización de los mismos. "Cada patrón describe un problema que ocurre infinidad de veces en nuestro entorno, así como la solución al mismo, de tal modo que podemos utilizar esta solución un millón de veces más adelante sin tener que volver a pensarla otra vez." (24) En 1979 publica otro libro titulado "The Timeless Way of Building" donde también hacía referencias a ellos. "Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software." (25)

No obstante, no fue hasta principios de la década de 1990 cuando los patrones de diseño tuvieron un gran éxito en el mundo de la informática a partir de la publicación del libro "Design Patterns escrito por el grupo Gang of Four (GoF) compuesto por Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides, en el que se recogían 23 patrones de diseño comunes. (2)

Los Patrones Generales de Asignación de Responsabilidades (General Responsibility Assignment Patterns, GRASP por sus siglas en inglés) describen los principios fundamentales de la asignación de responsabilidades a objetos, siendo esta, una de las tareas fundamentales en el diseño de un software.

Durante el diseño del módulo de Edición de Consulta para el SIGESC fueron utilizados varios de estos patrones GRASP.

El hecho de que la capa de *Acciones* pueda crear una instancia del *Negocio* y que esta a su vez no pueda realizar la misma operación con *Acciones*, está aplicando el uso del patrón Indirección y cuando la clase *Negocio* es la encargada de utilizar la clase de *AccesoDatos* estamos en presencia del patrón No Hables con Extraños. Todo esto conlleva a que las clases se especialicen solamente en lo que deben realizar garantizando de esta forma un Bajo Acoplamiento y una Alta Cohesión.

El patrón Polimorfismo se ve reflejado en las clases *ValorAsignacion*, quien es padre de *Valor*, *Variable* y *Campo*, redefiniendo estas últimas los métodos ofrecidos por su padre.

Los patrones de diseño se clasifican en: Creacionales, Estructurales y de Comportamiento. Los empleados en el diseño de la solución en cuestión son:

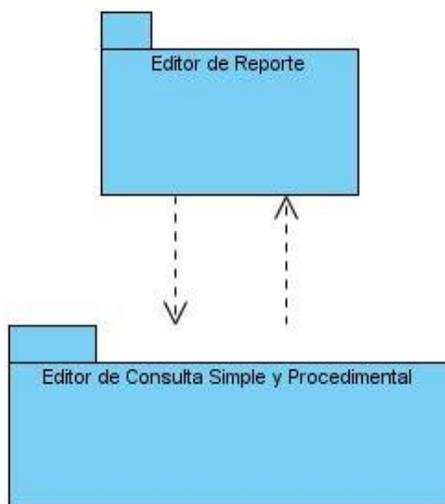
- Creacionales: Inicialización y configuración de objetos.
  - ✓ **AbstractFactory:** El problema a solucionar por este patrón es el de crear diferentes familias de objetos, como por ejemplo la creación de interfaces gráficas de distintos tipos (ventana, menú, botón, etc.). El prefijo PAF (ProxyAbstractFactory) es utilizado en clases que ejemplifican este patrón.
  - ✓ **Factory:** parte del principio de que las subclases determinan la clase a implementar. El proxy implementará el patrón “Factory” para obtener el “Puente”, permitiendo cambiarlo de ser necesario.
  - ✓ **Singleton:** (instancia única), garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. La clase *GestionarConsultaSimple* constituye un ejemplo del uso de este patrón.
- Estructurales: Separan la interfaz de la implementación. Se ocupan de cómo las clases y objetos se agrupan, para formar estructuras más grandes.
  - ✓ **Proxy:** el proxy (proxy remoto) es el encargado de abstraer la forma en que se obtiene la fábrica concreta utilizando un “Puente”. Las clases *FCNegocioConsultaSimple*, *FCInterfazConsultaSimple* y *FactoryAcople* hacen uso de este patrón.
  - ✓ **Puente:** el puente brinda un mecanismo dinámico, utilizando reflexión, para obtener objetos. Desacopla una abstracción de su implementación permitiendo modificarlas independientemente. Las Fábricas hacen uso de este patrón y las clases que lo implementan son: *PAFNegocioConsultaSimple*, *PAFInterfazConsultaSimple*, *PAFAcopleConsulta*.
  - ✓ **Fachada:** Cada capa expondrá una fachada (interfaz de acople) con el conjunto de funcionalidades que necesita la capa inmediata superior, un ejemplo de este patrón son las clases *INombre*.
- Comportamiento: Encapsula una petición en un objeto, permitiendo así parametrizar a los clientes con distintas peticiones, encolar o llevar un registro de las peticiones y poder deshacer las operaciones. Las clases de la capa de acciones son ejemplos del uso de este patrón.
  - ✓ **Comando:** las clases de la capa de “*Acciones*” son ejemplos del uso de este patrón.

### 3.3 Diagrama de Paquetes del Diseño

En un paquete de diseño se puede encontrar una colección de clases, relaciones y otros paquetes. Es usado para estructurar el modelo de diseño mediante su división en partes más pequeñas y agrupar elementos relacionados de dicho modelo con propósitos organizacionales. (8)

A continuación se muestra de forma general el diagrama de paquetes del diseño correspondiente al Módulo de Edición de Reportes ya existente y las nuevas funcionalidades a incorporar, las cuales se encuentran en el paquete Editor de Consulta Simple y Procedimental.

La relación entre ambos paquetes está dada precisamente por un fichero Sistema.xml (Ver Anexo) que se encuentra dentro del Módulo Estadística, el cual es inicializado por el método ConfigurarSistema().



**Figura 3.1 Diagrama de Paquetes del Diseño del Módulo de Edición de Reportes con nuevas funcionalidades incluidas.**

La figura 3.2 representa el diagrama de paquetes del paquete Editor de Reportes, al que se incluyeron dos paquetes nombrados Diseño de Plantillas y Diseño de Reportes que contienen parte de las nuevas funcionalidades a incorporar.

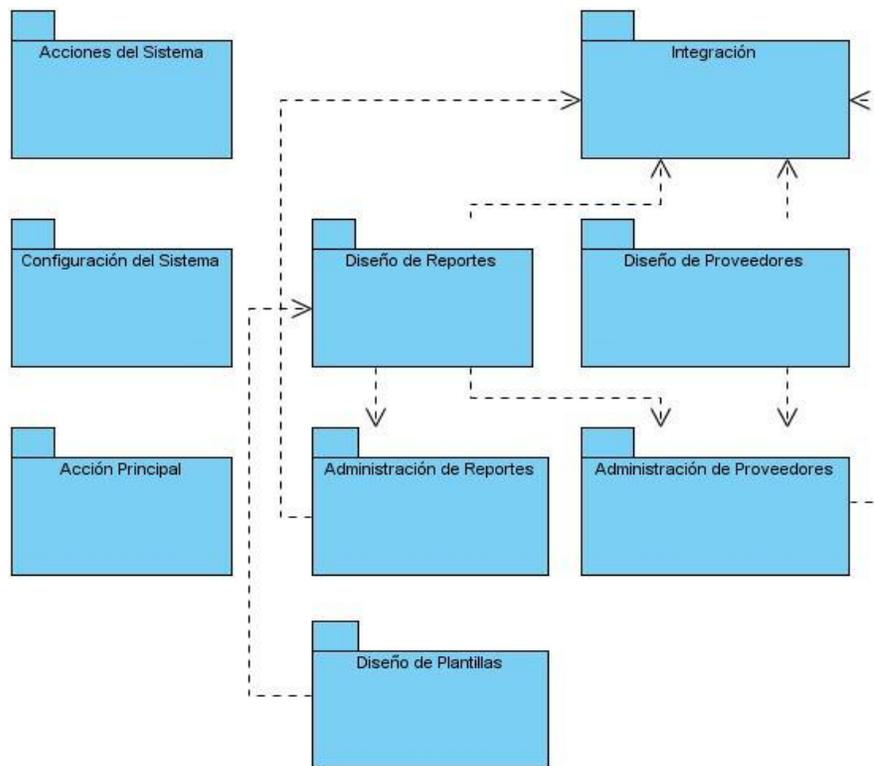


Figura 3.2 Diagrama de Paquetes del paquete Editor de Reportes.

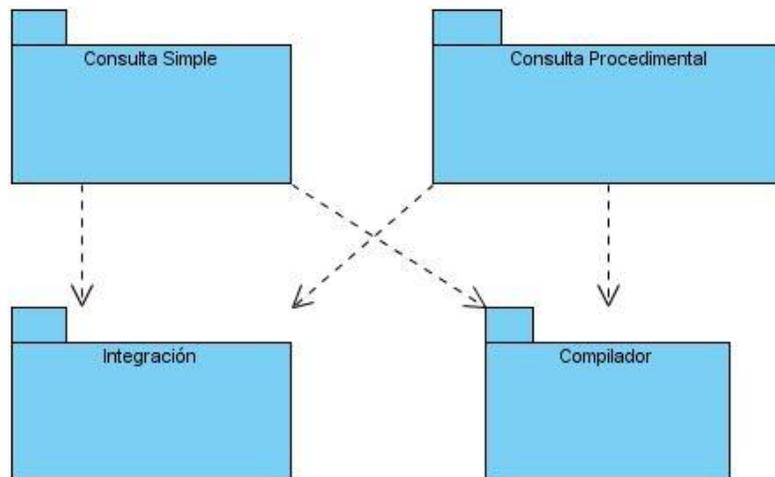
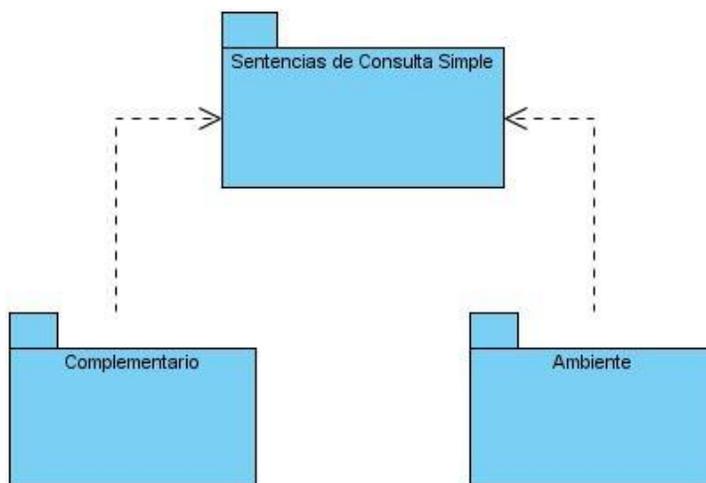


Figura 3.3 Diagrama de Paquetes del Paquete Editor de Consulta Simple y Procedimental.

Las figuras 3.4 y 3.5 representan diagramas de paquetes pertenecientes a los paquetes de Consulta Simple y Consulta Procedimental respectivamente, logrando un mayor entendimiento y comprensión de la lógica utilizada.



**Figura 3.4 Diagrama de paquetes del Paquete "Consulta Simple".**

### 3.4 Diagrama de Clases del Diseño

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación. Contiene clases, asociaciones y atributos, interfaces y operaciones. Otro de sus elementos son: métodos, navegabilidad y dependencias. (8)

Para una mejor comprensión de los diagramas de clases del diseño se propone una leyenda con una escala de colores representando las diferentes capas de la aplicación.

## Leyenda

	Clases Capa Interfaz
	Clases Capa Acciones
	Clases Capa Negocio
	Clases Capa Acceso a Datos
	Clases Capa Negocio
	Instancias de otros Paquetes

Los diagramas de clase del diseño se exponen en el Anexo 6 del presente documento.

### 3.5 Conclusiones Parciales

En este capítulo se expuso el diseño del sistema, contemplando todos los diagramas de clases del diseño del módulo Edición de Consulta, así como las relaciones entre clases y paquetes, quedando de esta forma diseñada la solución propuesta.

# Capítulo 4. Implementación

## 4.1 Introducción

En el presente capítulo se describirá la implementación del Módulo de Edición de Reportes. Como punto de partida se comenzó con el resultado del diseño y el objetivo fundamental es desarrollar la arquitectura y el sistema como un todo. La implementación es el centro durante las iteraciones de construcción.

Los propósitos de la implementación son:

- Planificar las integraciones de sistemas necesarias en cada iteración. Para ello se sigue un enfoque incremental, lo que da lugar a un sistema que se implementa en una sucesión de pasos pequeños y manejables.
- Distribuir el sistema asignando componentes ejecutables a nodos en el diagrama de despliegue. Esto se basa fundamentalmente en las claves activas encontradas durante el diseño.
- Implementar las clases y subsistemas encontrados durante el diseño. En particular las clases se implementan como componentes de fichero que contienen código fuente.
- Probar los componentes individualmente, y a continuación integrarlos compilándolos y enlazándolos en uno o más ejecutables, antes de ser enviados para ser integrados y llevar a cabo las comprobaciones de sistema. (8)

## 4.2 Diagrama de Despliegue

Un diagrama de despliegue modela la tipología del hardware sobre el que se ejecuta un sistema. Muestra la configuración de los nodos que participan en la ejecución y de los componentes que residen en ellos. Un nodo es un elemento físico que existe en tiempo de ejecución. Representa un recurso computacional que, por lo general, tiene memoria y capacidad de almacenamiento. Representa a un procesador o un dispositivo sobre el cual se despliegan los componentes. (4)

La aplicación de Estadística forma parte del sistema integrado SIGESC, por lo que se representa a continuación una parte de ese diagrama de despliegue donde interviene la aplicación en cuestión.

En el diagrama se representa una estación de trabajo por cada tipo de aplicación que compone al SIGESC, aunque varias aplicaciones del sistema pueden convivir en una misma estación de trabajo.

El diagrama de despliegue así como la explicación de cada nodo fue referenciado del Modelo de Despliegue –SIGESC v 1.2.

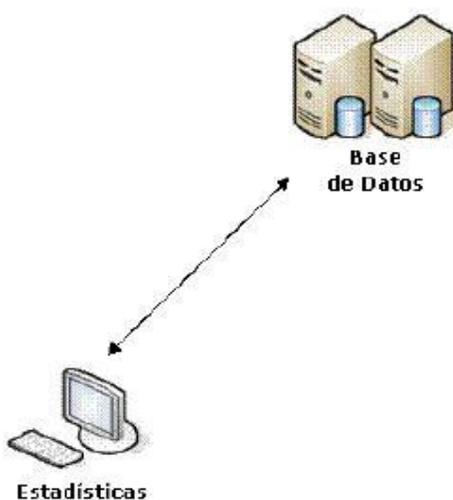


Figura 4.1 Diagrama de Despliegue de Estadística.

### 4.3 Descripción de Nodos

#### 4.3.1 Nodo Servidores de BD

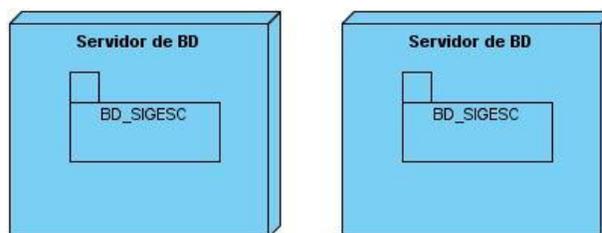


Figura 4.2 Servidores de BD.

Se recomienda la utilización de dos servidores en clúster para garantizar mayor vitalidad en el sistema de bases de datos.

Requerimientos de hardware:

- Tecnología Intel.
- 2 procesadores duales de 2.6 GHz
- 4 GB de memoria RAM.
- 2 discos de 72 GB en RAID1.
- 500 GB en RAID1 para el almacenamiento de los datos. Debe ser almacenamiento compartido si se utilizan dos servidores.

Requerimientos de software:

- Sistema Operativo Red Hat Linux Enterprise Advanced Server 4.0.
- Gestor de bases de datos Oracle Database 10g R2.
  - Oracle Real Application Clusters. Solo necesario si se usan dos servidores.
  - Oracle Spatial.

Otros requerimientos:

- Conexión mediante el protocolo TCP/IP a la red local.
- Dirección IP fija.
- Conexión de red a 1 Gigabit.
- Cada servidor debe cumplir todos los requerimientos antes mencionados.

### 4.3.3 Nodo PC Estadísticas

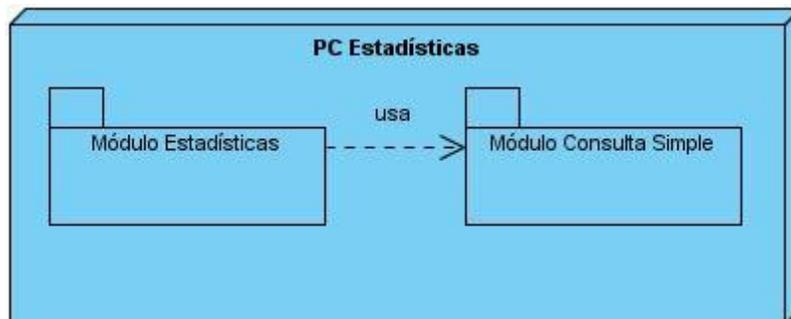


Figura 4.3 PC Estadística.

Requerimientos de hardware:

- Tecnología Intel.
- Procesador Pentium IV 3.0 GHz
- 512 MB de memoria de RAM.
- Espacio libre en disco: 125 MB.

Requerimientos de software:

- Sistema operativo Microsoft Windows XP SP2.
- Microsoft .NET Framework versión 2.0.
- Framework DMAX versión 2.0.
- Servicios DMAX versión 2.0.
- Cliente de conectividad a Oracle 10g.

Otros requerimientos:

- Conexión mediante el protocolo TCP/IP a la red local.

- Dirección IP fija.
- Conexión de red a 100 Megabit.
- Resolución gráfica de 1024x768 pixeles. (4)

#### **4.4 Diagrama de Componentes**

Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases en el diseño. (8)

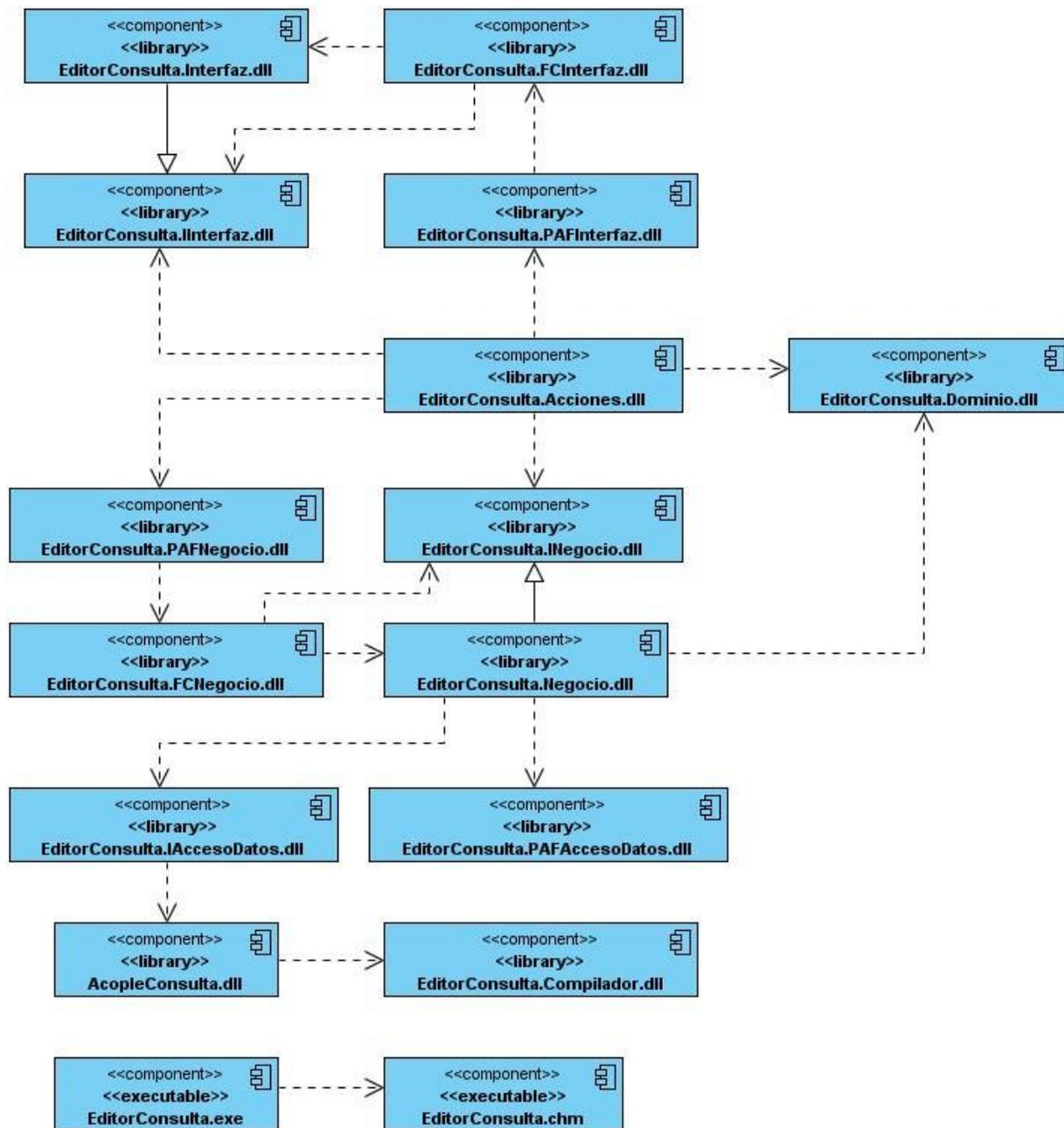


Figura 4.4 Diagrama de Componentes.

#### **4.5 Conclusiones Parciales**

Al concluir este capítulo se ha realizado el diagrama de despliegue, observándose la distribución física de cada uno de los componentes del módulo a desarrollar, así como su descripción. Se realizó además el diagrama de componentes.

## *Conclusiones Generales*

Al finalizar el trabajo de diploma Sistema de Gestión de Emergencia de Seguridad Ciudadana Módulo de Edición de Reportes, se concluye que se ha dado cumplimiento al objetivo general trazado durante el inicio del mismo. Se realizó un estudio acerca de las principales herramientas de generación de reportes existentes tomando de cada una de ellas sus aspectos positivos.

Se realizó el diseño de la gestión de plantillas y de la asignación de varios orígenes de datos a un mismo reporte permitiendo tener una mayor comprensión del tema para futuras implementaciones. Se obtuvo una aplicación para editar consultas simples. Se obtuvieron además una serie de artefactos que se fueron generando en cada fase del proceso de desarrollo del software.

## *Recomendaciones*

Implementar las iteraciones 2.0 y 3.0 descritas en el Capítulo 2 del presente documento.

Se recomienda la utilización de un área de modelado donde poder elaborar los procedimientos almacenados.

Integrar en un entorno único las aplicaciones que brindan las funcionalidades para edición de consultas, edición de procedimientos y la administración de los reportes.

Analizar la posible implementación del Módulo de Edición de Consulta con herramientas libres para contribuir en el proceso de migración que lleva a cabo el gobierno de la República Bolivariana de Venezuela.

Se recomienda la utilización del entorno para expandir sus funcionalidades e integrarlo con otros negocios o sistemas existentes y además otros gestores de bases de datos.

## Referencias Bibliográficas

1. **Pascal, Universidad Blas.** Todo Ambiente. *Todo Ambiente*. [En línea] 2010. [Citado el: 1 de Abril de 2011.] <http://www.ubp.edu.ar/todoambiente/seguridad.php>.
2. **Cruz, Javier Fernández.** *Sistema de Gestión de Emergencias de Seguridad Ciudadana (171) Entorno Integrado de Gestión Estadística*. Ciudad de la Habana : Universidad de las Ciencias Informáticas, 2010.
3. **Peña, Yahima Vigo Valdés y Yordan Nuez.** *Sistema de Gestión de Emergencias de Seguridad Ciudadana (171) Módulo de Despacho*. Ciudad de la Habana : Universidad de las Ciencias Informáticas, 2007.
4. **ALBET Ingeniería y Sistemas.** *Modelo de Despliegue*. 2007. CT-SW-DR-031701.
5. **Maranje Agramonte, Adrian y Alonso Riverón, Yisel.** *Modelo de Despliegue*. Ciudad de La Habana : s.n., 2007.
6. Diseño de consultas en Access. [En línea] [Citado el: 29 de Marzo de 2011.] <http://mortuux.wordpress.com/2009/10/26/disenio-de-consultas-en-access-qbe/>.
7. El Rincón del BI. Descubriendo el Business Intelligence... [En línea] [Citado el: 31 de Marzo de 2011.] <http://churriwifi.wordpress.com/2010/07/15/17-4-reporting-en-pentaho-con-pentaho-report-designer-otras-posibilidades-de-reporting-birt-y-jasperreports/>.
8. **Ivar Jacobson, Grady Booch y James Rumbaugh.** El Proceso Unificado de Desarrollo del Software. [En línea] 2000. [Citado el: 4 de Marzo de 2011.] <http://bibliodoc.uci.cu/pdf/reg00060.pdf>. ISBN: 84-7829-036-2.
9. *Metodologías de Desarrollo de Software*. [PDF]

10. **Larman, Craig.** *UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado.* 2da Edición en Español. México : Prentice Hall, 2003. pág. 520. ISBN 8420534382.
11. **Pressman, Roger S.** *Ingeniería del Software. Un enfoque Práctico.* [trad.] Darrel Ince. 5ta Edición. s.l. : Mc Graw Hill, 2001. pág. 614.
12. Resumen de la Metodología IDEF0. [En línea] [Citado el: 29 de Marzo de 2011.] <http://www.aqa.es/doc/Metodologia%20%20IDEF0%20Resumen.pdf>.
13. **Sánchez Tellez, Eddy.** *Especificación de la Arquitectura Base del Sistema de Gestión de Emergencia y Seguridad Ciudadana 171.* Ciudad de la Habana : Universidad de las Ciencias Informáticas, 2008.
14. **Catalinas, Enrique Quero.** *Sistemas operativos y lenguajes de programación.* s.l. : Paraninfo - España, 2002. ISBN: 8497321502.
15. **Microsoft Corporation.** [En línea] 2008. [Citado el: 5 de Abril de 2011.] <http://msdn.microsoft.com/es-es/library/hfa3fa08%28VS.80%29.aspx>.
16. Microsoft .Net su Framework por dentro y las bases para la Construcción de Aplicaciones. [En línea] [http://biblioteca.usac.edu.gt/tesis/08/08\\_5635.pdf](http://biblioteca.usac.edu.gt/tesis/08/08_5635.pdf).
17. emagister.com. [En línea] [Citado el: 20 de Marzo de 2011.] [http://grupos.emagister.com/documento/herramientas\\_de\\_reportes\\_para\\_net/1050-2035](http://grupos.emagister.com/documento/herramientas_de_reportes_para_net/1050-2035).
18. ¿Qué es un Sistema Gestor de Bases de Datos o SGBD? [En línea] 4 de Marzo de 2011. <http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd/>.
19. Oracle Support Polícies. [En línea] [Citado el: 5 de Mayo de 2011.] <http://www.oracle.com/us/support/library/057419.pdf>.
20. [En línea] [Citado el: 6 de Abril de 2011.] <http://www.wordreference.com/definicion/modelado>.
21. [En línea] [Citado el: 6 de Abril de 2011.] <http://www.wordreference.com/definicion/%C3%A1rea>.
22. **A. A. V., S. Ravi, and U. J. D.** *Compilers: principles, techniques, and tools.* Boston, MA, USA : Addison-Wesley Longman Publishing Co., Inc., 1986.

23. **ALBET Ingeniería y Sistemas.** *Especificaciones Suplementarias.* Ciudad de la Habana : s.n., 2007. CT-SW-DR-031601.
24. **Alexander, Christopher.** *A Pattern Language.* Nueva York : Oxford University Press, 1977.
25. **Tedeschi, Nicolás.** ¿Qué es un patrón de Diseño? [En línea] [Citado el: 1 de Abril de 2011.] <http://msdn.microsoft.com/es-es/library/bb972240.aspx>.
26. **DataDynamics.** [En línea] [Citado el: 5 de Abril de 2011.] <http://www.datadynamics.com/Downloads/ar2stdPDF.pdf>.

# Bibliografía

1. **Cabrera Díaz, Manfred Ramón.** *Arquitectura y Aplicaciones del SIGESC. Introducción a Microsoft .NET.* [Power Point] Ciudad Habana. : s.n., 2010.
2. **James Rumbaugh, Ivar Jacobson y Grady Booch.** *El Lenguaje Unificado de Modelado. Manual de Referencia.* s.l. : Mc Graw Hill, 2001.
3. Ingeniería del Software. [En línea] [Citado el: 3 de Marzo de 2011.] [http://www.um.es/docencia/barzana/IAGP/Enlaces/CASE\\_principales.html](http://www.um.es/docencia/barzana/IAGP/Enlaces/CASE_principales.html).
4. **Greenwald, Rick, Stackowiak, Robert y Stern, Jonathan.** *Oracle Essentials. Oracle Database 10g.*
5. **ALBET Ingeniería y Sistemas.** *Manual de Instalación.* 2007. CT-SW-DR-031801.
6. La plataforma Pentaho Open Source Business Intelligence . [En línea] [Citado el: 31 de Abril de 2011.] <http://egkafati.bligoo.com/content/view/219538/La-plataforma-Pentaho-Open-Source-Business-Intelligence.html>.
7. **Tresserras, Francesc Xavier Berjano.** *La Estrategia Open Source en Soluciones de Business Intelligence.* [PDF]
8. **Coulouris, George.** *Sistemas Distribuidos-Primera Edición.* Madrid : Addison Wesley, 2001. ISBN: 9788478290499 .
9. **Ballinger, David S. Platt y Keith.** *Microsoft .Net Introducing-Primera Edición.* s.l. : Microsoft Press, 2001.
10. Manual Generador de Reportes de Pentaho. [En línea] [Citado el: 3 de Abril de 2011.] [http://www.onuva.com/files/pentaho/Manual\\_PRD.pdf](http://www.onuva.com/files/pentaho/Manual_PRD.pdf).

11. **Rodríguez, Dr. Julio Cerezal Mezquita y Dr. Jorge Fiallo.** *¿Cómo investigar en Pedagogía?* Ciudad de la Habana : s.n., 2005.
12. Clikear.com. [En línea] [Citado el: 10 de Abril de 2011.]  
<http://www.clikear.com/manuales/csharp/c10.aspx>.

## *Glosario de Términos*

**SIGESC:** Sistema de Gestión de Emergencia de Seguridad Ciudadana.

**PL/SQL:** Lenguaje Procedimental/Query.

**AVL:** Localización Automática de Vehículos.

**GPS:** Sistema de Posicionamiento Global.

**PDF:** Formato de Documento Portátil.

**RTF:** Formato de texto enriquecido (Rich Text Format).

**TIFF:** Formato de archivo para almacenar imágenes (Tagged Image File Format).

**RUP:** Proceso Unificado de Rational.

**UML:** Lenguaje Unificado de Modelado (Unified Modeling Language).

**API:** Interfaz de Programación de Aplicaciones.

**IDE:** Entorno Integrado de Desarrollo.

**CLR:** Entorno Común de Ejecución para Lenguajes (Common Language Runtime).

**BCL:** Librería de Clases Base (Base Class Library).

**POO:** Programación Orientada a Objetos.

**ODBC:** Abrir conexión con Base de Datos (Open data base Connectivity).

**ASP:** Active Server Pages.

**DLL:** Biblioteca de vínculos dinámicos (Dynamic-Link Library).

**CASE:** Ingeniería de Software asistida por computadora (Computer Aided Software Engineering).

**ORM:** Mapeo Objeto-Relacional.

**CCS:** Hojas de Estilo en Cascada.

**XML:** Lenguaje de Marcas Extensible (eXtensible Markup Language).

**SGBD:** Sistema Gestor de Base de Datos.

**DBMA:** Sistema de Administración de Base de Datos (Data Base Management System).

**RDBMS:** Sistema de Gestión de Base de Dato Relacional (Relational Data Base Management System).