

Universidad de las Ciencias Informáticas

FACULTAD 6



Título: “Módulo de Reportes Estándares para SIGE”

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas.

Autor:

Victor Manuel Estrada Grillo.

Tutores:

Ing. Claudia García Suárez del Villar.

Ing. Héctor Luis Reyes Zaldívar.

Consultante:

Lic. Elena Leonila Fernández García.

La Habana, Junio de 2013

“Año 55 de la Revolución”



Aprende como si fueras a vivir toda la vida y vive como si fueras a morir mañana.

Charles Chaplin

Declaración de autoría

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis y reconozco 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 _____.

Victor Manuel Estrada Grillo

Firma del Autor

Claudia García Suárez del Villar

Firma del Tutor

Héctor L. Reyes Zaldívar

Firma del Tutor

Datos de contacto

DATOS DE CONTACTO

Victor Manuel Estrada Grillo

Universidad de las Ciencias Informáticas

e-mail: vmestrada@estudiantes.uci.cu

Tutor:

Claudia García Suárez del Villar

Ingeniera en Ciencias Informáticas

e-mail: cgarcias@uci.cu

Tutor:

Héctor Luis Reyes Zaldívar

Ingeniero en Ciencias Informáticas

e-mail: hlreyes@uci.cu.

Agradecimientos

AGRADECIMIENTOS

A mis tutores Claudia y Tico.

A Adrian.

A mis amigos y familiares.

Dedicatoria

DEDICATORIA

A mis padres.

Por cuidarme siempre.

RESUMEN

La plataforma para la toma de decisiones (PATDSI) perteneciente al centro de Gestión de Datos (DATEC) cuenta con un sistema para la gestión estadística (SIGE). Esta herramienta en su conjunto permite definir las formas en que se capturará información, dígame encuestas económicas y formularios estadísticos. Además posibilita el diseño de reportes de forma dinámica apoyada en el Generador Dinámico de Reportes (GDR). Los usuarios de SIGE requieren generar un tipo de reporte, el mismo cuenta con en formato estándar similar al formulario del cual se genera. Para la creación de estos reportes de tipo estándar se estableció un patrón en la creación los mismos. El objetivo del presente trabajo de diploma es desarrollar un módulo que permita crear reportes que cuentan con una información predefinida y que es de poca variedad. De esta forma los usuarios de SIGE pueden generar este tipo de reportes de forma automática sin la utilización de GDR. El uso de la herramienta GDR conlleva más trabajo, pues los reportes hay que diseñarlos manualmente y de cada formulario se generan varios reportes estándares. Como resultado se obtuvo un módulo integrado a SIGE que permite la creación de reportes estándares de forma automática y simple a los usuarios.

PALABRAS CLAVE

Reporte Estándar, SIGE, formularios, GDR.

Tabla de contenido

TABLA DE CONTENIDOS.	
DECLARACIÓN DE AUTORÍA	I
DATOS DE CONTACTO.....	2
AGRADECIMIENTOS.....	I
DEDICATORIA	2
RESUMEN.....	3
TABLA DE CONTENIDOS.	4
ÍNDICE DE FIGURAS.....	7
INTRODUCCIÓN.....	1
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA	4
1.1. Conceptos Estadísticos.....	4
1.2. Las Estadísticas en Cuba.	5
1.3. Sistema Integrado de Gestión Estadística (SIGE).....	6
1.4. Reportes Estándares.....	11
1.5. Lenguaje de programación.....	12
1.5.1. Lenguaje de programación del lado del servidor. PHP.	13
1.5.2. Lenguaje de Programación del lado del cliente. Java Script.	14
1.6. Metodología de desarrollo.	14
1.7. Proceso Unificado Abierto (OpenUP).	15
1.8. Lenguaje de Modelado. UML.	16
1.9. Herramienta CASE. Visual Paradigm.	16
1.10. Marco de trabajo de desarrollo.	17
1.10.1. Marco de trabajo de desarrollo del lado del cliente. ExtJS.	17
1.10.2. Marco de trabajo de desarrollo del lado del servidor. Symfony.	18
1.11. Entorno de desarrollo integrado (IDE). NetBeans.	19
1.12. Gestor de base de datos. PostgreSQL.....	20

Tabla de contenido

1.13. Aplicación gráfica para gestionar el gestor de bases de datos. PGAdmin III.....	20
Conclusiones parciales.	21
CAPÍTULO 2: ANÁLISIS Y DISEÑO DEL SISTEMA.	22
2.1. Modelo Dominio.....	22
2.2. Requisitos del Sistema.....	25
2.2.1. Requisitos funcionales.	25
2.3. Requisitos no funcionales.	27
2.4. Diagrama de Casos de Uso del Sistema.	28
2.5. Objetivos del diseño.....	32
2.6. Diagrama de Clases del Diseño.	33
2.7. Patrones arquitectónicos.....	35
2.8. Diagramas de Secuencia.....	37
2.9. Diagrama de clases persistentes.	39
2.10. Modelo de datos.	40
2.11. Vista de Despliegue.	41
Conclusiones parciales.	42
CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA.....	43
3.1. Modelo de Implementación.....	43
3.2. Diagrama de Componentes.....	43
3.3. Código Fuente.	45
3.4. Estándares de Codificación.....	45
3.5. Generación de Reportes Estándares.....	46
3.6. Pruebas.....	49
3.7. Casos de prueba.....	51
Conclusiones Parciales.....	52
CONCLUSIONES.....	53
RECOMENDACIONES	54

Tabla de contenido

REFERENCIAS BIBLIOGRÁFICAS.....	55
BIBLIOGRAFÍA.....	57
ANEXOS.....	59
Anexo1. Descripción de caso de uso CU_Imprimir Reporte.....	59
Anexo2. Diagrama de Clases del Diseño. CU_Imprimir Reporte Estándar.	60
Anexo3. Diagrama de Secuencia del CU_Imprimir Reporte.....	61
Anexo4. Diagrama de Componentes. CU_Imprimir Reporte Estándar.....	61
GLOSARIO.....	62

Índice de figuras

ÍNDICE DE FIGURAS.

FIGURA 1. MÓDULOS DE SIGE	20
FIGURA 2. MODELO DE DOMINIO.....	32
FIGURA 3. DIAGRAMA DE CASOS DE USO. CU_GENERAR REPORTES ESTÁNDARES.....	38
FIGURA 4. DIAGRAMA DE CLASES DEL DISEÑO. CU_GENERAR REPORTES ESTÁNDARES.....	42
FIGURA 5. DIAGRAMA DE SECUENCIA. CU_GENERAR REPORTES ESTÁNDARES.	45
FIGURA 6. DIAGRAMA DE CLASES PERSISTENTES.....	47
FIGURA 7. DIAGRAMA DE ENTIDAD RELACIÓN.....	48
FIGURA 8. DIAGRAMA DE DESPLIEGUE.....	49
FIGURA 9. DIAGRAMA DE COMPONENTES. CU_GENERAR REPORTE ESTÁNDARES.....	51
FIGURA 10. VISTA INTERFAZ DE USUARIO DE LA FUNCIONALIDAD SELECCIONAR PLANTILLAS..	54
FIGURA 11. VISTA INTERFAZ DE USUARIO DE LA FUNCIONALIDAD SELECCIONAR MAESTRO. ...	55
FIGURA 12. VISTA INTERFAZ DE USUARIO FUNCIONALIDAD SELECCIONAR ESPECIFICACIONES.	56
FIGURA 13. EJEMPLO DE REPORTE ESTÁNDAR TOTAL PROVINCIAL DE LA PROVINCIA HABANA.	57
FIGURA 14. DIAGRAMA DE CLASES DEL DISEÑO. CU_IMPRIMIR REPORTES ESTÁDARES.	68
FIGURA 15. DIAGRAMA DE SECUENCIA. CU_IMPRIMIR REPORTES ESTÁNDARES.	68
FIGURA 16. DIAGRAMA DE COMPONENTES. CU_IMPRIMIR REPORTES ESTÁNDARES.	69

Introducción

INTRODUCCIÓN

Las estadísticas oficiales desempeñan una función cada vez mayor en nuestras sociedades y administraciones públicas. Datos estadísticos de elevada calidad son ahora de fácil acceso en los planos nacional, regional y mundial. Los sistemas estadísticos nacionales a los que se ha encargado reunir y difundir estadísticas oficiales han evolucionado hasta convertirse en instituciones nacionales importantes en todos los países.

En Cuba la Oficina Nacional de Estadísticas e Información (ONEI) es la encargada de proponer, organizar, ejecutar y controlar, según corresponda, la aplicación de la política estatal en materia de estadísticas oficiales de la República de Cuba. El Consejo de Ministros, acuerda que toda empresa, organismo, organización, o institución identificada como centro informante y poseedor de información de interés para la toma de decisiones de los órganos de la administración del Estado está en la obligación de reportar su información estadística.

Con el objetivo de automatizar los procesos de gestión estadística en Cuba la ONEI en conjunto con el departamento de Soluciones Integrales perteneciente al Centro de Tecnologías de Gestión de Datos (DATEC) en la Universidad de las Ciencias Informáticas (UCI), desarrollan actualmente el Sistema Integrado de Gestión Estadística (SIGE). La herramienta en su conjunto permite definir las formas en que se capturará información, dígame encuestas económicas y formularios estadísticos. Además posibilita el diseño de reportes de forma dinámica apoyada en el Generador Dinámico de Reportes (GDR).

Sin embargo, existen una serie de reportes muy importantes para la toma de decisiones que cuentan con un formato estándar muy similar al del formulario a partir del cual se genera. Los mismos son resúmenes estadísticos cuyo contenido varía según rango de fecha y nivel (nacional, provincial, municipal o de base). Anualmente, acorde al movimiento económico y estructural de nuestro país, la ONEI emite el Sistema de Información Estadística Nacional (SIEN) que incluye todos los formularios estadísticos, entre los que se encuentran nuevos diseños asociados a los cuales deben diseñarse también una serie de reportes estándares. Esto implica una considerable inversión de horas/hombres cada vez que sea modificado el SIEN. Teniendo en cuenta el hecho de la existencia de una estandarización en el diseño de los reportes este proceso puede ser automatizado y por lo tanto garantizar un importante ahorro de tiempo y esfuerzo para el cliente.

Introducción

A partir de la situación descrita anteriormente, se presenta como **Problema de la investigación**: ¿Cómo crear reportes estándares para SIGE que faciliten el trabajo y la visualización de información estadística de uso frecuente? Este problema se define con el **Objeto de estudio**: los Sistemas de Gestión Estadística, el cual está delimitado por el **Campo de acción**: la generación de reportes.

Para resolver el problema planteado surge como **Objetivo general**: Desarrollar un módulo que permita crear reportes estándares para SIGE, facilitando el trabajo y la visualización de información estadística de uso frecuente.

Del Objetivo general se desglosan los siguientes **Objetivos específicos**:

- Realizar un estudio del marco conceptual relacionado con los sistemas de gestión estadística.
- Realizar el análisis y diseño del componente para la generación de reportes estándares.
- Implementar el componente para la generación de reportes estándares.
- Validar el componente para la generación de reportes estándares.

Para dar cumplimiento a los objetivos planteados, se deben realizar las siguientes **Tareas de investigación**:

- Revisión bibliográfica de las herramientas y metodología a utilizar en el desarrollo del módulo.
- Elaboración de los artefactos según la metodología de desarrollo de software y requisitos mínimos definidos en los lineamientos para las tesis de la facultad.
- Identificación de los requisitos funcionales y no funcionales para el correcto funcionamiento del módulo de reportes estándares para SIGE.
- Confección del modelo de casos de uso del módulo de reportes estándares para SIGE.
- Elaboración del modelo de diseño.
- Elaboración del modelo de implementación.
- Diseño de los casos de prueba definidos a partir de los requisitos funcionales.
- Realización de pruebas funcionales y otros tipos de pruebas para comprobar el correcto funcionamiento del módulo.

Introducción

Obteniendo el resultado esperado:

- Módulo que permita realizar reportes estándares para SIGE facilitando la visualización de información estadística de uso frecuente.

Estructuración del trabajo de diploma:

Capítulo 1: Fundamento Teórico

Este capítulo contiene los fundamentos teóricos para entender el problema a solucionar y conceptos fundamentales. Un análisis de las tecnologías, las herramientas y metodología a utilizar para desarrollar el sistema.

Capítulo 2: Análisis y Diseño.

Se presenta el desarrollo del módulo a través de la discusión de los artefactos resultados de las disciplinas Análisis y Diseño requeridos para la elaboración y construcción del mismo.

Capítulo 3: Implementación y pruebas de la solución.

Comienza con el resultado del diseño, implementando el sistema en términos de componentes, así como una revisión final de las especificaciones del diseño y la realización de pruebas, garantizando la calidad del software.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En este capítulo se presenta la definición del marco teórico de la investigación. Se exponen los conceptos estadísticos necesarios para una mejor comprensión del trabajo desarrollado. Incluyendo el análisis de las tecnologías, metodologías, lenguajes de programación y herramientas más idóneas para el desarrollo del tipo de aplicación que se implementará.

1.1. Conceptos Estadísticos.

Para la mejor comprensión del presente trabajo de diploma es necesario conocer los siguientes conceptos:

Sistema de Información Estadístico (SIE): El Sistema de Información Estadística permite producir y difundir información estadística de una forma distribuida, homogénea, intuitiva y asistida, sin necesidad de conocer herramientas específicas, lo que redundará por un lado en un incremento de la cantidad y calidad de la información con independencia del Departamento u Organismo que ejecute la operación, y por otro en agilizar la planificación de los Departamentos u Organismos Gestores (1).

Hoy en día es generalmente aceptado que un SIE sea un sistema en línea que permite las siguientes funciones:

- Consulta y recuperación de datos estadísticos.
- Consultas de metainformación estadística.
- Análisis matemático y gráfico de los datos.
- Navegación a través de conceptos (temáticos, temporales, territoriales) para la localización de información.
- Acceso asistido a través de un interface gráfico de usuario.
- Acceso a documentación sobre el diseño de la operación estadística.

Y que sus principales características sean:

- Diccionario de metainformación.
- Biblioteca de documentación sobre operaciones estadísticas.

Capítulo 1. Fundamentación teórica

- Conjunto de herramientas de gestión de datos y análisis.
- Herramientas de creación y modificación de metainformación/documentación
- Interface para acceder a otros sistemas de información.
- Matriz de seguridad para proteger la información confidencial.

Para entender los principios de diseño de un SIE es necesario conocer los siguientes conceptos:

Registros: para desarrollar el proceso captación y procesamiento de la información estadística, es necesario disponer de registros, que identifiquen los distintos centros informantes (2).

Clasificadores: permiten ordenar y expresar en un lenguaje adecuado, los distintos atributos a captar de tales centros informantes. Los clasificadores además ocupan un lugar en el orden conceptual y metodológico, pues permiten identificar el alcance de los objetos clasificados (2).

1.2. Las Estadísticas en Cuba.

En la sociedad actual el dominio de la información ha tomado un papel significativo. Tanto es así que el poseer el control de los datos estadísticos dentro de la infraestructura de una entidad socio-económica, dígase gobierno, empresa u otro tipo de organizaciones es de vital importancia para la toma de decisiones.

Cuba no se queda atrás, la Oficina Nacional de Estadísticas e Información (ONEI) es una institución gubernamental adscrita al Ministerio de Economía y Planificación: Su objetivo es organizar, dirigir, controlar y regular la actividad de la estadística en el ámbito nacional. Para la gestión de la información estadística se apoya en su Sistema Estadístico Nacional (SEN).

El SEN se divide en 3 subsistemas:

Sistema de Información Estadística Nacional (SIEN): incluye los formularios estadísticos recopilados por la Oficina Nacional de Estadísticas a través de sus dependencias y los ministerios con fines nacionales.

Sistema de Información Estadística Territorial (SIET): incluye los formularios estadísticos recopilados por la oficina de estadística territorial y entidades territoriales con fines territoriales aprobados por la ONEI.

Sistema de Información Estadística Complementaria (SIEC): incluye los formularios estadísticos recopilados por todos los ministerios y entidades para sus propios fines aprobados por la ONEI.

Capítulo 1. Fundamentación teórica

La ONEI tiene una estructura institucional distribuida territorialmente en las provincias y municipios del país. Existen 15 Oficinas Territoriales de Estadísticas (OTE) y 169 Oficinas Municipales de Estadísticas (OME). Esas oficinas tienen atención administrativa y metodológica por la oficina nacional. La información estadística la brindan las empresas, instituciones y organizaciones según lo previsto en el programa de captación de datos convenido con todas las entidades (2).

Microset NT

Microset NT fue desarrollado en MS-DOS en los años 70 por informáticos de la Oficina nacional de Estadística (ONE), para el control de la información estadística nacional. Permitía parte de las funcionalidades demandadas, pero no el total y dinámico trabajo con la información suministrada, condiciona la salida a un formato de tablas previamente parametrizados en el sistema. Dado que se desarrolló como aplicación para consola (con elementos visuales para el modo texto), no es todo lo amigable que se puede esperar de una aplicación actual. MicroSet NT no evolucionó como aplicación para Windows y actualmente presenta problemas para ejecutarse sobre cualquier Windows que no pertenezca a la familia 9x. La misma contaba con la funcionalidad para generar reportes estándares.

1.3. Sistema Integrado de Gestión Estadística (SIGE).

El Sistema Integrado de Gestión Estadística permite definir qué información se obtendrá a través de las encuestas económicas y los formularios estadísticos. Está diseñado como un sistema distribuido con arquitectura modular fomentando la flexibilidad y escalabilidad. De esta forma se garantiza un sistema adaptable a diferentes ámbitos del negocio estadístico, que no sólo sería útil para la ONEI sino también para otros negocios con una base metodológica de trabajo estadístico similar.

El sistema responde a los procesos fundamentales identificados en el marco de gestión de la información estadística. Tales procesos comprenden la creación de formularios y encuestas (MDF), la captura de la información a partir del diseño de formularios y encuestas (MED), la validación de la información captada, su almacenamiento en la base de datos y traspaso con posterioridad al almacén de datos para su consulta y procesamiento (3).

El sistema gestiona la seguridad a través del proveedor de identidad del Cedrux (ERP cubano), el componente Acaxia. El modelo de seguridad del sistema, se caracteriza por gestionar los conceptos rol, permiso y recurso, garantizando que determinados roles tengan las atribuciones o permisos que le

Capítulo 1. Fundamentación teórica

corresponden solamente sobre recursos específicos. El componente provee además un sistema de trazas que registra la información de las acciones que realiza el usuario en todo momento. La implementación de políticas de seguridad se deja a mano de la institución, recomendándose el uso de cortafuegos adecuadamente configurados entre los servidores y el resto de la red (3).

Módulos que integran SIGE:

Módulo Gestión de Configuración (MGC): es el encargado de la gestión de las empresas y de los conceptos o metadatos necesarios para su caracterización y organización a nivel de país. En este sentido, los metadatos gestionados constituyen la base para el diseño de los formularios y las encuestas económicas, en esencia marcan las pautas para conceptualizar los formularios y las encuestas.

Módulo Diseñador de Formularios (MDF): es una de las herramientas que compone la solución informática de SIGE. En este marco dicha solución garantiza el proceso de definición estructural de la captación de la información estadística.

El objetivo principal de esta herramienta es automatizar el proceso de diseño de los formularios y encuestas necesarios para la captación de la información estadística en las unidades de observación, lo que constituye entrada para el procesamiento y obtención de informes y reportes de comportamiento estadísticos para la toma de decisiones estratégicas en cada uno de los ministerios. Para ello, el módulo cuenta con dos funcionalidades: Diseñar Formularios y Diseñar Encuestas, mediante las cuales se obtiene un equivalente digital del facsímil en papel, utilizado con anterioridad para captar la información, denominado plantilla de formulario o de encuesta, según sea el caso.

Funcionalidad diseñar encuestas: esta funcionalidad garantiza el diseño de encuestas estadísticas; es muy similar a la funcionalidad de diseñar formularios y al igual que esta, permite la edición de encuestas previamente creadas y da la posibilidad de exportar e importar estas utilizando XML.

El procedimiento de diseñar una encuesta es casi idéntico al de diseñar un formulario. Contempla igualmente la definición conceptual de una encuesta, salvo que en este caso se cuenta con una etapa de menos debido a que las encuestas no se validan. La definición de las páginas es ligeramente distinta, ya que lo que se gestiona son las preguntas que va a contener cada página, en lugar de los indicadores y aspectos presentes en los formularios.

Funcionalidad diseñar formularios: esta funcionalidad contempla el diseño de formularios de nomenclatura

Capítulo 1. Fundamentación teórica

cerrada y abierta, así como la edición de formularios previamente creados y además la posibilidad de exportar e importar estos utilizando XML. El resultado final de esta operación es la obtención de la plantilla de formulario, para lo cual es necesaria su definición conceptual.

La definición conceptual de un formulario está dividida en cuatro etapas o secciones por las que se puede transitar no necesariamente en un orden específico:

1. Definición de datos generales: en esta sección se introducen datos como el nombre, número y subnúmero del formulario. Se define además, el pie de firma del formulario, las fechas de captación y la periodicidad de este, así como las variantes que puede poseer.
2. Definición de páginas: en esta sección se gestionan las páginas del formulario especificando los indicadores y aspectos de cada una por separado.
3. Definición de cuadros de validación: esta sección permite la definición de los cuadros de validación que permiten la validación del formulario una vez captada la información en el MED. El componente principal utilizado para la confección de los cuadros, representa un logro para la experiencia del usuario, logrando que estos no dependan del conocimiento previo de la gramática del lenguaje de cuadros estadísticos para un tránsito exitoso por esta etapa.
4. Definición de nota metodológica: Esta etapa es la más sencilla y fácil de transitar, básicamente contempla la redacción de la nota metodológica o metodología de llenado del formulario.

El tránsito del usuario por cada una de estas etapas se garantiza de una manera muy sencilla debido a que la funcionalidad consta de un asistente que guía al usuario a través de las mismas, agregando una quinta que permite ver la vista previa de cómo va quedando el diseño del formulario.

Una vez terminado el diseño del formulario puede ser publicado para su uso durante el proceso de captación de la información en las unidades de observación. Adicionalmente se puede imprimir la vista previa para ser utilizada como facsímil de captura manual de datos.

La captura de datos es el proceso primario del trabajo estadístico. La complejidad del mundo moderno está sostenida por muchos datos y dicha complejidad lógicamente se transmite a los métodos de obtención de estos. Si se añade el hecho de que el proceso se realice de forma manual, se estaría hablando de ineficiencia, inexactitud y riesgos. De ahí la necesidad de encontrar una solución informática a las técnicas actuales de captación de datos.

Capítulo 1. Fundamentación teórica

El Módulo de Entrada de Datos ha surgido con ese fin, sosteniendo así este proceso. Este módulo va desplegado principalmente en la base de la organización donde se encuentre instalada la solución informática, es decir, empresas o entidades generalizadas como unidades de observación en este caso. Por otra parte, su trabajo se desarrolla sobre los productos finales del Módulo Diseñador de Formularios, es decir, plantillas de formularios o encuestas en las que se definen el subconjunto de datos a captar de acuerdo a la naturaleza económica o social de la unidad de observación en cuestión. Este módulo es el responsable de la validación de la información que ha sido recopilada en cada formulario.

Para garantizar la captura y gestión de la información, MED cuenta con 5 funcionalidades:

- **Actualizar Formularios:** esta funcionalidad permite la gestión de las plantillas de formularios y su habilitación o inhabilitación para la captación de la información. Los formularios captados, entiéndase información asociada a una unidad de observación en una plantilla de formulario, son gestionados de acuerdo a sus estados, que pueden ser: Digitación (proceso de captación de la información), Validación (proceso de evaluación del contenido del formulario de acuerdo a las reglas definidas en su diseño) y Archivo (una vez que el formulario ha sido captado y validado). En cualquiera de estos estados, los formularios pueden ser eliminados del sistema, así como también exportados e importados hacia o desde archivos XML o MSET (formato similar al CSV, pero separado por espacios y con especificación a partir de la estructura de los formularios diseñados en el Módulo Diseñador de Formularios).
- **Actualizar Encuestas:** esta funcionalidad permite la gestión de las plantillas de encuestas y su habilitación o inhabilitación para la captación de la información. Las encuestas captadas, entiéndase información asociada a una unidad de observación en una plantilla de encuesta, son gestionadas de acuerdo a sus estados, que pueden ser: Digitación (proceso de captación de la información), y Archivo (una vez que la encuesta terminada). En cualquiera de estos estados, las encuestas pueden ser eliminadas del sistema, así como también exportadas e importadas hacia o desde archivos XML.
- **Captar Formularios:** esta funcionalidad permite cargar una plantilla de formulario para la captura de datos a partir del mismo. Para ello, presenta una sección de Captación Rápida que dinámicamente se adecúa al formulario en cuestión guiando el proceso a través de los diferentes aspectos (columnas del formulario), con lo que se agiliza y humaniza el trabajo del usuario final. Contiene también otras secciones como la Vista Formulario que muestra en formato HTML una vista del

Capítulo 1. Fundamentación teórica

formulario que se está captando con los datos que contiene hasta ese momento y la Nota Metodológica, que representa igualmente en formato HTML, la metodología asociada a la naturaleza y forma de captación de la plantilla.

- Captar Encuestas: esta funcionalidad es similar a la de Captar Formularios, sólo que sobre otro tipo de plantilla. Dado que las preguntas incluidas en una encuesta pueden ser sostenidas por diferentes formas de captación: campo de texto, campo numérico, campo de fecha, selección simple, selección múltiple y caja de texto, las mismas se cargan en una vista de impresión donde es capturada la respuesta de cada pregunta de acuerdo al caso correspondiente. También en este caso, existe una sección Nota Metodológica análoga a la de Captar Formularios.
- Validar Formularios: como se había mencionado anteriormente, cada plantilla de formulario define reglas para validar la información. Esta funcionalidad verifica el cumplimiento de estas reglas. Una vez que ha sido realizada dicha verificación, el formulario puede transitar por 2 estados: Archivo, en caso de que la validación haya resultado satisfactoria; o Digitación, de ocurrir errores durante la verificación de cada regla definida, por lo que el formulario deberá ser cargado en Captar Formularios para corregir los errores detectados.

Módulo Generador Dinámico de Reportes (GDR): incluido en la solución, es una herramienta desarrollada para la realización de los reportes ejecutivos. En el contexto de SIGE es usada para la consulta de la información estadística. Cubre el ciclo de vida de un reporte desde su creación, hasta su consulta y distribución. Entre las principales funcionalidades están: gestión de los modelos de datos para el reporte (fuente de datos), diseño visual de consultas, diseño de reportes parametrizables, visualización de los reportes, gestión de los reportes y distribución.

Por su contante evolución y mejora GDR agrega valor a SIGE con nuevas prestaciones como la generación de distintos tipos de gráficos, la incorporación de funciones estadísticas para consultar la información y de tipos especializados de reportes. Una cuestión adicional de este producto, es su capacidad de integración con sistemas de terceros a través de una Interfaz de Programación de Aplicaciones [*Application Programming Interface*] (API) lo que facilita que la consulta de reportes previamente elaborados pueda realizarse embebido en otro sistema web de una manera fácil (3).

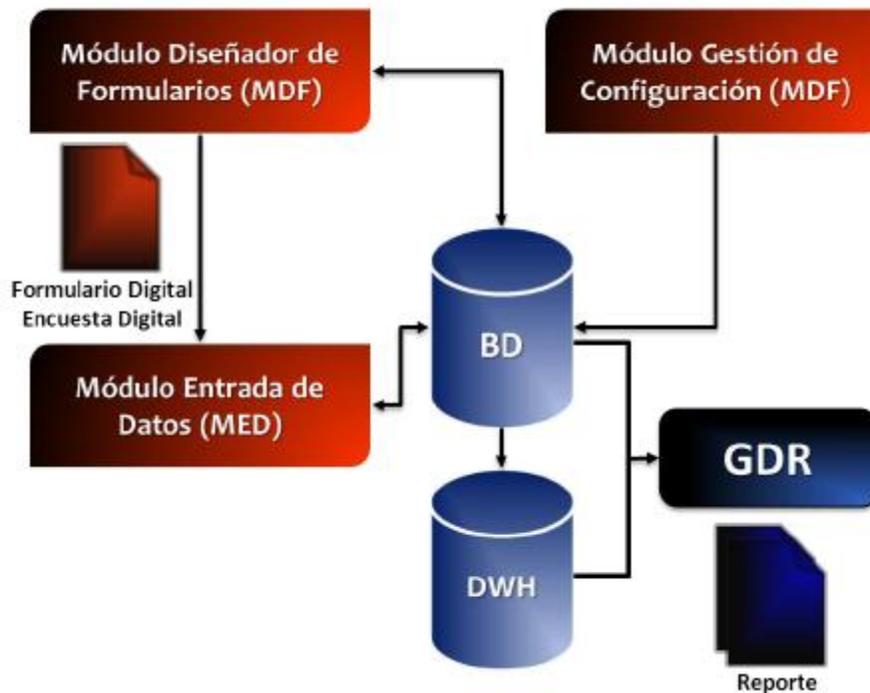


Figura 1. Módulos de SIGE

Los módulos que integran el sistema están altamente especializados y cohesionados, permitiendo a SIGE cumplir con su objetivo principal: captar la información estadística a nivel empresarial y mostrar los reportes solicitados por el cliente.

1.4. Reportes Estándares.

Los **reportes** son objetos que entregan información en un formato particular y que permiten realizar ciertas operaciones como, imprimirlos, enviarlos por email, guardarlos a un archivo, entre otras a partir de los datos almacenados en una base de datos.

Se define como **reporte estándar** al archivo, generado por un motor de reportes, con diferentes estructuras que contiene los datos de acuerdo a un interés específico. Sumatoria de valores y datos estadísticos solicitados por la ONEI a la fuente de datos, a partir de un formulario y un maestro. Además la información contenida en el reporte puede ser delimitada de forma opcional por los parámetros: rango de fecha, provincia, municipio y centro informante.

Capítulo 1. Fundamentación teórica

Para la creación de un reporte estándar se parte de un formulario y un maestro. El primero brinda la estructura que tendrá el reporte y los datos. Todos los formularios cuentan con atributos en común que juntos forman la llave primaria que lo identifica. Los restantes atributos del formulario son lo que se tomarán para diseñar el cuerpo del reporte. El maestro sirve para acotar la información seleccionada, este cuenta con la información de un grupo de centros informantes en cierto período de tiempo agrupados por una determinada característica en común. Además la información solicitada para mostrar en el reporte se le especificará opcionalmente el nivel (nacional, provincial, municipal y de base), rango de fecha y si la vista será de tipo total o desglose.

La vista total mostraría la información en una sola tabla. La de tipo desglose representa la información dividida en tablas de acuerdo al nivel. Por ejemplo una vista de tipo desglose del nivel nacional mostraría la información por provincias, donde cada tabla representa una provincia.

La cantidad de reportes estándares que se pueden generar partir de una fuente de datos varía en dependencia de la cantidad de formularios y maestros existentes. Además de las especificaciones seleccionadas por el usuario, opcionalmente este puede escoger entre un rango de fecha (fecha de inicio, fecha final), las provincias que componen Cuba y los municipios que componen la provincia seleccionada. De igual forma los centros informantes pertenecientes a ese municipio y si el reporte tendrá una vista total o por desglose.

Para hacer un reporte estándar utilizando GDR es necesario hacer 3 cosas: un diseño visual, una rutina o función en el gestor de base de datos y un reporte que es la unión entre el resultado de esa rutina y el diseño. Eso se tardaría tal vez todo un día o más además de que la rutina debe hacerla un especialista que domine el lenguaje SQL y tenga permiso para acceder al servidor de base de datos.

Con la utilización del Módulo Generador de Reportes Estándares, el usuario seleccionaría las opciones que precise y el sistema internamente elabora el reporte y lo visualiza de forma automática.

1.5. Lenguaje de programación.

Los lenguajes de programación son la vía de comunicación entre el hombre y la computadora. Un lenguaje de programación es una técnica estándar de comunicación. Permite expresar las instrucciones que han de ser ejecutadas en una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen un programa informático (4).

Capítulo 1. Fundamentación teórica

1.5.1. Lenguaje de programación del lado del servidor. PHP.

Un lenguaje del lado del servidor es aquel que se ejecuta en el servidor web, justo antes de que se envíe la página a través de Internet al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente (5).

PHP fue creado en 1994, su nombre es un acrónimo recursivo que significa PHP [*Hipertext Pre-processor*]. Se utiliza en el desarrollo de aplicaciones web dinámicas, ejecutadas en el servidor. Posee soporte para las bases de datos MySQL, PostgreSQL, Oracle, MS SQL Server, Sybase mSQL, Informix, entre otras. Se encuentra bajo licencia libre. Permite aplicar técnicas de Programación Orientada a Objetos. Es multiplataforma.

Toma prestado un poco de su sintaxis de otros lenguajes como C, Perl e incluso Java. Realmente es un lenguaje híbrido, tomando las mejores características de otros idiomas y creando un lenguaje fácil de usar y potente (6).

Ventajas:

- El código fuente escrito en PHP5 es invisible tanto para el navegador como para el cliente porque es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP5 sea segura y confiable.
- Permite aplicar técnicas de programación orientada a objetos.
- Biblioteca nativa de funciones bastante amplia e incluida.
- Es completamente expandible. Está compuesto de un sistema principal, un conjunto de módulos y una variedad de extensiones de código.
- Posee muchas interfaces distintas para cada tipo de servidor. Este lenguaje actualmente se puede ejecutar bajo Apache, IIS, AOLServer y Roxen.

Se selecciona PHP como lenguaje de programación del lado del servidor en su versión 5, teniendo en cuenta que se trabajara con Symfony como framework de desarrollo. Además de ser el lenguaje utilizado en SIGE.

Capítulo 1. Fundamentación teórica

1.5.2. Lenguaje de Programación del lado del cliente. Java Script.

Un lenguaje del lado cliente es totalmente independiente del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio. El código, tanto del hipertexto como de los scripts, es accesible a cualquiera y ello puede afectar a la seguridad (7).

Java Script es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas que permitan intercambiar con los usuarios. Se implementa como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas. Es interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con Java Script se pueden probar directamente en cualquier navegador que lo soporte, sin necesidad de procesos intermedios (8).

Ventajas:

- Los programas escritos en este lenguaje no requieren de mucha memoria ni tiempo adicional de transmisión, por ser pequeños y compactos.
- No requiere un tiempo de compilación, pues los scripts se pueden desarrollar en un período de tiempo relativamente corto.
- Es independiente del hardware o sistema operativo, este funciona correctamente siempre y cuando exista un navegador que lo soporte.
- Asegura la permanencia de una operación realizada y aunque falle el sistema esta no podrá deshacerse.
- Se selecciona Java Script como lenguaje de programación del lado del cliente, teniendo en cuenta que es el lenguaje utilizado en la versión anterior de SIGE.

1.6. Metodología de desarrollo.

Para facilitar el desarrollo de software en un grupo de trabajo y garantizar la calidad se utilizan las metodologías de desarrollo.

Una metodología de desarrollo de software es un marco de trabajo que muestra las tareas que se requieren para construir un software de alta calidad. Indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben cumplir. Guía a los desarrolladores en la realización de

Capítulo 1. Fundamentación teórica

las actividades, durante todo el proceso de creación de un producto (9).

Se clasifican en dos tipos: robustas que son usadas en grandes proyectos, proponen un ciclo de desarrollo con mayor énfasis en la planificación y control del proyecto. Las ágiles se utilizan para proyectos de corto plazo con menor tamaño, se caracterizan por tener la capacidad de proveer respuestas rápidas y ser adaptables al cambio. Proporcionan una serie de pautas y principios junto a técnicas pragmáticas, que puede que no curen todos los males, pero harán la entrega del proyecto menos complicada y más satisfactoria tanto para los clientes como para los equipos de entrega (10).

1.7. Proceso Unificado Abierto (OpenUP).

La metodología Proceso Unificado Abierto [*Open Unified Process*] (OpenUP), es un proceso de desarrollo unificado que está basado en el Proceso Unificado de Rational [*Rational Unified Process*] (RUP). Mantiene las mismas características de RUP pues está dirigido por casos de uso, centrado en la arquitectura y además es iterativo e incremental. Solo incluye su contenido necesario y fundamental, que pueden servir de base a procesos específicos. La mayoría de los elementos están declarados para fomentar el intercambio de información entre los equipos de desarrollo. Posibilita mantener un entendimiento compartido del proyecto, sus objetivos y alcance. Estructura el proyecto en cuatro fases: concepción, elaboración, construcción y transición. El ciclo de vida del proyecto brinda a los interesados un mecanismo de supervisión y dirección para controlar los fundamentos del mismo, su ámbito, la exposición a los riesgos, el aumento de valor entre otros aspectos (11).

OpenUp presenta las siguientes características:

- Proceso de desarrollo del software. Es completo en el sentido que puede ser manifestado como todo el proceso para construir un sistema.
- Es extensible pues en el proceso se puede agregar o adaptar según la necesidad de los sistemas.
- Es un proceso ágil.
- Es ligero y proporciona una comprensión detallada del proyecto, beneficiando a clientes y desarrolladores sobre productos a entregar y su formalidad.
- Se centra en una arquitectura temprana para reducir al mínimo los riesgos y organizar el desarrollo.

Capítulo 1. Fundamentación teórica

- Es la metodología utilizada por desarrolladores de alto nivel en casi todo el mundo por sus altas cualidades administrativas.

Como el tiempo de desarrollo es corto, el equipo de trabajo es pequeño y la metodología usada en el departamento Soluciones Integrales del centro DATEC, cuenta con resultados palpables, bajo la decisión del proyecto se decide utilizar esta la misma para guiar el proceso de desarrollo de software del sistema a desarrollar.

1.8. Lenguaje de Modelado. UML.

El Lenguaje de Modelación Unificado [*Unified Model Lenguaje*] (UML) es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema software. El modelado visual ayuda a mejorar la capacidad del equipo para gestionar la complejidad del software. Combina lo mejor de los métodos de análisis y diseño anteriores a él. Es un lenguaje de modelado de sistemas orientado a objetos de notación para el desarrollo de software. Se estandarizó su uso en la industria de desarrollo de software. Es interactivo y orientado a objetos (12).

1.9. Herramienta CASE. Visual Paradigm.

Visual Paradigm para UML (VP-UML) es una herramienta Ingeniería de Software Asistida por Computadora [*Computer Aided Software Engineering*] (CASE). Ofrece un paquete completo necesario para la captura de requisitos, la planificación del software, planificación de pruebas, modelado de clases y modelado de datos.

Esta herramienta permite aumentar la calidad del software, a través de la mejora de la productividad en el desarrollo y mantenimiento del software. También permite la reutilización del código, portabilidad y estandarización de la documentación. Además del uso de las distintas metodologías propias de la Ingeniería del Software (13).

Algunas características Visual Paradigm:

- Generación de bases de datos. Transformación de diagramas de Entidad-Relación en tablas de base de datos.
- Proporciona a los desarrolladores una plataforma que les permite diseñar un producto rápidamente y con la calidad requerida.

Capítulo 1. Fundamentación teórica

- Facilita la interoperabilidad con otras herramientas CASE y se integra con múltiples herramientas de desarrollo, como Eclipse/IBMWebSphere, Jbuilder, NetBeans IDE, Oracle Jdeveloper.
- Genera código y realiza ingeniería inversa para diez lenguajes de programación, entre ellos Java, C++, PHP y XML Schema.
- Genera documentación para el proyecto en HTML, MS Word y PDF.
- Licencia: gratuita y Comercial.
- Fácil de instalar y actualizar.

Su selección como herramienta CASE se justifica debido a que además de trabajar en el diseño y el modelado, posibilita la integración con otras herramientas y lenguajes utilizados en el proceso de desarrollo de SIGE. La versión que se utilizará es la 8.0.

1.10. Marco de trabajo de desarrollo.

Un Marco de trabajo [*Framework*] es una abstracción en el que se brinda al usuario funcionalidades genéricas, proporcionando así software reutilizable utilizado para desarrollar aplicaciones, productos y soluciones. Incluye programas de apoyo, compiladores, bibliotecas de código, una API y conjuntos de herramientas que reúnen a todos los diferentes componentes para permitir el desarrollo de un proyecto o solución. Aumenta la facilidad del trabajo. Promueve buenas prácticas de desarrollo como el uso de patrones.

1.10.1. Marco de trabajo de desarrollo del lado del cliente. ExtJS.

ExtJS es una librería Java Script que permite crear aplicaciones web interactivas del lado del cliente, para ello cuenta con una serie de componentes. Es ligera y de alto rendimiento, compatible con la mayoría de navegadores, evitando tener que validar el código para que funcione correctamente en cada navegador. Permite crear aplicaciones complejas utilizando componentes predefinidos (14).

Empezó siendo un conjunto de librerías y extensiones para Interfaz de usuario Yahoo! [*Yahoo! User Interface*], (YUI), librería desarrollada en JavaScript que explota las potencialidades del JavaScript Asíncrono y Xml [*Asynchronous JavaScript And XML*], (AJAX), para el desarrollo de Aplicaciones Ricas de Internet [*Rich Internet Applications*], (RIA). Con el tiempo se convirtió en un Marco de trabajo independiente y a principios de 2007 se creó una compañía para comercializar y dar soporte del Marco de

Capítulo 1. Fundamentación teórica

trabajo ExtJS (15).

Principales características de ExtJS:

- Alto rendimiento en ejecución debido a la optimización de código Java Script
- Controles de usuario personalizables.
- Modelo orientado a componentes, bien diseñado y extensible.
- Posee una API intuitiva y fácil de utilizar.
- Es distribuido bajo licencias libres y comerciales.

Luego del estudio de las características del marco de trabajo, y siguiendo la estructura propuesta por el equipo de desarrollo de SIGE, se decide utilizar como marco de trabajo de desarrollo del lado del cliente ExtJS en su versión 2.2.

1.10.2. Marco de trabajo de desarrollo del lado del servidor. Symfony.

Symfony es uno de los frameworks más populares para PHP en la actualidad. Se basa en el paradigma programación orientada a objetos. Su objetivo es optimizar el desarrollo de aplicaciones web. Es independiente del sistema gestor de bases de datos (16).

Es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y Microsoft SQL Server. Corre sobre las versiones 5 o superiores de PHP. Utiliza el patrón arquitectónico Modelo-Vista-Controlador, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. En cuanto a seguridad, internamente se ocupa de los ataques más comunes, tales como Inyección SQL, XSS o CSRF. Miembro de la familia del Software Libre e independiente de la plataforma.

Symfony fue diseñado para ajustarse a los siguientes requisitos:

- Fácil de instalar y configurar en la mayoría de plataformas.
- Independiente del sistema gestor de bases de datos. Permite cambiar con facilidad de Sistema Gestor de Base de Datos (SGBD) en cualquier fase del proyecto.
- Utiliza programación orientada a objetos, de ahí que sea imprescindible PHP 5 o superior.

Capítulo 1. Fundamentación teórica

- Sencillo de usar en la mayoría de casos, aunque es preferible para el desarrollo de grandes aplicaciones Web que para pequeños proyectos.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Fácil de extender, lo que permite su integración con las bibliotecas de otros fabricantes.
- Una potente línea de comandos que facilitan generación de código, lo cual contribuye a ahorrar tiempo de trabajo.

Partiendo de que la programación del lado del servidor en SIGE se realizó con Symfony 1.1, es el Marco de trabajo que se utilizará en el desarrollo de la aplicación.

1.11. Entorno de desarrollo integrado (IDE). NetBeans.

Un Entorno Integrado de Desarrollo es una aplicación que facilita el trabajo a los desarrolladores de software. Generalmente integra un editor de código, un compilador y un debugger.

La plataforma de desarrollo NetBeans provee de una estructura para los proyectos que se puede crear junto a este entorno de desarrollo integrado (IDE), a partir de un conjunto de componentes de software llamados módulos. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que estos pueden ser desarrollados de forma independiente. Las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software. Este IDE es gratuito, y de código abierto. El mismo tiene gran éxito con una gran base de usuarios y una comunidad en constante crecimiento.

Sun Microsystems fundó el proyecto NetBeans en junio del 2000 y continúa siendo el patrocinador principal de los proyectos. Se desarrolla en comunidad de forma exitosa. Permite programar en distintos lenguajes, es ideal para trabajar con el lenguaje de desarrollo Java (y todos sus derivados), además ofrece un excelente entorno para programar en PHP. Las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software (17).

Provee una estructura para los proyectos, propone un esqueleto para organizar código fuente, el editor conjuntamente integra los lenguajes como HTML, JavaScript, PHP y CSS (17).

Es la herramienta idónea para desarrollar el modulo, pues integra los lenguajes que a utilizar en la construcción del módulo. Se utiliza en su versión 7.0.

Capítulo 1. Fundamentación teórica

1.12. Gestor de base de datos. PostgreSQL.

PostgreSQL es un sistema gestor de base de datos objeto-relacional bajo licencia libre y gratuita. En sus últimas versiones posee muchas características que solo se podían ver en productos comerciales. Utiliza un modelo cliente/servidor y además usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Soporte nativo para los lenguajes más populares del medio: PHP, C, C++, Perl, Python. Incluye todas las características de una base de datos profesional (triggers, store procedures, funciones, secuencias, relaciones, reglas, tipos de datos definidos por usuarios, vistas, vistas materializadas) (18).

Principales ventajas:

- Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.
- Posee una gran escalabilidad, haciéndolo idóneo para su uso en sitios Web que atienden un gran número de solicitudes.
- Puede ser instalado un número ilimitado de veces sin temor de sobrepasar la licencia.
- Posee estabilidad y confiabilidad reconocida.
- Es extensible a través del código fuente disponible sin costos adicionales.
- Es multiplataforma, disponible en la mayoría de los sistemas operativos.
- Permite implementar reglas, vistas, disparadores, sub-consultas y funciones.
- Partiendo de la arquitectura que propone el proyecto SIGE para su nueva versión y el estudio de las ventajas antes mencionadas se decide utilizar como gestor de base de datos PostgreSQL en su versión 8.4.

1.13. Aplicación gráfica para gestionar el gestor de bases de datos. PGAdmin III.

Pgadmin es la más popular plataforma de código abierto para el desarrollo con PostgreSQL. Rico en funciones para la administración y desarrollo para PostgreSQL. Puede ser usado en las plataformas Linux, FreeBSD, Solaris, MacOSX y Windows. Está diseñado con el fin de satisfacer la necesidad de todos los usuarios, para escribir simples sentencias SQL y desarrollar complejas bases de datos. La interfaz gráfica

Capítulo 1. Fundamentación teórica

soporta todas las características de PostgreSQL y hace más fácil la administración. La aplicación además incluye un editor de sintaxis SQL, editor de código del lado del servidor y una consola. Las conexiones al servidor pueden ser realizadas usando TCP/IP o Unix Domain Sockets (para plataformas Unix). Además de tener la posibilidad de usar encriptado SSL para seguridad. No requiere drivers adicionales para conectarse con el servidor de base de datos. PgAdmin es desarrollado por expertos de la comunidad de PostgreSQL alrededor del mundo y se encuentra disponible en una docena de idiomas. Fue liberado bajo licencia libre PostgreSQL (19).

Debido al uso de PostgreSQL como gestor de base de datos, es seleccionado pgAdmin III como aplicación gráfica para el trabajo con la bases de datos.

Conclusiones parciales.

En el capítulo se analizaron los conceptos estadísticos y las herramientas que se emplearán en la solución, mostrando como resultado las características claves que servirán de base para el diseño de la propuesta de trabajo. Se decidió utilizar como metodología de desarrollo a OpenUp, como herramienta de modelado Visual Paradigm 8.0. Para el desarrollo de la aplicación se utilizan como lenguajes de programación del lado del servidor PHP 5 y JavaScript del lado del cliente. Como marcos de trabajo para el desarrollo de aplicaciones Web se seleccionaron a ExtJS 2.2 del lado del cliente y Symfony 1.1 del lado del servidor. PostgreSQL 8.4 para gestor de base de datos y como administrador de PostgreSQL se utiliza pgAdmin III. Para el trabajo con los lenguajes de programación seleccionados se utilizará el IDE NetBeans 7.0.

Capítulo 2. Análisis y diseño del sistema

CAPÍTULO 2: ANÁLISIS Y DISEÑO DEL SISTEMA.

En este capítulo se describen, a partir del modelo de dominio, la propuesta de solución para el problema de la investigación. Describiendo los principales conceptos del entorno que serán objeto de investigación para la realización de la fase de Análisis y Diseño del componente propuesto. Se obtienen los requisitos funcionales y no funcionales a tener en cuenta para la construcción del componente. Se identifican los actores, casos de uso y las relaciones existentes entre ellos. Además se diseñarán los diagramas de clases y de interacción para los casos de uso arquitectónicamente significativos. Se definirá con precisión la estructura física que presentará el módulo de reportes estándares mediante el modelo de despliegue.

2.1. Modelo Dominio.

El modelo dominio es una representación visual de los conceptos u objetos del mundo real significativos para un problema o área de interés. Se representa en UML como un diagrama de clases en el que se muestran conceptos u objetos del dominio del problema y asociaciones entre las clases conceptuales. En la fase de inicio se determinó que los procesos del negocio no están claramente definidos, por esta razón se decide representar la situación real del sistema mediante un Modelo de Dominio. Facilitando a través de un vocabulario común que ayude a usuarios, clientes, desarrolladores e interesados a comprender el argumento principal del sistema (20). La figura 2 muestra el modelo de dominio.

Capítulo 2. Análisis y diseño del sistema

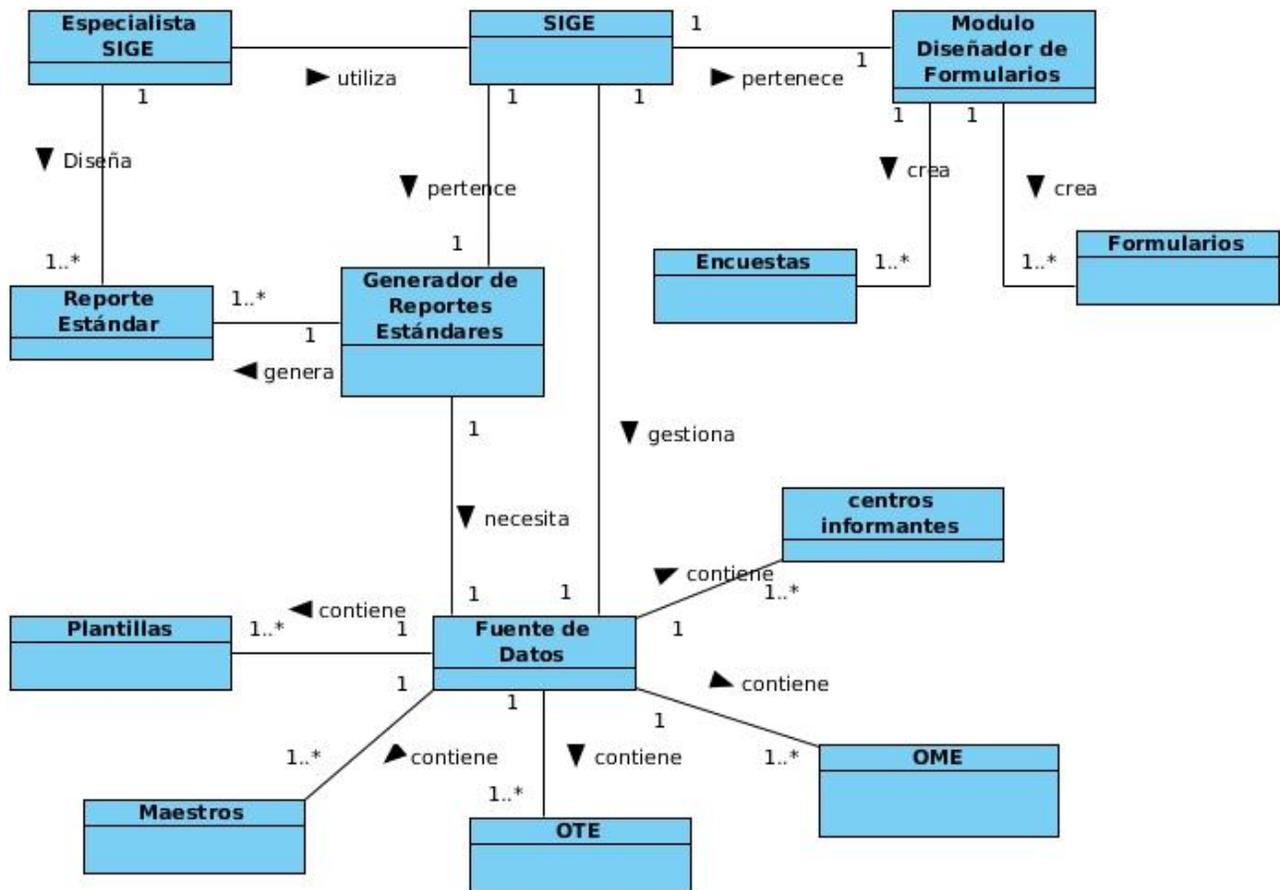


Figura 2. Modelo de Dominio.

Conceptos del Modelo de Dominio:

Especialista SIGE: persona capacitado para utilizar el sistema informático SIGE.

Reporte Estándar: archivos generados por SIGE a partir de información obtenida de la base de datos. Sumatoria de valores y datos estadísticos solicitados por la ONEI a la fuente de datos, a partir de un maestro y plantilla. Además la información contenida en el reporte puede ser acotada por los parámetros: provincia, municipio, rango de fecha y centro informante.

Generador de Reportes Estándares: módulo integrado SIGE que garantiza la generación de reportes estándares.

Fuente de datos: se compone por un conjunto de campos especificados por el usuario, asociados a un

Capítulo 2. Análisis y diseño del sistema

Modelo de datos. Se puede seleccionar como fuente de datos una tabla, vista o una función del modelo deseado, así como los indicadores obtenidos de la fuente de datos.

Plantillas: estructura definida, la cual es seleccionada por el usuario como punto de partida para generar los reportes estándares. Se genera a partir de un formulario publicado.

Maestros: los maestros guardan información referente a un conjunto de centros informantes en un periodo de tiempo dado.

SIGE: herramienta encargada de automatizar el proceso de control estadístico utilizada por la ONEI.

Módulo Diseñador de Formularios: es un componente de SIGE. Automatiza el proceso de diseño de los formularios y encuestas necesarios para la captación de la información estadística. La fuente de datos es llenada con la información capturada por este componente.

Formulario: plantilla utilizada para la captura de información a partir de los formularios diseñados por el Módulo Diseñador de Formularios. Con la información obtenida se nutre la fuente de datos.

Encuesta: herramienta utilizada para la captura de información a partir de las encuestas diseñadas. Por el Módulo Diseñador de Formularios.

OTE: Oficina Territorial de Estadísticas estructura que organiza el ámbito de la Oficina Nacional de Estadísticas en cuanto a la captación de la información. La ONEI tiene una estructura institucional distribuida territorialmente en las provincias. Existen 15 Oficinas Territoriales de Estadísticas (OTE), cada una corresponde a una provincia del país.

OME: Oficina Municipal de Estadística. Estructura que organiza el ámbito de la Oficina Territorial de Estadística en cuanto a la captación de la información. Las OTE están integradas por Oficinas Municipales de Estadísticas (OME). En cada municipio del país existe una OME.

Centro Informante: empresa, organismo, organización, o institución identificada como poseedor de información de interés para la toma de decisiones de los órganos de la administración del Estado.

En la ONEI se utiliza SIGE para la automatización de las estadísticas nacionales. Este sistema gestiona dichas estadísticas a partir de una fuente de datos que contiene la información recolectada por el Modulo Diseñador de Formularios, el cual puede crear formularios y encuestas estadísticas para la captura de la información proporcionada por los centros informantes.

Capítulo 2. Análisis y diseño del sistema

Los usuarios requieren los Reportes Estándares, para su elaboración a SIGE se le adicionará el Módulo Generador de Reportes Estándares. El cual obtiene la información requerida de la fuente de datos, diseña el reporte y lo visualiza.

2.2. Requisitos del Sistema.

Los requisitos del sistema no son más que esa condición que debe ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar u otro documento impuesto formalmente. Se denomina como: condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo, define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen (21).

Partiendo de la descripción de las clases más importantes dentro del contexto del sistema representado en el Modelo de Dominio, se determinaron los requisitos funcionales y no funcionales del componente.

2.2.1. Requisitos funcionales.

Se identificaron los siguientes requisitos funcionales:

Tabla 1. Descripción de los requisitos funcionales.

RF1. Seleccionar Plantilla	
Descripción	Seleccionar la plantilla requerida para crear el reporte a partir de una lista de todas las plantillas existentes en la base de datos.
Entrada	Las Plantillas almacenadas en la base de datos.
Salida	La seleccionada por el usuario
RF2. Seleccionar Maestro	
Descripción	Seleccionar el maestro requerido para crear el reporte a partir de una lista de todas los maestros existentes en la base de datos
Entrada	Los maestros almacenados en la base de datos
Salida	El maestro seleccionado por el usuario

Capítulo 2. Análisis y diseño del sistema

RF3. Seleccionar Especificaciones.	
Descripción	Seleccionar las especificaciones que acotarán el reporte, estas especificaciones funcionan como filtros. La selección de estas especificaciones es opcional en caso de que las requiera el reporte
Entrada	Provincia, municipio, rango de fecha, centro informante: Las especificaciones contenidas en la base de datos.
Salida	Provincia, municipio, fecha, empresa: Las seleccionadas por el usuario o ninguna.
RF4. Crear función para extraer información de la Base de Datos.	
Descripción	Con todas las indicaciones pedidas por el usuario se crea una función la cual extrae la información requerida de la base de datos. Filtrando por las especificaciones del usuario. La información se agrupa de acuerdo a indicadores comunes y se suma para obtener el resumen estadístico.
Entrada	Función
Salida	Información requerida para crear el reporte estándar
RF5. Elaborar Reporte Estándar.	
Descripción	Funcionalidad interna del sistema que toma la información obtenida de la base de datos y diseña el Reporte Estándar solicitado.
Entrada	Información solicitada para crear el reporte a la base de datos.
Salida	Información solicitada por el usuario
RF6. Visualizar el Reporte Estándar.	
Descripción	Obtenido y organizada toda la información que contendrá el Reporte Estándar es mostrado al usuario.
Entrada	Información solicitada por el usuario.
Salida	Reporte Estándar en formato HTLM.

Capítulo 2. Análisis y diseño del sistema

RF7. Imprimir Reporte Estándar.	
Descripción	Una vez obtenido el reporte estándar el usuario tiene opción de imprimirlo utilizando una impresora.
Entrada	Reporte Estándar.
Salida	Reporte Estándar impreso.

2.3. Requisitos no funcionales.

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Estas propiedades se describen como las características que hacen al producto atractivo, usable, rápido, confiable o eficiente con el hardware y software, con vista a las funcionalidades que el sistema debe brindar (21).

Usabilidad.

RNF1: El usuario podrá crear un reporte estándar automáticamente de forma ágil y rápida. Simplemente seleccionando los datos requeridos. Una interfaz amigable y fácil, donde se le visualizaran textos informativos indicándole al usuario los pasos que debe seguir.

Interfaces de usuario.

RNF2: El usuario utilizará el sistema como un instalador de un programa común.

Interfaces de Software.

RNF3: El servidor donde se instalará la aplicación debe cumplir con los siguientes requisitos de software:

- Sistema Operativo: Debian 6.0 o Ubuntu 10.10.
- Aplicación Servidora: Apache 2 con PHP5.
- Sistema Integrado de Gestión Estadística (SIGE) versión 1.7.
- Otras dependencias requeridas:
- php5-pgsql (paquete de extensión PHP para PostgreSQL).
- php5-xsl (paquete de extensión PHP para XSLT).

Capítulo 2. Análisis y diseño del sistema

- php-gd (paquete de extensión PHP para gráficos).

RNF4: El servidor donde se instalará la Base de Datos del sistema debe cumplir con los siguientes requisitos de software (puede ser el mismo donde estará la aplicación):

- Sistema Operativo: Debian 6.0 o Ubuntu 10.10.
- Gestor de Base de Datos: PostgreSQL 8.4.
- Cliente de Base de Datos: PgAdmin III.

RNF5: Estación de Trabajo:

- Sistema Operativo: Ubuntu 10.04 o superior
- Navegador: Mozilla Firefox 3.4 o superior.

Interfaces de hardware:

RNF6: Las PC clientes deben cumplir con los siguientes requisitos de hardware:

- Procesador Intel Pentium (I, II, III, IV) 1.7 GHz o superior, o AMD similar, 256 MB RAM.

Los servidores deben cumplir con los siguientes requisitos del hardware:

RNF7: Servidor de Base de Datos:

- Procesador Intel Pentium (IV) 1.7 GHz o AMD equivalente, 1 GB de RAM, 80 GB de espacio en disco duro.

RNF8: Servidor web:

- Procesador Intel Pentium (IV) 1.7 GHz o AMD equivalente, 1 GB de RAM, 40 GB de espacio en disco duro.

Portabilidad:

RNF9: El sistema será multiplataforma, razón por la cual podrá ser utilizado en cualquier sistema operativo y arquitectura de hardware.

2.4. Diagrama de Casos de Uso del Sistema.

El diagrama de casos de uso documenta el comportamiento de un sistema desde el punto de vista del

Capítulo 2. Análisis y diseño del sistema

usuario. Representa las relaciones existentes entre actores y casos de uso. Partiendo de los requisitos funcionales identificados se modeló el Diagrama de Casos de Uso, en el cual se evidencia el patrón de caso de uso CRUD Parcial. Este patrón alternativo modela una de las vías de los casos de uso como un caso de uso separado. Es preferiblemente utilizado cuando una de las alternativas de los casos de uso es más significativa, larga o más compleja que las otras (9). Se representan las relaciones entre los actores y casos de uso del sistema evidenciándose los principales procesos del sistema.

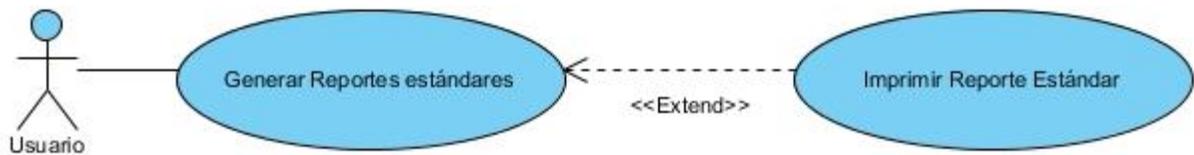


Figura 3. Diagrama de Casos de Uso. CU_Generar Reportes Estándares.

Tabla 2. Descripción de usuario del sistema.

Actor	Descripción
Usuario	El usuario es cualquier persona autorizada a utilizar el sistema para crear reportes estándares. Varía en dependencia del lugar donde se despliegue SIGE.

Tabla 3. Descripción de Casos de Uso. CU_Generar Reportes Estándares.

Caso de Uso:	Generar Reportes Estándares
Actores:	Usuario
Resumen:	El usuario desea generar un reporte estándar, el sistema muestra una interfaz visual con las opciones necesarias para crear un reporte o si el usuario cancela la operación antes de terminar.
Precondiciones:	Que el usuario este autenticado en el sistema y tenga los permisos necesarios

Capítulo 2. Análisis y diseño del sistema

	para generar un reporte estándar.
Referencias	RF1, RF2, RF3, RF4, RF5, RF6
Prioridad	Crítica
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El usuario solicita la creación de un reporte estándar.	2. El sistema solicita a la base de datos la lista de todas las plantillas publicadas que se encuentran en la misma. 3. Las plantillas son listadas para que el usuario seleccione una.
4. El usuario escoge la plantilla que desea y selecciona la opción Siguiente	5. El sistema almacena el número y el subnúmero de la plantilla seleccionada. 6. El sistema solicita a la bases de datos los la lista de maestros que contiene. 7. Se muestra un listado con los maestros enviados desde la base de datos.
8. El usuario escoge el maestro que desea y selecciona la opción Siguiente.	9. El sistema almacena el alias del maestro seleccionado. 10. El sistema muestra varias especificaciones las cuales el usuario puede escoger opcionalmente. Las mismas servirán de filtro para el reporte, estas son: <ul style="list-style-type: none"> ➤ Rango de fecha. ➤ Provincia.

Capítulo 2. Análisis y diseño del sistema

	<ul style="list-style-type: none">➤ Municipio.➤ Centro Informante. <p>11. El sistema muestra la opción para la vista del reporte, total o desglose, si el usuario no selecciona ninguna el sistema toma la vista total por defecto.</p>
<p>12. El usuario escoge los campos que necesita para su reporte estándar y selecciona la opción Crear Reporte.</p>	<p>13. El sistema guarda los valores escogidos por el usuario.</p> <p>14. El sistema toma todos los valores de almacenados a partir de la selección del usuario y los organiza en variables para crear una función SQL general para extraer la información requerida de la base de datos solicitada por el usuario.</p> <p>15. El sistema construye una función SQL para obtener información solicitada a partir de las selecciones del usuario de la base de datos. Esta función además suma los datos de acuerdo a plantilla seleccionada.</p> <p>16. Con la información obtenida de la base de datos el sistema confecciona el reporte estándar. Para ello tiene en cuenta el modo de vista seleccionado por el usuario. Si es total muestra una sola tabla con toda la información solicitada, en caso de ser desglose el sistema muestra varias tablas, cada una</p>

Capítulo 2. Análisis y diseño del sistema

	representando los subconjuntos en que se divide el total. 17. El sistema visualiza el reporte estándar requerido. Termina el caso de uso.
Flujos Alternos: el usuario cancela la operación	
Acción del Actor	Respuesta del Sistema
	2a. El sistema muestra mensaje: Desea cancelar esta operación: Si: cierra el proceso. No: continúa el proceso.
Poscondiciones	Se visualiza reporte estándar con los datos solicitados por el usuario

2.5. Objetivos del diseño.

El diseño tiene como principales objetivos comprender detalladamente los requisitos funcionales y no funcionales, sistemas operativos, tecnologías de distribución, restricciones relacionadas con el lenguaje o los de programación, tecnologías de interfaz de usuario. Además, crea un punto de partida para las actividades de implementación que siguen. Es el punto de mayor importancia al final de la fase de elaboración y el comienzo de las iteraciones de construcción. Permite facilitar una arquitectura estable y sólida, crear un plano muy cercano del modelo de implementación. Es necesario mantener el modelo de diseño a través de todo el ciclo de vida del software.

Propósitos del diseño:

- Adquirir una comprensión de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, tecnologías de distribución y concurrencia, sistemas operativos, componentes reutilizables y tecnologías de interfaz de usuario.
- Crear una entrada apropiada y un punto de partida para actividades de implementación, capturando los requisitos o subsistemas individuales, interfaces y clases.

Capítulo 2. Análisis y diseño del sistema

- Descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo.
- Capturar las interfaces entre los subsistemas antes en el ciclo de vida del software, lo cual es muy útil cuando utilizamos interfaces como elementos de sincronización entre diferentes equipos de desarrollo (12).

2.6. Diagrama de Clases del Diseño.

Un diagrama de clases del diseño es una representación concreta de los métodos y atributos de cada una de las clases del sistema que se debe implementar. Para mostrar de forma simple la colaboración y las tareas de cada una de ellas en relación al sistema que conforman. Estos diagramas representan la parte estática del sistema (9). La figura 4 muestra el diagrama de clases del diseño del caso de uso más significativo, generar reporte estándar.

Capítulo 2. Análisis y diseño del sistema

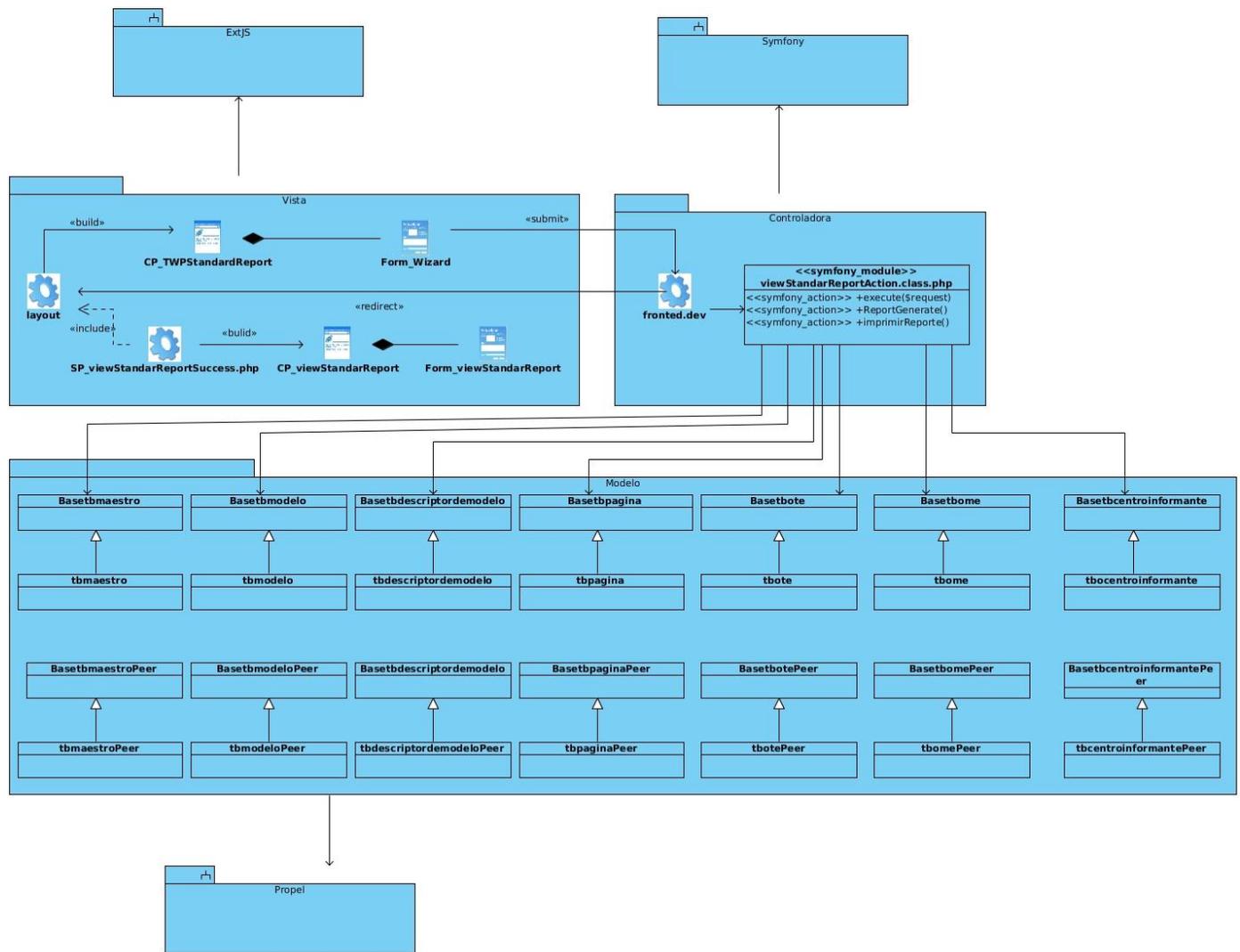


Figura 4. Diagrama de Clases del Diseño. CU_Generar Reportes Estándares.

Con el fin de facilitar la implementación de la aplicación se utilizó el framework Symfony que tiene entre sus patrones más utilizados: el patrón Modelo-Vista-Controlador (MVC), logrando la organización del código, la reutilización y la flexibilidad. El MVC se representa a través de tres elementos fundamentales: el modelo, la vista y el controlador (22).

El diagrama de clases del diseño del caso de uso más significativo (Figura 3) contiene tres paquetes, la Vista, el Modelo y el Controlador. En el primero de ellos se encuentran las clases TWPStandardReport.js

Capítulo 2. Análisis y diseño del sistema

construye la interfaz de usuario utilizando la librería ExtJS y `viewStandardReportSuccess.php` que es la encargada de construir la vista final del reporte. En el Modelo, se representan las clases asociadas al modelo de datos, en el diagrama se muestran las 4 clases que genera Propel por cada tabla de la base de datos. El Controlador contiene la clase `viewStandardReport.php` que es la encargada de la interacción con el Modelo y realizar las funcionalidades principales del caso de uso.

2.7. Patrones arquitectónicos.

Los patrones constituyen una guía para el diseño del software. Su objetivo es la solución de problemas que ocurren repetidamente dentro de un contexto muy bien definido. Además deben ser reusables, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias. Son de gran utilidad para describir las mejores prácticas, buenos diseños, y encapsulan la experiencia permitiendo su reutilización. Definen la estructura de un sistema de software, los cuales a su vez se componen de subsistemas con sus responsabilidades, también tienen una serie de directivas para organizar los componentes del mismo sistema, con el objetivo de facilitar la tarea del diseño de tal sistema (23).

Uno de los patrones más generalizados en el diseño web es el Modelo-Vista-Controlador (MVC). Este ya fue abordado en el epígrafe anterior. Fracciona el sistema en tres capas:

Modelo: representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.

Vista: transforma el modelo en una interfaz visual para interacción del usuario con el sistema.

Controlador: se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista (22).

Patrones generales de software para asignar responsabilidades (GRASP).

Experto: en la arquitectura de Symfony, específicamente en el modelo, existen dos tipos de clases fundamentales:

- Las clases de abstracción de datos: son las responsables de realizar todas las operaciones con la BD.
- Las de acceso a datos: encargadas de interactuar con las clases de abstracción de datos, devuelven los objetos que necesitan los controladores en su forma original.

El patrón experto en información es el principio básico de asignación de responsabilidades. Indica que la

Capítulo 2. Análisis y diseño del sistema

responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo se obtiene un diseño con mayor cohesión y así la información se mantiene encapsulada.

Al incluir Propel como ORM, se generan las clases para la gestión de las entidades con las responsabilidades debidamente asignadas. Esta es precisamente la solución que propone el patrón Experto, pues cada una de estas clases cuenta con un conjunto de funcionalidades relacionadas directamente con la entidad que representan (22).

Alta Cohesión: una de las características principales de Symfony es la organización del trabajo en cuanto a la estructura del proyecto, lo cual permite crear y trabajar con clases con una alta cohesión. En la solución se encuentran varias clases de tipo Actions que contienen varias funcionalidades que poseen un propósito único. Por ejemplo la clase `viewStandarReportAction.class.php` es la encargada de la principal funcionalidad del caso de uso, tiene la responsabilidad de definir las acciones para `viewStandarReportSuccess.php` (Template), que se encarga de la vista final del reporte. Las funcionalidades se encuentran estrechamente relacionadas proporcionando que el software sea flexible frente a grandes cambios (22).

Bajo Acoplamiento: en la capa del Modelo se encuentran las clases que implementan la lógica del negocio y de acceso a datos, estas clases tienen pocas asociaciones con otras de la Vista o el Controlador por lo que la dependencia en este caso es baja (22).

Controlador: todas las peticiones web son manejadas por un solo controlador frontal (`fronted.dev`), que es el punto de entrada único de toda la aplicación en un entorno determinado. Cuando el controlador frontal recibe una petición, utiliza el sistema de enrutamiento para asociar el nombre de una acción y el nombre de un módulo con la URL entrada por el usuario (22).

Patrones GoF.

Controlador Frontal [*Front Controller*]: El patrón Front Controller maneja todos los request en una aplicación. (23) Se evidencia en el `viewStandardReportActions.php`, pues es el punto de entrada a la aplicación que recibe todas las peticiones y decide el flujo a seguir, carga la configuración y determina la acción a ejecutarse.

Capítulo 2. Análisis y diseño del sistema

Patrón Decorador [Decorator]: Añade funcionalidad a una clase dinámicamente. Esto permite no tener que crear sucesivas clases que hereden de la primera incorporando la nueva funcionalidad, sino otras que la implementan y se asocian a la primera (23). Se evidencia cuando la clase layout, definida en el diagrama de clases del diseño, contiene todo el código HTML y a su vez decora todas las plantillas a usar, en dependencia de la petición del usuario.

2.8. Diagramas de Secuencia.

Los diagramas de secuencia ofrecen las interacciones entre objetos, ordenadas en secuencia temporal durante un escenario concreto. Describen el curso particular de los eventos de un caso de uso, proporcionando una realización física de comportamiento de los casos de uso en términos de clases del diseño (9). La figura 5 muestra el diagrama de secuencia del caso de uso más significativo Generar Reporte Estándar Reportes.

Capítulo 2. Análisis y diseño del sistema

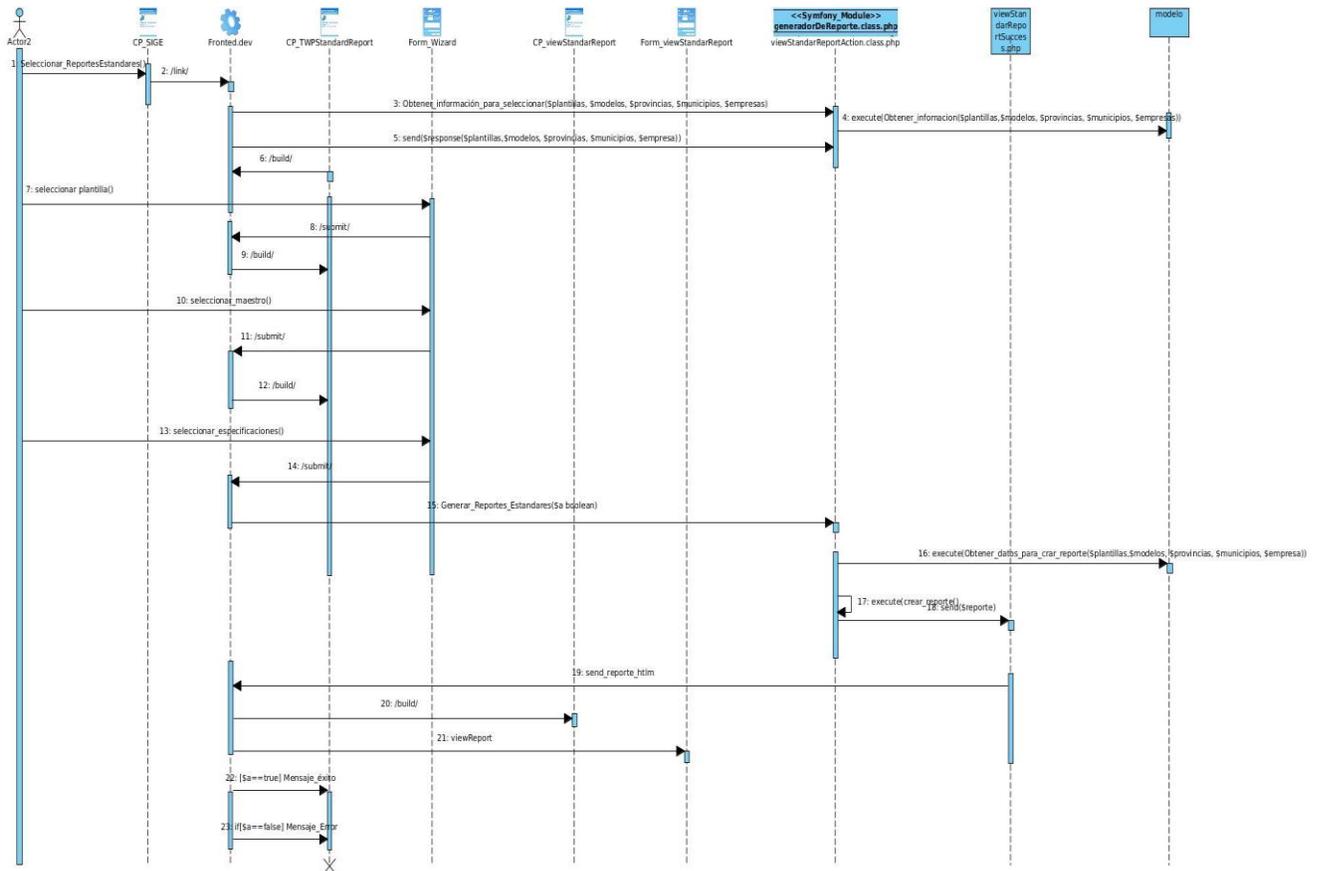


Figura 5. Diagrama de Secuencia. CU_Generar Reportes Estándares.

En el presente diagrama se muestran el flujo de acciones que ocurren en la aplicación al usuario ejecutar la acción “Reportes Estándares”, la página cliente de SIGE muestra el formulario Asistente Para Generar Reportes Estándares (wizard¹). El mismo consta de 3 partes la primera visualiza una lista de plantillas, estas son cargadas de la base de datos de forma automática y listadas. Seleccionada la plantilla corresponde la selección del maestro siguiendo los pasos anteriores. Se muestra el listado de maestros ofreciendo la posibilidad de seleccionar el que se utilizará. El siguiente paso consiste en ingresar las especificaciones del reporte. Las mismas acotan la información mostrada por el sistema, de acuerdo a lo solicitado por el usuario. Las especificaciones pueden ser rango de fecha, provincia, municipio o centro

1

Termino que se utiliza en la informática para definir a la interfaz visual que de forma sencilla y fácil guía al usuario en la realizar ejecución determinado proceso. Palabra nativa ingles que se traduce al español como hechicero.

Capítulo 2. Análisis y diseño del sistema

informante. Una vez obtenido los datos requeridos por el usuario la funcionalidad ReportGenerate () de la clase viewStandarReportAction.class.php creará el reporte el mismo será y luego envía la información a la clase viewStandarReportActionSucces. Esta última clase es la encargada de estructurar el reporte para quedar en forma de tabla con encabezado y pie de página. Por último la vista final del reporte es visualizado por el formulario Form_viewStandarReport.

2.9. Diagrama de clases persistentes.

La persistencia es la capacidad de un objeto de mantener su valor en el espacio y en el tiempo. Las clases persistentes constituyen información de larga duración, o sea, son los datos que necesitan ser almacenados para ser gestionados en cualquier momento. Una vez identificadas las clases cuyos estados son los más trascendentes en el tiempo de vida de la aplicación a desarrollar, se crea el diagrama de clases y se seleccionan las que constituyen una necesidad conservar su estado; quedando modelado el diagrama de clases persistentes. Las principales características que posee este diagrama es describir la estructura del sistema mostrando sus clases, atributos y las relaciones entre ellas. Generalmente, estas clases tienen como origen las clases clasificadas como entidad, porque modelan la información del sistema y el comportamiento asociado de algún fenómeno o concepto (24).

A continuación se muestra el diagrama para el sistema a desarrollar (Figura 6):

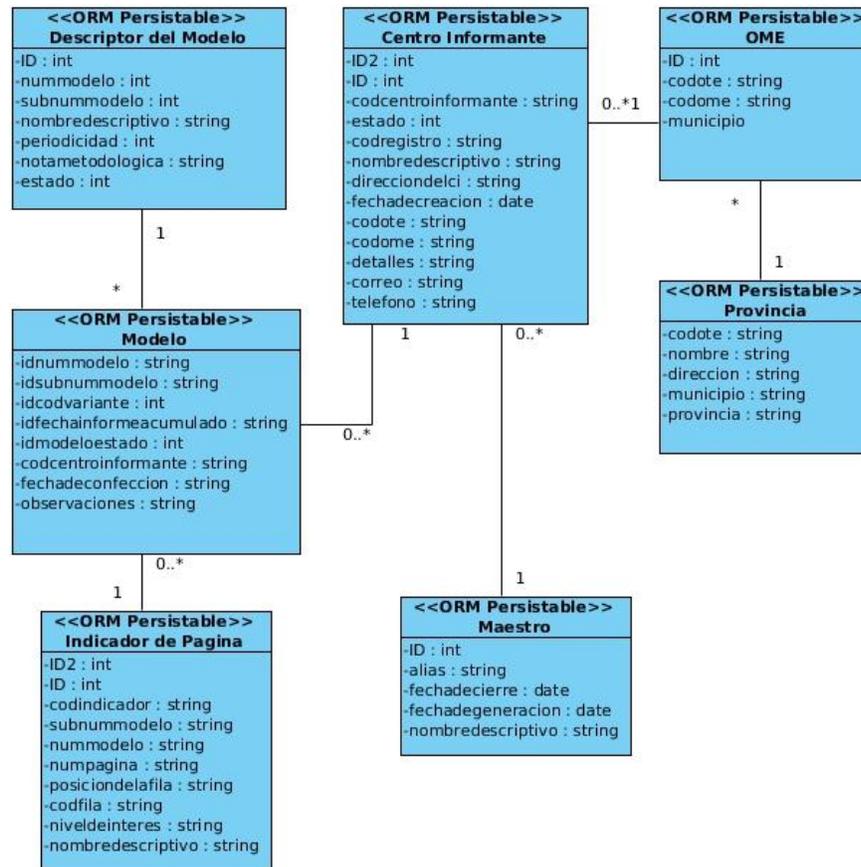


Figura 6. Diagrama de Clases Persistentes.

2.10. Modelo de datos.

Un modelo de datos es la estructura o representación física de las tablas de la base de datos obtenido a partir del diagrama de clases persistentes. Aporta la base conceptual para diseñar aplicaciones que hacen un uso intensivo de datos. De igual forma, permiten describir las estructuras de datos de la base, las restricciones de integridad y las operaciones de manipulación de los datos. Además brindan la base formal para las herramientas y técnicas empleadas en el desarrollo y uso de sistemas de información (24).

El diagrama muestra las entidades de la base de datos de SIGE que son utilizadas por el Módulo Generador de Reportes Estándares (Figura 7):



Figura 7. Diagrama de Entidad Relación.

2.11. Vista de Despliegue.

El modelo de despliegue muestra la configuración de nodos de procesamiento en tiempo de ejecución, los enlaces de comunicación entre ellos, y el componente de los casos y los objetos que residen en ellos. Además proporciona un modelo minucioso de la estructura en la que los componentes se desplegarán a lo largo de la infraestructura del sistema. El mismo los componentes de software ejecutables con los nodos de procesamiento. Además, tiene en cuenta los siguientes requerimientos: disponibilidad del sistema, rendimiento y escalabilidad (9). El módulo Generador de Reportes Estándares al pertenecer a SIGE presenta el siguiente modelo de despliegue en la figura 8.

Capítulo 2. Análisis y diseño del sistema

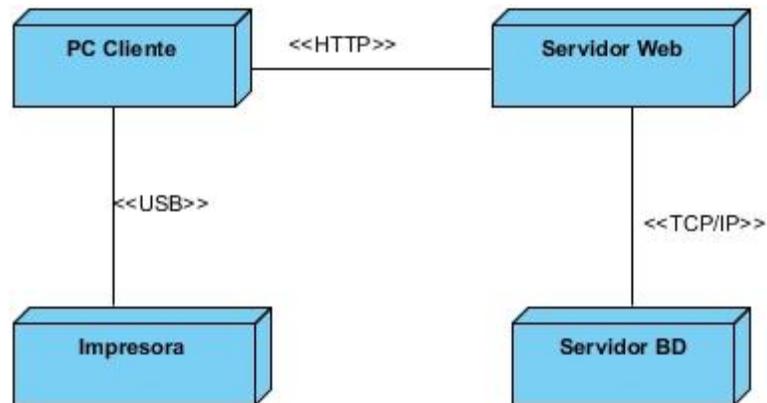


Figura 8. Diagrama de Despliegue.

PC Clientes (Estaciones de trabajo con la aplicación cliente): Se refiere a las estaciones de trabajo que el usuario utilizará para acceder a la aplicación Web y transcribir sus datos.

Servidor de Aplicación: Servidor de aplicación utilizado para la publicación de la aplicación; y para lograr la conexión del sistema con la PC Cliente se utiliza protocolo de transferencia de hipertexto [*Hypertext Transfer Protocol*] (HTTP) como protocolo de comunicación. Es la herramienta principal para ejecutar la lógica de negocio en el lado del servidor. Es el responsable de ejecutar el código de las páginas servidor. Se utiliza el servidor de aplicación Apache.

Servidor de Base de Datos: Se refiere a un servidor en el cual el almacenados todos los datos recopilados.

Impresora: Periférico que se utilizará para imprimir los reportes en papel.

Conclusiones parciales.

En el presente capítulo fueron descritos el análisis y el diseño de la aplicación comenzando por el modelo de dominio, los requisitos funcionales y no funcionales, quedando confeccionado el diagrama de casos de uso del sistema. Partiendo de la realización de los casos de uso, se crea el diagrama de clases del diseño dando paso a los diagramas de secuencia, reflejando un proceso lógico de acciones para resolver las peticiones del cliente. A través del diagrama de despliegue se describió como se realizará el despliegue de los componentes a lo largo de la infraestructura del sistema, para así dar comienzo a la implementación de la aplicación.

Capítulo 3. Implementación y prueba

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA

En el presente capítulo de las disciplinas implementación y pruebas se realizará una descripción de cómo los elementos del modelo de diseño se implementan en términos de componentes. Se llevarán a cabo las pruebas para validar el módulo generador de reportes estándares, arrojando los principales resultados obtenidos a través de los casos de pruebas realizados a cada caso de uso.

3.1. Modelo de Implementación.

El Modelo de Implementación describe cómo los elementos del modelo de diseño, se implementan en términos de componentes, ficheros de código fuente, ejecutables, entre otras. Describe también cómo se organizan los componentes de acuerdo a los mecanismos de estructuración disponibles en el entorno de implementación y lenguaje o lenguajes de implementación empleados, y cómo dependen los componentes unos de otros (9). Es de gran utilidad a la hora de implementar el sistema, pues facilita la organización del producto y lo hace más comprensible a los desarrolladores.

3.2. Diagrama de Componentes.

Dentro del Modelo de Implementación se encuentran los diagramas de componentes. Un diagrama de componentes representa cómo un sistema de software es dividido en componentes, mostrando las dependencias que existen entre ellos. Los componentes físicos incluyen: archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables y paquetes. Los diagramas de componentes prevalecen en el campo de la arquitectura de software, pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema; estos son utilizados para modelar la vista estática y dinámica de un sistema. Muestran la organización y las dependencias entre un conjunto de componentes (9). La figura 9 representa el diagrama de componentes de la aplicación a desarrollar.

La aplicación generador de reporte estándar está compuesta por 3 paquetes de componentes: el Web, Actions, Model.

La carpeta Web contiene las clases relacionadas con la interfaz de usuario del sistema. Están desarrolladas utilizando la librería ExtJS. La clase TWPstandardReport.js es la interfaz mediante la cual el usuario interactúa con el sistema. La misma está relacionada con el paquete Actions el cual acoge las clases de la lógica del negocio. Este paquete Actions se representan las clases del subsistema Symfony. En el mismo se encuentran los archivos viewStandardReportAction.php y viewStandardReportSucess.php.

Capítulo 3. Implementación y prueba

Estas clases son las encargadas de las funcionalidades principales del sistema.

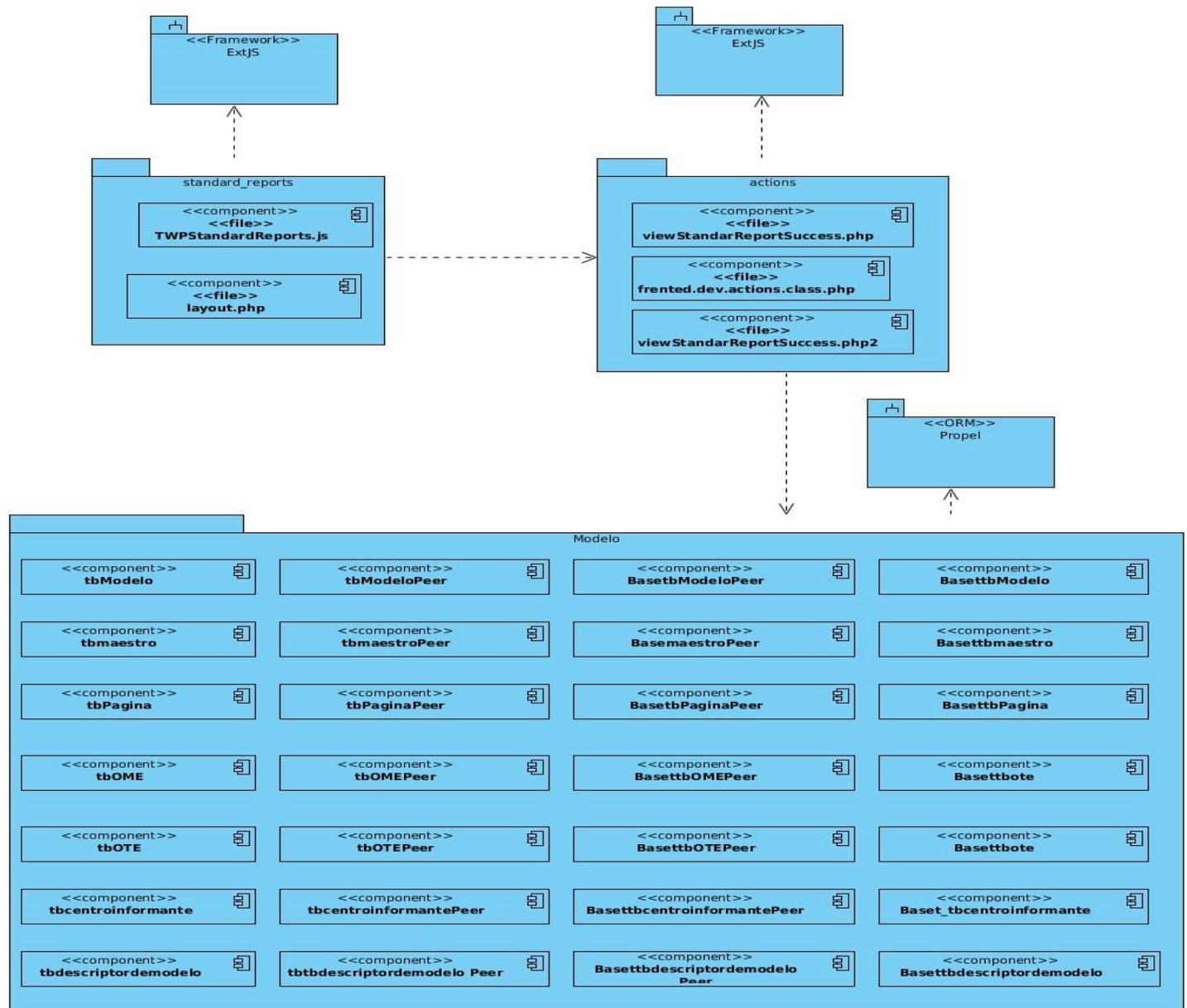


Figura 9. Diagrama de Componentes. CU_Generar Reporte Estándares.

La capa de acceso a datos está compuesta por las clases que son generadas por el ORM Propel, resultado del mapeo de la base de datos de SIGE. Propel genera automáticamente las clases de la capa del modelo, creando el esqueleto o estructura básica de las clases y genera automáticamente el código

Capítulo 3. Implementación y prueba

necesario. La abstracción de la base de datos es completamente invisible al programador, pues la realiza otro componente específico llamado Creole. En el diagrama se muestran las clases de la de SIGE utilizadas por el Módulo Generador de Reportes Estándares.

3.3. Código Fuente.

El código fuente es un conjunto de líneas de texto que son las instrucciones que debe seguir la computadora para ejecutar dicho programa. Por tanto, en el código fuente de un programa está descrito por completo su funcionamiento. Estas instrucciones son escritas en un lenguaje de programación que consiste en un conjunto de símbolos, reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones (26).

El siguiente ejemplo código muestra cómo se construye la vista HTML del reporte estándar:

```
73 <?php>
74 $resultados=$valoresSuma[$k];
75 $nombreDesgloce=(!$detail)?$nombreTabla:$resultados[0][0];
76 $informacion="<p style ='text-align:center;font-weight: bold'>".$nombreDesgloce."</p>";
77 echo $informacion;
78 $encabezado = '<tr>';
79 $cantidad = count($columnas);
80 for ($i = 0; $i < $cantidad; $i++) {
81     $fila = '<th style ="text-align:center">' . $columnas[$i] . '</th>';
82     $encabezado.=$fila;}
83 $encabezado.='</tr>';
84 echo $encabezado;
85 $filass = '<tr>';
86 $cantidadFilas = count($resultados);
87 for ($j = 0; $j < $cantidadFilas; $j++) {
88     $a= (!$detail?$resultados[$j][0]:$resultados[$j][1]);
89     $fila = '<td style ="text-align:center">' . $a . '</td>';
90     $filass.=$fila;
91     $array = $resultados[$j];
92     for ($i = (!$detail?1:2); $i < count($array); $i++) {
93         $fila = "<td style ='text-align:center'" . $array[$i] . "</td>";
94         $filass.=$fila;}
95     $filass.='</tr>|';}
96 echo $filass.($cantidadFilas==0?"<tr><td colspan=4>No existen datos captados.</td></tr>":""");
97 ?>
98
```

Figura 10. Ejemplo de código.

3.4. Estándares de Codificación.

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un

Capítulo 3. Implementación y prueba

código de alta calidad, es de gran importancia para la calidad del software y para obtener un buen rendimiento (26).

En la figura se muestra

Estilo de Codificación Utilizado:

- Todas las etiquetas php deben ser de tipo reducida (<?php>..... ?>).
- Los bloques de código siempre deben estar encerrados por llaves, si consta de una línea no es necesario utilizar llaves.
- Todas las funciones y clases deben estar comentariadas. Los comentarios deben ser añadidos de forma que resulten prácticos, para explicar el flujo del código y el propósito de las funciones o variables.

Tamaño = 4 (espacios) para:

- Declaraciones dentro de las clases.
- Enunciado dentro de métodos y funciones.
- Enunciados dentro de bloques de comandos.

3.5. Generación de Reportes Estándares.

A continuación se muestran algunas secciones de la interfaz gráfica de la aplicación encaminadas a lograr una visión agradable, sencilla y atractiva mediante la utilización del framework de presentación ExtJS.

La figura 10 muestra la interfaz a la gráfica de la aplicación correspondiente a la opción Seleccionar Plantilla, del caso de uso Generar Reporte Estándar. Donde se brinda la opción al usuario de seleccionar una plantilla dando comienzo a la creación del reporte estándar. A continuación se muestran a través de las interfaces los otros pasos para la generación de los reportes. La figura 11 refleja cómo se selecciona un maestro. Y como último paso se introducen las especificaciones como es reflejado en la figura 12.

Luego de realizar los pasos anteriores el reporte queda confeccionado como se muestra en la figura 13. En la misma se representa cómo queda estructurada el Reporte Estándar. El reporte está compuesto por un encabezado que contiene la información referente al formulario del cual se origina. El cuerpo del reporte consiste en una tabla con los valores solicitados por el usuario.

Capítulo 3. Implementación y prueba

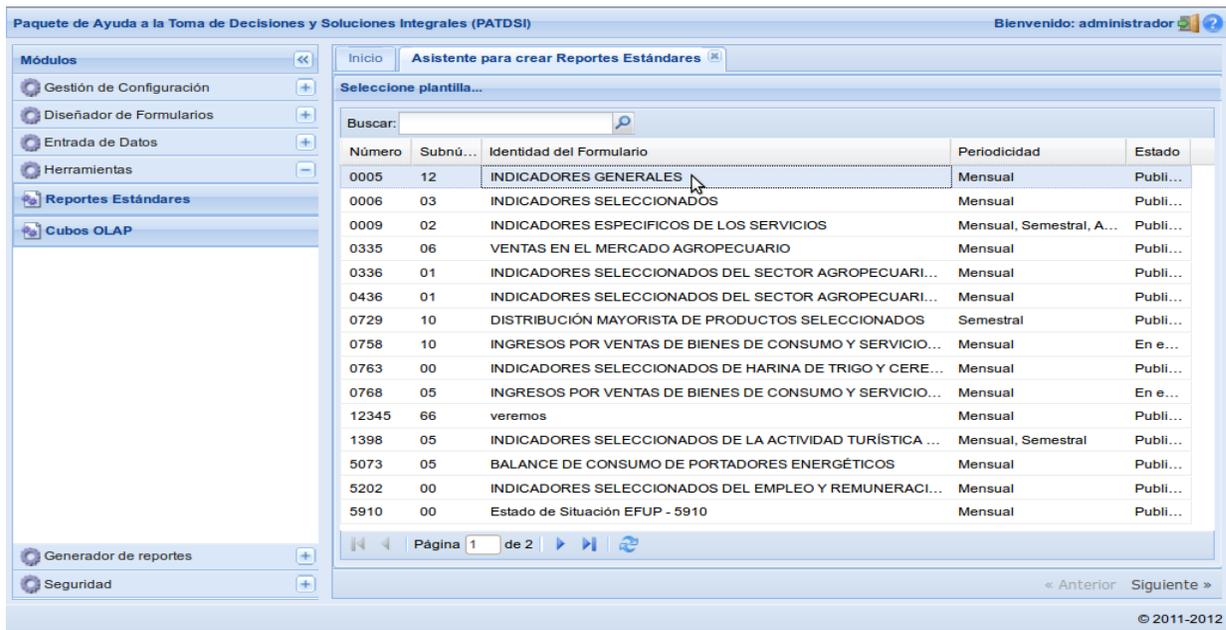


Figura 11. Vista interfaz de usuario de la funcionalidad Seleccionar Plantillas.

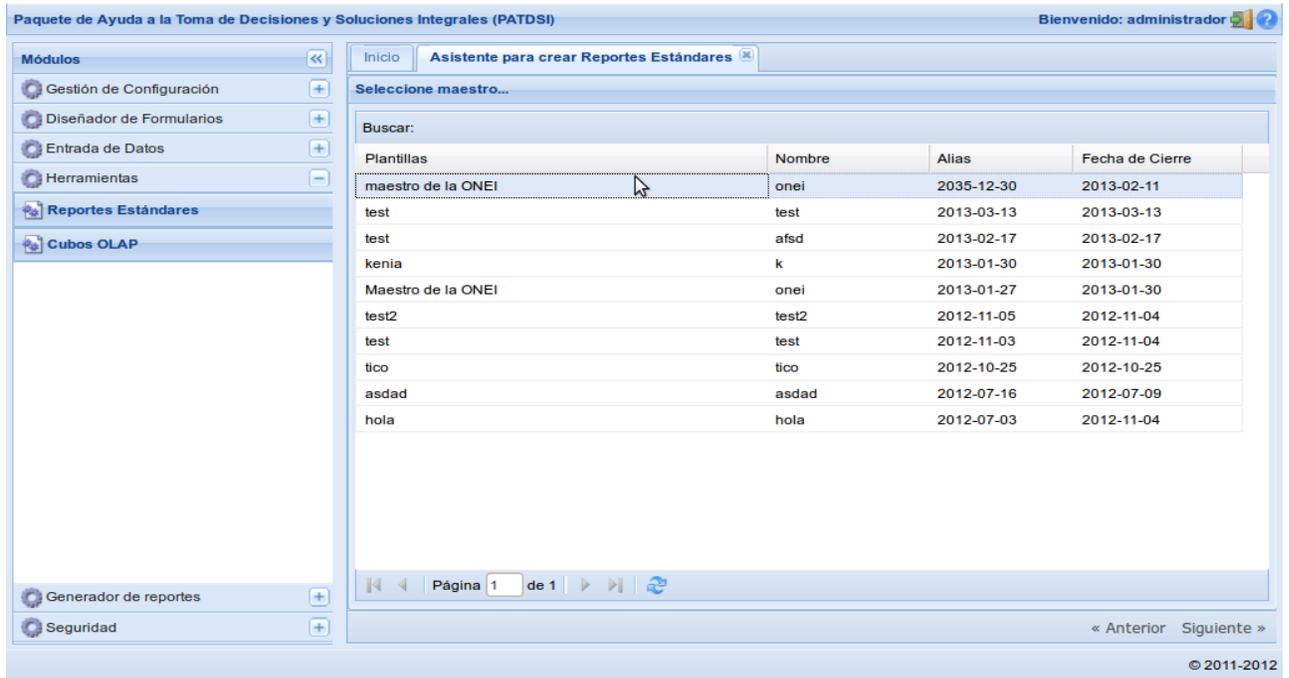


Figura 12. Vista interfaz de usuario de la funcionalidad Seleccionar Maestro.

Capítulo 3. Implementación y prueba

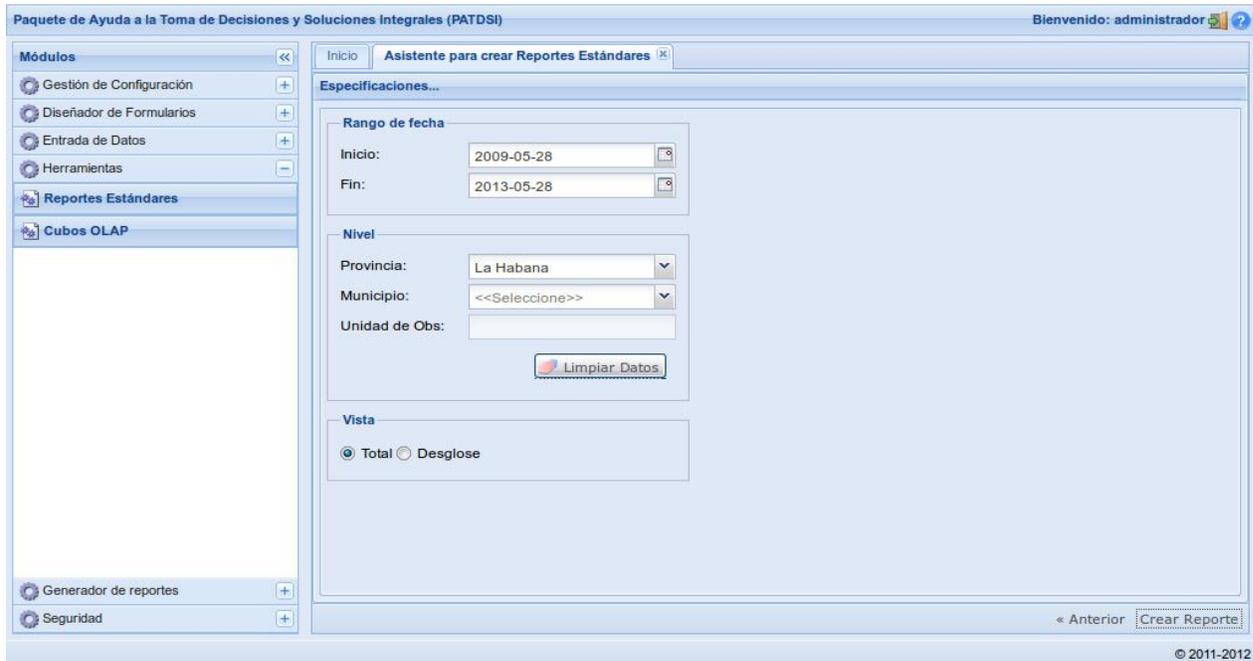


Figura 13. Vista Interfaz de Usuario funcionalidad Seleccionar Especificaciones.

Total Provincial La Habana			
codfila	m0005plan	m0005real	m0005aa
410	116.0	116.0	170.0
400	162.0	130.0	132.0
99999999	2.0	4.0	6.0
100	31.1	70.5	75.1
420	170.0	0.0	10.0
200	18.0	11.0	14.0
50	20.1	5.2	5.6

Figura 14. Ejemplo de Reporte Estándar Total Provincial de la provincia Habana.

El proceso se realiza de forma fácil y los pasos a seguir son evidentes, pues se explica mediante

Capítulo 3. Implementación y prueba

mensajes al usuario que debe hacer ante cada situación. Los escenarios sencillos no tienen muchas opciones, que puedan crear dudas o confusión en el usuario y lo conlleve a que se equivoque en los pasos y el reporte generado no coincida con el reporte precisado. Como se aprecia en las imágenes anteriores (figura 10, figura.11, figura 12 y figura 13) las interfaces de usuario solo cuentan con las opciones a seleccionar y los botones de navegación. Utilizando GDR un usuario demora en el proceso alrededor de 1 día de trabajo, mientras que con el uso de la herramienta este tiempo se simplifica a menos de 1 minuto.

3.6. Pruebas.

Las pruebas de software constituyen un pilar indispensable para evaluar y determinar la calidad de un producto. Además se puede definir como una actividad en el cual un sistema o componente es ejecutado bajo unas condiciones específicas, los resultados son observados y registrados. Las pruebas se aplican durante todo el ciclo de desarrollo del software para diferentes objetivos y en distintos niveles de trabajo. Para validar el componente implementado se ejecutan las pruebas a nivel de Pruebas de Desarrollador, ya que es la prueba diseñada e implementada por el equipo de desarrollo.

Cada nivel de prueba engloba una técnica de prueba específica según los atributos de calidad que se deseen verificar con las pruebas al software. Entre las técnicas de pruebas que se realizan en un sistema, se encuentra la técnica de Pruebas de Funcionalidad. Donde se utilizará el tipo Prueba Funcional que se encargará de evaluar el correcto desempeño de la aplicación. Este tipo de prueba, nos permite asegurar el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados.

Prueba Funcional.

Durante este nivel una de las pruebas que se aplicó fue la Prueba Funcional, comprobando que se permitiera la navegación por toda la aplicación a medida que se probara la correspondencia entre las funciones implementadas con los requisitos del cliente. Para ello se diseñaron casos de pruebas basados en los casos de uso y a su vez en los requisitos funcionales. De esta forma se compara cada funcionalidad implementada con la descrita, para verificar hasta qué punto cumplía con las necesidades del cliente. Se comprobó la correcta validación de los campos, verificando que cada cual aceptara exclusivamente los caracteres válidos. Por ejemplo en los campos donde su composición sólo era de números se verificó si el sistema permitía entrar letras o algún otro carácter, díjase /, *, -, +, %, entre otros. De esta forma se aplicó

Capítulo 3. Implementación y prueba

simultáneamente la técnica de Partición de Equivalencia del Método de Caja Negra, con la comprobación de las variables válidas e inválidas. Cada defecto encontrado se fue registrando en la plantilla de no conformidades, argumentando cada no conformidad y clasificándola según su grado de importancia en significativa o no. La siguiente tabla muestra un resumen de los resultados que se obtuvieron de la ejecución de las pruebas funcionales, donde a partir de los diseños de casos de prueba asociados a cada caso de uso se identificaron diferentes no conformidades. Estas dificultades que no corresponden con la calidad del software fueron revisadas continuamente durante el desarrollo del componente, trabajando sobre la solución de las mismas.

Tabla 4. No conformidades detectas.

Caso de Prueba	Cant. de No Conformidad	Fecha de Detección	Fecha de Resolución
CP_Generar Reporte Estándar	5	12/03/2013	18/05/2013
CP_Generar Reporte Estándar	1	19/04/2013	20/05/2013

Además de validar la completitud de los requisitos con las pruebas funcionales al software, se realizaron pruebas para evaluar el rendimiento de estos requisitos definidos, bajo condiciones prefijadas en los requisitos no funcionales.

Pruebas de Rendimiento.

Las pruebas de rendimiento se basan en comprobar que el sistema puede soportar el volumen de carga definido en la especificación, es decir, la eficiencia. Se evalúan los resultados de una prueba para un caso de uso comparándolo con varias ejecuciones de la misma y se examinan las estadísticas resumidas recaudadas para un caso de uso en busca de indicadores de variabilidad de las respuestas del sistema.

Para su diagnóstico se utilizan herramientas que monitorean el flujo de eventos de la aplicación y arrojan resultados estadísticos que tasan el desempeño del sistema. Los mismos controlan los tiempos de respuestas, la carga de usuarios, el uso de recursos, y comportamiento del software bajo determinadas condiciones de trabajo a las que se someta.

Para medir el rendimiento se utiliza el tipo de Pruebas de Carga. Una prueba de carga se realiza

Capítulo 3. Implementación y prueba

generalmente para observar el comportamiento de una aplicación bajo una cantidad de peticiones esperadas. Esta carga puede ser el número esperado de usuarios concurrentes utilizando la aplicación y que realizan un número específico de transacciones durante el tiempo que dura la carga. Esta prueba puede mostrar los tiempos de respuesta de todas las transacciones importantes de la aplicación.

Para el desarrollo de la prueba de carga se utilizó la herramienta Jmeter, la cual permite la simulación de un conjunto de usuarios accediendo simultáneamente a una misma URL. El entorno para la ejecución de las pruebas estuvo determinada por:

- PC Cliente: Pentium(R) 4, 3.0Ghz, memoria RAM de 1.0GB, disco duro de 15GB.
- Servidor de BD: Pentium(R) Dual-Core CPU T4400 2.20GHz, 3Gb de RAM, disco duro de 160GB.
- Funcionamiento de la red: 100 MBbps de velocidad de enlace.

Tras la ejecución de las pruebas para las secciones, exportar y visualizar plantilla de encuesta, fue posible comprobar el cumplimiento de los requisitos no funcionales definidos. En la siguiente tabla se muestra la prueba realizada en la aplicación JMeter.

Secciones	# Muestras	Media	Mediana	Mínimas	Máximas	% Error	Rendimiento
Generar Reporte Estándar	100	4904	1015	0	41951	20.00%	2.0/sec

3.7. Casos de prueba.

Un caso de prueba se diseña según las funcionalidades descritas en los casos de uso. El propósito que se persigue con este artefacto es lograr una comprensión común de las condiciones específicas que la solución debe cumplir. Se parte de la descripción de los casos de uso del sistema, como apoyo para las revisiones. Cada planilla de caso de prueba recoge la especificación de un caso de uso, dividido en secciones y escenarios, detallando las funcionalidades descritas en él y describiendo cada variable. Además quedan plasmados las revisiones realizadas al caso de prueba; así como un registro de todo aquello que no corresponde a la calidad del software. Se efectuaron los casos de pruebas a los 2 casos de uso del sistema, plasmándose en el expediente de proyecto de la fase de pruebas en la planilla de caso de pruebas del proyecto.

Capítulo 3. Implementación y prueba

Luego de aplicar las pruebas funcionales se arrojan diferentes resultados, pero en esencia, se busca mayor eficacia y eficiencia a la hora de encontrar defectos y verificar la calidad, de forma tal que se minimicen tiempos, costos, y se obtenga un resultado satisfactorio. El sistema desarrollado después de aplicarle las pruebas arrojó cuatro no conformidades significativas, las cuales fueron resueltas, culminando el presente trabajo con resultados satisfactorios.

Conclusiones Parciales.

Como resultado de este capítulo se obtuvieron los diagramas de componentes a través de la estructura que presenta los diagramas de clases del diseño. De esta forma se realizó la implementación del sistema en términos de componentes, ofreciendo solución a los requisitos especificados en el capítulo anterior. Se representó el modelo de datos del sistema, así como sus atributos y relaciones.

Conclusiones

CONCLUSIONES

Para el desarrollo del módulo generador de reportes estándares para SIGE:

- Se analizaron los preceptos teóricos y técnicos que sustentaron la investigación, para realizar el módulo generador de reportes estándares para SIGE, lo que permitió la selección de la metodología, las herramientas y tecnologías que se aplicaron durante la implementación del sistema.
- A partir de las funcionalidades descritas se realizó el análisis y el modelo de diseño del módulo generador de reportes estándares para SIGE, proporcionando el punto de partida para las actividades de implementación.
- Se implementó el componente para importar reportes posibilitando la creación de reportes estándares.
- Las pruebas realizadas para validar las funcionalidades que fueron implementadas, para desarrollar módulo generador de reportes estándares para SIGE, demostraron que los indicadores de calidad cumplieron satisfactoriamente con los requisitos, garantizando su correcto funcionamiento.

Recomendaciones

RECOMENDACIONES

Después de haber cumplido con los objetivos trazados en el presente trabajo de diploma se proponen la siguiente recomendación:

- Integrar Módulo Generador de Reportes Estándares a la herramienta SIGE.

Referencias bibliográficas

REFERENCIAS BIBLIOGRÁFICAS

1. Castaño, M. *El sistema de información estadística en el marco del enfoque sistémico*. Facultad de Informática Universidad Politécnica de Madrid: s.n
2. Sitio Oficial de la ONEI [En línea] 2007 [Citado el: 13 de Noviembre del 2012.] Disponible en: <http://onei.gob.cu>
3. Pérez, N. Nuñez, C. Monné, D. González, F., Lobo, A. Reyes, H García, O. *Informatización de los Procesos de Gestión Estadística en Cuba*.
4. A, A.; Ravi, S. y J, U. *Compilers: principles, techniques, and tools*. Boston, MA, USA: Addison- Wesley Longman Publishing Co., Inc, 1987. [En Línea] <http://cacs.usc.edu/education/cs653/CSCI565-Syllabus.pdf>
5. Sitio del Instituto Tecnológico de Veracruz. [En línea] 2011 [Citado el: 15 de Noviembre de 2012.]: <http://www.prograweb.com.mx/pweb/0203ladoServidor.html>
6. Gutmans A., Bakken, S., y Rethans, D. *PHP 5 Power Programming*. 2004. [En línea] <http://dl.acm.org/citation.cfm?id=123806>
7. Sitio del Instituto Tecnológico de Veracruz. [En línea] 2011 [Citado el: 20 de Noviembre del 2012.]: <http://www.prograweb.com.mx/pweb/0202ladoCliente.html>
8. Eguíluz, J. *Introducción a Java Script*. [En línea] 2007 [Citado el: 4 de Diciembre del 2012.] <http://www.librosweb.es/javascript>
9. Jacobson, I.; Grady, B. y Rumbaugh, J. *El proceso unificado de desarrollo de software*. USA: Pearson Addison-Wesley, 2000. ISBN 8478290362. [En línea] <http://en.scientificcommons.org/6960494>
10. Sitio Emuned. [En línea] 2010 [Citado el: 7 de Diciembre del 2012.]: <http://www.eumed.net/libros-gratis/2009c/584/Metodologias%20tradicionales%20y%20metodologias%20agiles.htm>
11. Asyles, K. *Openup-como-alternativa-metodologica*. [En línea] 2008. [Citado el: 7 de Diciembre del 2012.] kasyles.blogspot.com.
12. Pressman, R. *Software Engineering: a Practitioner's Approach*. The McGraw-Hill Companies, Inc. 2010 [En línea] http://guideforengineers.com/wordpress/wp-content/uploads/2009/10/se_notes.pdf
13. Sitio Oficial del Visual Paradigm. [En línea] 2009 [Citado el: 13 de Noviembre de 2012.]: <http://www.visual-paradigm.com/product/vpum!>

Referencias bibliográficas

14. Frederick, S.; Ramsay, C. y Blades, S. *Learning Ext JS*. Birmingham, UK: Packt Publishing Ltd, 2008. ISBN 978-1-847195-14-2
15. Villa, Crisfel. Ext JS: La herramienta que revolucionó el front-end. [En línea] 2012. sg.com.mx
16. Eguiluz, J. Symfony.es. [En línea] 2012. [Citado el: 13 de Noviembre de 2012.] <http://www.symfony.es/que-es-symfony>
17. Boudreau, T. *NetBeans: the definitive guide*. [En línea] 2002. books.google.com
18. Sitio PostgreSQL. [En línea] 2009. [Citado el: 20 de Noviembre de 2012.] <http://www.postgresql.org.es>.
19. Guia de Ubutuntu [Citado el: 13 de Noviembre de 2012.] http://www.guia-ubuntu.com/index.php?title=PgAdmin_III
20. Díaz Anton, Maria Gabriela. academia-interactiva.com. IV Congreso. [En línea] 2003. [Citado el: 5 de enero del 2013.] academia-interactiva.com.
21. Tecnología y Synergix. [En línea] 2009. [Citado el: 16 de Febrero de 2013.] <http://synergix.wordpress.com/2008/07/07/requisito-funcional-y-no-funcional/>.
22. Lobo, Armando Robert, Sierra, Julio Ernesto Ortiz y Pérez, Omar Ahmed García. *Sistema Integrado para la Gestión de Estadística en Cuba*. La Habana: s.n., 2008. 6ta Jornada Científica Estudiantil de la Universidad de las Ciencias Informáticas.
23. Gamma, E., Helm R., Johnson, R y Vlissides, J. *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading: Addison-Wesley, 1995 ISBN 0-201-63361-2 [En Línea] 2013 <http://www.weibnc.com/wp-content/uploads/brkpdfs/Design-Patterns-Elements-of-Reusable-Object-Oriented-Software-by-John-M.-VlissidesFoundational.pdf>
24. Bascón, E. Revista Acta Nova. [En línea] 2011. [Citado el: 16 de Febrero de 2013.]. ucbconocimiento.ucbcba.edu.bo.
25. Larman, Graig. *UML y Patrones. Introducción al análisis y diseño orientado a objetos*. La Habana: Felix Varela, 2004.

BIBLIOGRAFÍA

- JACOBSON, I.; G. BOOCH, et al. El proceso Unificado de Desarrollo de Software. p.xviii Addison Wesley Object Technology.
- LOBO, A. Arquitectura de Software para el Sistema Integrado de Gestión Estadística 2.0 Nuragas. Ciudad Habana: Trabajo de Diploma, 2009.
- PRESMAN, R. 2007. Ingeniería de software. Un enfoque práctico. 6ta Edición. Parte I. Prólogo y Capítulos 1, 2 y 21. Páginas 1-18, 22-45 y 641-657.
- WILLIAMS, I. Beginning xsl and xpath. Indianapolis, Indiana : Wiley Publishing, Inc, 2009. 978-0-470-47725-0.
- MANGANO, S. XSLT Cookbook, 2nd Edition. s.l. O'Reilly, 2005. 0-596-00974-7.
- ORTIZ, K.
<http://www.eumed.net/libros/2009c/583/Representacion%20del%20Modelo%20de%20Objetos%20de%20Dominio.html>. [En línea] [Citado el: 15 de enero de 2012.]
- POTENCIER, Fabien y François Zaninotto. Symfony La Guía Definitiva.
- TELLO, J. Diagramas de Casos de Uso. Universidad de Alcalá Departamento de Ciencias de Computación: s.n.
- Diseño, Patrones de. 2001. Patrones de Diseño. [En línea] 2001. <http://mit.ocw.universia.net/6.170/6.170/f01/pdf/lecture-12.pdf>.
- Propel. 2012. Propel project. [Citado el: 13 de Noviembre de 2012.] [En línea] 2009. <http://propel.phpdb.org/trac>.
- Sitio oficial del Visual Paradigm. Disponible en: <http://www.visual-paradigm.com/product/vpuml>.
- Hernández, Pedro Veloso. Uso de patrones de arquitectura.
- Bascón Pantoja, Ernesto. Revista Acta Nova. [En línea] 2011. ucbconocimiento.ucbcba.edu.bo.
- JACOBSON, I.; G. BOOCH, et al. El proceso Unificado de Desarrollo de Software. p.xviii Addison Wesley Object Technology.

Bibliografía

- PgAdmin_III. [En línea] guia-ubuntu.org.

ANEXOS

Anexo1. Descripción de caso de uso CU_Imprimir Reporte.

Caso de Uso:	Imprimir Reporte Estándar.
Actores:	Usuario
Resumen:	El usuario desea imprimir en papel un reporte estándar, el sistema muestra una interfaz visual con las opciones necesarias para imprimir un reporte o si el usuario cancela la operación antes de terminar.
Precondiciones:	Que el usuario este autenticado en el sistema y tenga los permisos necesarios para imprimir un reporte estándar. Que este creado el reporte estándar
Referencias	RF8
Prioridad	Secundario
Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
El usuario solicita exportar el reporte estándar.	El sistema brinda la opción de imprimir el reporte estándar en formato. El sistema muestra mediante una interfaz visual la opción para que el usuario configura para imprimir en papel el reporte.
El usuario selecciona "Aceptar".	e imprime el reporte en papel

Flujos Alternos al paso 2: El usuario cancela exportar el reporte estándar	
Acción del Actor	Respuesta del Sistema
	El sistema muestra mensaje: ¿Desea cancelar este proceso? Si: Se cancela el proceso. No: Continúa el proceso.
Poscondiciones	Se imprime el reporte en papel

Anexo2. Diagrama de Clases del Diseño. CU_Imprimir Reporte Estándar.

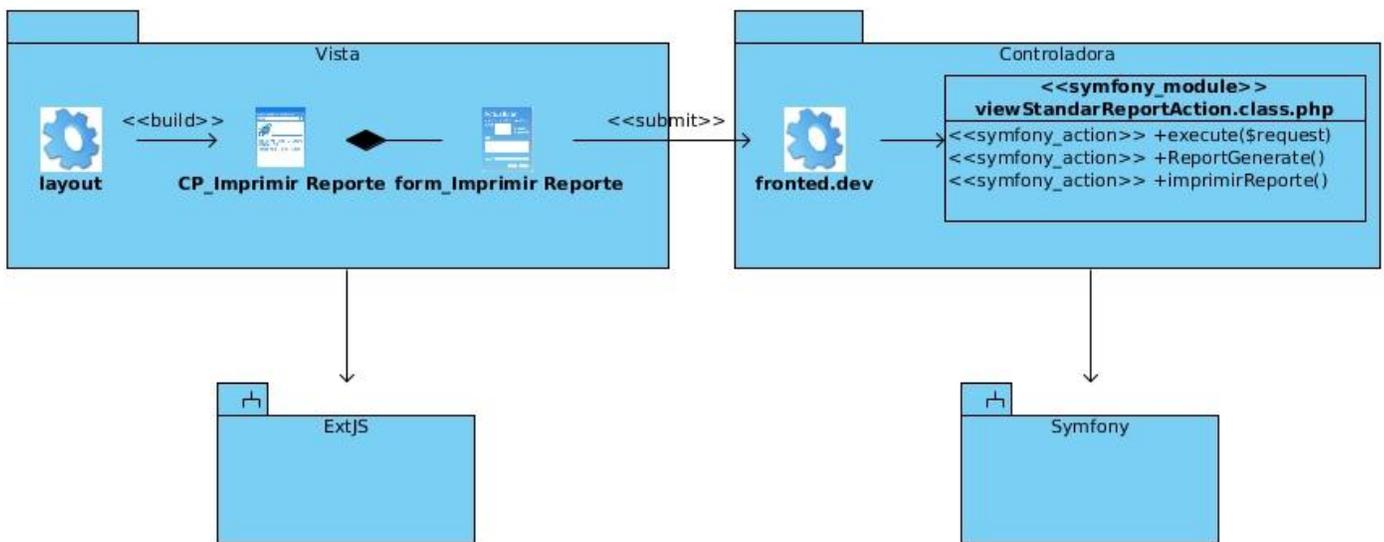


Figura 15. Diagrama de Clases del Diseño. CU_Imprimir Reportes Estádares.

Anexo3. Diagrama de Secuencia del CU_Imprimir Reporte.

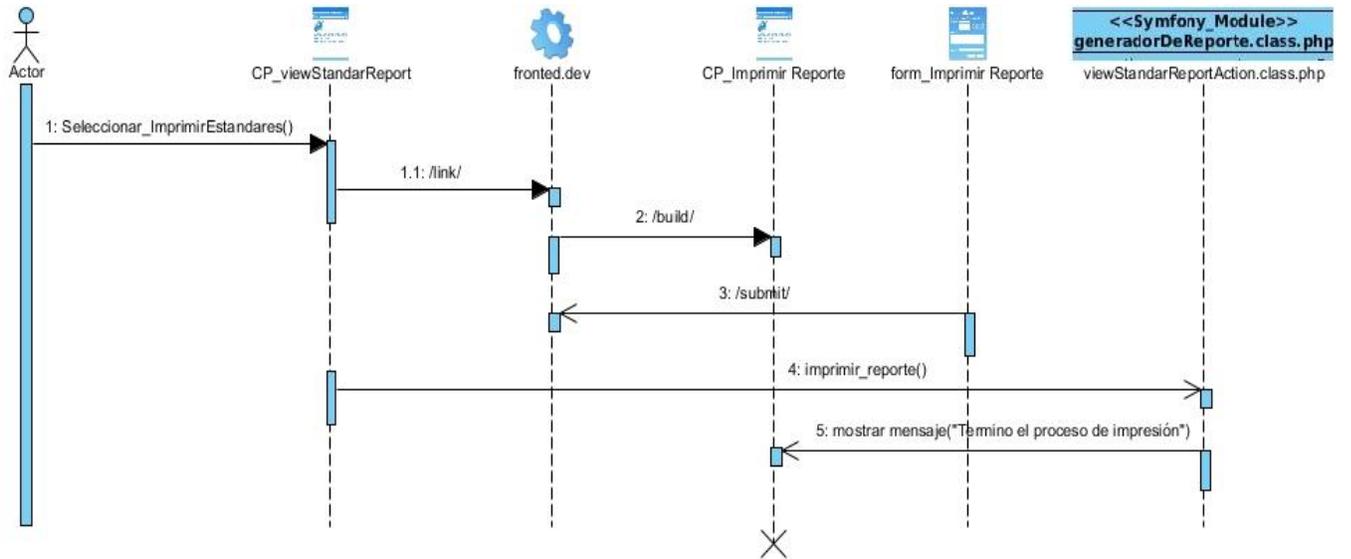


Figura 16. Diagrama de Secuencia. CU_Imprimir Reportes Estándares.

Anexo4. Diagrama de Componentes. CU_Imprimir Reporte Estándar.

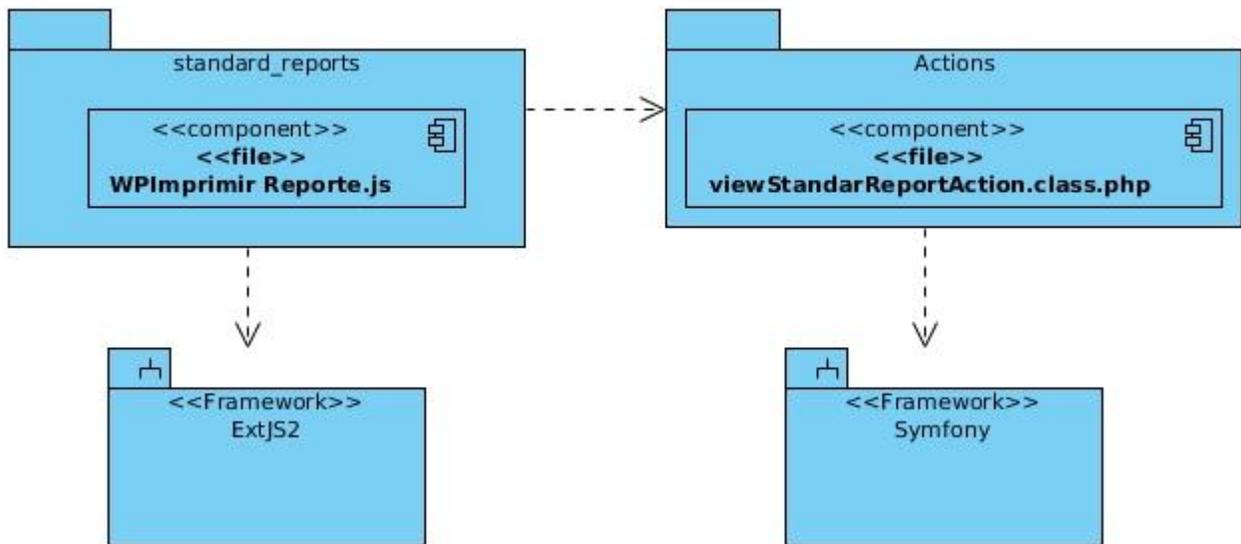


Figura 17. Diagrama de Componentes. CU_Imprimir Reportes Estándares.

Capítulo 2. Análisis y diseño del sistema

GLOSARIO

1. **BD:** Base de Datos.
2. **Framework:** Marco de trabajo.
3. **GDR:** Generador dinámico de reportes.
4. **IDE:** Entorno desarrollo integrado.
5. **OpenUp:** Metodología para el proceso del desarrollo del software.
6. **UML:** Unified modeling language (Lenguaje Unificado de Modelado).
7. **ORM:** Mapeo de Objetos a Bases de Datos.
8. **CASE:** Ingeniería de Software Asistida por Computación.
9. **GDR:** Generador de Reportes Dinámico
10. **GOF:** Acrónimo de Pandilla de los Cuatro [*Gang of Four*] formada por Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides. Son los autores del famoso libro "Design Patterns: Elements of Reusable Object Oriented Software".
11. **GRASP:** Acrónimo de Patrones de Software para la asignación General de Responsabilidades [*General Responsibility Assignment Software Patterns*]
12. **ONEI:** Oficina Nacional de Estadísticas e Información
13. **MVC:** Modelo – Vista – Controlador.
14. **XML:** (Extensible Markup Language) Lenguaje de Marcas Extensible, es un metalenguaje extensible de etiquetas.